

PRACTICAL MALWARE ANALYSIS

**The Hands-On Guide to
Dissecting Malicious
Software**

by Michael Sikorski and Andrew Honig



**no starch
press**

San Francisco

PRACTICAL MALWARE ANALYSIS. Copyright © 2012 by Michael Sikorski and Andrew Honig.

All rights reserved. No part of this work may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or by any information storage or retrieval system, without the prior written permission of the copyright owner and the publisher.

16 15 14 13 12 1 2 3 4 5 6 7 8 9

ISBN-10: 1-59327-290-1

ISBN-13: 978-1-59327-290-6

Publisher: William Pollock

Production Editor: Alison Law

Cover Illustration: Hugh D'Andrade

Interior Design: Octopod Studios

Developmental Editors: William Pollock and Tyler Ortman

Technical Reviewer: Stephen Lawler

Copyeditor: Marilyn Smith

Compositor: Riley Hoffman

Proofreader: Irene Barnard

Indexer: Nancy Guenther

For information on book distributors or translations, please contact No Starch Press, Inc. directly:

No Starch Press, Inc.

38 Ringold Street, San Francisco, CA 94103

phone: 415.863.9900; fax: 415.863.9950; info@nostarch.com; www.nostarch.com

Library of Congress Cataloging-in-Publication Data

A catalog record of this book is available from the Library of Congress.

No Starch Press and the No Starch Press logo are registered trademarks of No Starch Press, Inc. Other product and company names mentioned herein may be the trademarks of their respective owners. Rather than use a trademark symbol with every occurrence of a trademarked name, we are using the names only in an editorial fashion and to the benefit of the trademark owner, with no intention of infringement of the trademark.

The information in this book is distributed on an “As Is” basis, without warranty. While every precaution has been taken in the preparation of this work, neither the authors nor No Starch Press, Inc. shall have any liability to any person or entity with respect to any loss or damage caused or alleged to be caused directly or indirectly by the information contained in it.

BRIEF CONTENTS

About the Authors	xix
Foreword by Richard Bejtlich	xxi
Acknowledgments	xxv
Introduction	xxvii
Chapter 0: Malware Analysis Primer	1

PART 1: BASIC ANALYSIS

Chapter 1: Basic Static Techniques.....	9
Chapter 2: Malware Analysis in Virtual Machines.....	29
Chapter 3: Basic Dynamic Analysis	39

PART 2: ADVANCED STATIC ANALYSIS

Chapter 4: A Crash Course in x86 Disassembly	65
Chapter 5: IDA Pro	87
Chapter 6: Recognizing C Code Constructs in Assembly.....	109
Chapter 7: Analyzing Malicious Windows Programs.....	135

PART 3: ADVANCED DYNAMIC ANALYSIS

Chapter 8: Debugging.....	167
---------------------------	-----

Chapter 9: OllyDbg	179
Chapter 10: Kernel Debugging with WinDbg	205
PART 4: MALWARE FUNCTIONALITY	
Chapter 11: Malware Behavior	231
Chapter 12: Covert Malware Launching	253
Chapter 13: Data Encoding	269
Chapter 14: Malware-Focused Network Signatures.....	297
PART 5: ANTI-REVERSE-ENGINEERING	
Chapter 15: Anti-Disassembly.....	327
Chapter 16: Anti-Debugging	351
Chapter 17: Anti-Virtual Machine Techniques	369
Chapter 18: Packers and Unpacking	383
PART 6: SPECIAL TOPICS	
Chapter 19: Shellcode Analysis	407
Chapter 20: C++ Analysis	427
Chapter 21: 64-Bit Malware.....	441
Appendix A: Important Windows Functions	453
Appendix B: Tools for Malware Analysis.....	465
Appendix C: Solutions to Labs	477
Index	733

CONTENTS IN DETAIL

ABOUT THE AUTHORS xix

About the Technical Reviewer	xx
About the Contributing Authors	xx

FOREWORD by Richard Bejtlich xxi

ACKNOWLEDGMENTS xxv

Individual Thanks	xxv
-------------------------	-----

INTRODUCTION xxvii

What Is Malware Analysis?	xxviii
Prerequisites	xxviii
Practical, Hands-On Learning	xxix
What's in the Book?	xxx

0 **MALWARE ANALYSIS PRIMER** 1

The Goals of Malware Analysis	1
Malware Analysis Techniques	2
Basic Static Analysis	2
Basic Dynamic Analysis	2
Advanced Static Analysis	3
Advanced Dynamic Analysis	3
Types of Malware	3
General Rules for Malware Analysis	5

PART 1 BASIC ANALYSIS

1 **BASIC STATIC TECHNIQUES** 9

Antivirus Scanning: A Useful First Step	10
Hashing: A Fingerprint for Malware	10
Finding Strings	11
Packed and Obfuscated Malware	13
Packing Files	13
Detecting Packers with PEiD	14
Portable Executable File Format	14
Linked Libraries and Functions	15
Static, Runtime, and Dynamic Linking	15

Exploring Dynamically Linked Functions with Dependency Walker	16
Imported Functions	18
Exported Functions	18
Static Analysis in Practice	18
PotentialKeylogger.exe: An Unpacked Executable	18
PackedProgram.exe: A Dead End	21
The PE File Headers and Sections	21
Examining PE Files with PView	22
Viewing the Resource Section with Resource Hacker	25
Using Other PE File Tools	26
PE Header Summary	26
Conclusion	26
Labs	27

2 MALWARE ANALYSIS IN VIRTUAL MACHINES 29

The Structure of a Virtual Machine	30
Creating Your Malware Analysis Machine	31
Configuring VMware	31
Using Your Malware Analysis Machine	34
Connecting Malware to the Internet	34
Connecting and Disconnecting Peripheral Devices	34
Taking Snapshots	35
Transferring Files from a Virtual Machine	36
The Risks of Using VMware for Malware Analysis	36
Record/Replay: Running Your Computer in Reverse	37
Conclusion	37

3 BASIC DYNAMIC ANALYSIS 39

Sandboxes: The Quick-and-Dirty Approach	40
Using a Malware Sandbox	40
Sandbox Drawbacks	41
Running Malware	42
Monitoring with Process Monitor	43
The Procmon Display	44
Filtering in Procmon	44
Viewing Processes with Process Explorer	47
The Process Explorer Display	47
Using the Verify Option	48
Comparing Strings	49
Using Dependency Walker	49
Analyzing Malicious Documents	50
Comparing Registry Snapshots with Regshot	50

Faking a Network	51
Using ApatеDNS	51
Monitoring with Netcat	52
Packet Sniffing with Wireshark	53
Using INetSim	55
Basic Dynamic Tools in Practice	56
Conclusion	60
Labs	61

PART 2

ADVANCED STATIC ANALYSIS

4

A CRASH COURSE IN X86 DISASSEMBLY 65

Levels of Abstraction	66
Reverse-Engineering	67
The x86 Architecture	68
Main Memory	69
Instructions	69
Opcodes and Endianness	70
Operands	70
Registers	71
Simple Instructions	73
The Stack	77
Conditionals	80
Branching	80
Rep Instructions	81
C Main Method and Offsets	83
More Information: Intel x86 Architecture Manuals	85
Conclusion	85

5

IDA PRO 87

Loading an Executable	88
The IDA Pro Interface	89
Disassembly Window Modes	89
Useful Windows for Analysis	91
Returning to the Default View	92
Navigating IDA Pro	92
Searching	94
Using Cross-References	95
Code Cross-References	95
Data Cross-References	96
Analyzing Functions	97
Using Graphing Options	98

Enhancing Disassembly	100
Renaming Locations	100
Comments	100
Formatting Operands	100
Using Named Constants	102
Redefining Code and Data	103
Extending IDA with Plug-ins	103
Using IDC Scripts	104
Using IDAPython	105
Using Commercial Plug-ins	106
Conclusion	106
Labs	107

6 RECOGNIZING C CODE CONSTRUCTS IN ASSEMBLY 109

Global vs. Local Variables	110
Disassembling Arithmetic Operations	112
Recognizing if Statements	113
Analyzing Functions Graphically with IDA Pro	114
Recognizing Nested if Statements	114
Recognizing Loops	116
Finding for Loops	116
Finding while Loops	118
Understanding Function Call Conventions	119
cdecl	119
stdcall	120
fastcall	120
Push vs. Move	120
Analyzing switch Statements	121
If Style	122
Jump Table	123
Disassembling Arrays	127
Identifying Structs	128
Analyzing Linked List Traversal	130
Conclusion	132
Labs	133

7 ANALYZING MALICIOUS WINDOWS PROGRAMS 135

The Windows API	136
Types and Hungarian Notation	136
Handles	137
File System Functions	137
Special Files	138
The Windows Registry	139
Registry Root Keys	140
Regedit	140
Programs that Run Automatically	140
Common Registry Functions	141

Analyzing Registry Code in Practice	141
Registry Scripting with .reg Files	142
Networking APIs	143
Berkeley Compatible Sockets	143
The Server and Client Sides of Networking	144
The WinINet API	145
Following Running Malware	145
DLLs	145
Processes	147
Threads	149
Interprocess Coordination with Mutexes	151
Services	152
The Component Object Model	154
Exceptions: When Things Go Wrong	157
Kernel vs. User Mode	158
The Native API	159
Conclusion	161
Labs	162

PART 3

ADVANCED DYNAMIC ANALYSIS

8	
DEBUGGING	167
Source-Level vs. Assembly-Level Debuggers	168
Kernel vs. User-Mode Debugging	168
Using a Debugger	169
Single-Stepping	169
Stepping-Over vs. Stepping-Into	170
Pausing Execution with Breakpoints	171
Exceptions	175
First- and Second-Chance Exceptions	176
Common Exceptions	176
Modifying Execution with a Debugger	177
Modifying Program Execution in Practice	177
Conclusion	178

9	
OLLYDBG	179
Loading Malware	180
Opening an Executable	180
Attaching to a Running Process	181
The OllyDbg Interface	181
Memory Map	183
Rebasing	184
Viewing Threads and Stacks	185
Executing Code	186

Breakpoints	188
Software Breakpoints	188
Conditional Breakpoints	189
Hardware Breakpoints	190
Memory Breakpoints	190
Loading DLLs	191
Tracing	192
Standard Back Trace	192
Call Stack	193
Run Trace	193
Tracing Poison Ivy	193
Exception Handling	194
Patching	195
Analyzing Shellcode	196
Assistance Features	197
Plug-ins	197
OllyDump	198
Hide Debugger	198
Command Line	198
Bookmarks	199
Scriptable Debugging	200
Conclusion	201
Labs	202

10

KERNEL DEBUGGING WITH WINDBG 205

Drivers and Kernel Code	206
Setting Up Kernel Debugging	207
Using WinDbg	210
Reading from Memory	210
Using Arithmetic Operators	211
Setting Breakpoints	211
Listing Modules	212
Microsoft Symbols	212
Searching for Symbols	212
Viewing Structure Information	213
Configuring Windows Symbols	215
Kernel Debugging in Practice	215
Looking at the User-Space Code	215
Looking at the Kernel-Mode Code	217
Finding Driver Objects	220
Rootkits	221
Rootkit Analysis in Practice	222
Interrupts	225
Loading Drivers	226
Kernel Issues for Windows Vista, Windows 7, and x64 Versions	226
Conclusion	227
Labs	228

PART 4

MALWARE FUNCTIONALITY

11

MALWARE BEHAVIOR

231

Downloaders and Launchers	231
Backdoors	232
Reverse Shell	232
RATs	233
Botnets	234
RATs and Botnets Compared	234
Credential Stealers	234
GINA Interception	235
Hash Dumping	236
Keystroke Logging	238
Persistence Mechanisms	241
The Windows Registry	241
Trojanized System Binaries	243
DLL Load-Order Hijacking	244
Privilege Escalation	245
Using SeDebugPrivilege	246
Covering Its Tracks—User-Mode Rootkits	247
IAT Hooking	248
Inline Hooking	248
Conclusion	250
Labs	251

12

COVERT MALWARE LAUNCHING

253

Launchers	253
Process Injection	254
DLL Injection	254
Direct Injection	257
Process Replacement	257
Hook Injection	259
Local and Remote Hooks	260
Keyloggers Using Hooks	260
Using SetWindowsHookEx	260
Thread Targeting	261
Detours	262
APC Injection	262
APC Injection from User Space	263
APC Injection from Kernel Space	264
Conclusion	265
Labs	266

13	DATA ENCODING	269
The Goal of Analyzing Encoding Algorithms		270
Simple Ciphers		270
Caesar Cipher		270
XOR		271
Other Simple Encoding Schemes		276
Base64		277
Common Cryptographic Algorithms		280
Recognizing Strings and Imports		281
Searching for Cryptographic Constants		282
Searching for High-Entropy Content		283
Custom Encoding		285
Identifying Custom Encoding		285
Advantages of Custom Encoding to the Attacker		288
Decoding		288
Self-Decoding		288
Manual Programming of Decoding Functions		289
Using Instrumentation for Generic Decryption		291
Conclusion		294
Labs		295

14	MALWARE-FOCUSED NETWORK SIGNATURES	297
Network Countermeasures		297
Observing the Malware in Its Natural Habitat		298
Indications of Malicious Activity		298
OPSEC = Operations Security		299
Safely Investigate an Attacker Online		300
Indirection Tactics		300
Getting IP Address and Domain Information		300
Content-Based Network Countermeasures		302
Intrusion Detection with Snort		303
Taking a Deeper Look		304
Combining Dynamic and Static Analysis Techniques		307
The Danger of Overanalysis		308
Hiding in Plain Sight		308
Understanding Surrounding Code		312
Finding the Networking Code		313
Knowing the Sources of Network Content		314
Hard-Coded Data vs. Ephemeral Data		314
Identifying and Leveraging the Encoding Steps		315
Creating a Signature		317
Analyze the Parsing Routines		318
Targeting Multiple Elements		320
Understanding the Attacker's Perspective		321
Conclusion		322
Labs		323

PART 5

ANTI-REVERSE-ENGINEERING

15

ANTI-DISASSEMBLY 327

Understanding Anti-Disassembly	328
Defeating Disassembly Algorithms	329
Linear Disassembly	329
Flow-Oriented Disassembly	331
Anti-Disassembly Techniques	334
Jump Instructions with the Same Target	334
A Jump Instruction with a Constant Condition	336
Impossible Disassembly	337
NOP-ing Out Instructions with IDA Pro	340
Obscuring Flow Control	340
The Function Pointer Problem	340
Adding Missing Code Cross-References in IDA Pro	342
Return Pointer Abuse	342
Misusing Structured Exception Handlers	344
Thwarting Stack-Frame Analysis	347
Conclusion	349
Labs	350

16

ANTI-DEBUGGING 351

Windows Debugger Detection	352
Using the Windows API	352
Manually Checking Structures	353
Checking for System Residue	356
Identifying Debugger Behavior	356
INT Scanning	357
Performing Code Checksums	357
Timing Checks	357
Interfering with Debugger Functionality	359
Using TLS Callbacks	359
Using Exceptions	361
Inserting Interrupts	362
Debugger Vulnerabilities	363
PE Header Vulnerabilities	363
The OutputDebugString Vulnerability	365
Conclusion	365
Labs	367

17

ANTI-VIRTUAL MACHINE TECHNIQUES 369

VMware Artifacts	370
Bypassing VMware Artifact Searching	372
Checking for Memory Artifacts	373

Vulnerable Instructions	373
Using the Red Pill Anti-VM Technique	374
Using the No Pill Technique	375
Querying the I/O Communication Port	375
Using the str Instruction	377
Anti-VM x86 Instructions	377
Highlighting Anti-VM in IDA Pro	377
Using ScoopyNG	379
Tweaking Settings	379
Escaping the Virtual Machine	380
Conclusion	380
Labs	381

18

PACKERS AND UNPACKING 383

Packer Anatomy	384
The Unpacking Stub	384
Loading the Executable	384
Resolving Imports	385
The Tail Jump	386
Unpacking Illustrated	386
Identifying Packed Programs	387
Indicators of a Packed Program	387
Entropy Calculation	387
Unpacking Options	388
Automated Unpacking	388
Manual Unpacking	389
Rebuilding the Import Table with Import Reconstructor	390
Finding the OEP	391
Repairing the Import Table Manually	395
Tips and Tricks for Common Packers	397
UPX	397
PECompact	397
ASPack	398
Petite	398
WinUpack	398
Themida	400
Analyzing Without Fully Unpacking	400
Packed DLLs	401
Conclusion	402
Labs	403

PART 6 SPECIAL TOPICS

19

SHELLCODE ANALYSIS 407

Loading Shellcode for Analysis	408
--------------------------------------	-----

Position-Independent Code	408
Identifying Execution Location	409
Using call/pop	409
Using fnstenv	411
Manual Symbol Resolution	413
Finding kernel32.dll in Memory	413
Parsing PE Export Data	415
Using Hashed Exported Names	417
A Full Hello World Example	418
Shellcode Encodings	421
NOP Sleds	422
Finding Shellcode	423
Conclusion	424
Labs	425

20

C++ ANALYSIS

427

Object-Oriented Programming	427
The this Pointer	428
Overloading and Mangling	430
Inheritance and Function Overriding	432
Virtual vs. Nonvirtual Functions	432
Use of Vtables	434
Recognizing a Vtable	435
Creating and Destroying Objects	437
Conclusion	438
Labs	439

21

64-BIT MALWARE

441

Why 64-Bit Malware?	442
Differences in x64 Architecture	443
Differences in the x64 Calling Convention and Stack Usage	444
64-Bit Exception Handling	447
Windows 32-Bit on Windows 64-Bit	447
64-Bit Hints at Malware Functionality	448
Conclusion	449
Labs	450

A

IMPORTANT WINDOWS FUNCTIONS

453

B

TOOLS FOR MALWARE ANALYSIS

465

C **SOLUTIONS TO LABS**

477

Lab 1-1	477	Lab 13-1	607
Lab 1-2	479	Lab 13-2	612
Lab 1-3	480	Lab 13-3	617
Lab 1-4	481	Lab 14-1	626
Lab 3-1	482	Lab 14-2	632
Lab 3-2	485	Lab 14-3	637
Lab 3-3	490	Lab 15-1	645
Lab 3-4	492	Lab 15-2	646
Lab 5-1	494	Lab 15-3	652
Lab 6-1	501	Lab 16-1	655
Lab 6-2	503	Lab 16-2	660
Lab 6-3	507	Lab 16-3	665
Lab 6-4	511	Lab 17-1	670
Lab 7-1	513	Lab 17-2	673
Lab 7-2	517	Lab 17-3	678
Lab 7-3	519	Lab 18-1	684
Lab 9-1	530	Lab 18-2	685
Lab 9-2	539	Lab 18-3	686
Lab 9-3	545	Lab 18-4	689
Lab 10-1	548	Lab 18-5	691
Lab 10-2	554	Lab 19-1	696
Lab 10-3	560	Lab 19-2	699
Lab 11-1	566	Lab 19-3	703
Lab 11-2	571	Lab 20-1	712
Lab 11-3	581	Lab 20-2	713
Lab 12-1	586	Lab 20-3	717
Lab 12-2	590	Lab 21-1	723
Lab 12-3	597	Lab 21-2	728
Lab 12-4	599		

INDEX

733