

Topics

Chapters

1.1 Introduction to Malware Analysis	1–4
1.2 Malware Analysis Lab	1–24
1.3 Static Properties Analysis	1–38
1.4 Behavioral Analysis Essentials	1–52
1.5 Code Analysis Essentials	1–79
1.6 Exploring Network Interactions	1–112
2.1 Core Reversing Concepts	2–4
2.2 Reversing Functions	2–61
2.3 Control Flow In-Depth	2–98
2.4 API patterns in Malware	2–136
2.5 64-bit Code Analysis	2–145
3.1 Malicious PDF Files	3–3
3.2 VBA Macros in MS Office Documents	3–37
3.3 Examining Malicious RTF Files	3–80
3.4 Deobfuscating Malicious JavaScript	3–102
4.1 Recognizing Packed Malware	4–3
4.2 Getting Started with Unpacking	4–15
4.3 Using Debuggers for Dumping	4–27
4.4 Debugging Packed Malware	4–44
4.5 Analyzing Multi-Technology Malware	4–56
4.6 Examining .NET Malware	4–106
4.7 Understanding Code Injection	4–133
5.1 Debugger Detection and Data Protection	5–3
5.2 Unpacking Process Hollowing	5–43
5.3 Detecting the Analysis Toolkit	5–61
5.4 Handling Misdirection Techniques	5–94
5.5 Unpacking by Anticipating Actions	5–144

Function

AllocateVirtualMemory	5–27, 4–137
Assembly.LoadFile	4–115
Assembly.LoadFrom	4–115
BlockInput	5–75, 5–91
CheckRemoteDebuggerPresent	5–12
CreateMutex	2–143
CreateProcess	5–52, 2–86
CreateRemoteThread	4–136–150
CreateThread	4–137
CreateToolHelp32Snapshot	5–84, 5–91, 4–136
CreateUserThread	4–137
CryptDecrypt	4–51, 4–98, 4–102
EnumProcesses	4–137
FindResource	2–140
FindWindow	5–77
GetForegroundWindow	5–65
GetLocalTime	5–14
GetModuleHandle	5–69, 5–77
GetProcAddress	5–148
GetSystemTime	5–14
GetTempFileName	2–62
GetTempPath	2–121
GetTickCount	5–14, 5–65
HttpAddRequestHeaders	2–46
HttpOpenRequest	2–46
HttpSendRequest	2–46
InternetConnect	2–46
InternetOpen	2–55, 2–159

InternetReadFile	2–46
IsDebuggerPresent	5–6, 5–12
KdDebuggerEnabled	5–91
LoadLibrary	5–51, 5–145, 4–151
LoadResource	2–140
LockResource	2–140
OpenProcess	4–136, 4–143
OutputDebugString	5–12
Process32First	5–84, 4–136
Process32Next	5–84, 4–136
ProtectVirtualMemory	5–27
QueryInformationProcess	4–137
QuerySystemInformation	4–137
QuerySystemTime	5–14
ReadFile	4–46, 1–95
RegOpenKey	2–21, 4–87–90, 4–137
regsvr32	3–53
ResumeThread	5–52
RtlDecompressBuffer	5–27
SetWindowsHookEx	5–68–69
ShellExecute	2–153
SizeofResource	2–140
UnmapViewOfSection	5–52, 5–59
URLDownloadToFileA	3–45, 3–52
VirtualAlloc	3–73, 4–75, 4–92, 5–145
VirtualProtect	5–153
wcsicmp	2–106
WriteProcessMemory	5–53, 4–136–138
WriteVirtualMemory	5–27, 4–137

Instructions

Overview	
ADD	2–38
AND	2–38, 2–41
CALL	2–38, 2–40
CMP	2–41, 2–55
IMUL	2–38
Jcc instructions	2–42
JA	2–42
JB	2–42
JC	2–106
JE	2–42
JG	2–42
JL	2–42
JMP	2–22, 2–37, 2–39
JNGE	2–42
JZ	2–42
LEAVE	2–68
LOOP	2–39, 2–109
LOOPcc	2–109
LOOPnz	2–109
MOV	2–38, 2–76
NOP	3–91
OR	2–38
PUSH	2–38
RDTSC	5–14
RET	2–38, 2–68
SCASB	2–23
SHL	2–38

SHR	2–38
SUB	2–38, 2–41
TEST	2–41
XCHG	2–37
Register	
AH (8 Bit)	2–27
AL (8 Bit)	2–27
AX (16 Bit)	2–27
CS	2–26
DS	2–26
EAX (32 Bit)	2–25
EBP	2–25, 2–67
EBX	2–24
ECX	2–25
EDI	2–25
EDX	2–24
EIP	2–26
ES	2–26
ESI	2–25
ESP	2–25
FS	2–26
GS	2–26, 5–99, 5–129
RAX (64 Bit)	2–149
SS	2–26

#

1768	3–76
------------	------

A

AcroForm	3–4
ActiveXObject	3–106, 3–108
Addressing memory	2–32–33
Addressing mode	2–23
Administrative rights	1–41
Adobe Reader	3–4
ADS	39
advapi32.dll	4–80, 4–87
AI	1–21
Alternate Data Stream	39
AMSI	3–109–110
Analysis detection	5–92
And	2–123–124
Anti-analysis	5–166
Anti-Debugger	5–6
Countermeasure	5–9
Timing	5–14
Antimalware San Interface(AMSI)	3–109–110
Any.run	1–13
API	2–18
Hooking	4–135
Native	4–137
Archive (self-extracting)	3–98
Arguments	2–81–82
64-bits	2–149
Armadillo	4–8
ASLR	4–18, 5–54, 5–73

.NET assembly	4–107
Automated analysis	1–13
AutoRun	4–63, 4–65
AV detection	5–77
Avast Antivirus	5–81
avghookx.dll	5–77

B

Balbuzard toolkit	5–19
Base64	4–72
base64dump.py	3–71, 4–73, 3–74
BAT File	4–65
bbcrack.py	5–19
Beaconing	1–17
Behavioral Analysis	1–35
BeingDebugged (flag)	5–12
BHOs	2–147
Binary Ninja	2–6, 1–81
binee	1–83
64-bits	2–149
BlockInput	5–75
box-js	3–118
Branches	2–39–41
Breakpoint	1–91
Countermeasure	5–167
Hardware	5–106–109
Memory	5–158
Browser Helper Objects	2–147
brutxor.py	5–18
brxor.py	5–18
Burp	3–19
Byte (8 Bit)	2–28
Bytehist	4–11

C

C2	1–17, 1–113
Call Stack	5–32, 4–89
Calling convention	2–72
capa	5–45–46, 1–83, 1–87–89
CAPE	1–13
Capstone	2–6
cdecl	2–72, 2–74, 2–85
CFF Explorer	4–19, 5–73
ASLR	4–19–20
ChatGPT	1–21
Common Intermediate Language (CIL)	4–107
Clonezilla	1–34
Code analysis	1–35
Dynamic	1–82
Static	1–82
Code Injection	5–59
Code lifecycle	2–5
Command and control (C2)	1–17
Common Intermediate Language (CIL)	4–107
Compiler	2–5

Just-In-Time Compiler	4–107
Composite Document File V2 (CDFV2)	3–40, 3–65
Compound File Binary Format (CFBF)	3–40
Constant	2–36
CREATE_SUSPENDED	5–46
CrowdStrike	1–15
CryptDecrypt	1–98
CScript	3–110, 3–118
CurrentVersion/Run	1–45
Cutter	2–6
CyberChef	1–105–109

D

.data	2–12
Data Structures	2–34, 2–93
dd	1–34
de4dot	4–108
de4ot	4–129
Debugger	1–81
Detection	5–79
Decompiler vs Dissassembly	2–158
Deep Freeze	1–34
Deobfuscation	3–105
Dereferencing	2–29–30
Detect It Easy	5–96
Detect It Easy (DIE)	4–12, 1–49
Device Driver	2–148
diec	4–12
Disassembling	1–81
DLL	1–14, 2–137
DNS	2–45, 1–67
DNS leakage	1–19
dnSpyEx	4–108, 4–113, 4–117
Dump Variable	4–126
Locals	4–126
Modules	4–120
Do (loop)	2–112
DotDumper	4–108, 4–129
Dropper	2–140
Dumping	4–23
Debugger	4–27
Dword (32 Bit)	2–28
DynamicBase	5–73, 5–97
Dyre	5–65

E

EA → Effective Address	
Effective Address (EA)	2–31
EFLAGS	2–26
emulating code execution	1–82–84
Entropy	4–9
PeStudio	4–10
Entry Point	4–7, 4–24–25, 5–111, 5–120
Original	4–29, 4–37, 5–105
Epilogue	2–84, 2–91

evilclippy	3–57
Exception	
Frame-based	5–98
Ignore Exception x32dbg	5–102
Stack-based	5–98
Executable File	2–5
Exeinfo PE	4–12, 1–49
Exfiltration	1–17
ExifTool	1–53, 3–54
Exploit vs Vulnerability	3–85
Ghidra	
Exports	2–16

F

fakedns	1–68, 1–120
fastcall	2–73, 2–75
feh	3–43
Fiddler	3–19, 3–21, 1–124–125
File extension	4–66
File handle	3–95
FileScan.IO	1–13
FLOSS	5–25–26
Fog	1–34
For (loop)	2–112, 113
FoxIT Reader	3–4
Frame pointer	2–68
FLG_HEAP_ENABLE_FREE_CHECK	5–13
FS:[0]	5–99
FS:[30h]	3–94
FS:[30h] → Process Environment Block	5–12–13
FSG	4–8
FTP	1–120
Function	2–61, 2–65
Epilogue	2–67
Prologue	2–67
Function Call Graph	4–148–149
Function Call Trees	4–145
Ghidra	
Function Graph	2–14

G

GetEIP	3–92
Ghidra	2–6, 1–81
Bad instruction	5–119
Comment	2–44
Convert	5–22
Create Project	2–8
Data Convert	2–50
Data Type	2–48
Defined Strings	2–103
Equate	2–36
Exports	2–16, 2–139
Function Call Tree	2–56
Function Graph	2–14
Functions	2–58

Headers	2-12
Imports	2-17, 2-140
Key binding	5-23
Local Variable Conflict	2-94
Program Tree	2-12
Project Analysis	2-10
Propagate External Parameters	2-48
Sections	2-12
Show References	2-80
String Reference	2-105
Symbol reference	2-18
Symbol References	4-140
TLS	5-117
Global Descriptor Table (GDT)	5-127

H

Handle	1-95-96
Hardware breakpoint	4-52
HashSets	1-13
Ghidra	
Headers	2-12
Hex encoding	4-75
Honeyclient	3-19
Honeypot	1-20
Hopper	2-6, 1-81
Hot patching	2-66
HTML	3-106
HTTP	2-45
httpd	1-72
Hybrid Analysis	1-13, 1-15

I

IAT → Import Address Table	
IDA	2-6, 1-81
If-Else	2-99-102
ILSpy	4-108, 4-111, 4-113, 4-128
ImageBase	4-18
Import Address Table (IAT)	4-7, 2-17, 4-22, 4-24, 5-164
Fix (Scylla)	4-24, 4-40-41
Rebuild	5-112
Imports	1-47, 5-146
Fixer	4-22
Ghidra	
Imports	2-17
Indicator of Compromise (IOC)	1-17, 1-45
INetSim	1-120
logfile	1-123
Injection	4-150-151
Inline functions	2-77
Instructions	2-23, 2-38
Intermodular call	4-50
Internet access	1-18
Intezer Analyze	1-13
IP redirection	1-127

iptables	1-127
IRC	2-45, 1-120

J

JavaScript	
Beautify	3-104
javascript	4-68
Jcc instructions	2-42
js debugger	3-118
js eval	3-105
js-beautify.js	3-104
Jump table	2-131
Jumps	2-39-41
Just-In-Time Compiler	4-107

K

KdDebuggerFlag	5-79
----------------	------

L

libemu	3-75
Linker	2-5
Loader	2-5
logman	3-110
Loop	2-106-112
loop assembly	2-116-120

M

Macro	
AutoOpen	3-46
Debug	3-57
Enable Content	3-42
Extraction	3-44
Malware	1-5
64-bit Malware	2-147
Malware analysis	1-5
input/output	1-10
stage	1-9
Malware Hash Registry	1-13
Malware lab	1-25
Configuration	1-26
Isolation	1-29
Malware report	1-11
MalwareBazaar	1-13
Memory map	4-48
MetaDefender	1-13
mshta.exe	4-67-68
Multithread process	2-66
Mutant → Mutex	
Mutex	1-15, 2-143
MZ	2-142

N

NanoLocker	5-141
------------	-------

nslookup	1-68
NtGlobalFlag	5-13
NTVDM	2-148

O

Object Code	2-5
objects.js	3-107, 3-114-117
OLE1	3-81
OLE2	3-40
oledump	3-47-49, 3-66-70, 3-85-88
olevba.py	3-45
OllyDbg	5-78
OllyDumpEx	4-36, 5-111
OOXML	3-40
Open Threat Exchange	1-13, 1-16
Operand (of Instructions)	2-23
Operation (of Instructions)	2-23
Or	2-123, 2-125
Original Entry Point (OEP)	4-29
OSINT	1-17

P

P2P	2-45
Packing	4-5-6, 1-49
Unpacking .NET malware	4-131
Page Memory Rights	4-38, 4-147, 5-158
Patching	5-9, 5-130
PDF	
/AA	3-5, 6
/Annots	3-33-34
/EmbeddedFiles	3-4
/JS	3-4
/JavaScript	3-4
/Launch	3-4
/ObjStm	3-29
/OpenAction	3-8
/SubmitForm	3-4
/URI	3-4, 3-8
/XFA	3-4
File structure	3-5
keywords	3-4
Object	3-5
object references	3-34
Stream	3-6
pdf-parser.py	3-9-17, 3-32
pdfid.py	3-8
pdftk	3-35
PE (Portable Executable)	1-49
PE Tools	4-22
pe_unmapper	5-163
PECompact	5-96
peframe	1-48
Persistence	1-45
pestr	1-44
PEStudio	5-96
PeStudio	1-46

.NET	4-110
ASLR	4-18
Packed	4-10
Ressource Dumping	2-142
TLS	5-113
PeStudio (DLL)	2-138
Pinpoint	3-19
Pointer	2-29-30
POP3	1-120
Position-independent code (PIC)	2-149
PowerShell	4-72
Debug	4-76-78
WriteAllBytes	4-79
ProcDOT	1-53, 4-61-64, 1-64-66
Process Environment Block (PEB)	5-12-13, 3-94
Process Hacker	5-44, 1-53, 54
String	4-21
Process hollowing	
Dump	5-52-57
Process Monitor	5-44, 1-53, 1-62-63
Process Tree	5-45
Process replacement	5-52
Prologue	2-83, 2-90
PXE booting	1-34

Q

Qiling	1-83
qpdf	3-35
Quttera	1-13
Qword (64 Bit)	2-28

R

Radare	2-6, 1-81
.rdata	2-12
Reflective code loading	4-114-116
reg_export	4-69
Register	2-25-27
64-bits	2-149
Edit	5-9
Register-based CPU	2-23
Registers	2-23
x86 Registers	2-25
Registry	4-65, 4-68-69
Registry persistence	1-45
Regshot	1-53, 1-56, 1-60-61
regsvr32	3-53
.reloc	2-12
Return Pointer	2-66
RIP-relative addressing	2-149
ROL	1-104
RollBack Rx	1-34
RootKit	4-135, 2-147
ROR	1-104
RTF	3-80
Auto extraction	3-81

Controls/Group/Object	3-82
rtfdump.py	3-83-84, 3-90
rtfobj.py	3-84
RunPE	5-52

S

Sandbox detection	5-62, 5-65, 5-92
Sandboxie	5-81
scdbg	4-76, 4-79, 3-95-96
Debug	4-83
scdbg (PS emulator)	3-75
Scout	3-19
Scylla	4-23, 5-112
IAT	4-24, 5-164
ScyllaHide	5-15, 5-28
Sections	4-7, 1-47
Ghidra	
Sections	2-12
SecurityTrails	1-13, 1-16
Segment	2-25
Segment register	5-127
selector	5-127
Self-defense of malware	1-30
countermeasures	1-31
set-static-ip	1-126
SetBPX	1-93
setdllcharacteristics	4-19-20
Shodan	1-13
Single-step	4-29, 1-91
SMTP	1-120
Snapshot	1-33
Speakeasy	1-83, 1-86
SRE → Software Reversing Engineering	
SRP Stream	3-54
Stack	2-68-70
Pointer	2-68
Stack String → String,stack	5-21
Static properties	1-43
Analysis	1-35
stdcall	2-72, 2-74
strace	2-19
strdeob.pl	5-24
String	2-103
Process Hacker	4-21
Stack	5-21
Strings	2-34, 1-44
strlen	2-111
Structured Exception Handling (SEH)	5-98
Structured Storage (SS)	3-40
Data Structures	2-34
Switch	2-130-132
SYN	1-119
SysAnalyzer	5-81
System call	2-18

T

FLG_HEAP_ENABLE_TAIL_CHECK	5-13
.text	2-12
TFTP	1-120
Themida	4-8
thiscall	2-73, 2-75
Thread Information Block (TIB)	3-94, 5-99
Thread Local Storage (TLS)	5-115
ThreatAnalyzer	5-81
ThreatFox	1-13
Thug	3-19
Tinba	5-65
TitanMist	4-17
TOR	1-19
translate.py	3-76
TrickBot	5-163
Trid	1-48
trid	4-84
try/except	5-98

U

Unicode conversion	4-7
UnpacMe	4-17
Upatre	5-65
UPX	4-8
Unpacking (auto)	4-16-17
urlscan.io	1-13

V

FLG_HEAP_VALIDATE_PARAMETERS	5-13
Variable	
Global	2-54
Local	2-54, 5-152
Static	2-54
VBA → Visual Basic for Applications	3-38
VBScript	3-81
Viper	1-48
ViperMonkey (vmonkey)	3-56
Virtual size	4-10
Virtualization	1-27-28
VirusTotal	1-13, 14
Visual Basic for Applications (VBA)	3-38
Downloader	3-38
Vivisect	1-83
VPN	1-19
Vulnerability vs Exploit	3-85

W

WH_MOUSE_LL	5-69
While (loop)	2-112
Winbindx	1-13
WinDbg	1-81, 5-81
Windows Virtual PC	5-81
Wireshark	1-53, 1-67
TCP Stream	1-74
WML_MOUSEBUTTONDOWN	5-71
WML_MOUSEBUTTOUP	5-71

FOR610 – Reverse-Engineering Malware

WM_MOUSEMOVE	5–70
Word (16 Bit)	2–28
wow64	2–148
WPE Pro	5–81

X

x32dbg	
Dump	5–31, 5–34, 4–95-100, 5–162
Follow in Memory Map	5–56
Ignore Exception	5–102
Save Patch	5–11
SEH	5–100, 5–105
SetBPX	5–30
Stack edition	5–76
Strings	4–34
TLS	5–120
x64dbg	1–81, 1–90

cleardb	4–47
xAnalyzer	5–152
XOR	1–104
XOR Encoding	5–16
XOR Obfuscation	5–118
XORSearch	5–17, 3–92
xxd	1–76

Y

yara-rules	3–76
------------------	------

Z

zipdump.py	3–43
Zone Identifier	39