Index

- 302 Found, 68
- 400 Bad Request, 69
- 403 Forbidden, 69
- 401 Unauthorized, 69
- 500 Internal Server Error, 69

A

- access controls, 18–19, 217
  - attacking, 224–228
  - broken, 6
  - horizontal, 218
  - insecure methods, 223–224
  - securing, 228–234
  - testing
    - insecure access control methods, 698
    - limited access, 697–698
    - multiple accounts, 697
    - requirements, 696–697
  - vertical, 218
  - vulnerabilities, 218–219
    - identifier-based functions, 220–221
    - insecure access control methods, 223–224
    - multistage functions, 222
    - static files, 222–223
    - unprotected functionality, 219–220
- Accipiter DirectServer, 568
- ActiveX controls, 119–120
  - attacking, 454–455
  - exported functions, 122
  - inputs processed by controls, 123–124
  - managed code, decompiling, 123–124
  - reverse engineering, 120–122
  - vulnerabilities
    - finding, 455–456
    - preventing, 456–457
- administrators, alerting, 30–31
- AJAX (Asynchronous JavaScript and XML), 54, 389–390
  - leveraging, 461–463
    - asynchronous off-site requests, 463–464
- alerting administrators, 30–31
- Alibaba, 568
- Allaire JRun directory listing vulnerability, 569
- analyzing applications
  - identifying data entry points, 673
  - identifying functionality, 673
  - identifying the technologies used, 673–674
  - mapping attack surface, 674
- anti-DNS pinning, 464–466
  - DNS pinning, 466
    - attacks against, 466–467
- Apache, chunked encoding overflow, 567
- application pages, functional paths and, 76–78
- application service providers. *See* ASPs
- applications
  - analyzing
    - identifying data entry points, 673
    - identifying functionality, 673
    - identifying the technologies used, 673–674
    - mapping the attack surface, 674
  - mapping content
    - default content and, 671
    - hidden content and, 670–671
    - identifier-specified functions, 671–672
    - public resources and, 670
    - test for debug parameters, 672
    - visible content, 669–670
- arbitrary redirection, 583–584
  - testing for, 706
- architecture, tiered, 535–536
  - applying defense in depth, 542
  - attacking tiers, 539–540
  - exploiting trust relationships between tiers, 537–538
  - minimizing trust relationships, 540–541
  - segregating different components, 541–542
  - subverting tiers, 538–539
- archives, hidden content and, 73
- ASP, code injection and, 302–303
- ASP.NET, 50
  - APIs, dangerous, 596–600
  - environment, configuring, 600–601
  - session interaction, 595–596
  - user-supplied data, identifying, 594–595
  - ViewState, 102–106
- ASPs (application service providers), 542–543
  - shared application services, 543–544
- attack surfaces, 91–92
- attackers
  - alerting administrators, 30–31
  - audit logs and, 29–30
  - error handling, 27–29
  - reacting to attacks, 31–32
- audit logs, 29–30

COPYRIGHTED MATERIAL

- authentication, 16–17
  - ACEGI, 49
  - broken, 6
  - design flaws
    - brute-forcible login, 136–138
    - incomplete credential validation, 152
    - insecure distribution of credentials, 155
    - passwords, 135–136
    - passwords, change functionality, 144–145
    - passwords, forgotten functionality, 145–147
    - passwords, predictable initial, 154–155
    - remember me functionality, 148–149
    - user impersonation functionality, 149–151
    - usernames, non-unique, 152–153
    - usernames, predictable, 154
    - verbose failure messages, 139–141
    - vulnerable transmission of credentials, 142–143
  - HTTP, 47, 178–179
  - implementation flaws
    - fail-open login mechanism, 156–157
    - insecure storage of credentials, 161
    - multistage login mechanisms, defects, 157–161
  - JAAS, 49
  - securing
    - account recovery function, 170–171
    - brute-force attack prevention, 167–169
    - credentials, handling secretively, 163–164
    - credentials, strong, 162–163
    - credentials, validating, 164–166
    - information leakage prevention, 166–167
    - log, 172
    - monitor, 172
    - notify, 172
    - password change function, 170
  - smartcards and, 176
  - technologies, 134–135
- authentication mechanism, testing
  - account recovery function, 682
  - check for unsafe distribution of credentials, 685
  - check for unsafe transmission of credentials, 684–685
  - exploit any vulnerabilities to gain unauthorized access, 687–688
  - impersonation function, 683
  - logic flaws, 685–686
  - multistage mechanisms, 686–687
  - password quality, 680
  - predictability of auto-generated credentials, 684
  - remember me function, 682–683
  - resilience to password guessing, 681
  - understanding mechanism, 680
  - username enumeration, 680–681
  - username uniqueness, 683–684
- autocomplete, 460

B

- backdoor passwords, 584–585
- banner grabbing, 82
- Base64 encoding, 58
- Basic authentication, HTTP, 47
- bespoke automation
  - attack scripting, 476–477
  - Burp Intruder, 491–501
  - enumerating identifiers
    - approach, 474
    - hit detection, 474–476
    - HTTP status code, 474
    - location header, 475
    - response body, 475
    - response length, 475
    - set-cookie header, 475
    - time delays, 476
  - fuzzing and, 487–491
  - harvesting data, 484–487
  - JAttack, 477–483
  - uses for, 472–473
- black-box testing, 578–579
- blocked characters, bypassing filters and, 267
- blocked strings, bypassing filters and, 268
- boundary validation, 23–25
- browsing history, 459
- brute-forcible login, 136–138
- buffer overflows
  - detecting vulnerabilities, 527–528
  - heap overflows, 523–524
  - off-by-one vulnerabilities, 524–527
  - stack overflows, 522–523
  - vulnerabilities, 585–586
  - web server vulnerabilities, 566–567
- Burp Intruder, 69, 491–492
  - application fuzzing, 500–501
  - enumerating identifiers, 495–498
  - harvesting information, 498–500
  - payloads
    - choosing, 493–494
    - positioning, 492–493
  - response analysis, 494–495
- Burp Proxy, 97, 105
- Burp Spider, 62
- Burp Suite, 643–644
- bypassing filters
  - blocked characters, 267
  - blocked strings, 268
  - circumventing validation, 267
  - defective filters, 269–270
  - dynamic execution, 268–269
  - SQL comments, 268
- bypassing login, injecting into SQL, 243–244

C

- cached web content, 458–459
- canonicalization, 26–27
  - vulnerabilities, web server, 568–571
- capturing user data
  - HTML forms
    - disabled elements, 110–111
    - length limits, 106–108
    - script-based validation, 108–110
  - thick-client components, 111–112
    - ActiveX controls, 119–124
    - Java applets, 112–119
    - Shockwave Flash Objects, 124–128
- circumventing validation, bypassing filters and, 267
- Cisco ACS Acme.server, 568
- client, transmitting data via, 95–96
  - ASP.NET ViewState, 102–106
  - form fields, hidden, 96–98
  - HTTP cookies, 99
  - opaque data, 101–102
  - Referer header, 100–101
  - URL parameters, 99–100
- client-side attack, escalating
  - attack other network hosts, 398
  - capturing clipboard contents, 396
  - enumerating currently used applications, 397
  - exploit browser vulnerabilities, 399
  - logging keystrokes, 396
  - port scanning local network, 397–398
  - stealing history and search queries, 396
- client-side controls, testing
  - client-side controls over user input, 676
  - thick-client components, 677–679
  - transmission of data via client, 675–676
- client-side data
  - alerting, 131
  - logging, 131
  - transmitting data via client, 128–129
  - validating client-generated data, 129–130
- client-side information leakage, 517

- clipboard, capturing contents, 396
- code, tools for browsing, 619–620
- code injection, 237
  - bypassing filters
    - blocked characters, 267
    - blocked strings, 268
    - circumventing validation, 267
    - defective filters, 269–270
    - dynamic execution, 268–269
    - SQL comments, 268
  - compiled languages, 238
  - fingerprinting database, 255–256
  - inference, 277–278
    - Absinthe, 278–282
    - conditional errors, 282–283
    - time delays, 283–285
  - interpreted languages, 238–239
  - LDAP, 326–327
    - flaws, 329–330
    - modifying search filter, 328–329
    - preventing, 330
    - query attributes, 327–328
  - ODBC error messages, 262–266
  - OS command
    - ASP and, 302–303
    - injection flaws, 304–307
    - Perl and, 300–302
    - preventing, 307
  - out-of-band channels, 274–275
    - MS-SQL, 275
    - MySQL, 276–277
    - Oracle, 275–276
  - retrieving data as numbers, 273–274
  - second-order SQL injection, 271–272
  - SMTP, 321–322
    - command injection, 323–324
    - email header manipulation, 322–323
    - flaws, 324–325
    - preventing, 325–326
  - SOAP, 313–316
  - SQL, 240–241
    - bugs, 244–247
    - bypassing login, 243–244
    - DELETE statements, 250
    - exploiting basic vulnerability, 241–243
    - INSERT statements, 248–249
    - preventing, 296–300
    - SELECT statements, 248
    - UNION operator, 250–255
    - UPDATE statements, 249–250
  - web scripting languages
    - dynamic execution vulnerabilities, 307–310
    - file inclusion vulnerabilities, 310–312
    - script injection vulnerabilities, preventing, 312
  - XPath, 316–317
    - blind, 319–320
    - flaws, 320–321
    - preventing, 321
    - subverting application logic, 317–318
- code review
  - black-box testing, 578–579
  - methodology, 579–580
  - white-box testing, 578–579
- command injection, SMTP, 323–324
- comments, source code, 586–587
- common functions of web applications, 3–4
- COMRaider, 122
- configuration
  - vulnerable
    - default content, 555–558
    - default credentials, 554–555
  - web server, securing, 565–566
- content, default, 555–558
- Cookie header, HTTP request, 37
- cookies
  - domain restrictions, 203–205
  - HTTP, 43–44, 99
  - path restrictions, 205–206
  - persistent, 458
  - scope, 695–696
  - session tokens and, 178
- credentials
  - default, 554–555
  - incomplete validation, 152
  - insecure distribution, 155
  - insecure storage, 161
  - vulnerable transmission, 142–143
- cross-site scripting, 580–581. *See* XSS (cross-site scripting)
- custom, scripts, Stunnel, 663–664

D

- data handling, safe, 22–23
- data sanitization, 22
- database error messages, 510
- Database object relational mapping, Hibernate, 49
- databases
  - attacking
    - MS-SQL and, 286–287
    - MySQL and, 288–289
    - Oracle and, 288
  - code components
    - calls to dangerous functions, 618–619
    - SQL injection, 617–618
  - data extraction
    - MS-SQL, 260–262
    - Oracle, 257–260
  - fingerprinting, code injection and, 255–256
- debug messages, 508–509
- debugging
  - breakpoint, 121
  - functionality, 555–556
  - OllyDbg, 120
- defective filters, bypassing filters and, 268–269
- DELETE statements (SQL), code injection, 250
- design flaws in authentication mechanisms
  - brute-forcible login, 136–138
  - incomplete credential validation, 152
  - insecure distribution of credentials, 155
  - passwords, 135–136
    - change functionality, 144–145
    - forgotten functionality, 145–147
    - predictable initial, 154–155
  - remember me functionality, 148–149
  - user impersonation functionality, 149–151
  - usernames
    - non-unique, 152–153
    - predictable, 154
  - verbose failure messages, 139–141
  - vulnerable transmission of credentials, 142–143
- Digest authentication, HTTP, 47
- directory listings, web server, 559–560
- directory names, 86
- DOM-based attacks, checking for, 724–725
- DOM-based XSS vulnerabilities, 386–388
- domain restrictions, cookies, 203–205
- dynamic execution, bypassing filters and, 268–269

E

- EJB (Enterprise Java Bean), 49
- elements, HTML forms, disabled, 110–111
- email, XSS attacks and, 388
- encoding
  - Base64, 58
  - hex, 59
  - HTML, 57–58
  - Unicode, 57
  - URL, 56
  - vulnerabilities, web server, 568–571
- Enterprise Java Bean (EJB), 49
- entry points for user input, identifying, 80–81
- enumeration, identifiers, bespoke automation and, 472, 473–483
- environments, shared
  - attacking, 544–549
  - securing, 549–551
- ERP (enterprise planning) software, 4
- error handling, 27–29
- error messages, 505–506
  - database messages, 509–511
  - debug messages, 508–509

  - engineering informative, 512–513
  - information leakage and, 516–517
  - public information, 511–512
  - script error messages, 506–507
  - server messages, 509–511
  - stack traces, 507–508
- exported functions, 122

F

- fail-open login mechanism, 156–157
- failure messages, verbose, 139–141
- fields, hidden in forms, 96–98
- file extensions, 84–86
- file inclusion, testing for, 711
- files, static, 222–223
- fingerprinting database, code injection and, 255–256
- Firefox, 624–626
- Flash VM, 125
- format string vulnerabilities, 531–532, 586
  - detecting, 532–533
- forms, 52–53
  - fields, hidden, 96–98
  - HTML, length limits, 106–108
  - parsing, web spidering, 62
- frame injection, 438–439
  - checking for, 725
  - exploiting, 439–440
  - preventing, 440
- function-specific input vulnerabilities, testing for
  - LDAP injection, 715–716
  - native software vulnerabilities, 713–714
  - SMTP injection, 712–713
  - SOAP injection, 715
  - XPath injection, 716–717
- functions
  - exported, 122
  - identifier-based, 220–221
  - multistage, 222
  - web server configuration, 557–558
- fuzzing, bespoke automation and, 472, 487–491

G

- GET method, HTTP, 38
- getObsScore method, 112–113

H

- harvesting data, bespoke automation and, 472, 484–487
- HEAD method, HTTP, 39
- heap overflows, 523–524
- hex encoding, 59
- Hibernate, Database object relational mapping, 49
- hidden content, discovering, 67
  - brute-force, 67–70
  - inference, 70–72
  - leveraging the web server, 75–76
  - public information, use of, 72–74
- hidden parameters, 79
- hijacking, session tokens, client exposure to, 201–202
- history, stealing, 396
- horizontal access controls, 218
- horizontal privilege escalation, 218
- Host header, HTTP request, 37
- hosting
  - shared, 542–543
  - virtual hosting, 543
- HTML (hypertext markup language), 51
  - encoding, 57–58
  - forms
    - disabled elements, 110–111
    - length limits, 106–108
    - parsing, 62
    - script-based validation, 108–110
- HTTP authentication, 178–179
- HTTP cookies, 99
- HTTP fingerprinting, 82–83
- HTTP header injection, 705–706
  - HTTP response splitting, 436–438
  - injecting cookies, 435–436
  - vulnerabilities
    - exploiting, 434–438
    - preventing, 434–438
- HTTP (Hypertext Transfer Protocol), 4–5, 35–36
  - authentication, 47
  - cookies, 43–44
  - headers
    - general, 41
    - request, 41–42
    - response, 42
    - tamper-proof, 101
  - HTTPS, 45–46
  - methods
    - GET, 38
    - HEAD, 39
    - OPTIONS, 40
    - POST, 39
    - PUT, 40
    - TRACE, 39–40
  - proxies, 46
  - requests, 36–37
    - headers, 41–42
    - Referer header, 99–100
  - responses, 37–38
    - headers, 42
    - status codes, 44–45
  - URLs, 40–41
- HTTP methods, web server, 560–562
- Hydra, 660
- hyperlinks, 51–52

I

- identifier-based functions, 220–221
- identifiers, enumerating, bespoke automation and, 472, 473–483
- implementation flaws
  - fail-open login mechanism, 156–157
  - insecure storage of credentials, 161
  - multistage login mechanisms, defects, 157–161
- incomplete credential validation, 152
- inference, 514–516
- information leakage, 6
  - follow up, 726–727
  - preventing
    - generic error messages, 516–517
    - minimizing client-side, 517
    - protecting sensitive information, 517
- initial passwords, predictable, 154–155
- injecting code. *See* code injection
- input, validation, 23–25
- input-based vulnerabilities, testing for
  - file inclusion, 711
  - fuzz all request parameters, 699–702
  - OS command injection, 707–708
  - path traversal, 709–710
  - script injection, 711
  - SQL injection, 702–704
  - XSS injection, 704–707
- input handling
  - accept known good, 21–22
  - reject known bad, 21
  - safe data handling, 22–23
  - sanitization, 22
  - semantic checks, 23
- insecure distribution of credentials, 155
- insecure storage of credentials, 161
- INSERT statements (SQL), code injection and, 248–249
- integer overflows, 529
- integer vulnerabilities, 586
  - detecting, 530–531
  - integer overflows, 529
  - signedness errors, 529–530
- integrated testing suites, 627–628
  - application fuzzers and scanners, 636–637
  - Burp Suite, 643–644
  - feature comparison, 640–643
  - intercepting proxies
    - alternatives to, 646–647, 646–648
    - browser configuration, 628–629
    - common features, 631–633
    - intercepting proxies and HTTPS, 629–631

Tamper Data, 647–648
TamperIE, 647–648
manual request tools, 637–639
shared functions and utilities, 639–640
Paros, 644–645
web application spiders, 633–636
WebScarab, 645–646
intercepting server responses, 107
Internet Explorer, 624
interpreted languages, code injection and, 238–239
iPlanet search overflow, 567
ISAPI extensions, 567

J
JAR (Java ARchive) files, 116
JAttack, 477–483
Java
APIs, dangerous, 589–592
bytecode
decompiling, 114–117
obfuscation, 117–119
environment, configuring, 593–594
session interaction, 589
user-supplied data, identifying, 587–589
web containers, 49
Java applets, JAR files, 116
Java Platform, Enterprise Edition, 49–50
Java Servlets, 49
JavaScript, 54, 616–617
JSON (JavaScript Object Notation), 446
attacks against
implementing callback function, 448–449
overriding array constructor, 447–448
hijacking, 446–447
preventing, 450
vulnerabilities, 449

K
keystrokes, logging, 396

L
LDAP
code injection, 326–327
flaws, 329–330
modifying search filter, 328–329
preventing, 330
query attributes, 327–328
injection, testing for, 715–716
local network, port scanning, 397–398
local privacy attacks
autocomplete, 460
browsing history, 459
cached web content, 458–459
cookies, persistent, 458
preventing, 460–461
local privacy vulnerabilities, checking for, 726
logging, 131
Log4J, 49
session token disclosure, 196–198
logic flaws
abusing a search function example, 365–366
avoiding, 370–372
beating a business limit example, 360–362
breaking the bank example, 356–359
cheating on bulk discounts example, 362–363
erasing an audit trail example, 359–360
escaping from escaping example, 363–364
fooling password change function example, 351–352
nature of, 350
proceeding to checkout example, 352–354
racing against the login, 368–370
rolling your own insurance example, 354–356
snarfing debug messages example, 366–368
testing for
handling of incomplete input, 718–719
key attack surface, 717
multistage processes, 718
transaction logic, 719–720
trust boundaries, 719
login
brute-forcible, 136–138
bypassing, 243–244
fail-open login mechanism, 156–157
multistage login mechanisms, defects in, 157–161
sessions and, 176
Log4J, logging, 49

M
mapping, session tokens, 198–200
mapping application content
default content and, 670–671
hidden content and, 670–671
identifier-specified functions, 671–672
public resources and, 670
test for debug parameters, 672
visible content and, 669–670
McAfee Epolicy Orcestrator, 568
Microsoft IIS
ISAPI extensions, 567
Unicode path traversal vulnerabilities, 569–570
WebDav overflow, 567
minimizing client-side information leakage, 517
multistage functions, 222
multistage login mechanisms, defects in, 157–161
multistep validation, 26–27
MySpace, XSS attack, 388

N
native software bugs
buffer overflow vulnerabilities, 585–586
format string vulnerabilities, 586
integer vulnerabilities, 586
native software vulnerabilities, testing for
buffer overflows, 713
format string vulnerabilities, 714
integer vulnerabilities, 714
networks
hosts, attacking, 398
port scanning, 397–398
session token disclosure, 192–195
Nikto, 660
non-unique usernames, 152–153
NTLM authentication, HTTP, 47

O
ODBC error messages, 262–263
column names, enumerating, 263–265
extracting arbitrary data, 265
recursion, 266
table names, enumerating, 263–265
off-by-one vulnerabilities, 524–527
OllyDbg, 120
opaque data, 101–102
Opera, 626–627
OPTIONS method, HTTP, 40
Oracle PL/SQL exclusion list bypasses, 570–571
OS command injection, 584
testing for, 707–708
OS commands, code injection, 300–307

P
parameters, URL parameters, 99–100
Paros, 62, 97, 644–645
parsing, HTML forms, web spidering, 62
passwords
backdoor, 584–585
bad, 135–136
change functionality, 144–145
forgotten functionality, 145–147
initial, predictable, 154–155

- path traversal, 582–583
  - testing for, 709–710
  - vulnerabilities
    - circumventing obstacles to attacks, 339–343
    - common, 333–334
    - custom encoding, 342–343
    - detecting, 336–339
    - exploiting, 344
    - preventing, 344–346
    - targets for attack, locating, 335–336
    - web server, 568
- per-page tokens, 211
- Perl, 611–612
  - APIs, dangerous, 613–615
  - code injection and, 302–303
  - environment, configuring, 615–616
  - session interaction, 613
  - user-supplied data, identifying, 612
- phishing scams, 383
- PHP, 50–51
  - APIs, dangerous, 604–609
  - environment, configuring, 609–611
  - session interaction, 603
  - user-supplied data, identifying, 601–603
- POJO (Plain Old Java Object), 49
- port scanning network, 397–398
- POST method, HTTP, 39
- Pragma header, HTTP response, 38
- predictable initial passwords, 154–155
- Presentation layer
  - SiteMesh, 49
  - Tapestry, 49
- protecting sensitive information, 517
- proxies
  - HTTP, 46
  - web servers as, 562–564
- public information, error messages, 511–512
- published information, gathering, 513–514
- PUT method, HTTP, 40

R

- redirection attacks, 428
  - absolute prefix, 432
  - blocking absolute URLs, 431
  - vulnerabilities
    - finding and exploiting, 429–433
    - preventing, 433–434
- Referer header, HTTP request, 37, 99–100
- reflected request parameters, 704
- reflected XSS, 379, 705
- remember me functionality, 148–149
- request forgery, 440–441
  - OSRF (on-site request forgery), 441–442
  - XSRF (cross-site request forgery), 442–446
    - flaws, exploiting, 443–444
    - flaws, preventing, 444–446
- reverse engineering, 120–122
- robots.txt, 62

S

- safe data handling, 22–23
- same origin policy, 381
- sample functionality, 556–557
- sanitizing input, 22
- script-based validation, 108–110
- script injection, testing for, 711
- scripting, cross-site, 6
- scripts, custom, 661–662
  - Curl, 662–663
  - Netcat, 663
  - Wget, 662
- search engines, hidden content and, 72
- search queries, stealing, 396
- security, 5
  - future of, 12
  - problem factors, 9–10
- SELECT statements (SQL), code injection and, 248
- semantic checks, 23
- sensitive information, protecting, 517
- server error messages, 509
- Server header, HTTP response, 38
- server responses, intercepting, 107
- server-side functionality
  - application behavior, 90–91
  - requests, 88–90
- server-side technologies
  - banner grabbing, 82
  - directory names and, 86
  - file extensions and, 84–86
  - HTTP fingerprinting, 82–83
  - session tokens and, 86
  - third-party code components and, 87–88
- session fixation, 450–452, 694
  - vulnerabilities
    - finding and exploiting, 452–453
    - preventing, 453–454
- session management, 17–18, 175–176
  - alerts, 211–212
  - logging, 211–212
  - monitoring, 211–212
  - securing
    - strong tokens, 206–208
    - token protection, 208–211
- session management mechanism, testing, 688
  - check cookie scope, 695–696
  - check for disclosure of tokens in logs, 692
  - check for insecure transmission of tokens, 691–692
  - check for session fixation, 694
  - 7check for XSRF, 694–695
  - check mapping of tokens to sessions, 692–693
  - test session termination, 693–694
  - testing tokens for meaning, 689–690
  - testing tokens for predictability, 690–691
  - understanding mechanism, 689
- session termination, testing, 693–694
- session tokens, 86
  - concurrent logins, 209
  - disclosure in logs, 196–198
  - disclosure on network, 192–195
  - generation, weaknesses in, 180–191
  - handling, weaknesses in, 191–192
  - hijacking, client exposure to, 201–202
  - logout functionality, 209
  - mapping, 198–200
  - meaningful, 180–182
  - per-page, 211
  - predictable, 182–183
    - concealed sequences, 184–185
    - random number generation, 187–191
    - time dependency, 185–187
  - SSL and, 192
  - structured, 181
  - transmitting to URL, 209
- sessionless state mechanisms, 179
- sessions, 55, 176
  - alternatives to, 178–180
  - HTTP authentication and, 178–179
  - identifiers, 177
  - login, 176
  - sessionless state mechanisms, 179
  - termination, 200–201
  - tokens, 177
    - cookies and, 178
- Set-Cookie header, HTTP response, 38, 203
- shared application services, 543–544
- shared environments
  - attacking
    - attacks against mechanisms, 545–546
    - attacks between applications, 546–549
  - securing
    - secure customer access, 549–550
    - segregating components in shared application, 551
    - segregating customer functionality, 550

- shared hosting, 542–543
  - virtual hosting, 543
  - vulnerabilities
    - segregation between ASP-hosted applications, 721
    - segregation in shared infrastructures, 720
- Shockwave Flash objects, 123–124
- signedness errors, 529–530
- SiteMesh, Presentation Layer, 49
- smartcards, authentication and, 176
- SMTP, code injection, 321–322
  - command injection, 323–324
  - email header manipulation, 322–323
  - flaws, 324–325
  - preventing, 325–326
- SMTP injection, testing for, 712–713
- SOAP
  - code injection, 313–316
  - injection, testing for, 715
- software
  - native bugs
    - buffer overflow vulnerabilities, 585–586
    - format string vulnerabilities, 586
    - integer vulnerabilities, 586
  - security hardening, 573
- source code, comments, 586–587
- spidering. *See* web spidering
- SQL (Structured Query Language)
  - code injection, 240–241
    - bugs, 244–247
    - bypassing login, 243–244
    - DELETE statements, 250
    - exploiting a basic vulnerability, 241–243
    - INSERT statements, 248–249
    - preventing, 296–300
    - SELECT statements, 248
    - UNION operator, 250–255
    - UPDATE statements, 249–250
  - comments, bypassing filters and, 268
  - error messages, 292–295
  - injection, 6, 581–582
    - testing for, 702–704
  - syntax reference, 289–291
- SSL (Secure Socket Layer), 6, 7
  - ciphers, weak, 727
  - session tokens and, 192
- stack overflows, 522–523
- stack traces, 507–508
- state, 55, 176–177
  - sessionless state mechanisms, 179
- static files, 222–223
- status codes, HTTP responses, 44–45
- stored attacks, testing for, 706–707

T

- Tapestry, Presentation Layer, 49
- termination, sessions, 200–201
- testing access controls
  - insecure access control methods, 698
  - limited access, 697–698
  - multiple accounts, 697
  - requirements, 696–697
- testing authentication mechanism
  - account recovery function, 682
  - check for unsafe distribution of credentials, 685
  - check for unsafe transmission of credentials, 684–685
  - exploit any vulnerabilities to gain unauthorized access, 687–688
  - impersonation function, 683
  - logic flaws, fail-open conditions, 685–686
  - multistage mechanisms, 686–687
  - password quality, 680
  - predictability of auto-generated credentials, 684
  - remember me function, 682–683
  - resilience to password guessing, 681
  - understand mechanism, 680
  - username enumeration, 680–681
  - username uniqueness, 683–684
- testing client-side controls
  - client-side controls over user input, 676
  - thick-client components
    - ActiveX controls, 678
    - Java applets, 677
    - Shockwave Flash objects, 678–679
  - transmission of data via the client, 675–676
- testing for function-specific input vulnerabilities
  - LDAP injection, 715–716
  - native software vulnerabilities, 713–714
  - SMTP injection, 712–713
  - SOAP injection, 715
  - XPath injection, 716–717
- testing for input-based vulnerabilities
  - fuzz all request parameters, 699–702
  - test for file inclusion, 711
  - test for OS command injection, 707–708
  - test for path traversal, 709–710
  - test for script injection, 711
  - test for SQL injection, 702–704
  - test for XSS injection
    - arbitrary redirection, 706
    - HTTP header injection, 705–706
    - reflected request parameters, 704
    - reflected XSS, 705
    - stored attacks, 706–707
- testing for logic flaws
  - handling of incomplete input, 718–719
  - key attack surface, 717
  - multistage processes, 718
  - transaction logic, 719–720
  - trust boundaries, 719
- testing session management mechanism, 688
  - check cookie scope, 695–696
  - check for disclosure of tokens in logs, 692
  - check for insecure transmission of tokens, 691–692
  - check for session fixation, 694
  - check for XSRF, 694–695
  - check mapping of tokens to sessions, 692–693
  - test session termination, 693–694
  - test tokens for meaning, 689–690
  - test tokens for predictability, 690–691
  - understanding mechanism, 689
- thick-client components, 54–55, 111–112
  - ActiveX controls, 119–120
    - decompiling managed code, 124
    - exported functions, 122
    - inputs, fixing, 123–124
    - reverse engineering, 120–122
  - Java applets, 112–114
    - bytecode obfuscation, 117–119
    - decompiling Java bytecode, 114–117
- third-party code components, 87–88
- tiered architectures, 535–536
  - attacking
    - attacking tiers, 539–540
    - exploiting trust relationships between tiers, 537–538
    - subverting tiers, 538–539
  - securing
    - applying defense in depth, 542
    - minimizing trust relationships, 540–541
    - segregating different components, 541–542
- tokens. *See* session tokens
  - disclosure in logs, 692
  - insecure transmission, 691–692
  - mapping to sessions, 692–693
  - strong, session management and, 206–208
  - testing for meaning, 689–690
  - testing for predictability, 690–691
  - transmitting, HTTPS, 208–209

TRACE method, HTTP, 39
transaction logic, 719–720
Trojans, XSS and, 392–393
trust boundaries, 719

U

Unicode encoding, 57
UNION operator (SQL), code injection, 250–255
unprotected functionality, 219
UPDATE statements (SQL), code injection, 249–250
URLs, 40–41
- ASCII characters, 56
- encoding, 56
- parameters, 99–100

user access
- access control, 18–19
- authentication, 16–17
- session management, 17–18

user actions, inducing, 394
User-Agent header, HTTP request, 37
user-directed web spidering, 65–66
user impersonation functionality, 149–151
user input, 19–20
- canonicalization, 26–27
- entry points, identifying, 80–81
- input, validation, 23–25
- input handling
  - accept known good, 21–22
  - reject known bad, 21
  - safe data handling, 22–23
  - sanitization, 22
  - semantic checks, 23
- types, 20–21
- validation
  - boundary validation, 23–25
  - multistep, 26–27

usernames
- authentication and, 139
- non-unique, 152–153
- predictable, 154

users, input, 8–9

V

validation
- boundary validation, 23–25
- canonicalization, 26–27
- multistep, 26–27
- script-based, 108–110

vendor patches, software, 572
verbose failure messages, 139–141
vertical access controls, 218
vertical privilege escalation, 218
ViewState (ASP.NET), 102–106
virtual hosting, 543
- misconfigured, 564–565

vulnerability scanners
- challenges faced by, 653–656
- limitations, 651–653
- using, 658–659
- vulnerabilities detected, 649–651

vulnerable transmission of credentials, 142–143

W

web applications
- benefits of, 4–5
- common functions, 3–4
- evolution of, 2–5
- managing, 32–33

web browsers
- exploitation frameworks, 467–469
- Firefox, 624–626
- Internet Explorer, 624
- Opera, 626–627
- vulnerabilities, 399

web server
- buffer overflow, vulnerabilities, 566–567
- configuration
  - debug functionality, 555–556
  - default content, 555–558
  - default credentials, 554–555
  - functions, 557–558
  - sample functionality, 556–557
  - securing, 565–566
- directory listings, 559–560
- encoding and canonicalization vulnerabilities, 568–571
- flaws, finding, 571–572
- HTTP methods, dangerous, 560–562
- path traversal vulnerabilities, 568
- as proxy, 562–564
- software, securing, 572–574
- virtual hosting, misconfigured, 564–565
- vulnerabilities, 721
  - dangerous HTTP methods, 722–723
  - default content, 722
  - default credentials, 722
  - proxy functionality, 723
  - virtual hosting misconfiguration, 723
  - web server software bugs, 723–724

web sites, 2
web spidering, 62–64
- user-directed, 65–66

WebDAV (Web-based Distributed Authoring and Versioning), 561
WebScarab, 62, 97, 645–646
white-box testing, 578–579

X

XPath
- code injection, 316–317
  - blind, 319–320
  - flaws, 320–321
  - preventing, 321
  - subverting application logic, 317–318
- injection, testing for, 716–717

XSRF, 694–695
XSS (cross-site scripting), 6, 376–377
- chaining, 390–391
- client-side attack, escalating
  - attack other network hosts, 398
  - clipboard contents, 396
  - currently used applications, 397
  - exploit browser vulnerabilities, 399
  - history and search queries, 396
  - keystrokes, 396
  - port scanning local network, 397–398
- cross-site tracing, 421–423
- delivery mechanisms
  - reflected and DOM-based attacks, 399–400
  - stored attacks, 400–401
- entry points, 405
- HttpOnly cookies, 421–423
- inducing user actions and, 394
- injection, testing for, 704–707
- length limits, 411–413
- nonstandard content encoding
  - US-ASCII, 414
  - UTF-7, 414
  - UTF-16, 414
- preventing attacks, 423–428
- real-world attacks, 388–390
- reflected, 379
- request methods, 413–414
- sanitization, 409–411
- signature-based filters, 406–409
- trojans and, 392–393
- trust relationships, exploiting, 394–395
- virtual defacement and, 391–392
- vulnerabilities, 377–379
  - DOM-based, 386–388
  - exploiting, 379–383
  - finding and exploiting DOM-based, 417–421
  - finding and exploiting reflected, 401–415
  - finding and exploiting stored, 415–417
  - stored, 383–386