

INDEX

Note: Page numbers followed by *f*, *n*, or *t* indicate figures, notes, and tables respectively.

A

Address Resolution Protocol. *See* ARP
 cache poisoning
AdjustTokenPrivileges function, 142
AF_INET parameter, 10
ARP (Address Resolution Protocol)
 cache poisoning, 51–55
 adding supporting functions,
 53–54
 coding poisoning script, 52–53
 inspecting cache, 51
 testing, 54–55

B

BHPFuzzer class, 81–82
bing_menu function, 89–90
Bing search engine, 87–93
 defining extender class, 88–89
 functionality to parse results, 90–91
 functionality to perform query,
 89–90
 testing, 91–92, 91f–93f
bing_search function, 89–90
Biondi, Philippe, 47
BitBlt function, 116
Browser Helper Objects, 128–135
brute force attacks
 on directories and file locations,
 65–68
 applying list of extensions to
 test for, 67–68
 creating list of extensions, 68
 creating Queue objects out of
 wordlist files, 66
 setting up wordlist, 68
 testing, 68

 in HTML form authentication,
 69–74
 administrator login form,
 69–70
 general settings, 70–71
 HTML parsing class, 72–73
 pasting in wordlist, 73
 primary brute-forcing class,
 71–72
 request flow, 70
 testing, 74
build_wordlist function, 73
Burp Extender API, 75–99
 creating password-guessing
 wordlist, 93–99
 converting selected HTTP
 traffic into wordlist, 95–96
 functionality to display wordlist,
 96–97
 testing, 97–99, 97f–99f
 creating web application fuzzers,
 78–87
 accessing Burp documentation,
 78–81
 implementing code to meet
 requirements, 79–82
 loading extension, 83–84,
 83f–84f
 simple fuzzer, 82–83
 using extension in attacks,
 84–87, 85f–87f
 installing, 76–77, 77f
 interfacing with Bing API to show
 all virtual hosts, 87–93
 defining extender class, 88–89
 functionality to parse results,
 90–91
 functionality to perform query,
 89–90
 testing, 91–92, 91f–93f
 Jython standalone JAR file, 76, 77f
BurpExtender class, 79–80, 88–90

C

- Cain and Abel, 74
- CANVAS, 117, 117n
- channel method, 32
- ClientConnected message, 28–29
- code injection
 - offensive forensics automation, 156–161
 - Windows privilege escalation, 147–149
- config directory, 102
- connect_to_github function, 105–106
- Content-Length header, 127
- count parameter, 48
- createMenuItem function, 88–89
- createNewInstance function, 79–80
- CreateProcess function, 139
- CredRequestHandler class, 127
- ctypes module, 39–41

D

- data directory, 102
- Debug Probe tab, WingIDE, 8
- Destination Unreachable message, 42, 43f
- dir_bruter function, 67
- DirBuster project, 65
- display_wordlist function, 96–97

E

- easy_install function, 3
- El Jefe project, 139
- encrypt_post function, 129–130
- encrypt_string function, 130
- environment setup, 1–8
 - Kali Linux, 2–3
 - default username and password, 2
 - desktop environment, 2f
 - determining version, 2
 - downloading image, 2
 - general discussion, 2
 - WingIDE, 3–8
 - accessing, 4f
 - fixing missing dependencies, 4
 - general discussion, 3–4
 - inspecting and modifying local variables, 8, 8f
 - installing, 4

- opening blank Python file, 5f
- setting breakpoints, 5
- setting script for debugging, 6, 6f
- viewing stack trace, 6, 7f

- Errors tab, Burp, 84
- exfiltrate function, 132–133
- exfiltration, 128–135
 - encryption routines, 129–130
 - key generation script, 133–134
 - login functionality, 131
 - posting functionality, 132
 - supporting functions, 129
 - testing, 134–135
- Extender tab, Burp, 83, 84f, 99f
- extract_image function, 58–59

F

- feed method, 71–72
- Fidao, Chris, 59
- FileCookieJar class, 71–72
- filter parameter, 48
- find_module function, 107
- forward SSH tunneling, 30, 30f
- Frisch, Dan, 138

G

- GDI (Windows Graphics Device Interface), 115–116
- GetAsyncKeyState function, 120
- get_file_contents function, 106
- GetForegroundWindow function, 112–113
- getGeneratorName function, 79–80
- get_http_headers function, 58–59
- GetLastInputInfo function, 119
- get_mac function, 53–54
- getNextPayload function, 81–82
- GetOwner function, 140–141
- GET requests, 62
- GetTickCount function, 119
- get_trojan_config function, 106
- GetWindowDC function, 116
- GetWindowTextA function, 112–113
- GetWindowThreadProcessId function, 112–113
- get_words function, 95–96
- github3 module, 3
- GitHub-aware trojans, 101–109
 - account setup, 102

- building, 105–108
- configuring, 104
- creating modules, 103
- hacking import functionality, 107–108
- improvements and enhancements to, 109
- testing, 108–109
- GitImporter class, 107

H

- handle_client function, 12–13
- handle_comment function, 94–95
- handle_data function, 73, 94–95
- handle_endtag function, 73
- handle_starttag function, 72–73
- HashDump object, 155
- hashdump plugin, 153
- hasMorePayloads function, 80–82
- hex dumping function, 23–24
- hivelist plugin, 153
- HookManager class, 114
- HTML form authentication, brute forcing, 69–74
 - administrator login form, 69–70
 - general settings, 70–71
 - HTML parsing class, 72–73
 - pasting in wordlist, 73
 - primary brute-forcing class, 71–72
 - request flow, 70
 - testing, 74
- HTMLParser class, 69, 72–73, 94–95
- HTTP history tab, Burp, 85, 85f

I

- IBurpExtender class, 79–80, 88–89
- ICMP message decoding routine, 42–46
 - Destination Unreachable message, 42–43, 43f
 - length calculation, 44
 - message elements, 42
 - sending UDP datagrams and interpreting results, 44–45
 - testing, 45–46
- IContextMenuFactory class, 88–89
- IContextMenuInvocation class, 88–89
- Explore.exe* process, 128
- iface parameter, 48

- IIintruderPayloadGenerator class, 78–82
- IIintruderPayloadGeneratorFactory class, 78–80
- image carving script, 55–60
 - adding facial detection code, 59
 - adding supporting functions, 58–59
 - coding processing script, 56–57
 - testing, 59–60
- imageinfo plugin, 152
- IMAP credentials, stealing, 48, 50
- Immunity Debugger, 156–157, 156n
- imp module, 107
- __init__ method, 41
- inject_code function, 148–149
- input/output control (IOCTL), 37, 37n
- input tags, 72–73
- Internet Explorer COM automation, 123–135
 - exfiltration, 128–135
 - encryption routines, 129–130
 - key generation script, 133–134
 - login functionality, 131
 - posting functionality, 132
 - supporting functions, 129
 - testing, 134–135
 - man-in-the-browser attacks, 124–128
 - creating HTTP server, 127–128
 - defined, 124
 - main loop, 125–127
 - support structure for, 124–125
 - testing, 128
 - waiting for browser functionality, 126–127
- Intruder tab, Burp, 85, 86f
- Intruder tool, Burp, 78
- IOCTL (input/output control), 37, 37n
- IP header decoding routine, 38–42
 - avoiding bit manipulation, 39–40
 - human-readable protocol, 40–41
 - testing, 41–42
 - typical IPv4 header structure, 39f

J

- Janzen, Cliff, 138
- JSON format, 104
- Jython standalone JAR file, 76, 77f

K

- Kali Linux
 - default username and password, 2
 - desktop environment, 2f
 - determining version, 2
 - downloading image, 2
 - general discussion, 2
 - installing packages, 3
- KeyDown event, 114
- keylogging, 112–115
- KeyStroke function, 114
- Khrais, Hussam, 27n
- Kuczmariski, Karol, 107n

L

- LASTINPUTINFO structure, 119
- load_module function, 107
- login_form_index function, 124–125
- login_to_tumblr function, 131
- logout_form function, 124–125
- logout_url function, 124–125

M

- mangle function, 96–97
- man-in-the-browser (MitB) attacks,
 - 124–128
 - creating HTTP server, 127–128
 - defined, 124
 - main loop, 125–127
 - support structure for, 124–125
 - testing, 128
 - waiting for browser functionality,
 - 126–127
- man-in-the-middle (MITM) attacks,
 - 51–55
 - adding supporting functions,
 - 53–54
 - coding poisoning script, 52–53
 - inspecting cache, 51
 - testing, 54–55
- Metasploit, 117
- Microsoft. *See* Bing search engine; Internet Explorer COM automation
- MitB attacks. *See* man-in-the-browser attacks
- MITM attacks. *See* man-in-the-middle attacks
- module_runner function, 108

- modules directory, 102
- mutate_payload function, 82

N

- Nathoo, Karim, 127
- netaddr module, 44, 46
- netcat-like functionality, 13–20
 - adding client code, 15–16
 - calling functions, 14–15
 - command execution functionality,
 - 17–19
 - command shell, 17–19
 - creating main function, 14–15
 - creating primary server loop, 16–17
 - creating stub function, 16–17
 - file upload functionality, 17–19
 - importing libraries, 13
 - setting global variables, 13
 - testing, 19–20
- network basics, 9–33
 - creating TCP clients, 10–11
 - creating TCP proxies, 20–25
 - hex dumping function, 23–24
 - proxy_handler function, 22–23
 - reasons for, 20
 - testing, 25
 - creating TCP servers, 12–13
 - creating UDP clients, 11
 - netcat-like functionality. *See* netcat-like functionality
 - SSH tunneling, 30–33
 - forward, 30, 30f
 - reverse, 30–33, 31f, 33f
 - testing, 33
 - SSH with Paramiko, 26–30
 - creating SSH server, 28–29
 - installing Paramiko, 26
 - key authentication, 26
 - running commands on
 - Windows client over SSH,
 - 27–29
 - testing, 29–30
- network sniffers, 35–46
 - discovering active hosts on network segments, 36
 - ICMP message decoding routine,
 - 42–46
 - Destination Unreachable message, 42–43, 43f
 - length calculation, 44
 - message elements, 42

- sending UDP datagrams and interpreting results, 44–45
 - testing, 45–46
 - IP header decoding routine, 38–42
 - avoiding bit manipulation, 39–40
 - human-readable protocol, 40–41
 - testing, 41–42
 - typical IPv4 header structure, 39f
 - promiscuous mode, 37
 - setting up raw socket sniffer, 37
 - Windows versus Linux, 36–38
 - `__new__` method, 41

O

- offensive forensics automation, 151–161
 - direct code injection, 156–161
 - installing Volatility, 152
 - profiles, 152
 - recovering password hashes, 153–155
- online resources
 - Bing API keys, 88n
 - Burp, 76
 - Cain and Abel, 74n
 - Carlos Perez, 147n
 - creating basic structure for repo, 102
 - DirBuster project, 65n
 - El Jefe project, 139
 - facial detection code, 59
 - generating Metasploit payloads, 117n
 - hacking Python `import` functionality, 107n
 - Hussam Khrais, 27n
 - Immunity Debugger, 156n
 - input/output control (IOCTL), 37n
 - Joomla administrator login form, 69
 - Jython, 76
 - Kali Linux, 2
 - MessageBox shellcode, 157
 - `netaddr` module, 46
 - OpenCV, 56n
 - Paramiko, 26
 - PortSwigger Web Security, 76
 - privilege escalation example service, 138

- py2exe, 105n
 - PyCrypto package, 128n
 - PyHook library, 112n
 - Python GitHub API library, 102n
 - Python WMI page, 139n
 - PyWin32 installer, 138
 - Scapy, 48, 48n
 - socket module, 9n
 - SVNDigger, 65n
 - VMWare Player, 1n
 - Volatility framework, 152
 - `Win32_Process` class documentation, 141n
 - Windows GDI, 116n
 - WingIDE, 4
 - Wireshark, 35
 - OpenCV, 56, 59–60
 - `os.walk` function, 64
 - owned flag, 124–125

P

- packet capture file processing. *See* PCAP processing
- `packet.show()` function, 49
- Paramiko, 26–30
 - creating SSH server, 28–29
 - installing, 26
 - running commands on Windows client over SSH, 27–29
 - SSH key authentication, 26
 - testing, 29–30
- password-guessing wordlist, 93–99
 - converting selected HTTP traffic into wordlist, 95–96
 - functionality to display wordlist, 96–97
 - testing, 97–99, 97f–99f
- Payloads tab, Burp, 85, 86f
- PCAP (packet capture file) processing, 55–60
 - adding facial detection code, 59
 - adding supporting functions, 58–59
 - ARP cache poisoning results, 53
 - coding processing script, 56–57
 - image carving script, 55–60
 - testing, 59–60
- Perez, Carlos, 147n
- `pip` package manager, 3
- POP3 credentials, stealing, 48, 50

- populate_offsets function, 154–155
- PortSwigger Web Security, 76
- Port Unreachable error, 42
- Positions tab, Burp, 85, 86f
- post_to_tumblr function, 132
- privilege escalation, 137–149
 - code injection, 147–149
 - installing example service, 138
 - installing libraries, 138
 - process monitoring, 139–141
 - testing, 141
 - with WMI, 139–141
 - token privileges, 141–143
 - automatically retrieving
 - enabled privileges, 142–143
 - outputting and logging, 143
 - winning race against code
 - execution, 144–147
 - creating file monitor, 144–146
 - testing, 146–147
- prn parameter, 48
- process monitoring, 139–141
 - testing, 141
 - with WMI, 139–141
- process_watcher function, 140–141
- profile flag, 152
- proxy_handler function, 22–23
- Proxy tab, Burp, 85, 85f
- PSList class, 158–159
- py2exe, 105
- PyCrypto package, 128n, 130
- PyHook library, 112, 120
- Python GitHub API library, 102
- PyWin32 installer, 138

Q

- Queue objects, 63–64, 66–67

R

- random_sleep function, 131
- ReadDirectoryChangesW function, 144–146
- receive_from function, 23–24
- recvfrom() function, 11
- registerIntruderPayloadGeneratorFactory
 - function, 79–80
- RegistryApi class, 154–155
- Repeater tool, Burp, 78
- Request class, 62
- request_handler function, 23–25
- request_port_forward function, 32
- reset function, 81

- response_handler function, 23–25
- restore_target function, 53–54
- reverse_forward_tunnel function, 31–32
- reverse SSH tunneling, 30–33, 31f, 33f
- run function, 103

S

- sandbox detection, 118–122
- Scapy library, 47–60
 - ARP cache poisoning, 51–55
 - adding supporting functions, 53–54
 - coding poisoning script, 52–53
 - inspecting cache, 51
 - testing, 54–55
 - installing, 48
 - PCAP processing
 - adding facial detection code, 59
 - adding supporting functions, 58–59
 - ARP cache poisoning results, 53
 - coding processing script, 56–57
 - image carving script, 55–60
 - testing, 59–60
 - stealing email credentials, 48–50
 - applying filter for common mail
 - ports, 49–50
 - creating simple sniffer, 48–49
 - testing, 50
- Scope tab, Burp, 92, 93f
- screenshots, 115–116
- SeBackupPrivilege privilege, 142t
- Secure Shell. *See* SSH
- SeDebugPrivilege privilege, 142t
- SelectObject function, 116
- SeLoadDriver privilege, 142, 142t
- sendto() function, 11
- server_loop function, 16–17
- SetWindowsHookEx function, 112
- shellcode execution, 116–118
- SimpleHTTPServer module, 117–118
- Site map tab, Burp, 97f–98f
- SMTP credentials, stealing, 48, 50
- sniff function, 48
- SOCK_DGRAM parameter, 11
- socket module, 9–10
 - building TCP proxies, 20–21
 - creating TCP clients, 10–11
 - creating TCP servers, 12–13
 - creating UDP clients, 11
 - netcat-like functionality, 13
- SOCK_STREAM parameter, 10–11

- SSH (Secure Shell)
 - with Paramiko, 26–30
 - creating SSH server, 28–29
 - installing Paramiko, 26
 - key authentication, 26
 - running commands on
 - Windows client over SSH, 27–29
 - testing, 29–30
 - tunneling, 30–33
 - forward, 30, 30f
 - reverse, 30–33, 31f, 33f
 - testing, 33
 - ssh_command function, 26–27
 - Stack Data tab, WingIDE, 6–8
 - start_monitor function, 145–146
 - store_module_result function, 106
 - store parameter, 50
 - strip function, 94–95
 - subprocess library, 17
 - SVNDigger, 65

T

- tag_results dictionary, 72–73
- TagStripper class, 94–96
- Target tab, Burp, 92, 93f, 97f–98f
- TCP clients, creating, 10–11
- TCP proxies
 - creating, 20–25
 - hex dumping function, 23–24
 - proxy_handler function, 22–23
 - reasons for building, 20
 - testing, 25
- TCPServer class, 127
- TCP servers, creating, 12–13
- test_remote function, 64
- token privileges, 141–143
 - automatically retrieving enabled
 - privileges, 142–143
 - outputting and logging, 143
- transport method, 32
- trojans
 - GitHub-aware, 101–109
 - account setup, 102
 - building, 105–108
 - configuring, 104
 - creating modules, 103
 - hacking import functionality, 107–108
 - improvements and
 - enhancements to, 109
 - testing, 108–109

- Windows tasks, 111–122
 - keylogging, 112–115
 - sandbox detection, 118–122
 - screenshots, 115–116
 - shellcode execution, 116–118

- Tumblr, 128–135

U

- UDP clients, creating, 11
- udp_sender function, 44–45
- urllib2 library, 62, 116
- urlopen function, 62

V

- VMWare Player, 1
- Volatility framework, 151–161
 - direct code injection, 157–161
 - installing, 152
 - profiles, 152
 - recovering password hashes, 153–155

W

- wait_for_browser function, 126–127
- wb flag, 17
- web application attacks, 61–74
 - brute-forcing directories and file
 - locations, 65–68
 - applying list of extensions to
 - test for, 67–68
 - creating list of extensions, 68
 - creating Queue objects out of
 - wordlist files, 66
 - setting up wordlist, 68
 - testing, 68
 - brute-forcing HTML form
 - authentication, 69–74
 - administrator login form, 69–70
 - general settings, 70–71
 - HTML parsing class, 72–73
 - pastings in wordlist, 73
 - primary brute-forcing class, 71–72
 - request flow, 70
 - testing, 74
- GET requests
 - simple, 62
 - using Request class, 62

- web application attacks, *continued*
 - mapping open source web app installations, 63–65
 - socket library, 62
- web application fuzzers, 78–87
 - accessing Burp documentation, 78–81
 - implementing code to meet requirements, 79–82
 - loading extension, 83–84, 83f–84f
 - simple fuzzer, 82–83
 - using extension in attacks, 84–87, 85f–87f
- Win32_Process class, 140–141, 141n
- win32security module, 142–143
- Windows Graphics Device Interface (GDI), 115–116
- Windows privilege escalation, 137–149
 - code injection, 147–149
 - installing example service, 138
 - installing libraries, 138
 - process monitoring, 139–141
 - testing, 141
 - with WMI, 139–141
 - token privileges, 141–143
 - automatically retrieving enabled privileges, 142–143
 - outputting and logging, 143
 - winning race against code execution, 144–147
 - creating file monitor, 144–146
 - testing, 146–147
- Windows trojan tasks, 111–122
 - keylogging, 112–115
 - sandbox detection, 118–122
 - screenshots, 115–116
 - shellcode execution, 116–118
- WingIDE
 - accessing, 4f
 - fixing missing dependencies, 4
 - general discussion, 3–4
 - inspecting and modifying local variables, 8, 8f
 - installing, 4
 - opening blank Python file, 5f
 - setting breakpoints, 5
 - setting script for debugging, 6, 6f
 - viewing stack trace, 6, 7f
- wordlist_menu function, 95–96
- Wuergler, Mark, 139