

INDEX

A

Activity Monitor utility, 8, 30–31,
33–34, 46

`addObserver(selector:name:object:`
method, 286–287

advanced persistent threats (APTs),
13–14, 104–105.
See also persistence

AF sockets, 105

Alchemist attack framework, 103

AMFI. *See* Apple Mobile File Integrity

analyzing malware safely, xxvii–xxviii

anchor apple generic requirement, 94

anchor apple requirement, 94, 96–97

app IDs, registering, 254–255

Apple File System (APFS), xxviii

Apple Mobile File Integrity (AMFI),
181, 254

- disabling, xxvii, 160, 181
- entitlements and, xxvii

Application Services APIs, 17–19

APTs (advanced persistent threats),
13–14, 104–105.
See also persistence

ARC (automatic reference counting),
18, 211

arguments, process, 9–13, 197–199

ARM binaries, 32

Art of Mac Malware, The, Volume 1
(Wardle), xxv

audio monitoring

- Oversight tool, 282–285
- Shazam widget, 313–315

audit tokens, 5–6

- Endpoint Security process
monitor, 192
- mute inversion via, 210
- obtaining code object references
via, 95–96
- XPC and, 266–268

authorization events, Endpoint
Security, 213–222

- blocking Background Task
Management bypasses,
219–222
- checking binary origins, 217–219
- meeting message deadlines, 215–217
- origin of, 183
- subscribing to, 213–215

automatic reference counting (ARC),
18, 211

- bridging, 80

AutoRuns tool, 233

AVFoundationAudioObjectAddProperty
ListenerBlock API, 282

AVFoundation framework, 58

- adding a property listener,
283–284
- device enumeration, 281
- extracting property values, 285
- property listener block, 282
- removing a property listener, 294

B

Background Task Management (BTM),
2, 123–136

- accessing metadata, 134–135
- BlockBlock tool and, 261–265
- blocking bypasses, 219–222
- deserialization, 130–134
- DumpBTM* project, 130–137
- event monitoring logic, 263
- finding database path, 130–131
- identifying malware, 135–136
- `initWithCoder:` methods, 132–134
- interaction with database, 124–127
- ItemRecord class, 131–133
- `itemsByUserIdentifier`
dictionary, 131
- KnockKnock tool and, 241–242

- Background Task Management
 - (*continued*)
 - serialization, 126–127
 - sfltool utility, 127–130
 - behavior-based heuristics.
 - See also* heuristic-based detection approach
 - defined, xxii
 - false positives, 75
 - binaries. *See also* Mach-O binaries
 - encrypted, 70–71
 - packed binaries, 62–70
 - universal binaries, 39–50
 - Black Mirror* (TV show), 279
 - BlockBlock tool, 253–276
 - alerts, 257
 - Background Task Management, 261–265
 - DazzleSpy and, 319
 - Endpoint Security and, 181
 - entitlements, 254–256
 - launch daemon, 257–258
 - login item, 257
 - message deadlines and, 216
 - notarization mode, 213
 - plug-ins, 258–261
 - 3CX supply chain attack, 324–325
 - XPC, 265–276
 - blocking DNS traffic, 307–310
 - closing local flow, 309
 - closing remote flow, 309
 - extracting answers from A record, 308–309
 - name error, 309
 - NXDOMAIN response, 309–310
 - response packets, 308
 - saving DNS questions and answers to cache, 305
 - bridging, 80
 - browser extensions, 242–245
 - BTM. *See* Background Task Management
- C**
- CalendarFree.app*, 10–13
 - callback logic, 114–115
 - camera monitoring, 285–286.
 - See also* Oversight tool
 - webcam, 142, 279–280
 - case studies, 313–326
 - DazzleSpy malware, 315–319
 - Shazam, 313–315
 - 3CX supply chain attack, 319–326
 - certificate authority chain, 80
 - CFBundleCopyExecutableArchitectures
 - ForURL API, 250
 - chained supply chain attack, 320
 - checkSignature* project, 76, 79, 84, 88, 94–95
 - Chropex (ChromeLoader), 9
 - clients
 - Endpoint Security, 185, 199–200
 - XPC, 269–271
 - CloudMensis malware, 40–41, 44–49, 52, 54, 56
 - code signing, 75–76
 - ad hoc signatures, 81–82
 - Apple requirements for, xxvi–xxvii
 - defined, 24
 - disk images and, 78–84
 - Endpoint Security process monitor, 195–197
 - error codes, 97
 - false positives and, 75, 96–97
 - importance of in malware detection, 76–78
 - notarization, 77, 82–84
 - on-disk Mach-O binaries and, 93–95
 - in packages, 84–93
 - revocations, 77
 - running processes and, 95–96
 - XPC and, 268–271
 - codesign utility, 78–79, 85, 93, 129, 256
 - CoinMiner malware, 8, 33
 - CoinTicker malware, 317
 - Coldroot malware, 28–29, 63
 - com.apple.developer.endpoint-security.client*
 - entitlement, xxvii, 180
 - com.apple.quarantine* extended attribute, 316–317
 - Contents/Library/SystemExtensions/* directory, 299, 303
 - CoreMediaIO* framework, 285–286
 - adding a property listener, 285
 - core media I/O subsystem, 142, 151–152, 289–291
 - CPU utilization, processes, 35–36

- computing CPU percentage in use, 35–36
- flavor argument, 35
- streaming log messages, 151

CreativeUpdate malware, 79, 84

Crisis malware, 142, 280, 314

CrowdStrike, 9, 320

CSCCommon.h file, 97

D

DA* APIs, 56–57

data collection, 1–2. *See also* code signing; network state and statistics; parsing binaries; processes

- persistence, 119–137
 - Background Task Management, 123–136
 - DazzleSpy malware, 121–123
 - DumpBTM* project, 130–137
 - LaunchAgents* directories, 121–123
 - WindTail malware, 120–121

DazzleSpy malware, 7, 23, 33, 127, 220

- code signing, 324
- exploit detection, 315–317
- extracting symbols, 59–60, 62
- network access, 319
- persistence and, 121–123, 317–319

default mute set, 210

delegates and delegate methods

- DNSMonitor, 301
- network monitoring, 168, 172
- system extensions, 161, 163
- XPC, 266

dependencies, binary

- analyzing, 56–59
- finding dependency paths, 54–56
- packer detection and, 63

deserialization, 130–134

detection heuristics. *See* heuristic-based detection approach

device connections and disconnections, 286–288

disabling

- Apple Mobile File Integrity, xxvii, 160, 181
- System Integrity Protection, xxvii, 160, 181

DiskArbitration framework, 54, 56

disk images

- ad hoc signatures, 80
- certificate authority chain, 81
- code object references, 80–81
- code signing and, 78–84
- extracting code signing information, 79–82
- manually verifying signatures, 78–79
- notarization status, 82–84
- static code reference, 80
- verbose output, 79

dispatch_semaphore_wait API, 216

DNS cache dumping, 304–307

DNSMonitor, 297–311

- blocking DNS traffic, 307–310
- classifying endpoints, 310
- DNS cache dumping, 304–307
- domain name registrar, 310
- historical DNS records, 310
- interprocess communication, 303–304
- network extensions, 298–303
- printing DNS packet to universal log, 303–304
- provisioning profiles, 298–299

DNS monitoring, 157–169

- activating system extensions, 160–161
- identifying responsible processes, 168–169
- NetworkExtension* framework, 159–160
- parsing DNS requests, 164–165
- parsing DNS responses, 165–168
- writing system extensions, 162–169

DNSProxyProvider class, 303

Dock, 19, 301

Documents directory

- monitoring file-open events in, 211–212
- WindTail malware and, 227

domain name registrar, 310

Dummy malware, 102–103, 111, 117, 159, 169

DumpBTM project, 130–137

- accessing metadata, 134–135
- deserializing files, 131–134

- DumpBTM* project (*continued*)
 - finding database path, 130–131
 - identifying malicious items, 135–136
 - KnockKnock tool and, 241
 - using DumpBTM in your own code, 136–137
- dylld* cache, 86–87, 145
- dylld-shared-cache-extractor* tool, 87
- dylib hijacking, 217, 249
- dylib insertions, 246–248
- dylib proxying, 249–252

E

- Eleanor malware, 280
- Electron* framework, 249
- ElectroRAT, 8
- encrypted binaries, 70–71
- endianness, 41, 43, 50–51
- endpoints, DNSMonitor, 310
- Endpoint Security, 179–203
 - authorization events, 213–222
 - clients, 185
 - detecting removal of quarantine attribute, 316
 - entitlements, 254–256
 - events, 182–184
 - authorization events, 183, 213–222
 - event handling, 185–190
 - mute inversion, 209–212
 - muting, 206–212
 - printing out file-open Endpoint Security event, 212
 - proof-of-concept file protector, 223–228
 - file monitoring, 200–203
 - handler blocks, 185
 - header files, 182–183
 - mute inversion, 209–212
 - muting events, 206–212
 - prerequisites, 191
 - process monitor, 190–200
 - proof-of-concept file protector, 223–228
 - workflow, 180–190
- EndpointSecurity.h* header file, 182

- entitlements
 - applying for, 254
 - BlockBlock tool and, 254–256
 - com.apple.developer.endpoint-security.client*, xxvii, 180
 - enabling in Xcode, 255–256
 - provisioning profiles, 255
 - registering App ID, 254–255
- entropy
 - encrypted binaries, 70
 - packed binaries, 67–70
- enumerateProcesses* project, 4.
 - See also* processes
- environment information, processes, 19–24
 - converting process information into string object, 22–23
 - creating shared memory object, 20
 - declaring required variables, 20
 - extracting global data, 21
 - extracting size of response data, 22
 - resolving function pointer, 21
 - tracing process ID back to launch item property list, 23–24
- e_ppid* member, process hierarchies, 14–15
- error codes, code signing, 97
- ESClient.h* header file, 182
- ES_EVENT_TYPE_AUTH_DELETEEXTATTR event, 219
- ES_EVENT_TYPE_AUTH_* events, 201–202
- ES_EVENT_TYPE_NOTIFY_BTМ_LAUNCH_ITEM_ADD event, 261–262
- ES_EVENT_TYPE_NOTIFY_BTМ_LAUNCH_ITEM_REMOVE event, 261
- ES_EVENT_TYPE_NOTIFY_CLOSE event, 202
- ES_EVENT_TYPE_NOTIFY_CREATE event, 201
- ES_EVENT_TYPE_NOTIFY_EXEC event, 184–186, 193–194, 197–198, 221
- ES_EVENT_TYPE_NOTIFY_RENAME event, 202
- ES_EVENT_TYPE_NOTIFY_UNLINK event, 202
- es_invert_muting* API, 210
- eslogger* utility, 183–184, 186
- ESMessage.h* header file, 182, 185, 201
- es_message_t* structure, 185–186, 201, 216
- es_muted_paths_events* API, 210

- es_mute_process API, 208
- es_mute_process_events API, 208
- ESPlayground project, 180–182, 190, 205, 207, 211–213, 215, 223, 227
- ESTypes.h header file, 182, 206, 210, 215
- EvilQuest malware, 79, 84–85, 93
- executable packers, xxii, 62–67
- execution architecture, processes, 32–34
- execution state, processes, 32
- exfiltration of data, 157, 326
- exit status, Endpoint Security process monitor, 199
- exploit detection, 315–317

F

- false positives, code signing, 75, 96–97
- fat binaries. *See* universal binaries
- file monitoring
 - Endpoint Security, 200–203
 - 3CX supply chain attack, 320–322
- file protector, Endpoint Security, 223–228
 - allowing all file accesses, 224–225
 - denying all file accesses, 225
 - extracting process paths and filepaths, 225–226
 - granting file access for platform and notarized processes, 226–227
- file utility, 40
- filter data providers, 159, 170–176
 - enabling, 170–171
 - querying the flow, 173–174
 - running monitor, 174–176
 - writing extension for, 171–172
- Finder, 19, 212, 301
- Flashback malware, 246, 248
- FruitFly malware, xxii, 142, 279–280, 314
- fully qualified domain name (FQDN), 164

G

- Genieo malware, 11
- getaddrinfo API, 110
- GetProcessForPID API, 17–19

H

- HackingTeam installer, 70–71
- handler blocks, Endpoint Security, 185
- header files, Endpoint Security, 182–183
- heuristic-based detection approach.
 - See also* code signing;
 - Objective-See tools
 - code signing and, 76
 - CPU utilization, 35–36
 - detecting obfuscation, 62
 - downsides of, xxii
 - false positives, xxii, 75, 96–97
 - file monitoring, 200
 - hidden directories and, 6–7
 - network monitoring, 174
 - protecting files in user's home directory, 225
- hierarchies, process, 13–19
 - Endpoint Security process monitor, 193
 - parent, 14–17
 - retrieving information with Application Services APIs, 17–19
- historical DNS records, 310
- Hopper, 87, 145
- host-based data collection, 102
- How to Reverse Malware on macOS Without Getting Infected* (Stokes), xxviii

I

- Info.plist* file
 - browser extensions, 245–246
 - checking binary origins, 218
 - DNSMonitor, 299–300, 303
 - dynamic library insertion, 248
 - writing system extensions, 171
- integrated development environment (IDE), xxvi
- Intel binaries, 32
- Internet Protocol (IP) sockets, 107, 109–110
- interprocess communication (IPC)
 - AF sockets, 105
 - DNSMonitor, 303–304
 - XPC, 265

Invisible Internet Project (I2P), 8
IPStorm malware, 63, 66, 69, 142
iWebUpdate binary, 11, 158, 167–168

J

JSON

- building JSON-ified string, 240
- converting object properties to, 238–240
- output from KnockKnock, 247

K

KeRanger malware, 7
KERN_PROCARGS2 value, 11–12
KeySteal malware, 86, 93
kill system API, 32
kinfo_proc structure, process hierarchies, 14–15
KnockKnock tool, 233–252

- Background Task Management, 241–242
- browser extensions, 242–245
- building list of loaded libraries with, 249
- command line options, 235
- DazzleSpy and, 319
- determining whether item is a binary, 250
- dylib hijacking, 249
- dylib insertions, 246–248
- dylib proxying, 249–252
- enumerating dependencies of running processes, 251
- ItemBase class, 238
- persistent item types, 238–240
- plug-ins, 235–237, 240–252
- positive detections/antivirus engines, 240
- system_profiler approach, 247
- user interface, 234–235

kNStatSrcKeyRxBytes key, 117
kNStatSrcKeyTxBytes key, 117
kp_eproc structure, process hierarchies, 14–15
kSecCodeInfoCertificates key, 81–82
kSecCodeInfoFlags key, 82

L

LaunchAgents directories, 121–123
launchctl utility, 19–20
launch daemon, 121, 257–258.
See also persistence
Launch Services APIs, 243, 246
Lazarus APT group, 13–14
LC_SYMTAB load command, 60
leaf signature, 90
libproc APIs, 4
/Library/SystemExtensions/<UUID>/ library, 303
listeners, XPC, 265–266
load commands, Mach-O binaries, 53
loaded libraries

- building list of with KnockKnock, 249
- enumerating, 24–28

LoggingSupport framework, 145–146, 148, 152, 289
log monitoring, 141–152

- extracting log object properties, 148–151
- remote logins, 142
- resource consumption, 151–152
- streaming log data, 146–148
- TCC mechanism, 142–143
- unified logging system, 143–146
- webcam access, 142

lsof tool, 30–31
LSSharedFileListCreate API, 120
LSSharedFileListInsertItemURL API, 120
LuLu software, 78–79, 84, 170, 307, 319, 326

M

Macho* APIs, 47–50
Mach-O binaries

- code signing and, 93–95
- extracting dependencies, 54–59
- extracting symbols, 59–62
- load commands, 53
- Mach-O headers, 50–52
- slices, 40, 43, 47–50
- universal binaries, 39–50

mach_timebase_info API, 216

- MacStealer malware, 209–210, 212, 225
- malicious networking activity, 102–105
- Malware Removal Tool (MRT), 76–77
- management information base (MIB)
 - array, 11
- metadata, accessing, 134–135
- microphone, 282–285. *See also* audio
 - monitoring
- Microsoft AutoRuns tool, 233
- Mokes malware, 57–58, 142, 280, 314
- MRT (Malware Removal Tool), 76–77
- mute inversion, Endpoint Security,
 - 209–212
 - audit tokens and, 210
 - default mute set and, 210
 - monitoring directory access,
 - 211–212
- muting events, Endpoint Security,
 - 206–212

N

- name error, DNS traffic, 309
- names, process, 8–9
- NEDNSProxyManager object, 161–162
- NEFilterFlow objects, 172–174
- NEFilterManager object, 170–171
- NEFilterSocketFlow objects, 174
- ENetworkRule object, 172
- netbottom command line tool, 112
- Netiquette tool, 104
- nettop utility, 112, 156
- network access, DazzleSpy, 319
- network-centric data collection, 102
- network extension, DNSMonitor,
 - 302–303
- NetworkExtension* framework, xxiii,
 - 111–117, 159–160,
 - 297–301
 - activation, 159–160
 - DNS monitoring, 157–169
 - filter data providers, 169–175
 - identifying responsible process,
 - 168–169
 - methods, 163
 - prerequisites, 159, 298
- network monitoring, 155–176
 - DNS monitoring, 157–169
 - filter data providers, 169–175

- snapshots, 156–157
 - 3CX supply chain attack, 322–323
- network sockets, 106–111
- network state and statistics, 101–118.
 - See also NetworkStatistics*
 - framework
- capturing, 105–111
- extracting network sockets,
 - 106–107
- host-based vs. network-centric
 - collection, 102
- malicious networking activity,
 - 102–105
- retrieving process file
 - descriptors, 106
- socket details, 107–111
- NetworkStatistics* framework, 111–112
 - callback logic, 114–115
 - creating network statistic
 - managers, 113–114
 - kNStatSrcKeyRxBytes key, 117
 - kNStatSrcKeyTxBytes key, 117
 - linking to, 113
 - queries, 115
- notarization
 - detecting, 77
 - disk images, 82–84
 - packages, 91–92
- notification events, 183–184, 200–203
 - device added, 286–287
 - device removed, 286–287
- NSRunningApplication object, 8–9
- NSTask API, 26
- NStatManagerCreate API, 113
- NStatManagerQueryAllSources
 - Descriptions API, 156–157
- NSUserDefaults class, 292
- NSXPCCConnection class, 267
- NSXPCListenerDelegate protocol,
 - 265–266
- NukeSped malware, 7
- NX* APIs, 42–47
- NXDOMAIN response, DNS traffic, 309–310

O

- Objective-C language, xxvi, 59
 - extracting log object properties,
 - 148–151

- Objective-C language (*continued*)
 - performSelector: method, 134
 - private classes, 89
- Objective-See tools, xxiv, 231–232
 - BlockBlock tool, 253–276
 - DNSMonitor, 297–311
 - KnockKnock tool, 233–252
 - LuLu software, 78–79, 84, 170, 307, 319, 326
 - Oversight tool, 280–295
 - TaskExplorer, 25
- OceanLotus malware, 317
- open files, 28–31
 - lsuf tool, 30–31
 - proc_pidinfo API, 29–30
- oRAT malware, 33, 63, 104–105
- os_log_create API, 303
- OSLogEventProxy object properties, 150–151
- OSSystemExtensionRequest class, 161
- OSSystemExtensionRequestDelegate protocol, 161
- otool command
 - confirming code accuracy, 45
 - detecting encrypted binaries, 70
 - enumerating network
 - connections, 112
 - finding dependency paths, 56
 - Mach-O headers and, 52
 - reverse engineering log APIs, 145
- OverSight tool, 280–295
 - Block option, 280
 - camera monitoring, 285–286
 - device connections and
 - disconnections, 286–288
 - disabling, 293–294
 - executing user actions, 292–293
 - extracting property values, 285
 - filtering cmio and coremedia
 - messages, 290
 - LogMonitor class, 289–290
 - mic monitoring, 282–284
 - parsing messages to detect
 - responsible process, 291
 - predicate evaluation, 151–152
 - property listener, 281–286
 - responsible process identification, 288–291

- sample utility, 288
- scripts and, 291–293
- stopping, 293

P

- PackageKit* framework, 86–89
- packages
 - accessing framework functions, 88–89
 - code signing and, 84–93
 - notarization status, 91–92
 - reverse engineering pkgutil utility, 86–88
 - validating, 90–91
- packed binaries, 62–70
 - calculating entropy, 67–70
 - dependencies, 63
 - section and segment names, 63–67
 - symbols, 63
- packers (executable packers), xxii, 62–67
- Palomino Labs, 105
- Parallels, xxviii
- parent hierarchy, 14–17
- ParentPSN key, 19
- parsing binaries
 - extracting dependencies, 54–59
 - extracting symbols, 59–62
 - load commands, 53
 - Mach-O binaries, 50
 - packed binaries, 62–70
 - universal binaries, 39–50
- paths, process, 6–8
 - of deleted binaries, 7–8
 - identifying hidden files and
 - directories, 6–7
- persistence, 119–137
 - Background Task Management, 123–136
 - BlockBlock, 258–264
 - DazzleSpy malware, 121–123, 317–319
 - DumpBTM* project, 130–137
 - KnockKnock, 240–251
 - LSSharedFileListCreate API, 120
 - LSSharedFileListInsertItemURL API, 120
 - ProgramArguments key, 122

- RunAtLoad key, 122–123
- WindTail malware, 120–121
- persistence enumerator.
 - See KnockKnock tool
- persistence monitor. See BlockBlock tool
- persistent item types, KnockKnock tool, 238–240
- pkgutil utility, 78
 - package notarization, 91
 - reverse engineering, 86–89
 - verifying signature, 84–86
- plug-ins
 - BlockBlock tool, 258–261
 - KnockKnock tool, 235–237
 - base scan method, 236
 - initializing by name, 237
 - methods of base class
 - plug-in, 236
 - properties of base class
 - plug-in, 236
 - updating global list of
 - persistent items, 237
- positive detections/antivirus engines, 240
- processes, 3–38
 - arguments, 9–13
 - audit tokens, 5–6
 - code signing and, 24, 95–96
 - CPU utilization, 35–36
 - enumerating, 4–5
 - environment information, 19–24
 - execution architecture, 32–34
 - execution state, 32
 - loaded libraries, 24–28
 - open files, 28–31
 - paths, 6–8
 - process hierarchies, 13–19
 - start time, 34–35
 - validating names, 8–9
- process file descriptors, retrieving, 106
- ProcessInformationCopyDictionary API, 18–19
- process monitor, Endpoint Security, 190–200
 - arguments, 197–199
 - audit tokens, 192
 - binary architecture, 194–195
 - code signing, 195–197
 - exit status, 199
 - extracting process
 - information, 192
 - extracting process objects, 191–192
 - hierarchies, 193
 - process paths, 192–193
 - script paths, 193–194
 - stopping the client, 199–200
 - subscribing to events, 191
- process monitoring, 3CX supply chain attack, 323–325
- process serial numbers, 17–19
- procinfo command line option, 19–20
- proc_listallpids API, 4–5
- proc_pid* APIs, 102, 105–107, 111
- proc_pidinfo API, 29–30
- PROC_PIDPATHINFO_MAXSIZE constant, 6
- proc_pid_rusage API, 35
- ProgramArguments key, 122
- property listeners, 281–286
 - audio monitoring, 282–285
 - camera monitoring, 285–286
- provisioning profiles
 - BlockBlock tool, 255–256
 - DNSMonitor, 298–299
 - NetworkExtension framework, 160
- psi structure, 108

Q

- qtn_file_* APIs, 218

R

- ransomware, 7, 120, 139, 200
- redacted WHOIS data, 310
- remote access tools (RATs)
 - CoinMiner, 8
 - ColdRoot, 28–29, 63
 - ElectroRAT, 8
- remote connections, enabling, 271–272
- remoteEndpoint instance variable, 173–174
- remote logins, 142
- remote methods, XPC, 275–276
- request:actionForReplacingExtension:
 - withExtension: delegate method, 161

- request:didFailWithError: delegate method, 161
- request:didFinishWithResult: delegate method, 161
- requestNeedsUserApproval: delegate method, 161
- resources, xxix
- respondsToSelector: method, 89
- response packets, DNS traffic, 308
- responsibility_get_pid_responsible_for_pid API, 16–17
- responsible process identification, 16–19, 168–169, 174, 188–189, 193, 226, 288–291
- reverse engineering
 - Activity Monitor utility, 30
 - log APIs, 145–146
 - pkgutil utility, 86–89
- revocations, 77
- rShell malware, 33
- RunAtLoad key, 122–123

S

- Safari browser extensions, 243–245
 - enumerating, 243
 - parsing output containing, 245
 - URLsForApplicationsToOpenURL: method, 243
- sample utility, 288
- SCDynamicStoreCopyConsoleUser API, 211
- scripts, 193–194, 291–293
- SecAssessmentCreate API, 83, 94
- SecAssessmentTicketLookup API, 83, 91–92
- SecCodeCopyGuestWithAttributes API, 95
- SecCodeCopyPath API, 96
- SecCodeCopySigningInformation API, 81
- SecRequirementCreateWithString API, 82
- SecStaticCodeCheckValidity API, 81–83, 93–94, 97
- SecStaticCodeCreateWithPath API, 80
- section and segment names, packed binaries, 63–67
- SecTranslocateIsTranslocatedURL API, 217–218
- self-deleting malware, 325
- serialization, 126–127
- sf1tool utility, 127–130
- Shazam, 313–315
- Shlayer malware, 9, 213, 242
- SIGUSR1 signal, DNS traffic, 305–306
- SIP. *See* System Integrity Protection
- slices, Mach-O binaries, 40, 43, 47–50
- snapshots, xxviii, 101, 112, 115, 139, 155–157
- soi_proto structure, sockets, 108
- Spotlight service, 206–207, 246
- startSystemExtensionMode method, 162–163
- start time, processes, 34–35
- swap_* APIs, 43–44
- Swift language, xxvi
- symbols, binary
 - extracting, 59–62
 - packed binaries, 63
- sysctl API, 11, 15, 34
- sysctlbyname API, 4
- sysctlnametomib API, 34
- system extensions. *See also* *NetworkStatistics* framework
 - activating, 160–161
 - entitlements, 298–300
 - identifying responsible processes, 168–169
 - prerequisites for, 160
 - writing, 162–169
- System Integrity Protection (SIP), 254
 - disabling, xxvii, 160, 181
 - entitlements and, xxvii
 - re-enabling in Recovery Mode, xxviii
- system monitoring. *See* Endpoint Security; log monitoring; network monitoring
- System Preferences application, 123–124
- system_profiler, 247

T

- TAOMM repository, xxv
- TaskExplorer tool, 25

TCC (Transparency, Consent, and Control) mechanism, 142–143, 223, 292

TCP protocol
 querying for statistics about network events, 115
 sockets, 105, 108

3CX supply chain attack, 310, 319–326
 BlockBlock tool, 324–325
 code signing, 323–324
 DNS monitoring and, 158–159
 exfiltration, 326
 file monitoring, 320–322
 network monitoring, 322–323
 process monitoring, 323–325
 self-deletion, 325

translocation, 217–218

Transport Layer Security (TLS)
 package, 105

U

UDP protocol
 DNS traffic, 163
 querying for statistics about network events, 115
 sockets, 105, 108

universal binaries, 39–50
 fat_arch structures, 41–47
 FAT_CIGAM value, 41, 43–44
 fat_header structure, 40–41, 43–47
 inspecting, 40–42
 Macho* APIs, 47–50
 NX* APIs, 42–47
 parsing, 42
 swap_* APIs, 43–44

universal logging subsystem, 143–146
 DNSMonitor, 303–304
 manually interfacing with, 144–145
 Oversight tool and, 288–289
 reverse engineering log APIs, 145–146

URLsForApplicationsToOpenURL:
 method, 243

V

verifyReturningError: method, 90

virtual machines
 analyzing malware safely, xxvii–xxviii
 disabling SIM and AMFI, 160

VirusTotal, 168, 234, 239–240, 242, 319

vmmmap tool, 24–26

VMware, xxviii

W

webcam access, 142.

See also Oversight tool

WindTail malware, 95, 120–121, 227, 242

Wireshark, 127, 135, 158

workflow, Endpoint Security, 180–190
 clients, 185
 event handling, 185–190
 events of interest, 182–184
 handler blocks, 185

X

Xcode, xxvi, 255–256

XCSSET malware, 142, 223, 317

XPC, 265–275

 authorizing clients, 269–271
 client requirements, 270–271
 delegates, 266
 extracting audit tokens, 266–268
 initiating connections, 274
 listeners, 265–266
 methods, 272–274
 protocols, 271–273
 remote connections, 271–272
 remote methods, 275–276
 verifying clients, 268–271

XProtect, 150, 183, 280

Y

Yort malware, 13–14

Z

zombie processes, 32

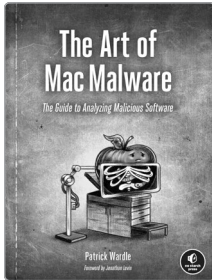
ZuRu malware, 24–28, 58–59, 63

The Art of Mac Malware, Volume 2, is set in New Baskerville, Futura, Dogma, and TheSansMono Condensed.

RESOURCES

Visit <https://nostarch.com/art-mac-malware-v2> for errata and more information.

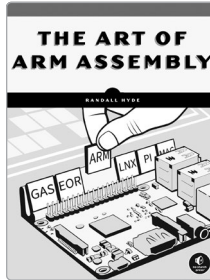
More no-nonsense books from  **NO STARCH PRESS**



THE ART OF MAC MALWARE, VOLUME 1

The Guide to Analyzing Malicious Software

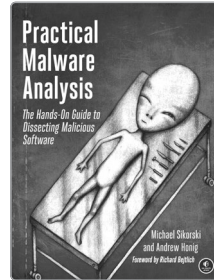
BY PATRICK WARDLE
328 PP., \$49.99
ISBN 978-1-7185-0194-2



THE ART OF ARM ASSEMBLY, VOLUME 1

64-Bit ARM Machine Organization and Programming

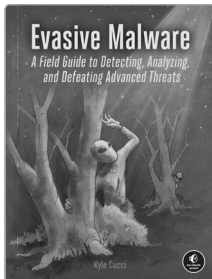
BY RANDALL HYDE
1,064 PP., \$89.99
ISBN 978-1-7185-0282-6



PRACTICAL MALWARE ANALYSIS

The Hands-On Guide to Dissecting Malicious Software

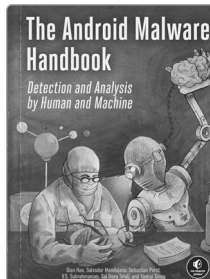
BY MICHAEL SIKORSKI AND ANDREW HONIG
800 PP., \$59.99
ISBN 978-1-59327-290-6



EVASIVE MALWARE

A Field Guide to Detecting, Analyzing, and Defeating Advanced Threats

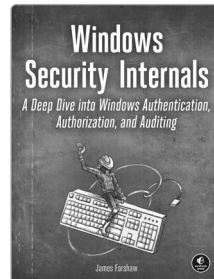
BY KYLE CUCCI
488 PP., \$69.99
ISBN 978-1-7185-0326-7



THE ANDROID MALWARE HANDBOOK

Detection and Analysis by Human and Machine

BY QIAN HAN ET AL.
328 PP., \$49.99
ISBN 978-1-7185-0330-4



WINDOWS SECURITY INTERNALS

A Deep Dive into Windows Authentication, Authorization, and Auditing

BY JAMES FORSHAW
608 PP., \$59.99
ISBN 978-1-7185-0198-0

PHONE:
800.420.7240 OR
415.863.9900

EMAIL:
SALES@NOSTARCH.COM
WEB:
WWW.NOSTARCH.COM



Never before has the world relied so heavily on the Internet to stay connected and informed. That makes the Electronic Frontier Foundation's mission—to ensure that technology supports freedom, justice, and innovation for all people—more urgent than ever.

For over 30 years, EFF has fought for tech users through activism, in the courts, and by developing software to overcome obstacles to your privacy, security, and free expression. This dedication empowers all of us through darkness. With your help we can navigate toward a brighter digital future.



LEARN MORE AND JOIN EFF AT EFF.ORG/NO-STARCH-PRESS



"If you want to study these techniques, you'd better learn from the best."





—Maria Markstedter, founder of Azeria Labs and
Forbes Person of the Year in Cybersecurity

As renowned Mac security expert Patrick Wardle notes in *The Art of Mac Malware*, Volume 2, the substantial and growing number of Mac users, both personal and enterprise, has created a compelling incentive for malware authors to ever more frequently target macOS systems. The only effective way to counter these constantly evolving and increasingly sophisticated threats is through learning and applying robust heuristic-based detection techniques.

To that end, Wardle draws upon decades of experience to guide you through the programmatic implementation of such detection techniques. By exploring how to leverage macOS's security-centric frameworks (both public and private), diving into key elements of behavioral-based detection, and highlighting relevant examples of real-life malware, Wardle teaches and underscores the efficacy of these powerful approaches.

Across 14 in-depth chapters, you'll learn how to:

-  Capture critical snapshots of system state to reveal the subtle signs of infection
-  Enumerate and analyze running processes to uncover evidence of malware

-  Parse the macOS's distribution and binary file formats to detect malicious anomalies
-  Utilize code signing as an effective tool to identify malware and reduce false positives
-  Write efficient code that harnesses the full potential of Apple's public and private APIs
-  Leverage Apple's Endpoint Security and Network Extension frameworks to build real-time monitoring tools

This comprehensive guide provides you with the knowledge to develop tools and techniques, and to neutralize threats before it's too late.

About the Author

PATRICK WARDLE is the founder of Objective-See, a nonprofit dedicated to creating free, open source macOS security tools and organizing the "Objective by the Sea" Apple security conference. Wardle is also the co-founder and CEO of DoubleYou, a cybersecurity startup focused on empowering the builders of Apple-focused security tools. Having worked at both NASA and the National Security Agency and having presented at countless security conferences, he is intimately familiar with aliens, spies, and talking nerdy.



THE FINEST IN GEEK ENTERTAINMENT™

www.nostarch.com