

# Index

## Numerics

3DES, 201

## A

**account lockouts, 451–454**

**AcquireRequestState event, 42**

**Active Directory. See AD (Active Directory)**

**Active Directory Application Mode. See ADAM (Active Directory Application Mode)**

**ActiveDirectoryMembershipProvider**

ActiveDirectoryMembershipUser,  
480–482

AD (Active Directory), 482–502

ADAM (Active Directory Application Mode),  
503–512

ChangePassword, 477

ChangePasswordQuestionAndAnswer,  
478

CreateUser, 477

DeleteUser, 478

FindUsersByEmail, 478

FindUsersByName, 478

functionality of provider, 477–480

GeneratePassword, 478

GetAllUsers, 478

GetNumberOfUsersOnline, 478

GetPassword, 478

GetUser, 478

GetUserNameByEmail, 479

overview, 465

partial trust, 512–515

provider configuration, 468–477

ResetPassword, 479

supported directory architectures, 465–468

UnlockUser, 479

UpdateUser, 479

ValidateUser, 479

**ActiveDirectoryMembershipUser**

IsApproved, 481

IsLockedOut, 481

overview, 480–481

ProviderUserKey, 482

**AD (Active Directory)**

container nesting, 486–487

overview, 482–484

SAM account names, 484–485

securing containers, 487–493  
self-service password reset, configuring,  
494–502

UPN-style usernames, 484–485

**ADAM (Active Directory Application Mode)**

application partition, using, 510–512

installation with an application partition,  
504–510

overview, 503–504

**AddUsersToRole method, 520**

## AddUsersToRoles method

---

**AddUsersToRoles method, 520, 545**

**AddUserToRole method, 520**

**AddUserToRoles method, 520**

**ADODB access, 131–133**

**AES (Advanced Encryption Standard), 201–204**

**allowOverride attribute, 146**

**AllowPartiallyTrustedCallers**

**Attribute class**

/bin directory and, 125–126

overview, 121–124

strongly named assemblies and, 124

**/app\_browsers path, 8**

**/app\_code path, 8**

**/app\_data path, 8**

**\$AppDir\$, 85**

**\$AppDirUrl\$, 85**

**/app\_globalresources path, 8**

**application domain policy, 92**

**application domain startup**

assembly locations, establishing, 19

auto-generated machine key, obtaining, 19–23

compilation system, initializing, 23

directory information, initializing, 16–17

identity, establishing, 16

overview, 15

trust level, setting, 17–19

**application domain trust level, 17**

**application impersonation, 26**

**application name, storing, 404–405**

**application partition, using, 510–512**

**ApplicationId, 406, 415, 554**

**ApplicationName, 518, 544, 577**

**applicationName, 475, 577**

**Application\_Start event, 25–28**

**/app\_localresources path, 8**

**applying new trust level, 99**

**App\_Offline.htm**

disabling a Website with, 24–25

origins of, 25

**/app\_webreferences path, 8**

**ASP.NET permission set, 86–87**

**ASP.NET per-request security**

asynchronous pipeline events and thread

identity, 43–48

DefaultAuthentication event, 54–56

DefaultAuthenticationModule, 54–56

HttpContext.Current.User, 48

IPrincipal, 48

operating system thread identity, establishing, 38–41

overview, 33–34

requests, security identity for, 34–37

Thread.CurrentPrincipal, 48, 54–56

**aspnet\_filter.dll, 5–6**

**AspNetHostingPermission**

code, used in your, 110–111

Full trust, 107

functionality of ASPNET and trust level of, 108

High trust, 107

implications outside of ASPNET of, 109

Low trust, 107

Medium trust, 107

Minimal trust, 108

overview, 106

trust levels, 107–108

**aspnet\_isapi.dll**

application domain, starting up an, 15–23

overview, 14

**aspnet\_regiis, 181–182**

**aspnet\_Roles\_BasicAccess, 563**

**aspnet\_Roles\_FullAccess, 563**

**aspnet\_Roles\_ReportingAccess, 563**

**assembly locations, establishing, 19**

**Assertion, 115**

**asynchronous page execution**

asynchronous PreRender processing, 69–71

automatically flowing identity to asynchronous work, 73–74

identity during, 69–74

overview, 69

PageAsyncTask, 71–73

**asynchronous pipeline events and thread identity, 43–48**

**asynchronous PreRender processing, 69–71**

**AuthenticateRequest event**

FormsAuthenticatonModule, 52–54

overview, 48–49, 192

WindowsAuthenticatonModule, 49–52

**authenticating classic ASP with ASP.NET**

cookieless forms authentication and, 273–274  
 overview, 272–273  
 passing data to ASP from ASP.NET, 274–276  
 passing username to ASP, 276

**Authorization Manager (AzMan), 573–576****authorization methods, 544****AuthorizationStoreRoleProvider**

ApplicationName, 577  
 applicationName, 577  
 Authorization Manager (AzMan), 573–576  
 cacheRefreshInterval, 577  
 directory-based policy store, 580–589  
 file-based policy store, 578–580  
 Membership, use of Role Manager and, 592–594  
 overview, 573  
 partial trusts, 589–592  
 provider design, 573–576  
 Role Manager, use of Membership and, 592–594  
 ScopeName, 577  
 scopeName, 577  
 supported functionality, 576–577

**AuthorizeRequest event**

FileAuthorizationModule, 58–60  
 overview, 58, 192  
 UrlAuthorizationModule, 60–65

**authorizing classic ASP with ASP.NET**

ConvertStringKeyToByteArray method, 281  
 hash helper, full code listing of, 284–285  
 HashStringValue method, 280  
 Helper class, 284–285  
 overview, 276–277  
 passing user roles to classic ASP, 277–278  
 sensitive data to classic ASP, safely passing, 278–284  
 ValidateHash method, 280

**AutoDetect attribute**

Internet Explorer, simulating in, 212–213  
 overview, 209, 210  
 phase 1, 210–211  
 phase 2, 211  
 phase 3, 211–212

subsequent authenticated access, 212

**auto-generated machine key**

AutoGenKey, 20  
 AutoGenKeyCreationTime, 20  
 AutoGenKeyFormat, 20  
 obtaining, 19–23

**AutoGenKey, 20****AutoGenKeyCreationTime, 20****AutoGenKeyFormat, 20****automatic unlocking, implementing, 454–458****automatically flowing identity to**

asynchronous work, 73–74

**AzMan (Authorization Manager), 573–576****B****basic configuration, 544****BeginRequest event, 42****/bin directory, 125–126****/bin path, 8****blocking restricted directories, 8–9****building a provider-based feature**

concrete provider, 359–361  
 configuration, 357–359  
 connection string, 362  
 initialization, 353–357  
 lifecycle of provider-based feature, 353  
 overview, 351–353  
 sample application, 363–365

**C****CachedListChanged property, 529****cacheRefreshInterval, 577****CacheRolesInCookie property, 518, 535****CAS policy, checking current, 93–94****case sensitivity, 414–415****central login application, 241–245****ChangePassword method, 386, 477****ChangePasswordQuestionAndAnswer method, 391, 478****character sets affecting URL authorization, 65****client impersonation, 26****clientSearchTimeout, 474****clock resets, 196–197**

**code**

- access security, places that define, 90–91
- AspNetHostingPermission, used in your, 110–111
- permission sets matched to, 88–90
- \$CodeGen\$, 85**
- <CodeGroup /> elements, 89**
- commandTimeout, 418, 555**
- Comment property, 371, 418**
- common database schema**
  - Application Id, 406
  - application name, storing, 404–405
  - case sensitivity, 414–415
  - common users table, 405–408
  - IsAnonymous, 408
  - LastActivityDate, 407–408
  - MobileAlias, 408
  - overview, 404
  - user records, linking custom features to, 410–413
  - versioning provider schema, 408–10
  - views, querying common tables with, 410
- common users table, 405–408**
- compilation system, initializing, 23**
- complex permissions, troubleshooting, 94–96**
- configProtectedData, 168**
- configuration**
  - IIS6 wildcard mappings, 261–268
  - MembershipProvider base class, 383–384
- configuration class, demanding permissions from, 165–166**
- configuration system security. See also protected configuration**
  - configuration used in partial trust, 161–166
  - <location /> element, 143–146
  - lock attributes, 146–153
  - reading and writing configuration, 153–161
- configuration used in partial trust**
  - configuration class, demanding permissions from, 165–166
  - design-time API and, 166
  - FileIOPermission, 166
  - overview, 161–163
  - requirePermission attribute, 163–164

- ConfigurationManager class, 153–154**
- connection string, changing, 425–426**
- connectionProtection, 470**
- connectionStringName, 418, 555**
- container nesting, 486–487**
- ControlPrincipal, 115–116**
- ControlThread, 116**
- ConvertStringKeyToByteArray method, 281**
- cookie domain, 226–227**
- cookie-based sessions**
  - cross-application cookie sharing, 292–293
  - overview, 291
  - protecting session cookies, 293
  - session ID reuse, 294
- cookie-based SSO-lite**
  - behavior of solution, list of desired, 236
  - central login application, 241–245
  - examples of, 245–246
  - overview, 234–238
  - sample participating application, 238–241
  - technical tips for, 246–247
- cookied cross-application behavior, 231–234**
- cookieless cross-application behavior, 228–231**
- cookieless forms authentication**
  - and authenticating classic ASP with ASPNET, 273–274
  - AutoDetect attribute, 209, 210–213
  - overview, 208–210
  - payload size and, 218–220
  - replay attacks with cookieless tickets, 215–216
  - unexpected redirect behavior and, 221–222
  - URLs in pages and, other, 216–218
  - UseCookies attribute, 209
  - UseDeviceProfile attribute, 209, 213–214
  - UseUri attribute, 209
- cookieless sessions, 294–296**
- CookiePath property, 530**
- cookieProtection, 536**
- cookieRequiresSSL, 536**
- cookiesSlidingExpiration, 536**

**cookie-specific security options**

HttpOnly cookies, 206–208  
 overview, 204  
 requireSSL attribute, 204–206  
 slidingExpiration attribute, 208

**cookieTimeout, 535–536****core provider classes**

System.Configuration classes,  
 347–350  
 System.Configuration.Provider  
 classes, 342–346  
 System.Web.Configuration classes,  
 346

**CreateDate, 417****createPersistentCookie, 535****CreateRole, 545****CreateUser method, 385, 390–391, 392,  
 477****CreationDate property, 371, 379****cross-application cookie sharing, 292–293****cross-application ticket passing. See also  
 cross-application ticket sharing**

cookie domain, 226–227  
 overview, 226

**cross-application ticket sharing. See also  
 cookie-based SSO-lite**

cookieless cross-application behavior,  
 228–231  
 how it works, 228–234  
 overview, 227–228  
 single sign on (SSO) products, 227–228

**cryptographicSettings, 168****custom encryption, 435–437****custom hash algorithms, 399–401****custom password generation, 432–435****custom password strength rules**

initialize method, 444  
 overview, 437–439  
 password history, implementing, 440–451  
 ValidatePassword event, 439–440

**custom provider, redirecting configuration  
 with, 184–190****custom trust level**

applying new trust level, 99  
 creating, 96–105  
 declarative permission representations,  
 determining, 97–99  
 OdbcPermission, customizing, 101–103  
 OleDbPermission, customizing, 100–101  
 overview, 96  
 policy file, creating, 96–97  
 WebPermission, 103–105

**D****data layer, authorizing roles in, 570–571****database schema. See also common database  
 schema**

Membership, 415–418  
 SqlRoleProvider, 553–556

**database security, 426–428, 563****DataList data control, 313****date-time values, 380–382****DBO user and database schemas, 428–430****declarative permission representations,  
 determining, 97–99****default security permissions defined by  
 ASP.NET**

AspNetHostingPermission, 106–111  
 DnsPermission, 111  
 EnvironmentPermission, 111–112  
 FileIOPermission, 112–113  
 IsolatedStorageFilePermission, 113  
 overview, 105  
 PrintingPermission, 114  
 ReflectionPermission, 114  
 RegistryPermission, 115  
 SecurityPermission, 115–116  
 SmtppPermission, 117  
 SocketPermission, 117  
 SqlClientPermission, 118  
 WebPermission, 118

**DefaultAuthentication event, 54–56****DefaultAuthenticationModule, 54–56****DefaultHttpHandler**

EndProcessRequest, 270–271  
 OverrideExecuteUrlPath, 270–271

### **DefaultHttpHandler (continued)**

- overview, 268–269
- using, 270–271
- defining protected configuration providers, 172**
- DeleteRole method, 520, 545**
- DeleteUser method, 385, 478**
- Denial of Service (DOS) attacks, 297–300**
- Description, 555**
- deserialization requirements for session state, 302–304**
- Design Patterns: Elements of Reusable Object-Oriented Software* (Gamma, Helm, Johnson & Vlissides), 332**
- design-time API, 166**
- DetailsView data control, 313**
- directory connection settings, 468–471**
- directory information, initializing, 16–17**
- directory schema mappings, 471–474**
- directory-based policy store, 580–589**
- disabling a Website with `App_Offline.htm`, 24–25**
- DLLs (dynamic link libraries), 10**
- DnsPermission, 111**
- DOS (Denial of Service) attacks, 297–300**
- DpapiProtectedConfigurationProvider**
  - `keyEntropy`, 173–174
  - overview, 172–173
  - `useMachineProtection`, 174–175
- dynamic applications, supporting, 458–463, 571–572**
- dynamic content**
  - ISAPI extension mappings, 10–13
  - MIME type mappings, 9–10
  - overview, 9
  - static content compared, 9–14
  - wildcard application mappings, 13–14
- dynamic link libraries (DLLs), 10**

**E**

- ECB (Extension Control Block), 32**
- Email property, 372, 416**
- Enabled property, 518**

- EnablePasswordReset property, 390, 476**
- EnablePasswordRetrieval property, 390**
- enableSearchMethods, 474**
- EndProcessRequest, 270–271**
- EndRequest event, 42, 74–75, 192, 534–535**
- enterprise trust level, 17**
- EnvironmentPermission, 111–112**
- error handling, 393–394**
- Execution, 116**
- expiration enforced by forms authentication, 194–196**
- Expired property, 530**
- ExpireDate property, 530**
- Extension Control Block (ECB), 32**

## **F**

- facade methods, 369–370**
- Facade pattern**
  - Membership, 341
  - overview, 341
  - Profile, 341
  - Role Manager, 341
  - session state, 342
  - Site Navigation, 342
  - Web parts personalization, 341–342
- Factory Method, 334–339**
- FailedPasswordAnswerAttemptCount, 417**
- FailedPasswordAnswerAttemptWindowStart, 418**
- FailedPasswordAttemptCount, 417**
- FailedPasswordAttemptWindowStart, 418**
- FileAuthorizationModule, 58–60**
- file-based policy store, 578–580**
- FileIOPermission, 112–113, 166**
- finding the trust policy file, 84**
- FindUsersByEmail method, 388, 478**
- FindUsersByName method, 387–388, 478**
- FindUsersInRole, 545**
- first request initialization**
  - application impersonation, 26
  - `Application_Start` event, 25–28

App\_Offline.htm, disabling a Website with, 24–25  
 client impersonation, 26  
 HttpRuntime, disabling a Website with, 24  
 overview, 23–24

**forms authentication. See also cross-application ticket passing**

AuthenticateRequest and, 192  
 AuthorizeRequest and, 192  
 cookieless forms authentication, 208–222  
 cookie-specific security options, 204–208  
 EndRequest and, 192  
 LoggedIn event, 249  
 LoggingIn event, 249  
 Membership and, 247–256  
 overview, 192  
 persistent tickets, 192–197  
 sharing tickets between ASP.NET 1.1 and ASP.NET 2.0, 222–223  
 single logons, enforcing, 247–255  
 single logouts, enforcing, 247–248, 255–256  
 ticket security, 198–204  
 UserData property, leveraging, 224–226

**FormsAuthenticationModule, 52–54**

**FormView data control, 313**

**fraudulent postbacks, 318–322**

**Full trust, 78, 80, 107**

**FullTrust permission set, 86**

**functionality of ASP.NET and trust levels, 108**

**functionality of provider, 477–480**

## G

**GAC (global assembly cache), 301**

**Gamma, Eric (*Design Patterns: Elements of Reusable Object-Oriented Software*), 332**

**GeneratePassword, 478**

**generating keys programmatically, 203–204**

**GetAllRoles, 545**

**GetAllUsers method, 387, 478**

**GetNumberOfUsersOnline method, 392, 478**

**GetPassword method, 391, 478**

**GetRoles method**

described, 523  
 working with multiple providers during, 537–542

**GetRolesForUser method, 520, 544**

**GetSection, 157**

**GetUser method, 387, 392, 478**

**GetUserNameByEmail method, 387, 479**

**GetUsersInRole, 545**

**GetWebApplicationSection, 157**

**global assembly cache (GAC), 301**

**GridView data control, 313**

## H

**handler execution, blocking requests during, 66–67**

**hash helper, full code listing of, 284–285**

**HashAlgorithmType property, 369**

**HashStringValue method, 280**

**headers, processing, 6–7**

**Helm, Richard (*Design Patterns: Elements of Reusable Object-Oriented Software*), 332**

**Helper class, 284–285**

**High trust, 79, 80, 107**

**HTTP handlers, 66–67**

**HttpContext.Current.User, 48**

**HttpOnly cookies, 206–208**

**HttpRuntime, disabling a Website with, 24**

**http.sys, 3–5**

## I

**identity, establishing, 16**

**Identity property, 523**

**<identity /> tag, 16, 171**

**IIS request handling**

aspnet\_filter.dll, 5–6

blocking restricted directories, 8–9

headers, processing, 6–7

http.sys, 3–5

overview, 2

registry settings, 5

**IIS5 ISAPI extension behavior, 260–261**

**IIS6 wildcard mappings**

configuration, 261–268

overview, 261

Verify that File Exists setting, 268

**impersonation token, 41**

**individual permissions, defining, 87–88**

**initialize method, 444**

**installation with an application partition,  
504–510**

**integration of ASP.NET security with classic  
ASP**

authenticating classic ASP with ASP.NET,  
272–276

authorizing classic ASP with ASP.NET,  
276–285

DefaultHttpHandler, 268–271

IIS5 ISAPI extension behavior, 260–261

IIS6 wildcard mappings, 261–268

overview, 259

**Internet Explorer, 212–213**

**IPrincipal, 48**

**IsAnonymous, 408**

**ISAPI extension mappings, 10–13**

**IsApproved property, 372, 417, 481**

**IsInRole method, 522–523**

**IsLockedOut property, 372, 379, 417, 481**

**IsolatedStorageFilePermission, 113**

**IsOnline property, 372, 379**

**IsRoleListCached property, 523**

**IssueDate property, 530**

**IsUserInRole method, 520, 544**

## J

**Johnson, Ralph (*Design Patterns: Elements of  
Reusable Object-Oriented Software*), 332**

## K

**keyContainerName, 176–179**

**keyEntropy, 173–174**

## L

**LastActivityDate property, 372, 407–408**

**LastLockoutDate property, 373, 379, 417**

**LastLoginDate property, 373, 417**

**LastPasswordChangedDate property, 373,  
379, 417**

**limited set of roles, 565–569**

**LinkDemand, 119–121**

**<location /> element**

allowOverride attribute, 146

overview, 143–145

path attribute, 145–146

**lock attributes**

lockAllAttributesExcept, 147–148

lockAllElementsExcept, 147, 149–151

lockAttributes, 147–148

lockElements, 147, 149–151

locking provider definitions, 151–153

overview, 146–147

**LoggedIn event, 249**

**LoggingIn event, 249**

**logon session compared to state session,  
287–290**

**Low trust, 79, 80, 107**

**LoweredEmail, 416**

**LoweredRoleName, 554**

## M

**machine trust level, 17**

**managing roles and role associations,  
544–545**

**MaxCachedResults property, 518, 536**

**MaxFieldLength registry setting, 5**

**MaxInvalidPasswordAttempts property,  
388–389, 476**

**MaxRequestBytes registry setting, 5**

**Medium trust, 79, 80, 107**

**Membership**

custom hash algorithms, use of, 399–401

Facade pattern, 341

and forms authentication, 247–256

Membership class, 368–371

MembershipProvider base class,  
382–394

MembershipUser class, 371–382

primary key for, 394–396

Role Manager and, 592–594

Strategy pattern, 333

supported environments, 396–399

**Membership class**

facade methods, 369–370

HashAlgorithmType property, 369

overview, 368–371  
 Provider property, 369  
 Providers property, 369  
 UserIsOnlineTimeWindow property, 369  
 utility methods, 370

### Membership database schema

ApplicationId, 415  
 commandTimeout, 418  
 Comment, 418  
 connectionStringName, 418  
 CreateDate, 417  
 Email, 416  
 FailedPasswordAnswerAttemptCount, 417  
 FailedPasswordAnswerAttemptWindowStart, 418  
 FailedPasswordAttemptCount, 417  
 FailedPasswordAttemptWindowStart, 418  
 IsApproved, 417  
 IsLockedOut, 417  
 LastLockoutDate, 417  
 LastLoginDate, 417  
 LastPasswordChangedDate, 417  
 LoweredEmail, 416  
 MobilePIN, 416  
 overview, 415  
 Password, 416  
 PasswordAnswer, 417  
 PasswordFormat, 416  
 PasswordQuestion, 416  
 PasswordSalt, 416  
 UserId, 416

### membership provider settings, 475–477

#### MembershipProvider base class

ChangePassword method, 386  
 ChangePasswordQuestionAndAnswer method, 391  
 configuration, 383–384  
 CreateUser method, 385, 390–391, 392  
 DeleteUser method, 385  
 EnablePasswordReset property, 390  
 EnablePasswordRetrieval property, 390  
 error handling, 393–394  
 FindUsersByEmail method, 388

FindUsersByName method, 387–388  
 GetAllUsers method, 387  
 GetNumberOfUsersOnline method, 392  
 GetPassword method, 391  
 GetUser method, 387, 392  
 GetUserNameByEmail method, 387  
 MaxInvalidPasswordAttempts property, 388–389  
 MinRequiredNonAlphanumericCharacters property, 384  
 MinRequiredPasswordLength property, 384  
 multiple users, retrieving and searching for, 387–388  
 online users, tracking, 392–393  
 OnValidatingPassword method, 386  
 overview, 382–383  
 password validation, 388–389  
 PasswordAttemptWindow property, 389  
 PasswordFormat property, 390  
 PasswordStrengthRegularExpression property, 384–385  
 RequiresQuestionAndAnswer property, 390  
 RequiresUniqueEmail property, 385  
 ResetPassword method, 391–392  
 single user, retrieving data for, 387  
 supporting self-service password reset or retrieval, 390–392  
 UnlockUser method, 389  
 UpdateUser method, 386, 392  
 user creation, 384–386  
 user updates, 384–386  
 ValidateUser method, 389, 392  
 validating user credentials, 388–389  
 ValidatingPassword property, 385

#### MembershipUser class

Comment property, 371  
 CreationDate property, 371, 379  
 date-time values, 380–382  
 Email property, 372  
 extending, 373–375  
 IsApproved property, 372  
 IsLockedOut property, 372, 379  
 IsOnline property, 372, 379

### MembershipUser class (continued)

LastActivityDate property, 372  
LastLockoutDate property, 373, 379  
LastLoginDate property, 373  
LastPasswordChangedDate property, 373, 379  
overview, 371–373  
PasswordQuestion property, 372, 379  
ProviderName property, 372, 379  
ProviderUserKey property, 371, 379  
updates and state of, 375–380  
UserName property, 371, 379  
UTC time and, 380–382

### MIME type mappings, 9–10

### Minimal trust, 79, 80, 108

**MinRequiredNonAlphanumeric Characters property, 384, 476**

**MinRequiredPasswordLength property, 384, 476**

**MobileAlias, 408**

**MobilePIN, 416**

**multiple users, retrieving and searching for, 387–388**

## N

**.NET Framework Configuration MMC, 316**

**new encryption options, 201–204**

**NIST (National Institute of Standards and Technology), 201**

**no-compile page, 135–137**

**non-ASP.NET file extensions, blocking access to, 67–68**

**Nothing permission set, 86**

## O

**OdbcPermission, customizing, 101–103**

**OleDbPermission**

allowing, 103

customizing, 100–101

**online users, tracking, 392–393**

**OnValidatingPassword method, 386**

**OOP state server, security options for, 306–307**

**OpenWebConfiguration, 157**

**operating system thread identity, establishing, 38–41**

**\$OriginHost\$, 85**

**OverrideExecuteUrlPath, 270–271**

## P

**page compilation, 314–318**

**PageAsyncTask, 71–73**

**partial trusts**

ActiveDirectoryMembershipProvider, 512–515

AllowPartiallyTrustedCallers Attribute class, 121–126

AuthorizationStoreRoleProvider, 589–592

LinkDemand, 119–121

overview, 118–119

ProcessRequestInApplicationTrust, 135–141

sandboxed assembly, 126–135

using protected configuration providers in, 182–184

**partially trusted non-ASP.NET applications, using providers in, 557–562**

**passing data to ASP from ASP.NET, 274–276**

**passing user roles to classic ASP, 277–278**

**passing username to ASP, 276**

**password. See also custom password strength rules**

formats, changing, 430–432

history, implementing, 440–451

validation, 388–389

**Password, 416**

**PasswordAnswer, 417**

**passwordAnswerAttemptLockout**

Duration, 476–477

**PasswordAttemptWindow property, 389, 476**

**PasswordFormat property, 390, 416**

**PasswordQuestion property, 372, 379, 416**

**PasswordSalt, 416**

**PasswordStrengthRegularExpression property, 384–385, 476**

**path attribute, 145–146**

**payload size, 218–220****permissions**

- demands, 93
- reading local configuration, required for, 155–157
- remote editing, required for, 159–161
- writing local configuration, required for, 157–159

**persistent tickets**

- clock resets, 196–197
- expiration enforced by forms authentication, 194–196
- overview, 192–194
- Universal Coordinate Time (UTC), 195

**PIA (primary interop assembly), 81****policy file, creating, 96–97****policyFile attributes, 84****PostAcquireRequestState event, 42****PostAuthenticateRequest event, 42, 57–58, 531–534****PostAuthorizeRequest event, 42, 65–66****PostMapRequestHandler event, 42****PostReleaseRequestState event, 42****PostRequestHandlerExecute event, 42****PostResolveRequestCache event, 42****PostUpdateRequestCache event, 42****PreRequestHandlerExecute event, 42, 65–66****primary interop assembly (PIA), 81****primary key, 394–396****PrintingPermission, 114****processing pipeline**

- AcquireRequestState event, 42
- AuthenticateRequest event, 42, 48–54
- AuthorizeRequest event, 42, 58–65
- BeginRequest event, 42
- EndRequest event, 42, 74–75
- overview, 41–42
- PostAcquireRequestState event, 42
- PostAuthenticateRequest event, 42, 57–58
- PostAuthorizeRequest event, 42, 65–66
- PostMapRequestHandler event, 42
- PostReleaseRequestState event, 42
- PostRequestHandlerExecute event, 42

## PostResolveRequestCache event, 42

## PostUpdateRequestCache event, 42

## PreRequestHandlerExecute event, 42, 65–66

## ReleaseRequestState event, 42

## ResolveRequestCache event, 42

## UpdateRequestCache event, 42

**processModel, 168****ProcessRequestInApplicationTrust, 135–141****Profile**

## Facade pattern, 341

## Strategy pattern, 333

**protected configuration**

## aspnet\_regiis, 181–182

## custom provider, redirecting configuration with a, 184–190

## defining protected configuration providers, 172

## DpapiProtectedConfiguration Provider, 172–175

## &lt;identity /&gt; section, 171

## not protected, list of configuration systems that are, 168

## overview, 166–168

## partial trust, using protected configuration providers in, 182–184

## RsaProtectedConfigurationProvider, 175–180

## selecting a protected configuration provider, 169–171

## System.Configuration.DPAPI ProtectedConfiguration Provider, 167

## System.Configuration.RsaProtected ConfigurationProvider, 167

**protecting session cookies, 293****provider configuration**

## applicationName, 475

## clientSearchTimeout, 474

## connectionProtection, 470

## directory connection settings, 468–471

## directory schema mappings, 471–474

## enablePasswordReset, 476

## enableSearchMethods, 474

## maxInvalidPasswordAttempts, 476

### provider configuration (continued)

- membership provider settings, 475–477
- minRequiredNonalphanumeric Characters, 476
- minRequiredPasswordLength, 476
- overview, 468
- passwordAnswerAttemptLockout Duration, 476–477
- passwordAttemptWindow, 476
- passwordStrengthRegularExpression, 476
- provider settings for search, 474–475
- requiresQuestionAndAnswer, 476
- requiresUniqueEmail, 475
- serverSearchTimeout, 474

### provider design, 573–576

#### provider model

- building a provider-based feature, 351–365
- core provider classes, 342–350
- Facade pattern, 341–342
- Factory Method, 334–339
- patterns found in, 332–342
- reasons for providers, 329–332
- Singleton Pattern, 339–341
- Strategy pattern, 332–334

#### Provider property, 369, 518

#### provider security

- overview, 556
- partially trusted non-ASP.NET applications, using providers in, 557–562
- trust-level requirements and configuration, 557–562

#### provider settings for search, 474–475

#### ProviderBase, 342–343

#### ProviderCollection, 344–346

#### ProviderException, 344

#### ProviderName property, 372, 379, 523

#### Providers property, 369, 518

#### ProviderSettings, 347–349

#### ProviderSettingsCollection, 349–350

#### ProviderUserKey property, 371, 379, 482

## R

### reading and writing configuration

- ConfigurationManager class, 153–154
- GetSection, 157
- GetWebApplicationSection, 157
- OpenWebConfiguration, 157
- overview, 153–155
- permissions required for reading local configuration, 155–157
- permissions required for remote editing, 159–161
- permissions required for writing local configuration, 157–159
- WebConfigurationManager class, 154–155

#### ReflectionPermission, 114

#### regenerateExpiredSessionId, 297

#### registry settings, 5

#### RegistryPermission, 115

#### ReleaseRequestState event, 42

#### RemotingConfiguration, 116

#### RemoveUserFromRole method, 520

#### RemoveUserFromRoles method, 520

#### RemoveUsersFromRole method, 520

#### RemoveUsersFromRoles method, 520, 545

#### replay attacks with cookieless tickets, 215–216

#### request validation, 310–311

#### requests, security identity for, 34–37

#### requirePermission attribute, 163–164

#### RequiresQuestionAndAnswer property, 390, 476

#### requiresSSL attribute, 204–206

#### RequiresUniqueEmail property, 385, 475

#### ResetPassword method, 391–392, 479

#### ResolveRequestCache event, 42

#### role cache cookie settings and behavior, 535–537

#### Role Manager

- Facade pattern, 341
- Membership and, 592–594
- overview, 517
- RoleManagerModule, 531–542
- RolePrincipal class, 521–531

RoleProvider class, 542–545  
 Roles class, 517–520  
 Strategy pattern, 333  
 WindowsTokenRoleProvider, 546–551

**RoleExists, 545****RoleId, 554, 555****RoleManagerModule**

cacheRolesInCookie, 535  
 cookieProtection, 536  
 cookieRequiresSSL, 536  
 cookieSlidingExpiration, 536  
 cookieTimeout, 535–536  
 createPersistentCookie, 535  
 EndRequest, 534–535  
 GetRoles, working with multiple providers  
   during, 537–542  
 maxCachedResults, 536  
 overview, 531  
 PostAuthenticateRequest, 531–534  
 role cache cookie settings and behavior,  
   535–537

**RoleName, 554****RolePrincipal class**

CachedListChanged property, 529  
 CookiePath property, 530  
 Expired property, 530  
 ExpireDate property, 530  
 GetRoles method, 523  
 Identity property, 523  
 IsInRole method, 522–523  
 IsRoleListCached property, 523  
 IssueDate property, 530  
 overview, 521–522  
 ProviderName property, 523  
 SetDirty method, 523  
 Version property, 523

**RoleProvider class**

AddUsersToRoles, 545  
 ApplicationName, 544  
 authorization methods, 544  
 basic configuration, 544  
 CreateRole, 545  
 DeleteRole, 545  
 FindUsersInRole, 545  
 GetAllRoles, 545

GetRolesForUser, 544  
 GetUsersInRole, 545  
 IsUserInRole, 544  
 managing roles and role associations,  
   544–545  
 overview, 542–544  
 RemoveUsersFromRoles, 545  
 RoleExists, 545

**Roles class**

AddUsersToRole method, 520  
 AddUsersToRoles method, 520  
 AddUserToRole method, 520  
 AddUserToRoles method, 520  
 ApplicationName property, 518  
 CacheRolesInCookie property, 518  
 DeleteRole method, 520  
 Enabled property, 518  
 GetRolesForUser method, 520  
 IsUserInRole method, 520  
 MaxCachedResults property, 518  
 overview, 517–518  
 Provider property, 518  
 Providers property, 518  
 RemoveUserFromRole method, 520  
 RemoveUserFromRoles method, 520  
 RemoveUsersFromRole method, 520  
 RemoveUsersFromRoles method, 520

**RsaProtectedConfigurationProvider**

keyContainerName, 176–179  
 overview, 175–176  
 synchronizing key containers across  
   machines, 180  
 useMachineContainer, 179

**S****SAM account names, 484–485****sample participating application in cookie-based SSO-lite, 238–241****sandboxed assembly**

ADODB access, 131–133  
 overview, 126–131  
 SqlClientPermission access, 133–135

**ScopeName, 577****scopeName, 577**

### **securing containers, 487–493**

#### **security. See also ASP.NET per-request**

##### **security**

fraudulent postbacks, 318–322

IIS per-request security, 32–33

impersonation token, 41

page compilation, 314–318

request validation, 310–311

session state, features of, 289

Site Navigation, 322–327

Universal Naming Convention (UNC), 32

viewstate protection, 311–313

#### **SecurityPermission, 115–116**

#### **selecting a protected configuration provider, 169–171**

#### **self-service password reset, configuring, 494–502**

#### **sensitive data to classic ASP, safely passing, 278–284**

#### **serialization requirements of session state, 302–304**

#### **serverSearchTimeout, 474**

#### **session data partitioning, 290–291**

#### **session ID reuse**

and expired sessions, 296–297

overview, 294

#### **session state**

cookie-based sessions, 291–294

cookieless sessions, 294–296

deserialization requirements, 302–304

DOS attacks and, 297–300

Facade pattern, 342

global assembly cache (GAC), 301

logon session compared, 287–290

OOP state server, security options for, 306–307

regenerateExpiredSessionId, 297

security features, 289

serialization requirements, 302–304

session data partitioning, 290–291

session ID reuse and expired sessions, 296–297

SQL Server session state, database security for, 304–306

Strategy pattern, 334

trust levels and, 300–304

#### **SetDirty method, 523**

#### **sets of permissions, defining, 86–87**

#### **SHA1, 198–201**

#### **sharing tickets between ASP.NET 1.1 and ASP.NET 2.0, 222–223**

#### **signed tickets, security for, 198–201**

#### **single logons, enforcing, 247–255**

#### **single logouts, enforcing, 247–248, 255–256**

#### **single sign on (SSO) products, 227–228**

#### **single user, retrieving data for, 387**

#### **Singleton Pattern, 339–341**

#### **Site Navigation**

Facade pattern, 342

security, 322–327

Strategy pattern, 334

#### **slidingExpiration attribute, 208**

#### **SmtpPermission, 117**

#### **SocketPermission, 117**

#### **SQL Server Express**

connection string, changing, 425–426

overview, 419–424

sharing issues with, 424–425

#### **SQL Server session state, database security for, 304–306**

#### **SQL server-specific provider configuration options, 555**

#### **SqlClientPermission access, 118, 133–135**

#### **SqlMembershipProvider**

account lockouts, 451–454

automatic unlocking, implementing, 454–458

custom encryption, 435–437

custom password generation, 432–435

database security, 426–428

DBO user and database schemas, 428–430

dynamic applications, support for, 458–463

enforcing custom password strength rules, 437–451

Membership database schema, 415–418

overview, 403

password formats, changing, 430–432

password history, implementing, 440–451

SQL Server Express, 419–426

**SqlRoleProvider**

- ApplicationId, 554
- aspnet\_Roles\_BasicAccess, 563
- aspnet\_Roles\_FullAccess, 563
- aspnet\_Roles\_ReportingAccess, 563
- commandTimeout, 555
- connectionStringName, 555
- data layer, authorizing roles in, 570–571
- database schema, 553–556
- database security, 563
- Description, 555
- dynamic applications, supporting, 571–572
- limited set of roles, 565–569
- LoweredRoleName, 554
- overview, 553
- provider security, 556–563
- RoleId, 554, 555
- RoleName, 554
- SQL server-specific provider configuration options, 555
- transaction behavior, 556
- UserId, 555
- Windows authentication, 563–565

**SSO (single sign on) products, 227–228****stack trace, 94–95****static content**

- dynamic content compared, 9–14
- ISAPI extension mappings, 10–13
- MIME type mappings, 9–10
- overview, 9
- wildcard application mappings, 13–14

**StopBinFiltering, 9****StopProtectedDirectoryFiltering, 9****Strategy pattern**

- Health Monitoring, 334
- Membership, 333
- overview, 332–333
- Profile, 333
- Role Manager, 333
- session state, 334
- Site Navigation, 334
- Web parts personalization, 334

**string replacements in policy files, 85–86****strongly named assemblies, 124****subsequent authenticated access, 212****supported directory architectures, 465–468****supported environments, 396–399****supported functionality, 576–577****supporting self-service password reset or retrieval, 390–392****synchronizing key containers across machines, 180****System.Configuration classes**

- overview, 347

- ProviderSettings, 347–349

- ProviderSettingsCollection, 349–350

**System.Configuration.DPAPI**

- ProtectedConfiguration**

- Provider, 167**

**System.Configuration.Provider classes**

- overview, 342

- ProviderBase, 342–343

- ProviderCollection, 344–346

- ProviderException, 344

**System.Configuration.RsaProtected**

- ConfigurationProvider, 167**

**System.Web.Configuration classes, 346****System.Web.HttpForbiddenHandler, 67****System.Web.HttpMethodNotAllowed Handler, 67****System.Web.HttpNotFoundHandler, 67****System.Web.HttpNotImplemented Handler, 67****T****technical tips for cookie-based SSO-lite, 246–247****Thread.CurrentPrincipal, 48, 54–56****3DES, 201****ticket security**

- Advanced Encryption Standard (AES), 201–204

- generating keys programmatically, 203–204

- new encryption options, 201–204

- overview, 198

- SHA1, 198–201

- signed tickets, security for, 198–201

**transaction behavior, 556**

### trust levels

- anatomy of, 83–91
- \$AppDir\$, 85
- \$AppDirUrl\$, 85
- application domain policy, 92
- ASPNet permission set, 86–87
- CAS policy, checking current, 93–94
- code, how permission sets are matched to, 88–90
- code access security, places that define, 90–91
- \$CodeGen\$, 85
- complex permissions, troubleshooting, 94–96
- configuring, 80–83
- custom trust level, 96–105
- default security permissions defined by ASPNET, 105–118
- different trust levels, working with, 81–83
- finding the trust policy file, 84
- Full trust, 78, 80
- FullTrust permission set, 86
- High trust, 79, 80
- individual permissions, defining, 87–88
- Low trust, 79, 80
- Medium trust, 79, 80
- Minimal trust, 79, 80
- Nothing permission set, 86
- \$OriginHost\$, 85
- overview, 78–80
- permission demands, 93
- policyFile attributes, 84
- requirements and configuration, 557–562
- and session state, 300–304
- sets of permissions, defining, 86–87
- setting, 17–19
- stack trace, 94–95
- steps for, 92–94
- string replacements in policy files, 85–86
- user code calls into protected framework class, 92–93

**<trust /> tag, 17–18**

## U

- unexpected redirect behavior, 221–222**
- Universal Coordinate Time (UTC), 195, 380–382**
- Universal Naming Convention (UNC), 32**
- UnlockUser method, 389, 479**
- UpdateRequestCache event, 42**
- updates, 375–380**
- UpdateUser method, 386, 392, 479**
- UPN-style usernames, 484–485**
- URL authorization, character sets affecting, 65**
- UrlAuthorizationModule, 60–65**
- URLs in pages, 216–218**
- UrlSegmentMaxCount registry setting, 5**
- UrlSegmentMaxLength registry setting, 5**
- UseCookies attribute, 209**
- UseDeviceProfile attribute, 209, 213–214**
- useMachineContainer, 179**
- useMachineProtection, 174–175**
- user code calls into protected framework class, 92–93**
- user creation, 384–386**
- user records, linking custom features to, 410–413**
- user trust level, 17**
- user updates, 384–386**
- UserData property, leveraging, 224–226**
- UserId, 416, 555**
- UserIsOnlineTimeWindow property, 369**
- UserName property, 371, 379**
- UseUri attribute, 209**
- UTC (Universal Coordinate Time), 195, 380–382**
- utility methods, 370**

## V

- ValidatePassword event, 439–440**
- ValidateUser method, 389, 392, 479**
- validating user credentials, 388–389**
- ValidatingPassword property, 385**
- Verify that File Exists setting, 268**
- Version property, 523**
- versioning provider schema, 408–410**

**views, querying common tables with, 410****viewstate protection**

- DataList data control, 313
- DetailsView data control, 313
- FormView data control, 313
- GridView data control, 313
- overview, 311–313
- ViewStateEncryptionMode attribute, 312–313

**Vlissides, John (*Design Patterns: Elements of Reusable Object-Oriented Software*), 332****W****Web parts personalization**

- Facade pattern, 341–342
- Strategy pattern, 334

**WebConfigurationManager class, 154–155****WebPermission, 103–105, 118****wildcard application mappings, 13–14****Windows authentication, 563–565****WindowsAuthenticationModule, 49–52****WindowsTokenRoleProvider, 546–551**

















