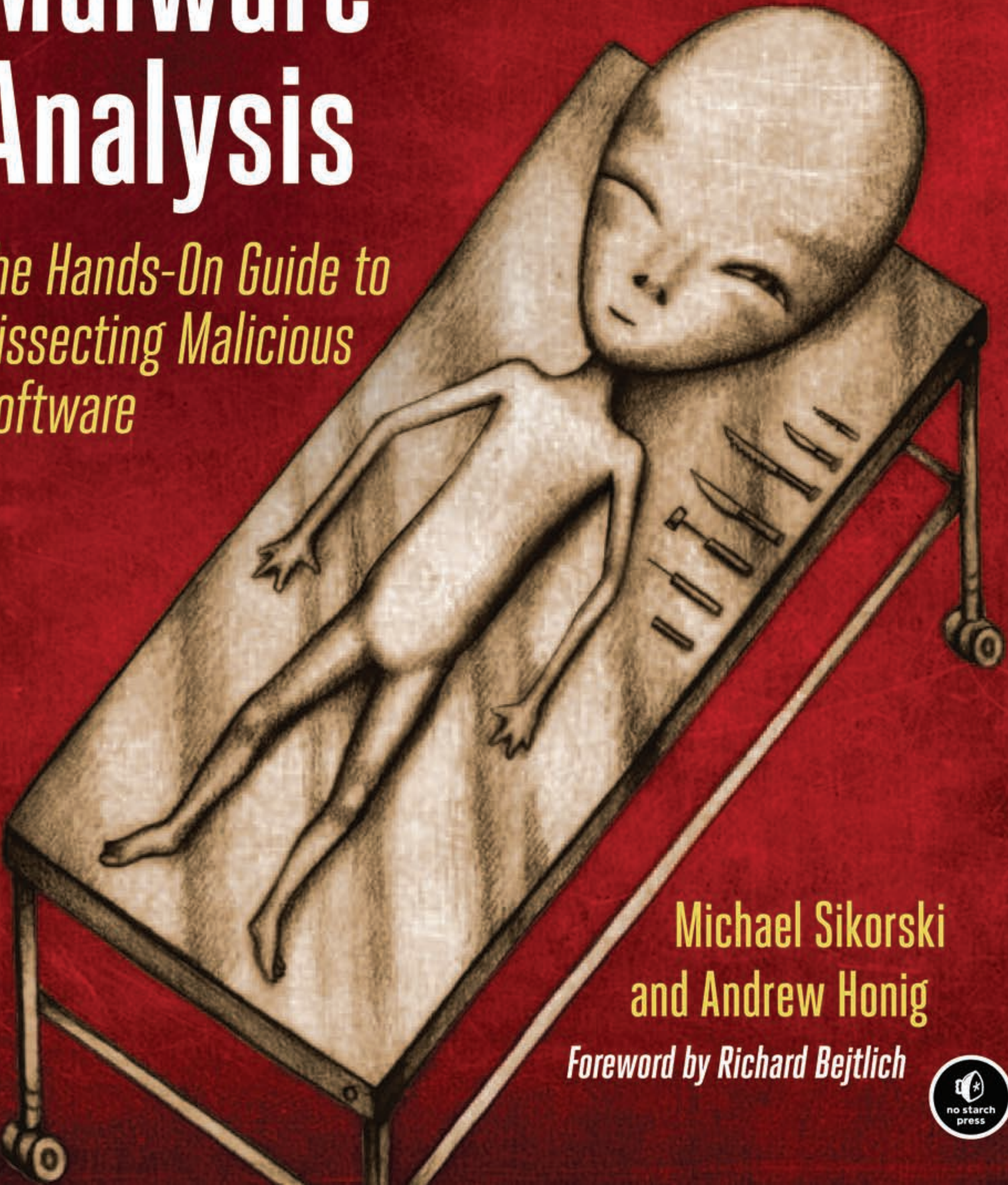
Practical Malware Analysis

The Hands-On Guide to Dissecting Malicious Software



Michael Sikorski
and Andrew Honig

Foreword by Richard Bejtlich

no starch press

INDEX

Symbols and Numbers

! (bang symbol), 305
-- operation, 112
% operation, 112
% symbol, 423
| (pipe symbol), in Snort, 304
++ operation, 112
010 Editor, 468
32-bit applications, WOW64 and, 448
32-bit rotate-right-additive hash, 418
64-bit malware, 441–449
  clues to functionality, 448
  labs, 450–451
    solutions, 723–732

A

A, at end of Windows function name, 17
absolute addresses, 443
  vs. relative addresses, in OllyDbg, 184–185
abstraction levels, in x86 disassembly, 66–67
accept function, 143, 144, 454
access token, 246
accuracy, vs. expediency, 304
active window, logging, 239
ADD encoding algorithm, 276
add instruction, 74, 349
AddCodeXref function (IDC), 342
address space, loading executable into another process's, 595
address space layout randomization (ASLR), 184
AddressOfNameOrdinals array, 416
AddressOfNames array, 416
AdjustTokenPrivileges function, 246, 247, 454, 730
administrator privileges, for malware launchers, 254
Adobe Reader
  CVE-2010-0188 critical vulnerability, 424
  overflow in, 705
ADS (Alternate Data Streams) feature, 139
Advanced Encryption Standard (AES), 618
  decrypting, 625–626
advapi32.dll, 17
  imports from, 20, 480, 481
  obtaining handle to, 237
advertisements, pop-up, 560–561
AES (Advanced Encryption Standard), 618
  decrypting, 625–626
Agobot, 376
air-gapped networks, 29
_alloca_probe function, 522
alphabetic encoding, shellcode decoder with, 697
Alternate Data Streams (ADS) feature, 139
ALU (arithmetic logic unit), 68
AMD64 architecture, 441
"Analysis of the Intel Pentium's Ability to Support a Secure Virtual Machine Monitor" (Robin and Irvine), 373
AND logical operator, in x86 architecture, 75
anti-debugging, 351–366
  checks, 656
  defeating techniques, 660
  labs, 367–368
    solutions, 655–669

anti-debugging, *continued*
NTGlobalFlag flag, 659–660
PhantOm protection from checks, 658, 659
ProcessHeap flag, 658–659
timing checks, 665–669
GetTickCount function, 668–669
with QueryPerformanceCounter, 667–668
rdtsc function, 669
anti-disassembly, 327–349
basics, 328–329
defeating disassembly algorithms, 329–334
flow-oriented disassembly, 331–334
linear disassembly, 329–331
false conditional branch, 336, 645, 647, 653
labs, 350
solutions, 645–655
malware awareness of debugger, 351
manually repaired code, 648–649
obscuring flow control, 340–346
adding missing code cross-references in IDA Pro, 342
function pointer problem, 340–341
misusing structured exception handlers, 344–346
return pointer abuse, 342–343
signs of, 645–646
techniques, 334–340
impossible disassembly, 337–340
jump instruction with constant condition, 336
jump instructions with same target, 334–335
NOP-ing out instructions with IDA Pro, 340
thwarting stack-frame analysis, 347–349
anti-virtual machine (anti-VM) techniques, 369–380, 500, 678
finding using strings, 679–683
highlighting anti-VM in IDA Pro, 377–378
impact on malware analysis, 673–677
labs, 381–382
solutions, 670–684
process replacement, 683–684
tweaking settings, 379–380
VMware artifacts, 370–373
vulnerable instructions, 373–379, 678–679
No Pill technique, 375
querying I/O communication port, 375–377
Red Pill anti-VM technique, 374–375
antivirus programs, and kernel patching, 227
antivirus scanning, 10
antivirus signatures, scan against, 478
Anubis, 40
ApateDNS, 51–52, 57, 465, 483, 485
malware DNS requests and, 489
APC (asynchronous procedure call), 263
APC injection, 262–265
AppInit_DLLs, 241–242, 572
for persistence, 575
applications, access to device objects, 206
arguments in malware, OllyDbg to debug, 532
arithmetic instruction, 74–76
arithmetic logic unit (ALU), 68
arithmetic operations
disassembly, 112–113
in WinDbg, 211
arrays, disassembling, 127–128
arrows window, in IDA Pro, 90
The Art of Assembly Language (Hyde), 68
ASCII strings, 11
loading on stack, 724–725
ASLR (address space layout randomization), 184
ASPack, 398
assembly code, for process replacement, 258
assembly language, 67. *See also* C code constructs in assembly
if statement, 113–114
for loop, 117
switch statement, 124
while loop, 118

assembly-level debuggers, vs. source-level, 168
asynchronous procedure call (APC), 263
`AT_INFO` structure, 547–548
`AttachThreadInput` function, 454
attackers
  identifying investigative activity, 299
  safely investigating online, 300–302
Autoruns tool, 140, 241, 465–466

B

backdoor, 3, 121, 232–234, 479, 519, 538
  analysis, 537–538
  `CreateProcess` and `Sleep` functions for, 479
  evading detection, 308–311
  HTTP reverse, 539
  implementing, 524
  indications of, 493
  reading configuration file, 723
  sandbox and, 41
backup images, of operating systems, 30
"Bad or Unknown 32-bit Executable File" error, 363
bang symbol (`!`), 305
base addresses
  finding with PEview, 545
  of *kernel32.dll*, finding with assembly code, 415
  for PE files in Windows, 184
Base64 cipher, 277–280, 622
  custom substitution cipher, 280
  identifying and decoding, 278–280
Base64 encoding
  decoding, 624–625
  identifying in URL, 611
  padding, 610, 630
  Python program to decode string, 289
  static pattern within, 631
`base64_encode` function, 610
basename, 535
BCDEdit, 227
beaconing, 309, 611
  client-initiated, 311
  determining generation, 628–629
  packet structure, 643
  request from initial malware run, 627–628
  sending by malware, 633–634, 639–640
  string decoding, 636
beep driver, 214
behavior of malware. *See* malware behavior
`BeingDebugged` flag, 354, 657–658
  checking, 353–354
Berkeley compatible sockets, 143–144
BFK DNS logger, 302
BHOs (Browser Helper Objects), 157
big-endian, 70
binary data
  Base64-encoding conversion, 277
  static analysis, 503–507
Binary File option, in IDA Pro, 88
binary translation by VMware, in kernel mode, 373
`bind` function, 143, 144, 454
BinDiff, 466
BinNavi, 466
BitBlaze, 40
`BitBlt` function, 454
blacklists, of IP addresses, 301
`Blink` pointers, 414
block cryptography algorithms, 626
blue screen, in Windows, 158
Bochs (debugger), 467
Bookmarks plug-in, in OllyDbg, 199–200
boot.ini file, 207–208, 226
botnet controller, 234
botnets, 3, 234, 376
`bp` command, in WinDbg, 211
branching, in x86 architecture, 80–81
breakpoints
  in debuggers, 171–175
    conditional, 175
    hardware execution, 174–175
    software execution, 173–174
  deferred, 212–213, 554
  hardware vs. software, 687
  for kernel activity, 548
  in OllyDbg, 188–191, 391
    command-line to set, 199
  scanning code for, 357
  and self-decoding, 289

breakpoints, *continued*
  setting, 357
  setting on stack, 690
  in WinDbg, 211–212
bridged network adapter, 34
Browser Helper Objects (BHOs), 157
brute-force XOR encoding, 271–273
`bu` command, in WinDbg, 212
`bu $iment` command, in WinDbg, 213, 554
buffer, malware placement of value in, 538
buffer-overflow attack, 421
Burp Suite, 467
Buster Sandbox Analyzer, 473
byte array initialization, 680–681
bytecode, 67

C

C code constructs in assembly, 109–132
  arithmetic operations disassembly, 112–113
  array disassembly, 127–128
  function call conventions, 119–121
  global vs. local variables, 110–112
  `if` statements, 113–116
  labs, 133–134
    solutions, 501–513
  linked list traversal, 130–132
  loops, 116–118
    `for` loops, 116–118
    `while` loops, 118
  structures, identifying, 128–130
  `switch` statements, 121–126
    if style for, 122–123, 124
    jump table, 123–126
C programming language, 110
  function pointers in, 340
  `main` method and offsets, in x86 architecture, 83–84
  pseudocode for process replacement, 258
  standard library, IDA Pro catalog of named constants, 102
The C Programming Language (Kernighan and Ritchie), 110
C++ analysis, 427–438
  labs, 439–440
    solutions, 712–723
  object-oriented programming, 427–432
    inheritance and function overriding, 432
    overloading and mangling, 430–431
    `this` pointer, 428–430
  objects creation and destruction, 437
  virtual vs. nonvirtual functions, 432–436
Caesar cipher, 270
`call` instruction, 119, 333, 386, 409
  and finding OEP, 391–392
  position dependence, 408
  for quick analysis, 521–522
  with target based on `DWORD` pointer, 396
`call` *`memory_location`*, 77
call stack trace, in OllyDbg, 193
callback type, 136
calling conventions, x64 architecture differences, 443–447
`CallNextHookEx` function, 260, 261, 454
Canvas penetration-testing tool, 380
Capture BAT, 467
capturing events
  network traffic, 580
  stopping procmon from, 44
capturing screen, function for, 615
CBC (Cipher Block Chaining), 626
`cdecl` calling convention, 119–120
cell phone malware, 88
central processing unit (CPU)
  threads and, 149
  in x86 architecture, 68
`CertOpenSystemStore` function, 454
CF (carry) flag, 72
CFB (Cipher Feedback), 626
CFF Explorer, 467
`cfile.read` command, 293
chained encoding algorithm, 277
`CheckRemoteDebuggerPresent` function, 352, 454
child classes in C++, 432
  functions from parent class, 436
chunk size, dependency with entropy score, 284
Cipher Block Chaining (CBC), 626
Cipher Feedback (CFB), 626

ciphers, 270–280
  Base 64, 277–280
  Caesar cipher, 270
  other encoding schemes, 276–277
  XOR cipher, 271–276
cisvc.exe
  PEview of original and trojanized versions, 584–585
  writing shellcode into, 583–584
class identifiers (CLSIDs), 155
  and COM functionality, 518
classes, in object-oriented code, 428
`classtype` keyword, in Snort, 304
client side of network, 144–145
client-initiated beaconing, 311
client/server framework, Component Object Model as, 154
`CloseHandle` function, 526
`CloseServiceHandle` function, 554
cloud services, 300
Cloudburst, 380
CLSIDs (class identifiers), 155
  and COM functionality, 518
cmd.exe, 544
`cmp` instruction, 80, 348, 502
`CoCreateInstance` function, 155–156, 313, 454, 518
code
  in memory, 69
  performing checksums, 357
  redefining in IDA Pro, 103
  understanding surrounding, 312–313
code construct, 109
code cross-references, 95–96
code entry point, unpacking stub and, 384
code libraries, linking, 15
COFF (Common Object File Format), IDA Pro support for, 87
`CoInitialize` function, 313
`CoInitializeEx` function, 154
colors in IDA Pro navigation band, 93
COM (Component Object Model), 154–157, 313, 626
  related functions, 518
  server malware, 157
command-line
  analysis of binary, 94
  arguments in malware, 526
  check for arguments, 525–529
  encoded, 636
  option analysis, 535–537
  running malware from, 493
Command Line plug-in, for OllyDbg, 198–199, 657–658
  launching, 660
command processing, and malware signature, 644
command shell, thread input to, 636
comments
  in HTML, 506
    command character parsed from, 509
    to send commands to malware, 507
  in IDA Pro, 100
Common Object File Format (COFF), IDA Pro support for, 87
Comodo Instant Malware Analysis, 40
comparing strings, in Process Explorer, 49
compilation, 67
Component Object Model (COM), 154–157, 313, 626
  related functions, 518
  server malware, 157
compression algorithm, packers and, 384
`compsb` instruction, 82
ComSpec environmental variable, 636
conditional branches, 348
  false, 645, 647
  flow-oriented disassembly and, 333
conditional breakpoints, 175
  in OllyDbg, 188, 189–190
conditional jump, 80–81, 113, 116, 354
conditionals, in x86 architecture, 80
configuration information, Windows Registry for, 139
`connect` function, 143, 144, 313, 454, 727
connect mode, in Netcat, 52
`ConnectNamedPipe` function, 455
console programs, `IMAGE_SUBSYSTEM_WINDOWS_CUI` value for, 23
constructor, 437
`content` keyword, in Snort, 304

content-based countermeasures, 298, 302–307
control unit, 68
ControlService function, 455, 549
convention, 72
CopyFile function, 526
countermeasures
  content-based, 302–307
  network-based, 297
covert launching techniques, 253–265
  APC injection, 262–265
  Detours, 262
  hook injection, 259–261
  labs, 266–267
    solutions, 586–607
  launchers, 253–254
  process injection, 254–257
  process replacement, 257–259
CPU (central processing unit)
  threads and, 149
  in x86 architecture, 68
cpuid instruction, virtual machine and, 374
crashing virtual machine, from procmon, 44
CreateFile function, 137, 215, 219, 455, 520, 527, 583, 640
  debugger and, 171
CreateFileMapping function, 137–138, 455, 520, 527, 583
CreateMutex function, 152, 455, 522
CreatePipe function, 233
CreateProcess function, 147–149, 233, 455, 479, 524, 544, 590, 642, 727
  parameters, 728
CreateRemoteThread function, 256, 262, 423, 455, 586, 600, 730
  arguments for, 603–604
  and direct injection, 257
  for DLL injection, 255
CreateService function, 153, 243, 455, 514–516, 549, 550, 554
CreateThread function, 150–151
CreateToolhelp32Snapshot function, 255, 263, 455, 498
CreateWindowEx function, 137
credential stealers, 234–241
  GINA interception, 235–236, 570–571
  hash dumping, 236–238
  keystroke logging, 238–241
cross-references (xref), 124
  checking for gethostbyname, 495
  for global variables, 547
  graphs of, 98, 99
    for function, 498
    for installer export, 572–573
  in IDA Pro, 95–97
    adding missing code, 342
    navigating, 92–93
  and virtual functions, 436
CryptAcquireContext function, 455
cryptographic algorithms, 280–285
  recognizing strings and imports, 281–282
  search for cryptographic constants, 282–283
  search for high-entropy content, 283–285
cryptography, drawbacks, 281
CWSandbox, 40

D

da command, in WinDbg, 210
data
  hard-coded vs. ephemeral, 314–315
  overlaying onto structure, 214
  Python script for converting to string, 500–501
  redefining in IDA Pro, 103
data buffers, instructions for manipulating, 81
data cross-references, 96–97
data encoding, 269–294
  cryptographic algorithms, 280–285
    recognizing strings and imports, 281–282
    search for cryptographic constants, 282–283
    search for high-entropy content, 283–285
  custom, 285–288
  decoding, 288–294
    instrumentation for generic decryption, 291–294
    manual programming of functions, 289–290
    self-decoding, 288–289

goal of analyzing algorithms, 270
identifying and leveraging steps, 315–317
labs, 295–296
solutions, 607–626
simple ciphers, 270–280
Base64, 277–280
Caesar cipher, 270
other encoding schemes, 276–277
XOR cipher, 271–276
Data Execution Prevention (DEP), 578
data section in main memory, 69
`.data` section in PE file, 22
size of, 24
`DataDirectory` array, 364
`db` command, in WinDbg, 558
`dd` command, in WinDbg, 210, 218, 564
DDoS (distributed denial-of-service) attack, 234
malware to launch, 517
debuggers, 167–178. *See also* anti-debugging; Ollydbg; WinDbg
exceptions, 175–177
first- and second-chance, 176
identifying behavior, 356–359
`INT` scanning, 357
performing code checksums, 357
timing checks, 357–359
interference with functionality, 359–363
exceptions, 361–362
inserting interrupts, 362–363
TLS callbacks, 359–361
just-in-time, 411
kernel vs. user mode, 168–169
Microsoft symbols, 212–215
modifying program execution with, 177
source-level vs. assembly-level, 168
using, 169–175
breakpoints, 171–175
single-stepping, 169–170
stepping-over vs. stepping-into, 170–171
vulnerabilities, 363–365
Windows debugger detection, 352–356
manually checking structures, 353–356
with Windows API, 352–353
decoding, 288–294
anti-debugging routine in, 663
instrumentation for generic decryption, 291–294
manual programming of functions, 289–290
self-decoding, 288–289
stack-formed strings, 540–541
XOR-encoded strings, 542–543
decryption
of AES, 625–626
instrumentation for generic, 291–294
requirements for, 622
Deep Freeze, 467
default view for IDA Pro, returning to, 92
default web browser, malware determination of, 699–703
deferred breakpoint, 212–213, 554
`delete` operator, 437
`DeleteFile` function, PyCommand to prevent execution, 201
Delphi programs, compile time, 23
DEP (Data Execution Prevention), 578
Dependency Walker (*depends.exe*), 16–17, 49, 468, 480
destructor, 437
Detail filter, in procmon, 45
Detours, 262
device drivers, 206
analysis, 562
finding in kernel, 217
finding in memory, with WinDbg, 563
IDA Pro to open, 551
loading, 226
tool for loading, 470–471
WinDbg for viewing, 551–553
device object
obtaining handle to, 216
viewing in kernel, 218
\Device\PhysicalDisk1, 138
\Device\PhysicalMemory, 139
`DeviceIoControl` function, 206, 216, 219, 455, 561–562, 565–566

!devobj command, in WinDbg, 220
digital logic, 66
digital signatures, 48
direct injection, 254–257
disassembler, 3, 67. *See also* anti-disassembly; IDA Pro (Interactive Disassembler Professional)
Disassembler window, in OllyDbg, 182
disassembly, 65. *See also* x86 disassembly
  enhancing in IDA Pro, 100–103
  of Hello World program, 412
distance Snort rule keyword, 305
distributed denial-of-service (DDoS) attack, 234
  malware to launch, 517
div instruction, 75
divide-by-zero exception, 653, 668
DLL display window, in Process Explorer, 48
DLL injection, 286-289, 621, 676, 730
    DLL load-order hijacking, 244–245
DllCanUnloadNow function, 157, 456
DllEntryPoint function, 495
DllGetClassObject function, 157, 456
DllInstall function, 157, 456
DllMain function, 43, 146, 254, 401, 495
  determining number of functions called by, 499
DLL_PROCESS_ATTACH, 573
DllRegisterServer function, 157, 456
DLLs. *See* dynamic link libraries (DLLs)
DllUnregisterServer function, 157, 456
DNS (Domain Name System)
  attackers tunneling information, 309
  attackers' use of, 309
  server, malware access to, 34
  tools for controlling responses, 465
DNS requests
  ApateDNS response to, 51–52
  checking for, 57
documentation manuals, for x86 architecture, 85
domain
  blacklists, 301
  getting information, 300–302
  and malicious activity, 299
Domain Name System. *See* DNS (Domain Name System)
DomainTools, 301
double-packed malware, 397
downloaders, 3, 231–232
  malware as, 481–482
downloading malware, opening URL for, 651–652, 654
driver objects
  finding, 220–221
  getting list, 552
  structure in Windows, 206
driver signature, 64-bit versions of Windows and, 227
DriverEntry function, 206, 551
DriverInit function, 214, 563
DriverUnload command, 563
!drvobj command, in WinDbg, 217
dt command, in WinDbg, 217, 552, 563, 565
du command, in WinDbg, 210
dummy names, 100
  changing, 111
Dummy service, in INetSim, 56
dump command, in OllyDbg, 658
dumping executable from memory, 469
  OllyDump for, 390
dwo command, in WinDbg, 211
DWORD
  call instruction with target based on, 396
  in Windows API, 136
dynamic analysis, 39–60, 65. *See also* debuggers
  advanced, 3
  basic, 2–3
  basic tools in practice, 56–59
  benefits of, 39
  Capture BAT for, 467
  combining with static analysis, 307–321
  comparing Registry snapshots with Regshot, 50–51
  faking network, 51–53
  INetSim, 55–56
  labs, 61–62
    solutions, 482–493
  packet sniffing with Wireshark, 53–55
  Process Explorer for viewing processes, 47–50

Process Monitor (procmon), 43–46
running malware, 42–43
sandboxes, 40–42
drawbacks, 41–42
dynamic link libraries (DLLs), 12, 17, 145–147
analyzing in IDA Pro, 521–525
base address different from preferred, 88
basic structure, 146–147
Detours to add new to existing binaries, 262
injection, 254–257
debugger view, 256
launching, 42
loading in OllyDbg, 191–192
load-order hijacking, for persistence, 244–245
malware as, 42
memory addresses for, 184–185
memory map to locate, 546–547
packed, 401
Process Explorer for finding injection, 589
dynamic linking, 16
dynamic unpacking programs, automated, 389
dynamically linked functions, exploring with Dependency Walker, 16–17

E

EA (effective address), in IDAPython scripts, 105
Eagle, Chris, *The IDA Pro Book*, 88
EAT (export address table), hooking method and, 248
EAX register, 75, 646
EBP register, 77
ECB (Electronic Code Book), 626
Eckel, Bruce, *Thinking in C++*, 428
ECX register, `this` parameter and, 429
`.edata` section, in PE file, 22
EDI register, 81
EDX register, 75
effective address (EA), in IDAPython scripts, 105
EFLAGS register, 72
EIP (instruction pointer), 73
Electronic Code Book (ECB), 626
ELF (Executable and Linking Format), IDA Pro support for, 87
EM64T architecture, 441
email-stealing malware, 572
Emerging Threats list of signatures, 304
`EnableExecuteProtectionSupport` function, 456
encoding. *See* data encoding
encoding functions, 614, 725
encrypted files
first bytes of, 286
writing, 614
encrypted write, function graph of, 287
encryption
decoding algorithm with OllyDbg, 616
indications of, 618
relationship of functions, 621–622
endianness, in x86 architecture, 70
`enter` instruction, 77
entropy calculation, for packed executables, 387–388
entropy score, dependency with chunk size, 284
`EnumProcesses` function, 456, 730
`EnumProcessModules` function, 456
epilogue
64-bit code, 446–447
in functions, 77
`EPROCESS` structure
changing, 566
examining in WinDbg, 565
error message strings
finding in binary, 503–504
indicators of malware's likely functions, 718
ESI register, 81
ESP register, 77, 348
event capture, toggling on and off in procmon, 749
event flow, in Windows with and without hook injection, 259
`Ex` suffix, for Windows functions, 17
exception handlers
in 64-bit systems, 445, 447
building, 653
misusing structured, 344–346
in OllyDbg, 194–195
properly disassembled code, 654

ExceptionHandler function, 345
EXCEPTION_REGISTRATION data structure, 344
exceptions, 344, 361–362
  in debuggers, 175–177
    first- and second-chance, 176
  in Windows, 157–158
exclusive OR cipher. *See* XOR cipher
.exe files, program infecting, 519
Executable and Linking Format (ELF), IDA Pro support for, 87
executables. *See also* packed executables
  dumping from memory, 469
  function import by ordinal, 16–17, 43
  loading, 384–385
    into address space of another process, 595
    in IDA Pro, 88–89
  opening in OllyDbg, 180–181
  PEiD plug-ins running of, 14
  searching for strings in, 11
  shellcode as, 407
  termination, 656–657
exit, analysis of immediate, 724
expediency, vs. accuracy, 304
exploits, 245
explorer.exe
  code search for, 732
  writing path into process, 588
export address table (EAT), hooking method and, 248
export data, in IMAGE_EXPORT_DIRECTORY array, 416
exported functions, 18
  absence of, 521
Exports window, in IDA Pro, 91
$EXTERNAL_NET variable, in Snort, 303

F

fake services, 55
FakeDNS, 469
faking networks, 51–53
  Netcat (nc) for monitoring, 52–53
false positives, in Snort, 306
Fast Library Identification and Recognition Technology (FLIRT), 88
  signature detection, 541
fastcall calling convention, 120
fibers, in Microsoft systems, 151
"File contains too much data" error, in OllyDbg, 364
file mappings, 137–138
file signatures, 10
File system filters, in procmon, 46
file system functions, in Windows API, 137–138
FILE_BOTH_DIR_INFORMATION structure, 558–559
FileInformation structure, 558–559
FileInsight, 468
FileMon tool, 43
files
  brute-forcing many, 273
  checking names, 541
  hidden, 558–559
    recovering, 559–560
  malware creation of, 612
  malware modification of, 527–529
  malware opening of, 714–716
  malware uploading of, 716
  transferring from virtual machine, 36
  writing from kernel space, 215
Filter dialog in Process Monitor, 484
filters
  in procmon, 44–46
  in Wireshark, 53
Find OEP plug-in (Section Hop), 391
FindCrypt2, 283
  output, 619
FindFirstFile function, 20, 456, 478–479, 520, 527, 715
finding
  networking code, 313
  original entry point (OEP), 391–395
    with automated tools, 391–392
    manually, 392–395
  strings, 11–13
findKernel32Base function, 419, 697, 707
FindNextFile function, 20, 478–479, 520, 715
FindResource function, 254, 456, 596, 600, 609
findSymbolByHash function, 418, 419, 697, 707

FindWindow function, 456, 662–663
  to search for debugger, 356
firewall
  and kernel patching, 227
  for virtual machine, 33
firmware, 66
flags, 72–73
fldz instruction, 412
FlexHEX, 468
Flink pointers, 414
FLIRT (Fast Library Identification and Recognition Technology), 88
  signature detection, 541
floating-point instruction, 130
flow chart, of current function, 98
flow control, obscuring, 340–346
  adding missing code cross-references in IDA Pro, 342
  function pointer problem, 340–341
  misusing structured exception handlers, 344–346
  return pointer abuse, 342–343
flow-oriented disassembly, 329, 331–334
flow Snort rule keyword, 305
fnstenv instruction, structure for, 411–412
for loops, 116–118
ForceFlags field, in heap header, 355
format string, identifying, 505
formatting operands, in IDA Pro, 100
FPU (x87 floating-point unit), 411–413
FpuSaveState structure, 411
frame functions, 446
FS segment register, and SEH chain, 344, 354
fsgina.dll, 235
fstenv instruction, structure for, 411–412
FtpPutFile function, 456, 714
FtpSetCurrentDirectory function, 714
function pointers, 435
  problem, 340–341
functions
  analysis to determine stack frame construction, 347
  analyzing in IDA Pro, 97–98
    graphically, 114
  call conventions, 119–121
  decision to skip analysis, 526
  disassembly and memory dump, 174
  executable import by ordinal, 16–17, 43
  executable use of, 15–18
  exported, 18
  finding connection between, 622
  finding that installs hook, 223
  graphing cross-references, 498
  graphs of calls, 98
  hard-coded locations for calls, 410
  identifying at stored memory location, 695
  imported, 18, 19
  naming conventions, 17
  overloading in object-oriented programming, 430–431
  program termination by, 656–657
  recursive, 527
  search for information on, 19
  stepping-over vs. stepping-into, 394–395
  virtual vs. nonvirtual, 432–436
    vtables, 434–435
Functions window, in IDA Pro, 91

G

g (go) command, in WinDbg, 211
GCC (GNU Compiler Convention), calling conventions, 121
GDI32.dll, 17
  importing from, 20
GDT (global descriptor table), 374
GDT register (GDTR), 374
general registers, 71–72
  in x64 architecture, 443
GET request, 309
  and malicious activity, 299
  malware construction of, 539
GetAdaptersInfo function, 456
  dynamic resolution, 680
getaddrinfo function, 313
GetAsyncKeyState function, 239, 457, 581, 585
GetCommandLineA function, 395
  breakpoint on, 400
getContent function, 615
GetCurrentProcessId function, 547
GetCurrentThreadId function, 575

GetDC function, 457
GetFileSize function, 708
GetForegroundWindow function, 239–240, 457, 581, 585, 598–599
GetHash function, 236
gethostbyname function, 313, 314, 457, 495–496, 727
gethostname function, 457, 611, 650
GetKeyState function, 240, 457
GetModuleBaseNameA function, 587
GetModuleFileName function, 457, 515, 531, 541, 575
GetModuleHandle function, 395, 457, 609
  breakpoint on, 400
GetProcAddress function, 13, 15, 224, 237, 256, 387, 413, 457, 520
  setting breakpoints on, 395
  unpacking stub import of, 385
GetStartupInfo function, 457
GetSystemDefaultLangId function, 457, 498
GetSystemDefaultLCID function, 178
GetTempPath function, 457, 604
GetThreadContext function, 458, 590, 591
GetTickCount function, 313, 314, 315, 358–359, 458, 668–669
GetVersion function, 395
GetVersionEx function, 458
GetWindowsDirectory function, 458
GFI Sandbox, 40–41
GINA (Graphical Identification and Authentication) interception, 235–236
  indications of, 567–571
global descriptor table (GDT), 374
global values in memory, 69
global variables, 587
  cross-references for, 547
  vs. local, 110–112
GlobalAlloc function, 609
globally unique identifiers (GUIDs), 155
GNU Compiler Collection (GCC), calling conventions, 121
gnuunx (GNU C++ UNIX) libraries, 102
GrabHash function, 237
graph
  of encrypted write, 287
  from IDA Pro Entropy Plugin, 284–285
graph mode, in IDA Pro, 89–90, 98–99
Graphical Identification and Authentication (GINA) interception, 235–236
  indications of, 567–571
Gray Hat Python (Seitz), 201
GUI manipulation functions, 20
GUI programs, IMAGE_SUBSYSTEM_WINDOWS_GUI value for, 23
GUIDs (globally unique identifiers), 155

H

hal.dll, malicious drivers and, 207
handles
  for device objects, 220
    obtaining, 216
  for injecting malicious DLL, 255
  locating for PDF document, 708
  obtaining to *samsrv.dll* and *advapi32.dll*, 237
  for service, OpenService function for, 550
  in Windows API, 137
  to Winlogon, opening, 603
handles type (H) type, in Windows API, 136
Handles window, in Process Explorer, 48
hard-coded headers, 637
hard-coded locations, for function calls, 410
hardware breakpoints, 357
  in OllyDbg, 188, 190
  vs. software, 687
hardware level, in x86 architecture, 66
hash dumping, 236–238
  identifying method, 238
hash function, 418
hashed exported names, for symbol resolution, 417–418
hashing, 10
headers
  hard-coded, 637
  in PE file format, 21–26
Heads function, 105

heap, 69
heap space, creating, 437
heap spray, 705
heap structures, information for creating, 355
Hello World program, 418–421
  disassembly, 412
help, in OllyDbg, 197
heuristics, 10
Hex Editor Neo, 468
hex editors, 468
hex window, in Wireshark, 53
Hex Workshop, 468
HexEdit, 468
Hex-Rays Decompiler plug-in, 106, 347, 468–469
hidden files, 558–559
  recovering, 559–560
hidden process, 566
Hide Debugger plug-in, 354
  for OllyDbg, 198
Hidedebug plug-in, 354
high-entropy content, search for, 283
high-level language, 66, 67
high-level remote hooks, 260
HKEY, 139
`HKEY_CLASSES_ROOT`, 140
  `\http\shell\open\command`, 699
`HKEY_CURRENT_CONFIG`, 140
`HKEY_CURRENT_USER` (`HKCU`), 140
`HKEY_LOCAL_MACHINE` (`HKLM`), 140
`HKEY_LOCAL_MACHINE\Software` registry key, 448
  `\Microsoft\Cryptography\RNG\Seed`, 549
    `RegSetValue`, 484
  `\Microsoft\Windows\CurrentVersion\Run`, 485
  `\Microsoft\Windows NT\CurrentVersion\SvcHost`, 488
`HKEY_USERS`, 140
`HlpGetPrimaryCredential` function, 238
`$HOME_NET` variable, in Snort, 303
honeypots, 369
hook function, `NtQueryDirectoryFile` function as, 556–558
hook injection, 259–261
  assembly code, 261
hooking
  examining in OllyDbg, 579–580
  function, 598
  inline, 248–250
  keylogger and, 239
  local and remote, 260
  looking for code, 223
  low-level operation, 579
  malware, installing code for, 577
  System Service Descriptor Table (SSDT), 221–222
    checking for, 222
host-based signatures, 2
hostname
  Base64 string for encoding, 611
  function to obtain, 722
  of local machine, loading buffer, 650
host-only networking, 32–33
hotkeys, registering, 20
HTML (HyperText Markup Language) comments, 506
  command character parsed from, 509
  to send commands to malware, 507
`htons` function, 727
HTTP (HyperText Transfer Protocol)
  attackers' use of, 309
  port 80 and, 232
  reverse backdoor, 539
HTTP server
  backdoor indicators, 493
  malware access to, 34
  simulating, 56
`HTTPOpenRequest` function, 313
`$HTTP_PORTS` variable, in Snort, 303
`HTTPQueryInfo` function, 313
HTTPS, attackers' use of, 309
HTTPS server, simulating, 56
`HTTPSendRequest` function, 313
Hungarian notation, 136
Hyde, Randall, *The Art of Assembly Language*, 68
HyperText Markup Language (HTML) comments, 506
  command character parsed from, 509
  to send commands to malware, 507
HyperText Transfer Protocol (HTTP)
  attackers' use of, 309
  port 80 and, 232
  reverse backdoor, 539

I

IAT (import address table), hooking method and, 248
ICE (In-Circuit Emulator) breakpoint, 363
IDA Pro (Interactive Disassembler Professional), 87–106, 469
  adding `IP_ADAPTER_INFO` structure, 680–681
  analyzing functions, 97–98
  analyzing functions graphically, 114
  applying structure in, 547–548
  comparison plug-in for, 466
  consecutive jump instructions in, 335
  cross-references, 95–97
  enhancing disassembly, 100–103
  FindCrypt2, 283
  graphing options, 98–99
    `for` loop, 117
    of parsing routines, 318–319
    view for tail jump, 392–394
  highlighting anti-VM in, 377–378
  identifying XOR loops in, 274–276
  interface, 89–95
    disassembly window modes, 89–90
    returning to default view, 92
    windows for analysis, 91–92
  labs, 107–108
    solutions, 494–501
  listing imported with cryptographic functions, 282
  loading executable, 88–89
  looking at user-space code in, 215–216
  manually switching bytes between data and instructions, 334
  navigating, 92–94
    colors in navigation band, 93
    exploring history, 93
    jumping to location, 93–94
    links and cross-references, 92–93
  to open driver, 551
  packed program and, 387
  plug-ins for extending, 103–106
    commercial plug-ins, 106
    IDAPython, 105–106
    IDC scripts, 104–105
  search for x86 instructions vulnerable to VM detection, 670–672
  searching, 94–95
  searching packed executable for tail jump, 392
  for TLS callback function analysis, 360
  toggling between graph and traditional view, 495
  vs. WinDbg, 553
  `WinMain` function in, 561
The IDA Pro Book (Eagle), 88
IDA Pro Entropy Plugin, 283–284, 619–620, 622
  graph from, 284–285
IDA Pro database (*idb*), 88
IDA Pro Free, 87
idaapi module in IDAPython, 105
IDAPython, 105–106
`.idata` section, in PE file, 22
idautils module in IDAPython, 105
idb (IDA Pro database), 88
idc module in IDAPython, 105
IDC scripts, 104–105
IDEA (International Data Encryption Algorithm), 283
identifying malware, hashing for, 10
IDSs (intrusion detection systems), 298
  signature-based, 302
  with Snort, 303–304
IDT (Interrupt Descriptor Table), 225, 374
IDT register (IDTR), 374
`if` statements
  for active Internet connection, 510
  recognizing, 113–116
IIDs (interface identifiers), 155
  and COM functionality, 518
image base, 184
`IMAGE_DATA_DIRECTORY` structure, 364
`IMAGE_DOS_HEADER` structure, 22, 594
`IMAGE_EXPORT_DIRECTORY` array, export data in, 416
`IMAGE_FILE_DLL`, to modify PE header, 43
`IMAGE_FILE_HEADER`, in PE file, 22
`IMAGE_NT_HEADERS` structure, 22, 594
`IMAGE_OPTIONAL_HEADER`, in PE file, 23
`IMAGE_OPTIONAL_HEADER` structure, 363–364
`IMAGE_SECTION_HEADER` structure, 23, 594

IMAGE_SUBSYSTEM_WINDOWS_CUI value, for console programs, 23
IMAGE_SUBSYSTEM_WINDOWS_GUI value, for GUI programs, 23
$iment command, in WinDbg, 213
ImmDbg (Immunity Debugger), 179, 292–294, 469, 616–617
  Python scripts for, 200
immediate operands, 69
imm.getRegs function, 293
imm.remoteVirtualAlloc command, 293
imm.setBreakpoint function, 293
Immunity Debugger (ImmDbg), 179, 292–294, 469, 616–617
  Python scripts for, 200
Immunity security company, 179
imm.writeLong function, 293
imm.writeMemory command, 293
import address table (IAT), hooking method and, 248
Import Reconstructor (ImpRec), 390–391, 469
import table
  absence of, 480
  modification, 262
  rebuilding with Import Reconstructor, 390–391
  repairing manually, 395–397
imported functions, 15, 18, 19
  examining list, 513–517
  packer resolving of, 385
Imports window, in IDA Pro, 91
ImpRec (Import Reconstructor), 390–391, 469
In-Circuit Emulator (ICE) breakpoint, 363
in instruction (x86), 376
indexing service, malware starting, 582
indirection tactics, 300
inet_addr function, 458, 522
INetSim, 55–56, 57, 469, 634
  logs for requests, 58
information-stealing malware, 4
infrastructure, attackers' use of existing, 311
inheritance, in object-oriented programming, 432
.ini files, 139
InInitializationOrderLinks list of structures, 414
initialization function, 214
injected code, 64-bit version, 442
inline hooking, 248–250
  function installing, 574–575
input function, and decoding, 286
input/output system (I/O), in x86 architecture, 68
inserting interrupts, 362–363
installer export, graph of cross-references, 572–573
installing
  inline hook, 574–575
  VMware Tools, 31
InstallService, 43
instance of class, 428
instruction pointer, 68, 71
  debugger to change, 177
instruction pointer–relative data addressing, in x64 architecture, 443–444
instruction set, 67
instructions
  bytes as part of multiple, 338
  in x86 architecture, 69–70
    anti-VM, 377
INT 0x2E instruction, 158
INT 2D anti-debugging technique, 363
INT 3 instruction
  exception and, 176
  inserting, 362
INT scanning, 357
Interactive Disassembly Professional. *See* IDA Pro (Interactive Disassembly Professional)
interface identifiers (IIDs), 155
  and COM functionality, 518
International Data Encryption Algorithm (IDEA), 283
Internet connection
  if construct for active, 510
  malware and, 29, 34
  malware check for active, 501
Internet Explorer, third-party plug-ins for, 157
Internet functions, graph for functions connected with, 634–635
Internet Relay Chat (IRC), 309
Internet services, simulating, 55

InternetCloseHandle function, 504, 634
InternetConnect function, 313
InternetGetConnectedState function, 501, 502
InternetOpen function, 145, 313, 458, 504, 505, 514, 634, 639–640, 650
InternetOpenUrl function, 145, 313, 458, 504, 505, 514, 634
InternetReadFile function, 145, 313, 458, 504, 505, 634
InternetWriteFile function, 313, 458
interpreted languages, 67
interprocess coordination, with mutexes, 151–152
Interrupt Descriptor Table (IDT), 225, 374
interrupts
  anti-debugging with, 362–363
  rootkits and, 225
intrusion detection systems (IDSs), 298
  signature-based, 302
  with Snort, 303–304
intrusion prevention systems (IPSs), 298
I/O communication port, query of, 375–377
IoConnectInterrupt function, 225
IoCreateDevice function, 562
IoCreateSymbolicLink function, 562
IoGetCurrentProcess function, 565
  import for, 560
IopInvalidDeviceRequest function, 564
IP addresses
  blacklists of, 301
  getting, 300–302
  and malicious activity, 299
IP_ADAPTER_INFO structure, adding to IDA Pro, 680–681
IPRIP service, malware installed as, 488
IRC (Internet Relay Chat), 309
IRP_MJ_DEVICE_CONTROL function, 218
  code listing, 219
  locating function for, 218
IRP_MJ_READ function, 219
Irvine, Cynthia, 373
isdataat Snort rule keyword, 305
IsDebuggerPresent function, 352, 458
IsNTAdmin function, 458
IsWow64Process function, 448, 459
  effort to dynamically resolve, 729–730
Itanium, 441
IWebBrowser2 interface, Navigate function, 155

J

JavaScript
  in PDF files, 704–705
  to profile user's system, 423
jmp instruction, 80, 329, 517
  consecutive in IDA Pro, 335
  with constant condition, 336
  with same target, 334–335
jnz instruction, 408–409
Joe Sandbox, 40
jump instructions, 386
jump table, for switch structure, 641–642
jumping to location, in IDA Pro, 93–94
just-in-time debugger, 411
  OllyDbg as, 696
jz instruction, 653
  false conditional for, 647–648
  target of, 646

K

KANAL (Krypto ANALyzer), 283, 610, 619
KeInitializeApc function, 264–265
KeInsertQueueApc function, 264–265
kernel-based keyloggers, 238
kernel code, 206
  64-bit malware and, 442
  breakpoints, 548
  malware creation of file, 554
kernel debugging
  looking at user-space code, 215–216
  setting up for VMware, 207–210
  WinDbg and, 205
kernel driver, creating service to load, 216
kernel mode
  binary translation by VMware, 373
  for debuggers, vs. user mode, 168–169
  in Windows, 158–159

kernel patch protection, 227
kernel space, APC injection from, 264–265
kernel32.dll, 17, 20, 159
  assembly code to find base address, 415
  finding in memory, 413–415
  imported functions, 16
  imports from, 480, 481
  name change by malware, 520, 529
  shellcode and, 413
  viewing imports from, 478
`KERNEL_DRIVER` service type, 153
kernel-mode APC, 263
kernel-mode code, looking at, 217–220
Kernighan, Brian, *The C Programming Language*, 110
`KeServiceDescriptorTable` function, 559
`KeTickCount` function, 549
key
  for cryptographic algorithms, 281
  in registry, 139
key initialization code, identifying, 622
keyboard inputs, 20
keyloggers, 4, 238–241
  analysis, 585, 597–599
  hooks for, 260
  indications of, 491, 581
KMixer.sys, 217
`KnownDLLs` registry key, 245
Krypto ANALyzer (KANAL), 283, 610, 619

L

lab environments, malware and, 298
labeling, in OllyDbg, 197
labs
  64-bit malware, 450–451
    solutions, 723–732
  anti-debugging, 367–368
    solutions, 655–669
  anti-disassembly, 350
    solutions, 645–655
  anti-virtual machine (anti-VM) techniques, 381–382
    solutions, 670–684
  C++ analysis, 439–440
    solutions, 712–723
  C code constructs in assembly, 133–134
    solutions, 501–513
  covert launching techniques, 266–267
    solutions, 586–607
  data encoding, 295–296
    solutions, 607–626
  dynamic analysis, 61–62
    solutions, 482–493
  IDA Pro, 107–108
    solutions, 494–501
  malware behavior, 251–252
    solutions, 566–586
  network signatures, 323–324
    solutions, 626–645
  OllyDbg, 202–203
    solutions, 530–548
  packers, 403
    solutions, 684–695
  shellcode analysis, 425–426
    solutions, 696–712
  static analysis, 27–28
    solutions, 477–481
  WinDbg, 228
    solutions, 548–566
  Windows malware, 162–163
    solutions, 513–529
last in, first out (LIFO) structure, 77
launchers, 4, 231–232. *See also* covert launching techniques
`LdrGetProcAddress` function, 15
`LdrLoadDll` function, 15, 459
LDT (local descriptor table), 374
LDT register (LDTR), 374
`lea` instruction (load effective address), 73–74
leaf functions, 446
`leave` instruction, 77
left rotation (`rol`), 76
legacy graphs, in IDA Pro, 98
libdisasm disassembly library, 330
LIFO (last in, first out) structure, 77
linear disassembly, 329–331
  vs. flow-oriented, 331–332
linked libraries, executable use of, 15–18
linked list traversal, 130–132
links, navigating in IDA Pro, 92–93
Linux virtual machine, 57

`listen` function, 143, 144
listen mode, in Netcat, 52
`LIST_ENTRY` structure, 414, 565
little-endian data, 69
`lm` command, in WinDbg, 212, 223, 553, 555
`ln` command, in WinDbg, 213
loaddll.exe, 401
  OllyDbg use of, 191
loader, 232. *See also* launchers
loading
  device drivers, 226
  executable, 384–385
    in IDA Pro, 88–89
`LoadLibrary` function, 13, 15, 261, 387, 413, 417, 459, 520, 521, 545, 546, 547, 585
  finding last call, 692
  unpacking stub import of, 385
`LoadResource` function, 254, 459, 596, 600, 609
loc links, in IDA Pro, 93
local administrator, user running as, 245
local descriptor table (LDT), 374
local hooks, 260
local machine, loading buffer with hostname, 650
Local Security Authority Subsystem Service (LSASS) process, 236
local user accounts, password hashes of, 236
local variables, vs. global, 110–112
locally unique identifiers (LUIDs), 238, 247
locations, name changes in IDA Pro, 100
`LockResource` function, 596, 609
logging
  active window, 239
  of credentials, 570–571
  errors in malware, 674
  in OllyDbg, 197
logical operators, 75
logon, credential stealers, 234–241
long pointer (LP) type, in Windows API, 136
`LookupPrivilegeValueA` function, 247, 730
loopback encoding algorithm, 277
loops
  in C code, 116–118
  setting breakpoints after, 394
LordPE, 469
`LowLevelKeyboardProc` export, 20
low-level language level, 66, 67
`LowLevelMouseProc` export, 20
low-level remote hooks, 260
`LsaEnumerateLogonSessions` function, 459
lsaext.dll, 236
LSASS (Local Security Authority Subsystem Service) process, 236
lsass.exe, 236
LUIDs (locally unique identifiers), 238

M

MAC address, for virtual machine, 371
machine code, 67
magic constant, 283
magic number, 376
`main` function
  determining start, 530
  starting analysis at, 501–502
main memory, in x86 architecture, 68, 69
major function table, 218
  analyzing functions of, 564–566
  finding, 220
Malcode Analyst Pack, 469–470
malicious documents, Process Explorer to analyze, 50
`malloc` function, 578
malware. *See also* Windows malware
  64-bit, 441–449
  analyzing without unpacking, 400–401
  attempts to delete itself, 531
  double-packed, 397
  hashing for identifying, 10
  observing in natural habitat, 298
  packed and obfuscated, 13–14
  running, 42–43
  safe environment for running, 14
  searching for evidence of encoding, 608
  self-deletion scripting code, 674
  types, 3–4
malware analysis
  creating machine for, 31–33
  danger of overanalysis, 308

general rules, 4–5
goals, 1–2
risks of using VMware for, 36–37
techniques, 2–3. *See also* dynamic analysis; static analysis
tools, 465–475
malware behavior, 231–250
backdoor, 232–234
botnets, 234
credential stealers, 234–241
GINA interception, 235–236
hash dumping, 236–238
keystroke logging, 238–241
downloaders and launchers, 231–232
indications of, 298–299
labs, 251–252
solutions, 566–586
persistence, 241–245
DLL load-order hijacking, 244–245
trojanized system binaries, 243–244
Windows Registry for, 241–243
privilege escalation, 245–247
`SeDebugPrivilege`, 246–247
remote administration tool (RAT), 233–234
user-mode rootkits, 247–250
IAT hooking, 248
inline hooking, 248–250
Mandiant
ApateDNS, 51–52
Red Curtain, 388
mangling, 430–431
manual unpacking, 389–397
`MapViewOfFile` function, 137–138, 459, 520, 527, 583
`MapVirtualKey` function, 459
mass malware, 4
MD5 (Message-Digest Algorithm 5), 10
media files, shellcode stored within, 423
`memcmp` function, 497
`memcpy` function, 596
memory
addresses for global variables, 111
allocation for objects, 437
checking for VMware artifacts, 373
copying PE sections into, 593–594
dumping executable from, 390, 400, 469
finding device driver in, with WinDbg, 563
finding *kernel32.dll* in, 413–415
function dump, 174
processes and, 147
memory-access violations, 177
memory address operands, 69
memory breakpoint, in OllyDbg, 188, 190–191
Memory dump window, in OllyDbg, 183
memory map, to locate DLLs, 546–547
Memory Map window, in OllyDbg, 183–185
memory window, WinDbg reading from, 210–211
Memoryze, 470
message box, malware creation of, 586
Message-Digest Algorithm 5 (MD5), 10
message flow, in Windows with and without hook injection, 259
Metasploit, 245, 418
methods
in C++ class, 427
overloading, 430–431
microcode, in x86 architecture, 66
Microsoft. *See also* Windows
Component Object Model (COM), 154–157
documentation, 453
firewall, 33
Hyper-V, 31
Software Data Execution Prevention (DEP), 345
symbols, 212–215
Virtual PC, 31
Visual Studio, 16
calling conventions, 121
wide character string, 11
Microsoft Developer Network (MSDN), 414
Microsoft signed binary, verifying, 48
MIME (Multipurpose Internet Mail Extensions) standard, Base64 and, 277
`MmGetSystemRoutineAddress` function, 224, 459

mneumonics, in instructions, 69
`Module32First` function, 459
`Module32Next` function, 459
modules
  getting name of, 602
  listing in WinDbg, 212
modulo operation, 75, 112, 113
`mov` instruction, 73, 76, 79, 338, 500
  position dependence, 409
`movsb` instruction, 82
`movsd` instruction, 528
`movsx` instruction, 81
MS-DOS Stub Program, 22
MSDN (Microsoft Developer Network), 414
MSDN online, 19
`msg` keyword, in Snort, 304
msgina.dll, and GINA, 235
msvcrt.dll, imports from, 480
`mul` instruction, 75
multibyte encoding algorithm, 276
Multipurpose Internet Mail Extensions (MIME) standard, Base64 and, 277
multithreaded version, of Windows reverse shell, 233
mutants, 151
mutexes, 58, 482, 513
  creating, 483, 515
  interprocess coordination with, 151–152
  malware creation of, 585
  malware use of, 517
MZ header, in PE executable, 594

N

named constants, 102–103
named pipes, watching for input on, 634
names
  conventions for functions, 17
  hashed exported, for symbol resolution, 417–418
  for lab files, 27
  of locations, changing in IDA Pro, 100
  for malicious DLL, 257
  of malware, string comparison, 666
  mangling in C++, 431
  of modules, getting, 602
  for mutexes, 151
Names window, in IDA Pro, 91
namespaces, files accessible via, 138–139
NAT (Network Address Translation), 311
  for VMware, 34
Native API, in Windows, 159–161
native applications, 161
`Navigate` function, 155, 313
`nc`. *See* Netcat (`nc`)
Nebbett, Gary, *Windows NT/2000 Native API Reference*, 160
nested `if` statements, 113, 114–116
`net start cisvc` command, 582
`net start` command, 43, 152, 581
Netcat (`nc`), 52–53, 470, 483, 634
  examining results, 485
  output when listening on port 80, 504
  reverse shells, 232–233
`NetScheduleJobAdd` function, 459, 547
`NetShareEnum` function, 459
network adapter, bridged, 34
Network Address Translation (NAT), 311
  for VMware, 34
network countermeasures, 297
Network filter, in procmon, 46
network interface cards (NICs), virtual, 371
network signatures, 2, 297–322
  analysis, 631–632
  attacker's perspective and, 321–322
  creating, 490
  creating for Snort, 317
  creating XOR brute-force, 273
  Emerging Threats list of, 304
  generating, 643
  labs, 323–324
    solutions, 626–645
  for malware infection detection, 2
  User-Agent field for, 637
networking APIs, 143–145
networks
  analysis, 538–539
  capturing traffic, 580
  faking, 51–53
  finding code, 313

host-only, 32–33
indications of functioning, 572
knowing sources of content, 314
server and client sides, 144–145
virtual, 32
`new` operator, 435, 437, 712
nibble, 278
NICs (network interface cards), virtual, 371
No Pill technique, 375. *See also* `sldt` instruction (No Pill)
`NopBytes` function, 339
nonleaf functions, 446
nonprivileged mode, 177
nonvirtual functions, vs. virtual, 432–436
NOP instruction, in x86 architecture, 76
NOP sequence, 337
NOP sled, shellcode and, 422–423
NOP-ing out instructions with IDA Pro, 340
Norman SandBox, 40
Norton Ghost, 30
`noscript` tags, malware commands from, 638
NSPack, 388
NT namespace, 138
`NtContinue` function, 161, 386
`NtCreateFile` function, 215, 224
ntdll.dll, 17, 159, 352, 414
`NTGlobalFlag` flag, 355, 659–660
`ntohl` function, 191
ntoskrnl.exe, 159
malicious drivers and, 207
`NtQueryDirectoryFile` function, 459, 559
as hook function, 556–558
`NtQueryInformationFile` function, 160
`NtQueryInformationKey` function, 160
`NtQueryInformationProcess` function, 352, 460
`NtQueryInformationThread` function, 160
`NtQuerySystemInformation` function, 160
`NtReadFile` function, 160
`NtSetInformationProcess` function, 460
`NtWriteFile` function, 160, 215
NULL bytes, avoiding in shellcode, 421
NULL-preserving single-byte XOR encoding, 273–274
NULL terminator, 11
Number of Opcode Bytes option, 335
NXDOMAIN option, 52

O

`!object` command, in WinDbg, 552
object-oriented programming, 427–432
overloading and mangling, 430–431
`this` pointer, 428–430
objects, creating and destroying in C++, 437
OEP. *See* original entry point (OEP)
OfficeMalScanner, 470
offset links, in IDA Pro, 93
`OleInitialize` function, 154, 460, 518
OllyDbg, 168, 179–201, 364, 470
analysis, 691
assistance features, 197
breakpoints, 188–191
choosing to debug arguments, 532
debug window from, 173
default settings for exceptions, 362
disassembly view, 533
examining hook in, 579–580
exception handling, 194–195
executing code, 186–187
finding function addresses with, 410
forcing code disassembly, 689
interface, 181–183
as just-in-time debugger, 411, 696
labs, 202–203
solutions, 530–548
loading DLLs, 191–192, 401
loading malware, 180–181, 656
loading packed executable in, 389
memory map to examine DLL load locations, 546
Memory Map window, 183–185
opening malware with, 538
`OutputDebugString` format string vulnerability, 365
packed program and, 387
patching, 195–196
pausing before TLS callback, 361
plug-ins, 197–200, 354

OllyDbg, *continued*
  premature termination of program in, 662
  rebasing, 184–185
  Run Trace option, 395
  screen capture decoding with, 616
  scriptable debugging, 200–201
  shellcode analysis, 196–197
  `strncmp` function in, 663
  tracing, 192–194
  viewing threads and stacks, 185–186
  vulnerabilities in, 363–365
  WinUpack and, 400
OllyDump, 198, 389–390
  dumping unpacked program, 694
  Find OEP by Section Hop (Trace Into), 686
  Find OEP by Section Hop (Trace Over), 685, 687
  forcing code disassembly, 686
opcodes, in x86 architecture, 67, 70
open source sniffer, 53
`OpenMutex` function, 152, 460, 522
`OpenProcess` function, 460
`OpenProcessToken` function, 247, 730
`OpenSCManager` function, 153, 460, 514, 515, 549, 550, 554
`OpenService` function, 549, 550
OpenSSL, 281
operands
  formatting in IDA Pro, 100
  in x86 architecture, 69, 70
operating systems (OSs), backup images of, 30
Operation filter, in procmon, 45
operational replication, 308
operations security (OPSEC), 299
`or` instruction, 76
OR logical operator, in x86 architecture, 75
ordinal, executable import of functions by, 16–17, 43
original entry point (OEP)
  code around, 399
  in DLLs, 401
  finding, 391–395
    with automated tools, 391–392
    manually, 392–395
  indications of, 694
  transferring execution to, 386
  unpacking stub and, 384
orphaned process, 490–491
OSR Driver Loader, 470–471
OSs (operating systems), backup images of, 30
Outlook Express, 579–580
output functions, tracing from, 286
`OutputDebugString` function, 353, 460, 664
overanalysis, danger of, 308
overloading, 430–431

P

packed DLLs, 401
packed executables
  detecting, 23
  entropy calculation for, 387–388
  identifying, 387–388
  loading in OllyDbg, 389
  repairing import table for, 390
packed files
  indications of, 480
  strings and, 483
packed malware, 13–14
  detecting with PEiD, 14
packers, 383–402
  anatomy, 384–387
  labs, 403
    solutions, 684–695
  resolving imports, 385
  tail jump, 386
  tips and tricks for common, 397–400
  unpacking illustrated, 386–387
packet listing, in Wireshark, 53
packet sniffing, with Wireshark, 53–55
packing algorithm, program to run in reverse, 389
padding characters, Base64 string and, 279
Parallels, 31
parent classes in C++, 432
  child class functions from, 436
parent-child relationships, in classes, 432
parsing routines
  analyzing, 318–320
  IDA Pro graph of, 318–319
pass-the-hash attacks, 236

password check function, 533
  testing if disabled, 534
passwords, 661
  getting correct, 665
  sniffing, 53
`PatchByte` function, 337, 339
PatchGuard, 227
patching, in OllyDbg, 195–196
payload rule options, in Snort, 303
PCRE (Perl Compatible Regular Expression) notation, in Snort, 305, 316
`pcre` Snort rule keyword, 305
`.pdata` section, in PE file, 22
PDF Dissector, 471
.pdf documents, 704–712
  analyzing with Process Explorer, 50
  objects created for, 716
PDF Tools, 471
PE Explorer, 26, 471
  unpacking plug-ins, 388
PE file format. *See* Portable Executable (PE) file format
PEB (Process Environment Block) structure, 352, 591–592
  documented, 354
PEBrowse Professional, 26
PECompact, 397–398
`PeekNamedPipe` function, 460, 634
PEiD, 471, 478, 479–480
  detecting packers with, 14
  KANAL output, 610
peripheral devices, connecting and disconnecting, 34–35
Perl Compatible Regular Expression (PCRE) notation, in Snort, 305, 316
persistence, 241–245, 572
  `AppInit_DLLs` for, 575
  DLL load-order hijacking, 244–245
  of registry, 139
  trojanized system binaries, 243–244
  Windows Registry for, 241–243
Petite, 398
PEview, 471, 478
  examining PE files with, 22–24
  finding base address with, 545
  original and trojanized versions of *cisvc.exe*, 584–585
PhantOm plug-in, 354, 658, 659, 665
Phatbot, VMware detection, 375–376
phishing, targeted, 299
PIC (position-independent code), 408–409
pipe symbol (|), in Snort, 304
plug-ins
  for extending IDA Pro, 103–106
  in OllyDbg, 197–200, 354
  PEiD, running of executables, 14
  third-party, for Internet Explorer, 157
pointers, handles vs., 137
Poison Ivy, 189, 234
  tracing, 193–194
  use of `VirtualAlloc` function, 189–190
polling, 239
polymorphism, 434
`pop` instruction, 77, 79
  after `call`, 409–411
  and tail jump, 394
pop-up ads, 560–561
`popa` instruction, 79, 244
`popad` instruction, 79
port 80, backdoor and, 232
Portable Executable (PE) file format, 14–15, 396
  copying sections into memory, 593–594
  examining file structure, 486
  header vulnerabilities, OllyDbg, 363–365
  headers and sections, 21–26
    summary information, 26
  IDA Pro support for, 87
  indications in, 729
  packed executables formatting of, 385
  parsing export data, 415–417
  PEview for examining, 22–24
  rebasing and, 184
  Resource Hacker tool for viewing, 25–26
  resource section, 254, 567
  section headers, and OllyDbg crash, 364
  `.tls` section, 360, 662
ports, malware use of, 52
position-independent code (PIC), 408–409

`POST` method, 309
`printf` function, 120
  call compiled for 32-bit processor, 445
  call compiled for 64-bit processor, 446
  IDA Pro problems recognizing, 502
privilege escalation, 245–247
  `SeDebugPrivilege`, 246–247
privileged mode, 177
ProcDump, 400
Process activity filter, in procmon, 46
process context, 158
Process Environment Block (PEB) structure, 352, 591–592
  documented, 354
Process Explorer, 58, 472, 483
  comparing strings, 49
  Dependency Walker, 49
  for finding DLL injection, 589
  Verify option, 48–49
  viewing processes with, 47–50
Process Hacker, 472
Process Monitor (procmon), 43–46, 472, 483
  boot logging options, 46
  display, 44
  Filter dialog, 484
  filtering in, 44–46
  filters on toolbar, 46
  reviewing results, 58
  toggling event capture on and off, 749
Process Name filter, in procmon, 45
Process Properties window, Strings tab, 49
process replacement, 48–49, 257–259
`Process32First` function, 255, 263, 460
`Process32Next` function, 255, 263, 460
processes
  creating, 147–149, 590
  dumping from memory, 390, 400
  dynamically resolving enumeration imports, 600–601
  EBX register of suspended newly created, 591
  enumerating, 601
  for following running malware, 147–149
  function to open and manipulate, 20
  hidden, 566
  interprocess coordination with mutexes, 151–152
  Properties window for, 48
  resuming suspended, 595
  starting and replacing, 596
`ProcessHeap` flag, in PEB structure, 355
procmon. *See* Process Monitor (procmon)
programs. *See* executables
prologue
  64-bit code, 446–447
  in functions, 77
Properties window, in Process Explorer, 48
protocols, attackers mimicking existing, 309–310
psapi.dll, 586, 600
`push` instruction, 77, 79, 244, 329, 689
  vs. `mov`, 120
  with `return` instruction for tail jump, 399
  to start functions in disassembly, 394
Pwdump, 236
PyCommand Python script, 200–201
PyCrypto cryptography library, 290, 625
  potential pitfalls, 626
Python, 472
  IDAPython, 105–106
  program to decode Base64-encoded string, 289
  PyCommand script, 200–201
  script for converting data to string, 500–501

Q

query, of I/O communication port, 375–377
`QueryPerformanceCounter` function, 358–359, 460, 667–668
`QueueUserAPC` function, 263, 460

R

radio-frequency identification (RFID) tokens, 235
`RaiseException` function, 157, 344

Random function, 313, 314
random number generator seed, 484
RAT (remote administration tool), 233–234
raw data, translating to Base64, 277–278
RC4 algorithm, 283
RCPT command (SMTP), 572
.rdata section, in PE file, 21
rdtsc function, 669
rdtsc instruction, for timing check, 358
read breakpoints, for finding tail jump, 394
ReadFile function, 137, 219
  origin of handle passed to, 623
ReadProcessMemory function, 460, 590
rebasing, 88
  in OllyDbg, 184–185
receiving data, and code analysis, 312
recovery of hidden files, 559–560
recursive function, 527
recv function, 143, 144, 313, 461
Red Pill anti-VM technique, 374–375. *See also* sidt instruction (Red Pill)
reference Snort rule keyword, 305
RegCreateKeyEx function, 448
RegDeleteKeyEx function, 448
Regedit (Registry Editor), 140–141
RegGetValue function, 141
Regional Internet Registries (RIRs), 301
register operands, 69
RegisterClassEx function, 20
RegisterHotKey function, 20, 461
registers, 68
  shifting, 75
  in x64 architecture, 443
  in x86 architecture, 71–73
Registers window, in OllyDbg, 182
registries, for Internet addresses, 301
Registry (Windows), 139–143
  analyzing code, 141–142
  common functions, 141
  defining services, 242
  function for string search, 679
  indications of modification, 508
  for persistence, 241–243
  root keys, 140
  scripting with *.reg* files, 142–143
  snapshots with Regshot, 50–51
  VMware artifacts in, 371
Registry Editor (Regedit), 140–141
Registry filter, in procmon, 46
registry keys, 20
  malware and, 42
  references to debuggers, 356
\Registry\Machine strings, 549
RegMon tool, 43
RegOpenKey function, 461
RegOpenKeyEx function, 141, 142, 448, 508
RegSetValueEx function, 141, 508
Regshot, 50–51, 56, 472, 487–488
regular expressions, for identifying malware patterns, 631
relative addresses, vs. absolute addresses, in OllyDbg, 184–185
relative virtual addresses (RVAs), for PE files, 416
ReleaseMutex function, 151
.reloc section, in PE file, 22
remote administration tool (RAT), 233–234
remote hooks, 260
remote machine, program receiving commands from, 522
remote process, VirtualAllocEx function and, 255
remote shell session function, 497
remote socket, program connecting to, 727
rep instructions, in x86 architecture, 81–83
REP MOVSx instruction, 536
replication, operational, 308
resource extraction import functions, 567
Resource Hacker, 25–26, 472, 482, 554, 596–597
resource section
  executable file stored in, 555
  loading data from, 481
resources
  imports for manipulating, 600
  obfuscated with single-byte XOR encoding, 609

resources management, processes for, 147
ResumeThread function, 259, 461
ret instruction, 77, 386, 409
retn instruction, 342–343, 693
return instruction, for tail jump, push instruction with, 399
return pointer, abuse, 342–343
rev keyword, in Snort, 304
reverse-engineering, 3
  network protocols, 53
  in x86 disassembly, 67–68
reverse-engineering environment, 466
reverse IP lookups, 301
reverse shell, 232–233
  analysis, 544
  creating, 703
reversible cipher, 271
RFID (radio-frequency identification) tokens, 235
right rotation (ror), 76
Rijndael algorithm, 618
RIP-relative addressing, 443
RIRs (Regional Internet Registries), 301
Ritchie, Dennis, *The C Programming Language*, 110
Robin, John, 373
RobTex, 302
rogue byte, 337
ROL encoding algorithm, 276
rol instruction, 76
Roman Empire, Caesar cipher and, 270
root key, in registry, 139
rootkits, 4, 221–225
  finding, 555–556
  interrupts and, 225
  user-mode rootkits, 247–250
ROR encoding algorithm, 276
ror instruction, 76
ROT encoding algorithm, 276
rotation, instruction for, 76
.rsrc section, in PE file, 22, 25–26
RtlCompareMemory function, 557–558
RtlCreateRegistryKey function, 461, 549, 553
RtlInitUnicodeString function, 219, 559
RtlWriteRegistryValue function, 461, 549, 553
rtutils.dll, comparing trojanized and clean versions, 243
rule options, in Snort, 303
Run subkey, for running programs automatically, 140
run trace, in OllyDbg, 193
rundll32.exe, 42–43, 488
  filter for process, 572
  for running DLL malware, 42–43
running process, attaching OllyDbg to, 181
running services, listing, 152
runtime linking, 15
RVAs (relative virtual addresses), for PE files, 416

S

safe environment, 29. *See also* virtual machines
SafeSEH, 345
SAM (Security Account Manager), password hashes of local user accounts, 236
SamIConnect function, 237, 461
SamIGetPrivateData function, 237, 461
SamQueryInformationUse function, 461
SamrQueryInformationUser function, 237
samsrv.dll library, obtaining handle to, 237
sandboxes, 40–42, 473
Sandboxie, 473
sc command, 555
scareware, 4
scasb instruction, 82
scasx instruction, 81
ScoopyNG, 379
screen capture, function for, 615
ScreenEA function, 105
scriptable debugging, in OllyDbg, 200–201
scripts, IDC, 104–105
searching
  default order for loading DLLs in Windows XP, 245
  in IDA Pro, 94–95
  for symbols, 212–213
Section Hop, 391
Secure Hash Algorithm 1 (SHA-1), 10

Security Account Manager (SAM), password hashes of local user accounts, 236
security descriptor, 246
SeDebugPrivilege privilege-escalation procedure, 603
segment registers, 71
SEH (Structured Exception Handling), 157, 665
  chain, 345
  misusing, 344–346
Seitz, Justin, *Gray Hat Python*, 201
self-decoding, 288–289
self-deletion scripting code, 674
send function, 143, 144, 313, 461
  installing inline hook, 574
sending data, and code analysis, 312
server side of network, 144–145
ServiceMain function, 673
services
  defining in Registry, 242
  function creating, 677
  functions indicating creation, 549
  handles for, OpenService function for, 550
  malware creation, 514
  malware installed as, 487
  program creating, 561
  sc command for information about, 555
  in Windows, 152–154
SetColor function, 105
setdll tool, 262
SetFilePointer function, 709
SetFileTime function, 461
SetThreadContext function, 259, 461, 590, 595
SetWaitableTimer function, 516
SetWindowsHookEx function, 20, 239, 260, 261, 462, 597
SetWindowText function, 20
SF (sign) flag, 72
sfc_os.dll, 604
SfcTerminateWatcherThread function, 462, 604
sgdt instruction
  virtual machine and, 374
  and VMware detection, 375
SHA-1 (Secure Hash Algorithm 1), 10
shared files, 138
shared folders, 36
  in VMware, 380
shell, connecting pipe to output, 624
Shell32.dll, 20
shellcode
  64-bit version, 442
  decoder with alphabetic encoding, 697
  finding, 423–424
  hash array, 700–701
  locating open handle to PDF, 708
  payload, 698
  writing into *cisvc.exe*, 583–584
shellcode analysis, 407–424
  dynamic, 706–707
  encodings, 421–422
  identifying execution location, 409–413
  labs, 425–426
    solutions, 696–712
  loading code for, 408
  manual symbol resolution, 413–418
    finding *kernel32.dll* in memory, 413–415
    parsing PE export data, 415–417
    using hashed exported names, 417–418
  NOP sled, 422–423
  in OllyDbg, 196–197
  position-independent code (PIC), 408–409
shellcode_launcher.exe, 408, 411, 696
ShellExecute function, 462, 636
shifting registers, 75
shl instruction, 75, 76
ShowWindow function, 20
shr instruction, 75
sid keyword, in Snort, 304
sidt instruction (Red Pill), 375, 670, 671
  virtual machine and, 374
signature-based IDSs, 302
signatures. *See* network signatures
simple ciphers, 270–280
  Base64, 277–280
  Caesar cipher, 270
  other encoding schemes, 276–277
  XOR cipher, 271–276
simple instructions, in x86 architecture, 73–76

single-byte XOR encoding, 271
single-stepping
  in debuggers, 169–170, 176
  and icebp instruction, 363
  in OllyDbg, 187
sinkhole, 297
Size of Raw Data, 23–24
SizeOfRawData field, in PE header, 365
SizeofResource function, 254, 596, 609
sldt instruction (No Pill), 670, 672
  and VMware detection, 375
Sleep function, 239, 263, 329, 479
  in loop, 629
  parameter for, 499
  sandboxes and, 41
Sleuth Kit, The (TSK), 473
smart cards, 235
snapshots
  comparing with Regshot, 50–51, 58
  of registry, 487–488
  of virtual machines, 35–36
Snort, 473
  analyzing parsing routines, 318–320
  creating signature, 317
  false positives in, 306
  intrusion detection with, 303–304
  Perl Compatible Regular Expression (PCRE) notation in, 305
  signature for rule, 632
  targeting multiple elements, 320–321
sockaddr_in structure, 543, 702
socket function, 143, 144, 313
  symbolic constants for, 500
sockets
  Berkeley compatible, 143–144
  code for creating, 701–702
  program connecting to remote, 727
SoftICE, 168
software, modifying execution with debugger, 177
software breakpoints, 357
  vs. hardware, 687
  in OllyDbg, 188–189
Software Data Execution Prevention Software (DEP), 345
source-level debuggers, vs. assembly-level, 168
spam-sending malware, 4
spear-phishing, 299
special files, in Windows API, 138–139
sprintf function, annotated code for arguments, 628–629
spyware, 20
SSDT (System Service Descriptor Table)
  checking for, 222
  hooking, 221–222
stack, 69
  addresses for local variables, 111
  ExceptionHandler code and, 345
  fixing for function, 506–507
  identifying parameters pushed onto, 502–503
  objects created on, 437
  viewing in OllyDbg, 185–186
  in x64 architecture, differences in usage, 443–447
  in x86 architecture, 77–80
    function calls, 77–78
    layout, 78–80
stack overflow, 158
stack pointer, negative number for, 348
stack variables, automatically naming, 100
Stack window, in OllyDbg, 182–183
stack-formed strings, decoding, 540–541
stack-frame analysis, thwarting, 347–349
standard back trace, in OllyDbg, 192–193
StartAddress function, 516
START_PENDING, as service status, 517
StartService function, 153, 549, 550, 554
StartServiceCtrlDispatcher function, 462, 514
STARTUPINFO structure, 148, 233
  manipulating, 544
static analysis, 9–26, 65
  advanced, 3
  basic, 2
  combining with dynamic analysis, 307–321
  Dependency Walker for, 468
  example, *PotentialKeylogger.exe*, 18–21

labs, 27–28
solutions, 477–481
techniques, 482–485
static IP addresses, 632
static libraries, 145
static linking, 15
static unpacking programs, automated, 389
static values in memory, 69
status flags, 71
`STATUS_BREAKPOINT` exception, 362
`stdcall` calling convention, 120
stepping, in OllyDbg, 187
stepping-into, in debuggers, 170–171
stepping-over, in debuggers, 170–171, 187
Storm worm, 375
`stos`*x* instruction, 81
`str` instruction, 670, 671–672
to detect VMware, 377–378
and virtual machine detection, 377
`strcat` function, risk in using, 421
`strcpy` function, risk in using, 421
`stricmp` function, 527
string instructions, 81
strings
comparing in Process Explorer, 49
comparison of malware names, 666
concatenation functions, 535
decoding stack-formed, 540–541
decoding XOR encoded, 542–543
finding, 11–13
finding anti-VM techniques using, 679–683
functions for manipulating, 715
in malware, 487
obfuscated comparison, 640–641
packed files and, 483
Python script for converting data to, 500–501
recognizing in cryptographic algorithms, 281–282
sending to debugger for display, 353
strings listings, identifying keyloggers in, 240–241
Strings tool, 473
to search executable, 11–12
Strings window, in IDA Pro, 91
`strncmp` function, 256, 523, 524, 715
for module name comparison, 666
in OllyDbg, 663
`strncpy` function, 611
`strrchr` function, 541, 725
`strstr` function, 640
Structured Exception Handling (SEH), 157, 665
chain, 345
misusing, 344–346
structures
applying in IDA Pro, 547–548
`AT_INFO`, 547–548
`EPROCESS`
changing, 566
examining in WinDbg, 565
identifying, 128–130
`InInitializationOrderLinks` list of, 414
`LIST_ENTRY`, 414, 565
manually checking, 353–356
Microsoft symbols and viewing information on, 213–214
overlaying data onto, 214
`sockaddr_in`, 543, 702
`STARTUPINFO`, 148, 233, 594
`SYSTEMTIME`, 516
time-related, manipulating, 516
`UNICODE_STRING`, for Windows kernel, 219
Structures window, in IDA Pro, 92
SUB encoding algorithm, 276
sub links, in IDA Pro, 93
subkey, in registry, 139
subtraction, instruction for, 74
suspended process, resuming, 595
suspended state, creating process in, 258
`SuspendThread` function, 462
SvcHost DLLs, 242–243
svchost.exe, 257–258
malware launch from, 488
running as orphaned process, 490–491
`switch` statement, 121–126, 722–723
graph indicating, 509–510
if style for, 122–123, 124
jump table for, 123–126, 641–642
symbolic constants, for `socket` function, 500

symbolic links, creating, 562
symbols, 212–215
  configuring, 215
  searching for, 212–213
  and viewing structure information, 213–214
SYSCALL instruction, 158, 221
SYSENTER instruction, 158
Sysinternals, Autoruns program, 241
SYSTEM account, 152
system binaries, trojanized, for persistence, 243–244
system calls, filtering on, 45
system function, 462
system memory. *See* memory
system residue, checking for, 356
System Service Descriptor Table (SSDT)
  checking for, 222
  hooking, 221–222
SystemFunction025 function, 237
SystemFunction027 function, 237
SYSTEMTIME structure, 516
SystemTimeToFileTime function, 516

T

tail jump, 386
  eliminating code as, 693
  examining code for, 687–688
  and finding OEP, 392
  for program packed with UPack, 399
targeted malware, 4
targeted phishing, 299
TCP handshake, capturing, 59
TCPView, 473
TEB (Thread Environment Block), 344
TerminateProcess function, IAT hooking of, 248
test instruction, 80
text mode, in IDA Pro, 90–91
.text section, in PE file, 21, 22
TF (trap) flag, 72
The Sleuth Kit (TSK), 473
Themida, 400
Thinking in C++ (Eckel), 428
this pointer, 428–430, 712–713, 719
  in disassembly, 430
thread context, 149
Thread Environment Block (TEB), 344
thread identifiers (TID), 575–576
Thread Information Block (TIB), 344
thread local storage (TLS) callbacks, 359–361
Thread32First function, 462
Thread32Next function, 462
threads
  program accessing context of, 591
  targeting, 261
  viewing in OllyDbg, 185–186
  in Windows, 149–151
ThreatExpert, 40
TIB (Thread Information Block), 344
TID (thread identifiers), 575–576
Time Date Stamp description, in PE file, 22–23
time-related structures, manipulating, 516
timestomping, 535
timing checks, 357–359
  GetTickCount function, 668–669
  with QueryPerformanceCounter, 667–668
  rdtsc function, 669
TLS (thread local storage) callbacks, 359–361
Toolhelp32ReadProcessMemory function, 462
Tor, 300, 474
tracing, in OllyDbg, 192–194
traffic logs, of malware activities, 312
transferring files, from virtual machine, 36
trap flag, 176–177
trojanized system binaries, for persistence, 243–244
Truman, 474
TSK (The Sleuth Kit), 473
type library, loading manually in IDA Pro, 102
types, in Windows API, 136

U

u (unassemble) command, in WinDbg, 212
Ultimate Packer for eXecutables. *See* UPX (Ultimate Packer for eXecutables)

unconditional jump, 80, 517
undo feature, snapshots as, 35
unescape function (JavaScript), 423, 705–706
unhandled exception, 344
UnhookWindowsHookEx function, 261
Unicode strings, 11–12
UNICODE_STRING structure, for Windows kernel, 219
uniform resource locators (URLs), opening to download malware, 651–652, 654
unload function, analysis in WinDbg vs. IDA Pro, 553
UnMapViewOfSection function, 592
unpacking, 14, 685–686
  analyzing malware without, 400–401
  example, 386–387
  manual, 389–397
unpacking stub, 383, 384, 389, 692
  size of, 399
UPack, 388, 398
UPX (Ultimate Packer for eXecutables), 14, 388, 389, 475
  packing with modified version, 684–685
  tips and tricks, 397
UPX-packed malware, 479
URLDownloadToCacheFile function, 232, 606, 626, 628, 642
URLDownloadToFile function, 313, 462, 482
URLs (uniform resource locators), opening to download malware, 651–652, 654
USB flash drives, 206
user mode
  calls from application, 206–207
  for debuggers, vs. kernel mode, 168–169
  in Windows, 158–159
user space
  APC injection from, 263–264
  keyloggers, 239–240
  looking at code, 215–216
user32.dll, 17, 20, 545
User-Agent, 312, 317
  dynamically generated, 511
  for malware, 303, 310, 628
  string for signature, 643
user-mode APC, 263
user-mode rootkits, 247–250
  IAT hooking, 248
  inline hooking, 248–250

V

value entry, in registry, 140
variables, global vs. local, 110–112
VERA (Visualizing Executables for Reversing and Analysis), 475–476
victim information, malware gathering of, 722
viewing processes, with Process Explorer, 47–50
virtual addresses, automatically naming, 100
virtual function tables, 434–435, 715
  recognizing, 435–436
virtual functions, vs. nonvirtual, 432–436
virtual machines, 29–38. *See also* anti-virtual machine (anti-VM) techniques
  crashing from procmon, 44
  disconnecting network, 32
  escaping, 380
  hiding precise location, 300
  malware detection on, 42
  malware efforts to detect, 369, 670–672
  option to boot debugger-enabled version of OS, 208
  setting up, 580
  structure, 30–31
  taking snapshots, 35–36
  transferring files from, 36
  using multiple, 33
virtual machine team, 33
virtual networking, 32, 57
Virtual Size, 23–24
VirtualAlloc function, 596
  Poison Ivy use of, 189–190

`VirtualAllocEx` function, 255, 256, 423, 462, 586, 588, 730
  and direct injection, 257
  and process injection, 254
`VirtualProtectEx` function, 462
`VirtualSize` field, in PE header, 365
virus, 4
  language setting and, 177
VirusTotal, 10, 475, 478, 479
Visualizing Executables for Reversing and Analysis (VERA), 475–476
VMcat, 380
VMchat, 380
VMdrag-n-hack, 380
VMdrag-n-sploit, 380
VMftp, 380
VMware, 30
  artifacts, 370–373
  configuring, 31–33
  configuring to create virtual connection with host OS, 208–209
  disk space use, 31
  kernel debugging setup, 207–210
  movie-capture feature, 37
  Network Address Translation (NAT) mode, 34
  record/replay, 37, 170
  risks of using for malware analysis, 36–37
  settings to avoid detection, 379–380
  Snapshot Manager, 35
VMware Player, 30
VMware Tools
  installing, 31
  stopping service, 371
VMware Workstation, 30–31, 475
VMwareService.exe, 370
VMwareTray.exe, 370
VMwareUser.exe, 370
.vmx file, 379
Volatility Framework, 475
Von Neumann architecture, 68
vtables, 434–435
  recognizing, 435–436

W

W, at end of Windows function name, 17
`WaitForMultipleObjectsEx` function, 263
`WaitForSingleObject` function, 151
`WaitForSingleObjectEx` function, 263
Watches window, in OllyDbg, 197
web applications, Burp Suite for testing, 467
web browser, malware determination of default, 699–703
WEP (Wired Equivalent Privacy), 34
`while` loops, 118
`WH_KEYBOARD` procedures, 260
`WH_KEYBOARD_LL` procedures, 260
whois requests, for domains, 301–302
whosthere-alt, 238
wide character string, 11
`WideCharToMultiByte` function, 462
Wi-Fi Protected Access (WPA), 34
Win32 device namespace, 138
`WIN32_SHARE_PROCESS` type, 153
WinDbg, 168, 205–227, 475
  arithmetic operators, 211
  breakpoints, 211–212
  connecting to virtual machine with, 209–210
  `EPROCESS` structure examined with, 565
  finding device driver in memory, 563
  vs. IDA Pro, 553
  for kernel debugger, 552
  labs, 228
    solutions, 548–566
  loading drivers, 226
  module listing, 212
  output, 726
  reading from memory, 210–211
  rootkits, 221–225
  SSDT viewed in, 222
  system breakpoint and, 361
  viewing driver, 551–553
window modes, in IDA Pro, 89–90
Windows
  blue screen, 158
  Component Object Model (COM), 154–157
  device drivers, 206
  executables, common sections, 22
  following running malware, 145–158
    dynamic link libraries (DLLs), 145–147

exceptions, 157–158
interprocess coordination with mutexes, 151–152
processes, 147–149
services, 152–154
threads, 149–151
functions for importing linked functions, 15
kernel vs. user mode, 158–159
Native API, 159–161
reverse shell, 233
tool for dumping process, 400
as virtual OS, 31
Windows 7, kernel issues in, 226–227
Windows 32-bit on Windows 64-bit (WOW64) subsystem, 447
Windows API, 136–139
code calling functions, 526
debugger detection with, 352–353
file system functions, 137–138
handles, 137
IDA Pro catalog of named constants, 102
networking APIs, 143–145
special files, 138–139
Windows debugger detection, 352–356
manually checking structures, 353–356
with Windows API, 352–353
Windows File Protection, 604, 605–606
Windows functions, 453–463
`Ex` suffix for, 17
Windows Internet (WinINet) API, 145, 313, 504, 639–640
advantages and disadvantages, 633
Windows malware, 135–161
labs, 162–163
solutions, 513–529
Windows NT/2000 Native API Reference (Nebbett), 160
Windows Registry. *See* Registry (Windows)
Windows Sockets (Winsock) API, 313
Windows Update binary
malware creation of handler, 605–606
moving to temporary directory, 605
string to temporary move, 606
Windows virtual machine, 57
Windows Vista, kernel issues for, 226–227
Windows XP
default search order for loading DLLs, 245
disabled firewall, 549
`WinExec` function, 462, 482
WinGraph32 application, 98
WinHex, 468, 596–597, 609
WinINet (Windows Internet) API, 145, 313, 504, 639–640
advantages and disadvantages, 633
wininet.dll, 17, 501
imports from, 480
Winlogon, opening handle to, 603
Winlogon Notify, 242
`WinMain` function, analysis, 640
WinMD5 calculator, 10, 11
WinObj Object Manager, 138
Winsock (Windows Sockets) API, 313
Winsock libraries, 143
WinUpack, 398–400, 691–695
Wired Equivalent Privacy (WEP), 34
Wireshark, 57, 475, 483
DNS and HTTP example, 54
Follow TCP Stream window, 54
packet sniffing with, 53–55
reviewing capture, 59
Witty worm, 138
Wlx, function names beginning with, 235
`WlxLoggedOnSAS` function, 463
Word documents, analyzing with Process Explorer, 50
`WORD` type, in Windows API, 136
worm, 4
WOW64 (Windows 32-bit on Windows 64-bit) subsystem, 447
`Wow64DisableWow64FsRedirection` function, 448, 463
WPA (Wi-Fi Protected Access), 34
`WriteFile` function, 137, 215, 219, 585
origin of handle passed to, 623
`WriteProcessMemory` function, 255, 256, 423, 463, 586, 590, 593
and direct injection, 257
and process injection, 254
ws2_32.dll, 17, 144, 483
imports from, 521
`WSAGetLastError` function, 144, 313

WSASocket function, 542, 727
WSAStartup function, 144, 313, 463, 542, 727
wshtcpip.dll, 483
WSock32.dll, 17
wupdmgr.exe, 604
  launching, 606

X

x command, WinDbg, 213
x64 architecture, 441
  differences in calling convention and stack usage, 443–447
  exception handling, 445
  malware with component for, 729
x64 Windows, kernel issues for, 226–227
x86-64 architecture, 441
x86 architecture, 68–85
  branching, 80–81
  C main method and offsets, 83–84
  code types and data access, 408
  conditionals, 80
  documentation manuals, 85
  instructions, 69–70
  instruction set, general-purpose register for, 409
  main memory, 69
  NOP instruction, 76
  opcodes and endianness, 70
  operands, 70
  registers, 71–73, 374
  rep instructions, 81–83
  search for vulnerable instructions, 670–672
  simple instructions, 73–76
  stack, 77–80
    function calls, 77–78
    layout, 78–80
x86 disassembly, 65–85
  levels of abstraction, 66–67
  reverse-engineer, 67–68
x87 floating-point unit (FPU), 411–413
Xen, 31
XOR cipher, 271–276
  brute-forcing, 271–273
  identifying loops in IDA Pro, 274–276
  NULL preserving single-byte, 273–274
XOR encoded strings, decoding, 542–543
XOR encoding loop, 620–621
xor instruction, 76, 596
  forms, 275
  searching for, 612–613
  searching for nonzeroing, 608
XOR logical operator, in x86 architecture, 75
xref. *See* cross-references (xref)
Xrefs window, in IDA Pro, 96

Y

YARA, 475
Yuschuk, Oleh, 179

Z

Zero Wine, 475
zero-day exploit, 33, 245
ZF (zero) flag, 72, 80
zombies, 234
ZwContinue function, 386
ZwCreateFile function, 219
ZwDeviceIoControlFile function, inline hooking of, 249–250
ZwUnmapViewOfSection function, 258
Zynamics BinDiff, 106