Index

Page numbers followed by an *f* or *t* indicate figures or tables, respectively.

: (colon), 139*t*
` (grave accent), 140*t*
? (question mark), 139*t*
; (semicolon), 140*t*
.asp file extension, 183
.aspx file extension, 183
.cfm file extension, 183
.NET for Web servers, 168
.nsf file extension, 183
.php file extension, 183
.pl file extension, 183
(pound sign), 139*t*
% (percent), 139*t*
& (ampersand), 139*t*
* (star character), 140*t*
+ (plus sign), 139*t*
= (equal sign), 139*t*
@ (at sign), 139*t*
| (pipe character), 140*t*
~ (tilde), 139*t*

A

Achilles, 398, 400–402, 408–413
ACL (Access Control List), 65
Active Server Pages
  ActiveX, 15*t*, 32–33
  background, 27–28
  coding examples, 28–29
  database connectivity, 29–32
  global file, 31
  query example for SQL, 178
  security implications, 30, 33
  URL file extensions, 183
  URL signature identification, 187
ActiveX control, 15*t*, 32–33
Allaire, 25, 45
ampersand (&), 139*t*
Apache
  background, 54–55
  CGI and, 59
  handlers, 59–60
  interfacing with PHP, 171–172
  interfacing with ServletExec, 172
  server side includes, 59
  URL signature identification, 187
  virtual hosts
    IP-based mechanism, 57–58
    name-based mechanism, 55–57
API (application programming interface), 167, 169, 177–178
Applets, Java
  breakdown, 41*t*
  coding examples, 39–40
  security implications, 40
applets and objects in a Web resource, 238
Application Error, 375
Application Servers, Java, 45
application state diagrams, 356–358
application systems, Web. *See* Web application systems
ASP+, 183
ATG Dynamo Application Server, 184
Atkinson, Bill, 13
at sign (@), 139*t*

B

back-end application server, 171
Basic authentication, 253
BEA. *See* WebLogic
Berners-Lee, Tim, 13, 119
BlackIce Defender, 429
Black Widow, 214, 223–224

blind stress testing, 377–382
browser source code. *See* HTML source code
brute forcing
  of authentication, 253–257, 392–394, 398
  tools, 267–269
Brutus, 267–268, 394–398, 399*f*
buffer overflows
  assembly language
    assembler instructions, 373, 374*t*
    described, 370–371
    general purpose registers, 371–372
    pointer registers, 372–373
    the stack, 373
  bytes tracking, 373–375
  countermeasures, 382
  defined, 370
  vulnerabilities examples
    blind stress testing, 377–382
    Code Red, 143
    disassembly, 376–377
bytecodes, 40

C

Carello shopping cart, 103–104
Cart32 shopping cart, 105
CERT (Carnegie-Mellon Emergency Response Team), 384
CFML (Cold Fusion Markup Language), 25–27
CGI (Common Gateway Interface), 19
  background, 33–34
  coding examples, 34
  environmental variables, 35, 36–37*t*
  historic vulnerabilities, 320
  query strings specifications, 136–138
  ScriptAlias, 59
  server-side includes, 35, 37, 38*t*
CHECKBOX, 152*t*
checkout process, shopping cart systems, 105–106. *See also* electronic shopping
client-based Java
  Applets
    breakdown, 41*t*
    coding examples, 39–40
    security implications, 40
  JavaScript, 41–44, 209
  Jscript, 44–45
  security implications, 42–44
client-side scripts
  in HTML source code, 208–209
  resource analysis in site linkages, 236
  validation bypass capability, 290, 292–296, 303–305
  in a Web resource, 238
Code Red worm, 143
  extent of spread, 421
  history of first attack, 418–421
  ISAPI filter and, 61
  new system-level variation, 421–423
ColdFusion
  Application Server, 25
  background, 25
  coding examples, 26–27
  Markup Language, 25–27
  security implications, 25
  Studio, 27
  URL file extensions, 183
colon (:), 139*t*
COM (Component Object Model), 26, 32
Common Gateway Interface. *See* CGI
CONNECT command in HTTP, 126*t*
Cookie Pal, 348, 402–413
cookies
  associated application servers, 188
  control of, 362, 363
  Cookie Pal for Web hacking, 402–413
  replacement, 350–351

in a Web resource, 238
crawlers, web
grep, 213–214
GUI-driven, 214–215
wget, 210–213, 222–223
crawling a Web site
described, 221
HTTP response header information, 221–222
redirection of crawlers, 227–228
tools
Black Widow, 223–224
Funnel Web Profiler, 224–226
grep, 213–214
GUI-driven, 214–215
Sam Spade, 226, 228
Teleport, 226, 228
wget, 210–213, 222–223
credit card fraud
countermeasures, 107–108
hacking countermeasures, 317–318
hacking example
ID parameter alteration, 311–316
input validation, 311
password search, 316–317
remote command execution capability, 298–301
vulnerability basis, 297–298, 305–306
cyber graffiti
brute-forcing tools for HTTP, 267–269
countermeasures
directory browsing disable, 272–273
password strengthening, 272
reverse proxy disable, 270–272
defacement case study
background of target site, 246–247
brute forcing of authentication, 253–257
directory browsing, 258–262
e-mail header, 248
reverse proxy technique, 252–253
reverse-server tactic, 250–251
security measures, 249, 253, 261
security vulnerabilities, 266–267
target network mapping, 249–253
upload of defaced pages, 262–265
cyber shoplifting. *See* e-shoplifting

D

database access
hacking countermeasures, 317–318
hacking example
ID parameter alteration, 311–316
input validation, 311
password search, 316–317
remote hacking, 308–310
database connectivity
with ASP, 29–32
with JSP, 46–47
databases and Web application systems
connecting with, 175, 177
JDBC use, 179–180
native APIs use, 177–178
ODBC use, 179
purpose of the database, 169
query examples, 178–179
servers identification, 190–192
database servers. *See* Oracle; SQL Server
Datarescue, 376
Data Source Names (DSN), 29
DCShop shopping cart, 104
defacement case study
background of target site, 246–247
brute forcing of authentication, 253–257
directory browsing, 258–262
e-mail header, 248
reverse proxy technique, 252–253
reverse-server tactic, 250–251

security measures, 249, 253, 261
security vulnerabilities, 266–267
target network mapping, 249–253
upload of defaced pages, 262–265
default stored procedures in SQL, 72, 73–75*t*, 75
DELETE command in HTTP, 126*t*
DHTML (Dynamic HTML), 16
Digest authentication, 254
disassembly of a binary executable program, 376
Discover Financial Services, 108
Domino server, 173–175, 183
Double-decode vulnerability, 146–147
"Dr. Watson error," 375
DSN (Data Source Names), 29
DSOs (Dynamic Shared Objects), 170
DTDs (Document Type Definitions), 16

E

electronic shopping
customer interface challenges, 92–93
database access (*see* database access)
elements, 277–279, 280*f*
increasing use of automation, 94–96, 97*f*, 279–280
individuals' ability to accept electronic payment, 115–116
payment system implementation issues, 114–115
process of, 96, 98*f*
purchase and delivery process, 93–94
security vulnerabilities (*see* e-shoplifting)
shopping cart systems
ability to change selections, 99
application implementation (*see* shopping cart applications)
checkout process, 105–106
content selection by customers, 98–99
credit card fraud protection, 106, 107–108
order confirmation, 106, 109
payment gateway interface, 109–110
payment gateway interface example, 110–113
payment processing, 105–106, 108*f*, 109*f*
poorly integrated carts examples, 103–105
purchase processing, 99–100
scope and lifetime per customer, 97–98
total cost tracking, 99, 101*f*, 102*f*
transaction database interface, 110
verification need, 106
e-mail addresses and usernames
in HTML source code, 206–207
impersonation hack (*see* session hijacking)
e-mail harvesting, 206–207
Entercept, 382
equal sign (=), 139*t*
error forcing
of databases, 196*f*
examples for servers, 188–189, 190*f*, 193*f*
error messages from HTTP response codes, 122*f*
e-shoplifting. *See also* electronic shopping
credit card fraud case study
remote command execution capability, 298–301
vulnerability basis, 297–298
price change case studies
background of target site, 242–243, 281–283

client-side validation bypass, 290, 292–296, 303–305
hidden fields vulnerabilities, 283–290, 301–303
vulnerabilities summary, 301–306
security vulnerabilities, 115
Explorer HTML elements viewing instructions, 199–200
eXtensible Markup Language (XML), 16–17

F

findstr, 213
firewalls, 118
front-end Web server, 165–167
Funnel Web Profiler, 224–226

G

general purpose registers, 371–372
GET
HTML forms parameter passing, 151, 153–157
HTTP, 121*t*, 126*t*
security implications, 14*t*
Google, 291
Gosling, James, 38
grave accent (`), 140*t*
grep, 213–214
Gutmans, Andi, 23

H

hacker psychology, 135, 322–325
hacking examples. *See* credit card fraud; defacement case study; identity stealing case; Java, server attack examples; price change case studies; worms
hacking tools. *See* Web hacking with automated tools
handler forcing, 328, 335
Hassan Consulting shopping cart, 104
HEAD command in HTTP, 121*t*, 126*t*
header definitions, HTTP, 124*t*, 128*t*
hexadecimal ASCII encoding, 141–143, 440–441
HIDDEN command in HTML, 152*t*
hidden fields
in HTML source code, 208
revealed in search engines, 291, 292*f*
session hijacking and, 363
vulnerabilities in electronic stores, 283–290, 301–303
hijacking. *See* session hijacking
HTML
attributes and security implications, 14–15*t*
background, 13–14
forms
browser-side handling of, 148
components and concepts, 149–150
input elements, 151, 152*f*
parameter passing, 151, 153–157
security vulnerabilities, 148
in a Web resource, 238
site linkage analysis and, 218–219
HTML source code
automated source sifting techniques
grep, 213–214
GUI-driven, 214–215
wget, 210–213, 222–223
browser display, 197–198
comments analysis
client-side scripts, 208–209

cross-references to files, 202
developer/author details, 202
e-mail addresses and usernames, 206–207
hidden fields, 208
insertions from servers, 204
keywords and meta tags, 207
old code, 204–205
purpose of comments, 200–201
reminders/placeholders, 203–204
revision history information, 202
internal and external hyperlinks, 205–206
viewing instructions
Explorer, 199–200
Netscape Navigator, 198–199
HTTP
brute-forcing tools, 267–269
dynamic content evolution, 320–321
hack example
hacker's activities, 2–10
security vulnerabilities, 5, 6
session tracking, 358–360
stateless vs. stateful applications, 360–361
version 1.0
background, 119–120
header field definitions, 124*t*
request steps, 120*f*, 121*t*
response steps, 121, 122*t*, 123
version 1.1
background, 123, 125
header field definitions, 128*t*
request/response steps, 125, 126–127*t*
HTTPS, 128–130
HyperCard, 13
hyperlinks in HTML source code, 205–206
hypertext, 218–219
HyperText Markup Language. *See* HTML
Hypertext transfer protocol. *See* HTTP

I

IBM, 45, 181
identity stealing case
application analysis, 348–349
cookie replacement, 350–351
situation, 344–347
IDS (intrusion detection systems)
accuracy in reporting, 430
countermeasures
SSL decryption, 446
URL decoding, 447
evasion techniques, 445
false positive generation, 444–445
hacking routes, 430–431
host-based, 429
for networks, 429
polymorphic URLs
adding fake paths, 442
described, 439–440
hexadecimal encoding, 440–441
illegal unicode/superfluous encoding, 441
inserting slash-dot-slash strings, 442
mixing techniques, 443
multiple slashes, 443
nonstandard path separators, 443
purpose and description, 428–429
SSL and
attacks using, 431–433
intrusion detection using, 435
sniffing traffic, 435–439
tunneling attacks using, 434
IIS Web server
Code Red attack
extent of spread, 421

history of first attack, 418–421
new system-level variation, 421–423
interfacing with Netscape Enterprise Server, 173–175
interfacing with ServletExec, 172–173
security components and liabilities
ISAPI applications, 60–61, 62*t*, 63
sample files, 64–65, 66*f*
virtual directories, 63–64
virtual hosts, 65, 67–70
SSI and, 37
unicode vulnerability, 144–146, 176–177
impersonation hack. *See* identity stealing case
Interactive Disassembler Professional, 376
Internet Information Server. *See* IIS Web server
Internet Server Application Programming Interface. *See* ISAPI applications
Internet Services Manager, ISAPI, 61
Internet shopping. *See* electronic shopping
IP-based virtual host, 57–58
ISAPI applications
back and security vulnerabilities, 60–61, 63
extensions for Web servers, 170
filters, 62*t*
interfacing with ServletExec, 172–173

J

J2EE for Web servers, 168
JAD (Java Decompiler), 40
Java
background, 38–39
basis of technology, 321–322
client-based (*see* client-based Java)
server architecture, 322, 323*f*, 324*f*
server attack examples
countermeasures, 322–325, 339–341
FileServlet invocation, 329–331
hacker psychology, 322–325
handler forcing, 328, 335
header response, 326
JSPServlet force, 332–337, 338*f*
security vulnerabilities, 331, 335
servlets and handlers, 327–328
SSIServlet invocation, 331–332
target site background, 325–326
server-based (*see* server-based Java)
for Web servers, 168
Java Database Connectivity (JDBC), 179–180
Java Decompiler (JAD), 40
JavaScript, 41–44, 209
JavaSoft, 49
Java Web Server, 45
security weaknesses, 48–49, 50
URL signature identification, 184
JDBC (Java Database Connectivity), 179–180
JDeveloper, 45
JHTML
background, 49
security implications, 50
JRun, 45
Jscript, 44–45
JSP
arbitrary command execution, 48–49
background, 45
database connectivity, 46–47
security implications, 47, 48
source code disclosure, 47–48
JSPServlet force, 332–337, 338*f*

K

Kouznetsov, Pavel, 40

L

languages of the Web. *See also specific languages*
  Active Server Pages
    ActiveX, 32–33
    background, 27–28
    coding examples, 28–29
    database connectivity, 29–32
    global file, 31
    security implications, 30, 33
  CGI
    background, 33–34
    coding examples, 34
    environmental variables, 35, 36–37*t*
    server-side includes, 35, 37, 38*t*
  ColdFusion
    Application Server, 25
    background, 25
    coding examples, 26–27
    Markup Language, 25–27
    security implications, 25
    Studio, 27
  DHTML, 16
  HTML
    attributes and security implications, 14–15*t*
    background, 13–14
  Java
    background, 38–39
    client-based (*see* client-based Java)
    server-based (*see* server-based Java)
  Perl
    background, 18
    in a CGI code, 34
    coding examples, 18–20
    security implications, 20–22
  PHP
    background, 22–23
    coding examples, 23–24
    security implications, 24–25
  XHTML, 17–18
  XML, 16–17
Lerdorf, Rasmus, 22
Linux, 362
Listener service, Oracle
  security vulnerabilities, 89–90
  status request, 82–89
Lotus Domino, 173–175, 183

M

majordomo, 18
meta-characters, 138, 140, 140*t*
Microsoft
  Explorer HTML elements viewing instructions, 199–200
  IIS Web server (*see* IIS Web server)
  SQL Server (*see* SQL Server)
Miva Merchant, 115
Morris worm, 418

N

name-based virtual host, 55–57
n-commerce (IBM), 181
Net.Data (IBM), 181, 184

Netcat, 182, 388–390
.NET for Web servers, 168
Netscape Enterprise Server
  interfacing with IIS and Domino, 173–175
  Unicode attack example, 176–177
  URL signature identification, 184
Netscape Navigator HTML elements
  viewing instructions, 198–199
network retrievers
  grep, 213–214
  wget, 210–212, 222–223
NewAtlanta, Inc., 170
Nimda worm, 61, 423–426
NSAPI modules, 170
NTFS file permissions, 65
NTLM authentication, 254
NTOMax, 377

O

objects and applets in a Web resource, 238
ODBC (Open Database Connectivity), 179
one-time use credit card, 107–108
"OnMouseOver," 16
Opera, 362
OPTIONS command in HTTP, 126–127*t*
Oracle, 45
  claim of "unbreakable," 80
  query example from PHP, 178–179
  security vulnerabilities
    Listener service, 82–89
    passwords, 81
    privileges, 82
    system tables, 80–81

P

packet sniffer in IDS, 430–431
PASSWORD command in HTML, 152*t*
passwords
  cracking with Brutus, 394–398, 399*f*
  Oracle security and, 81
  recommended policies, 81
  SQL Server security and, 79
PayFlow Pro, 110–113, 115
payment systems. *See also* shopping cart systems
  gateway interface, 109–110
  gateway interface example, 110–113
  implementation issues, 114–115
  individuals' ability to accept electronic payment, 115–116
  payment processing, 105–106, 108*f*, 109*f*
PayPal, 115–116
percent (%), 139*t*
Perl
  background, 18
  coding examples, 18–20
  security implications, 20–22
  URL file extensions, 183
"Personal Home Page". *See* PHP
PHP
  background, 22–23
  coding examples, 23–24
  interfacing with Apache, 171–172
  query example for Oracle, 178–179
  security implications, 24–25
pipe character (|), 140*t*
plug-in application engines, 238
plug-in frameworks, 167, 169
plus sign (+), 139*t*
pointer registers, 372–373
poisoning, SQL, 70–71

POP (Post Office Protocol), 26
port 443 (SSL), 157
port 80 (HTTP), 157
POST
  HTML forms parameter passing, 151, 153–157
  HTTP, 121*t*, 127*t*
  security implications, 14*t*
pound sign (#), 139*t*
Practical Extraction and Report Language. *See* Perl
price change case studies
  background of target site, 242–243, 281–283
  client-side validation bypass, 290, 292–296, 303–305
  hidden fields vulnerabilities, 283–290, 301–303
  vulnerabilities summary, 301–306
product catalog in electronic shopping, 100–101
protocols of the Web. *See* HTTP; HTTPS
Psionic PortSentry, 429
PUT command in HTTP, 127*t*

Q

Query String
  codes, 139–140*t*
  parameter passing, 136–138
question mark (?), 139*t*

R

RADIO command in HTML, 152*t*
Raggett, Dave, 14
RealSecure, 429
remote command execution capability. *See* Java

S

Sam Spade, 214, 226, 228
ScriptAlias, CGI, 59
scripting languages for Web application servers, 168
search engines and hidden fields, 291, 292*f*
Secure Electronic Transaction (SET), 107
SecureNet, 429
Secure Sockets Layer. *See* SSL
SELECT/OPTION command in HTML, 152*t*
semicolon (;), 140*t*
server-based Java
  JHTML
    background, 49
    security implications, 50
  JSP
    arbitrary command execution, 48–49
    background, 45
    database connectivity, 46–47
    security implications, 47, 48
    source code disclosure, 47–48
servers, Web. *See* Web servers
server-side includes. *See* SSI
server-side scripts
  processing of HTML forms, 148
  security vulnerabilities, 265–267
  in a Web resource, 238
Servlet, 322
ServletExec
  extensions for Web servers, 170
  interfacing with Apache DSO, 172
  interfacing with IIS, 172–173

session hijacking
attack analysis
application state diagrams, 356–358
session tracking over HTTP, 358–360
stateless vs. stateful applications, 360–361
cookies
associated application servers, 188
control of, 362, 363
Cookie Pal for Web hacking, 402–413
replacement, 350–351
in a Web resource, 238
described, 354–356
hidden fields, 363
identity stealing case
application analysis, 348–349
cookie replacement, 350–351
situation, 344–347
session and state tracking guidelines, 363–365
SET (Secure Electronic Transaction), 107
SGML (Standard Generalized Markup Language), 13
shareware, 116
shopping cart applications
database interfacing, 102
overview, 100, 103*f*
payment gateway integration, 102–103
product catalog, 100–101
security vulnerabilities, 101, 102
session management, 101–102
shopping cart systems
ability to change selections, 99
content selection by customers, 98–99
credit card fraud case study, 305–306
credit card fraud protection, 106, 107–108
implementation of an application (*see* shopping cart applications)
order confirmation, 106, 109
payment gateway interface, 109–110
payment gateway interface example, 110–113
payment processing, 105–106, 108*f*, 109*f*
poorly integrated carts examples, 103–105
purchase processing, 99–100
scope and lifetime per customer, 97–98
total cost tracking, 99, 101*f*, 102*f*
transaction database interface, 110
verification need, 106
shopping on the Internet. *See* electronic shopping
Simple Mail Transfer Protocol (SMTP), 26
Single-Use Credit Card, 108
site linkage analysis
crawling a Web site
described, 221
HTTP response header information, 221–222
redirection of crawlers, 227–228
tools, 222–228
group analysis, 231–232
group creation, 228–231
HTML and, 218–219
methodology overview, 219–220
resource analysis
applet and object identification, 235–236
client-side scripts, 236
comments and e-mail addresses, 236–237
extensions, 233
form determination, 235
matrix information, 237
session, 234–235
URL path, 233
resource inventory, 238–239
Snort, 429, 432, 433, 439, 441

source code disclosure. *See* HTML source code
source sifting, automatic
grep, 213–214
GUI-driven, 214–215
wget, 210–213, 222–223
SQL Server
commands, 71, 72*f*
database access (*see* database access)
poisoning, 70–71
query example from ASPs, 178
security vulnerabilities
default databases, 76
default stored procedures, 72, 73–75*t*, 75
default system and meta-data functions, 77–78
default system tables, 76–77
information schema views, 78–79
passwords, 79
SSI (server-side includes)
CGI and, 35, 37, 38*t*
security implications, 59
for shopping cart systems, 99
SSL (secure sockets layer)
credit card fraud and, 107
in HTTPS, 128–130
IDSs and
attacks using, 431–433
intrusion detection using, 435
sniffing traffic, 435–439
tunneling attacks using, 384, 434
implementation issues, 114
security issues, 130
stack in buffer overflows, 373
Standard Generalized Markup Language (SGML), 13
star character (*), 140*t*
static content in a Web resource, 238
StoryServer (Vignette), 181, 184
Studio, 27
SUBMIT command in HTML, 152*t*
Sun Microsystem, 45, 49, 50
Superfluous decode, 146–147
Suraski, Zeev, 23

T

technologies, Web. *See* Web application systems
Teleport, 214, 226, 228
Teleport Pro, 413–414
TEXTAREA command in HTML, 152*t*
TEXT command in HTML, 152*t*
tilde (~), 139*t*
tools for Web hacking. *See* Web hacking with automated tools
TRACE command in HTTP, 127*t*
Transact-SQL, 77
tunneling, SSL, 384, 434

U

UCS (Universal Character Set), 142
Unicode bug, 144–146
Unicode Consortium, 142, 143
Unicode encoding
attack example, 176–177
described, 141–143
in polymorphic URLs, 441
Uniform Resource Identifier (URI). *See* URL
Uniform Resource Locator. *See* URL
Universal Character Set (UCS), 142
UNIX, 57–58
URL
attack example, 142, 143
double-decode vulnerability, 146–147

encoding
abusing of, 143–144
meta-characters, 138, 140, 140*t*
special characters, 138, 139*t*
specifying special characters, 139, 139*t*, 141
file extensions, 183
hacker's approach to corrupting, 135
hack examples, 2–10, 142, 160–162
HTML forms
browser-side handling of, 148
components and concepts, 149–150
input elements, 151, 152*f*
parameter passing, 151, 153–157
security vulnerabilities, 148
in a Web resource, 238
identifying Web application systems components
examples, 184–188
technology identification, advanced, 188–189, 190*f*
technology identification basics, 182–183
parameter passing, 136–138
polymorphic
adding fake paths, 442
described, 439–440
hexadecimal encoding, 440–441
illegal unicode/superfluous encoding, 441
inserting slash-dot-slash strings, 442
mixing techniques, 443
multiple slashes, 443
nonstandard path separators, 443
structure and components, 133, 134*t*
types, 133, 134*t*, 135
unicode encoding, 141–143
unicode vulnerability, 144–146
UTF-8 encoding, 142. *See* Unicode encoding

V

VBScript, 209
VeriSign, 110
Vignette, 181, 184
virtual hosts
IIS Web server, 65
multiple websites, 68, 70
secondary IP addresses, 67–68, 69*f*
security vulnerabilities, 67
IP-based mechanism, 57–58
name-based mechanism, 55–57
Visual Basic, 27, 28

W

Wall, Larry, 18
Web application systems
components, 166*f*
database servers, 169
execution environment, 168–169
front-end Web server, 165–167
components identification from URLs
examples, 184–188
technology identification, advanced, 188–189, 190*f*
technology identification basics, 182–183
Unicode hack example, 176–177
countermeasures to hacking, 192–193
databases and
connecting with, 175, 177
JDBC use, 179–180
native APIs use, 177–178
ODBC use, 179
query examples, 178–179
database servers identification, 190–192

file extensions used in URLs, 183
interfacing schemes, 169–171
interfacing schemes examples, 171–175
internal proxying, 171
specialized servers, 180–181
URL mapping, 171
WebCracker, 268–269
Web hacking with automated tools
Achilles, 398, 400–402, 408–413
Brutus, 394–398, 399*f*
Cookie Pal, 402–413
Netcat, 388–390
security recommendations, 414–415
Teleport Pro, 413–414
Whisker, 390–391, 392*f*
Whisker and brute force, 392–394
Web languages. *See* languages of the Web
WebLogic, 45
hack example
countermeasures, 339–341
FileServlet invocation, 329–331
handler forcing, 328, 335
header response, 326
JSPServlet force, 332–337, 338*f*
security vulnerabilities, 331, 335
servlets and handlers, 327–328
SSIServlet invocation, 331–332
target site background, 325–326
security weaknesses, 47–48, 50
URL signature identification, 184, 187
Web servers. *See* Apache; IIS Web server; Web application systems
Web sites
Achilles, 388
Allaire, 25
Apache, 23
BlackIce Defender, 429
Black Widow, 214
Brutus, 267
Carello shopping cart, 103
Cart32 shopping cart, 105
CERT, 384
Cookie Pal, 348
DCShop shopping cart, 104
defaced sites index, 273
defacement stories, 267
Entercept, 382
fragrouter, 431
front-end Web servers, 167
Funnel Web Profiler, 224
GNU, 214
Hassan Consulting shopping cart, 104
HTML elements, 197
HTML specification and validation, 14
HTTP specification, 119
HTTP user agents, 35
Internet RFCs, 133
Java interpreter, 40
JSP syntax, 45
Netcat, 388
network IDSs, 429
NTLM brute forcing, 398
NTOMax, 377
one-time use credit card, 108
Oracle security, 81
PayFlow Pro exploitations, 115
PayPal, 115–116
Perl script security, 22
PHP, 25
port-binding win32 shellcode, 380
Psionic PortSentry, 429
RFCs, 133
SoftByteLabs, 223
SQL Server security, 79
SSL security, 130
Sun Microsystem, 49
Teleport, 214, 388
Tidy program, 14
tnscmd.pl script, 83
Unicode, 146
Unicode Consortium, 142

W3C, 13
WebCracker, 268
wget, 214
Whisker, 388
XHTML, 18
WebSphere, 45
wget, 210–213, *222–223*
Whisker, 193, 390–391, *392f*
Whisker and brute force, 392–394
World Wide Web Consortium (W3C), 13
worms
Code Red
extent of spread, 421
history of first attack, 418–421
new system-level variation, 421–423
defined, 418
example, 384–385
Nimda worm, 423–426
threat from, 425–426

X

XHTML, 17–18
XML (eXtensible Markup Language), 16–17