

Contents

Acknowledgements	xi
About the Author	xiii

PART I OVERVIEW

CHAPTER 1 Introduction	3
The Problem of Rootkits and Other Types of Malware	4
Why Do You Need This Book?	6
How This Book Is Organized	6
How This Book Is Different from Other Books on Rootkits	7
Terminology Used in This Book	9
Technology Background: An Overview	10
Managed versus Unmanaged Code	11
Managed Code Environments: An Overview	12
Summary	21
CHAPTER 2 Managed Code Rootkits	23
What Can Attackers Do with Managed Code Rootkits?	24
Common Attack Vectors	26
Maintaining Access after Successful Attacks	27
The Trusted Insider	28
Malware	30
Why Are Managed Code Rootkits Attractive to Attackers?	30
MCRs Have a Large Attack Surface	30
MCRs Have a Single Control Point	31
MCRs Can Act as a Universal Rootkit	31
MCRs Are an Ideal Place to Hide Malicious Code	32
Security Products Do Not Understand Intermediate Language Bytecode	32
Developers' Backdoors Are Hidden from Code Review Audits	32
Attackers' Backdoors Can Be Planted as Deliberate Security Holes	33
Managed Code Becomes Part of the OS	34
MCRs Provide Low-Level Access to Important Methods	35
Object-Oriented Malware Has Many Implications	35
Summary	35
Endnotes	36

PART II MALWARE DEVELOPMENT

CHAPTER 3	Tools of the Trade	39
	The Compiler	40
	The Decompiler.....	42
	The Assembler.....	46
	The Disassembler.....	49
	The Role of Debuggers	52
	The Native Compiler.....	56
	File Monitors.....	60
	Summary	61
CHAPTER 4	Runtime Modification	63
	Is It Possible to Change the Definition of a Programming	
	Language?	63
	Attacking the Runtime Class Libraries	66
	Attacking the JIT Compiler.....	66
	Abusing Runtime Instrumentation Features.....	67
	Walkthrough: Attacking the Runtime Class Libraries.....	71
	Case Study: The .NET Runtime	72
	Component Analysis	73
	Disassembling the Binaries	79
	Modifying the IL Code.....	80
	Reassembling the Code	82
	Deployment	83
	Case Study: The Java Runtime.....	90
	Case Study: The Dalvik Runtime.....	94
	Summary	99
CHAPTER 5	Manipulating the Runtime	101
	Manipulating the Runtime According to Our Needs.....	101
	Logical Manipulation	102
	Execution Flow Manipulation	113
	Literal Value Manipulation.....	122
	Reshaping the Code.....	129
	Referencing External Methods and Class Members	129
	Injecting References.....	130
	Max Stack Size.....	131
	Setting the Labels	134
	Code Injection Points	137
	Code Generation.....	139
	Summary	142

CHAPTER 6	Extending the Language with a Malware API	143
	Why Should We Extend the Language?	143
	Extending the Runtime with a Malware API	146
	Sending Data to the Attacker's Machine	146
	Omitting Items from Data Containers	153
	Locating Specific Items	156
	Calling Native Code Functions	160
	Deploying Files on the Victim's Machine	162
	Launching Executables	166
	Creating a Remote Reverse Shell Tunnel	171
	Creating Denial-of-Service (DoS) Code	175
	Downloading Content to the Victim's Machine	178
	Summary	179
	Endnote	180
CHAPTER 7	Automated Framework Modification	181
	What is ReFrameworker?	182
	ReFrameworker Modules Concept	184
	The <i>Item</i> Module	185
	The <i>Payload</i> Module	189
	The <i>Method</i> Module	190
	The <i>Class</i> Module	190
	The <i>Reference</i> Module	190
	Example: Single Module Injection	191
	Using the Tool	196
	Step-by-Step Usage of ReFrameworker	196
	The Workspace Directory	205
	Developing New Modules	206
	The Modules Directory	207
	Setting Up the Tool	212
	Installation	213
	Prerequisites	213
	Configuration	213
	Current Version	216
	Summary	216
CHAPTER 8	Advanced Topics	219
	"Object-Oriented-Aware" Malware	220
	Constructors	220
	Inheritance	223
	The <i>Object</i> Class	226

Polymorphism	228
Destructors.....	231
Thread Injection	231
State Manipulation	237
Covering the Traces as Native Code	247
Cached Image Manipulation: Rebinding	
Native Code Images.....	248
Summary	257

PART III COUNTERMEASURES

CHAPTER 9 Defending against MCRs	261
What Can We Do about This Kind of Threat?	261
Awareness: Malware Is Everybody's Problem.....	263
IT System Administrators.....	263
Security Auditors.....	264
Computer Forensic Investigators.....	265
Security Product Vendors	265
OS Vendors	266
Developers	267
End Users	267
The Prevention Approach.....	268
Obfuscation and Other Antireversing Techniques.....	268
Randomized Runtime Binaries.....	271
The Detection Approach	272
Software-Based Approach.....	273
Hardware-Based Approach.....	279
The Response Approach.....	284
Looking for Clues.....	284
Gathering Evidence and Restoring the Machine.....	286
Investigating How It Got There in the First Place.....	288
Summary	289
Endnote	290

PART IV WHERE DO WE GO FROM HERE?

CHAPTER 10 Other Uses of Runtime Modification	293
Runtime Modification As an Alternative Problem-Solving	
Approach	293
Hardening the Runtime Internals	294
Virtual Patching for Applications and Bug Fixing	294

Acting from the Inside.....	295
Runtime Optimizations	296
Runtime Hardening	297
Disabling Dangerous Methods and Operations.....	298
Enforcing a Secure Coding Best Practices Policy.....	302
Setting “Secure by Default” Values.....	304
Defense in Depth.....	305
Masking Web Application Technology Using Runtime Camouflaging.....	306
Summary	310
Index.....	311

For source code and to download the ReFrameworker tool, please visit
<http://www.managedcoderoothkits.com>.