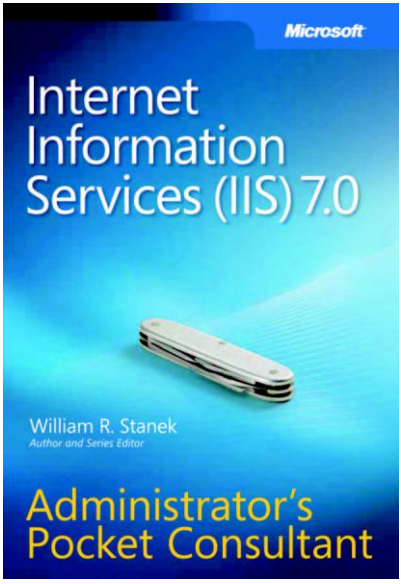

Internet Information Services (IIS) 7.0 Administrator's Pocket Consultant

William R. Stanek

To learn more about this book, visit Microsoft Learning at http://www.microsoft.com/MSPress/books/10442.aspx

9780735623644

Microsoft Press

Table of Contents

Chapter 1

IIS 7.0 Administration Overview

Let's start with the bad news right up front: Internet Information Services (IIS) 7.0 isn't what you think it is. Although IIS 7.0 *is* the latest release of Internet Information Services, it *isn't* what it seems. IIS does look a lot like its predecessors, but this is deceiving because under the surface, the architecture is completely different. So much has changed, in fact, that perhaps it might have been better if Microsoft had given IIS 7.0 a new product name. That way you'd know that IIS 7.0 was completely different from its predecessors, allowing you to start with a fresh perspective and a reasonable expectation of having to learn a whole new bag of tricks. Seasoned IIS pros also are going to have to unlearn some old tricks; and that's not only going to be difficult, it might be the single biggest obstacle to mastering IIS 7.0.

IIS 7.0 provides the core services for hosting Web servers, Web applications, and Microsoft Windows SharePoint Services. Throughout this book, I'll refer to administration of IIS, Web applications, and Windows SharePoint Services as *Microsoft Web administration* or simply *Web administration*. As you get started with Microsoft Web administration, you should concentrate on these key areas:

- What's new or changed in IIS 7.0
- How IIS 7.0 configuration schema and global configuration architecture are used
- How IIS 7.0 works with your hardware
- How IIS works with Windows-based operating systems
- Which administration tools are available
- Which administration techniques you can use to manage and maintain IIS

Working with IIS 7.0: What You Need to Know Right Now

Microsoft fully integrated Microsoft ASP.NET and the Microsoft .NET Framework into IIS 7.0. Unlike IIS 6, IIS 7.0 takes ASP.NET and the .NET Framework to the next level by integrating the ASP.NET runtime extensibility model with the core server architecture, allowing developers to fully extend the core server architecture by using ASP.NET and the .NET Framework. This tighter integration makes it possible to use existing ASP.NET features such as .NET Roles, Session Management, Output Caching, and Forms Authentication with all types of content.

IIS 7.0 has generalized the Hypertext Transfer Protocol (HTTP) process activation model that IIS 6 introduced with application pools and made it available for all protocols through an independent service called the Windows Process Activation Service, and developers can use Windows Communication Foundation (WCF) protocol adapters to take advantage of the capabilities of this service. You also should know up front that IIS 7.0 includes a metabase compatibility component that allows your existing scripts and applications to continue running but does not use a metabase to store configuration information. Instead of a metabase, IIS 7.0 uses a distributed configuration system with global and application-specific configuration files that are based on a customizable set of Extensible Markup Language (XML) schema files. These XML schema files define the configuration elements and attributes in addition to valid values for those elements and attributes, providing you precise control over exactly how you can configure and use IIS.

Microsoft built the configuration system around the concept of modules. *Modules* are standalone components that provide the core feature set of an IIS server. Microsoft ships more than 40 independent modules with IIS 7.0. Either these modules are IIS 7.0–native modules that use a Win32 DLL or IIS 7.0–managed modules that use a .NET Framework Class Library contained within an assembly. Because all server features are contained within modules, you can modify the available features easily by adding, removing, or replacing a server's modules. Further, by optimizing the installed modules based on the way an IIS server is used, you can enhance security by reducing the attack surface area and improve performance by reducing the resources required to run the core services.

> **Note** Because modules are such an important part of IIS 7.0, you'll find much discussion about them and how they are used in this book. Chapter 2, "Deploying IIS 7.0 in the Enterprise," introduces all the available native and managed modules. Chapter 5, "Managing Global IIS Configuration," details how to install and manage modules. The appendix, "Comprehensive IIS 7.0 Module and Schema Reference," provides a complete guide to using modules and schemas.

IIS 7.0 is more secure than IIS 6 because of built-in request filtering and rules-based Uniform Resource Locator (URL) authorization support. You can configure request filtering to reject suspicious requests by scanning URLs sent to a server and filtering out unwanted requests. You can configure URL authorization rules to require logon and allow or deny access to specific URLs based on user names, .NET roles, and HTTP request methods. To make it easier to resolve problems with the server and Web applications, IIS 7.0 includes new features for diagnostics, real-time request reviewing, and error reporting. These features allow you to:

- View the current running state of the server.
- Trace failed requests through the core server architecture.
- Obtain detailed error information to pinpoint the source of a problem.

IIS 7.0 has many other new and enhanced features, but few are as important as the new set of administration tools, including new graphical, command-line, and scripting administration tools. The new graphical administration tool uses a browser-like interface and adds features for delegated administration, remote administration over Secure HTTP (HTTPS), and extensibility through custom user interface components. The new command-line administration tool makes it possible to perform most configuration tasks with a single line of command text. With ASP.NET, you can manage IIS configuration through the .NET Framework by using the Microsoft.Web.Administrators application programming interface (API). With scripting, you can manage IIS configuration through the IIS 7.0 Windows Management Instrumentation (WMI) provider.

Because of the many changes, much of what you know about IIS is obsolete or irrelevant. But there's a light at the end of the tunnel—well, it's more like a freight train coming right at you—but it's there. The changes in IIS 7.0 are well worth the time and effort you'll spend learning the new architecture and the new techniques required to manage Web servers. Our dependence on ASP.NET and the .NET Framework will only grow over time, and the more you learn about the heart of the .NET architecture—IIS 7.0—the better prepared you'll be for now and for the future.

With IIS 7.0, key components that were a part of previous IIS releases are no longer available or work in different ways than they did before. Because IIS 7.0 does not use a metabase, applications designed for IIS 6 will not run on IIS 7.0 without special actions being taken. To run IIS 6 applications, you must install the IIS 6 compatibility and metabase feature. To manage IIS 6 applications and features, you must install IIS 6 Manager, IIS 6 scripting tools, and IIS 6 WMI compatibility. Additionally, IIS 7.0 does not include Post Office Protocol version 3 (POP3) or Simple Mail Transfer Protocol (SMTP) services. With IIS 7.0, you can send e-mail messages from a Web application by using the SMTP E-mail component of ASP.NET.

IIS Manager is the graphical user interface (GUI) for managing both local and remote installations of IIS 7.0. To use IIS Manager to manage an IIS server remotely, Web Management Service (WMSVC) must be installed and started on the IIS server you want to manage remotely. WMSVC is also required when IIS site or application administrators want to manage features over which they've been delegated control.

The Web Management Service provides a hostable Web core that acts as a standalone Web server for remote administration. After you install and start WMSVC on an IIS server, it listens on port 8172 on all unassigned IP addresses for four specific types of requests:

- **Login Requests** IIS Manager sends login requests to WMSVC to initiate connections. On the hostable Web core, login requests are handled by Login.axd. The authentication type is either NT LAN Manager (NTLM) or Basic, depending on what you select when you are prompted to provide credentials in the connection dialog box.

- **Code Download Requests** If login is successful, WMSVC returns a list of user interface (UI) modules for the connection. Each IIS Manager page corresponds to a specific UI module. If there's a module that IIS Manager doesn't have, it will request to download the module binaries. Code download requests are handled by Download.axd.
- **Management Service Requests** After a connection is established, your interactions with IIS Manager cause management service requests. Management service requests direct module services in WMSVC to read or write configuration data, runtime state, and providers on the server. Management service requests are handled by Service.axd.
- **Ping Requests** Ping requests are made from within the WMSVC service to the hostable Web core. Ping requests are made by Ping.axd to ensure that the hostable Web core continues to be responsive.

The Web Management Service stores a limited set of editable configuration values in the registry. Each time the service is started, the Web configuration files are regenerated in the following directory: *%SystemRoot%*\ServiceProfiles\LocalService \AppData\Local\Temp\WMSvc. To enhance security, WMSVC requires SSL (HTTPS) for all connections. This ensures that data passed between the remote IIS Manager client and WMSVC is secure. Additionally, WMSVC runs as Local Service with a reduced permission set and a locked down configuration. This ensures that only the minimal set of required modules are loaded when the hostable Web core starts. See Chapter 3, "Core IIS 7.0 Administration," for more information.

> **Note** *%SystemRoot%* refers to the SystemRoot environment variable. The Windows operating system has many environment variables, which are used to refer to user- and system-specific values. Often, I'll refer to environment variables in this book using this syntax: *%VariableName%*.

Introducing IIS 7.0 Configuration Architecture

You can use IIS 7.0 to publish information on intranets, extranets, and the Internet. Because today's Web sites use related features, such as ISAPI filters, ASP, ASP.NET, CGI, and the .NET Framework, IIS bundles these features as part of a comprehensive offering. What you need to know right now about IIS 7.0 is how IIS 7.0 uses the configuration schema and its global configuration system. In Chapter 2, you'll learn about the available setup features and the related configuration modules.

IIS 7.0 Configuration Schema

Unlike IIS 6, in which the main configuration information is stored in metabase files, IIS 7.0 has a unified configuration system for storing server, site, and application settings. You can manage these settings by using an included set of managed code, scripting APIs, and management tools. You can also manage these settings by directly

editing the configuration files themselves. Direct editing of configuration files is possible because the files use XML and are written in plain-language text files based on a predefined set of XML schema files.

> **Note** IIS 7.0 always takes the master state for configuration from the configuration files. This is a dramatic change from IIS 6, in which the master state was taken from the in-memory configuration database, which was flushed periodically to disk.

Using the XML schema to specify the configuration settings ensures that the related configuration files are well-structured XML, which is easy to modify and maintain. Because configuration values are stored using easy-to-understand text strings and values, they are easy to work with. By examining the schema itself, you can determine the exact set of acceptable values for any configuration option. IIS shares the same schema with ASP.NET configuration, ensuring that configuration settings for ASP.NET applications are just as easy to manage and maintain.

On an IIS server, schema files are stored in the *%SystemRoot%*\System32\Inetsrv\Config\Schema directory. The four standard schema files are:

- **IIS_schema.xml** This file provides the IIS configuration schema.
- **ASPNET_schema.xml** This file provides the ASP.NET configuration schema.
- **FX_schema.xml** This file provides the .NET Framework configuration schema (providing features beyond what the ASP.NET schema offers).
- **rscaext.xml** This file provides the Runtime Status and Control API (RSCA) configuration schema, providing dynamic properties for obtaining detailed runtime data.

IIS reads in the schema files automatically during startup of the application pools. The IIS schema file is the master schema file. Within the IIS schema file, you'll find configuration sections for each major feature of IIS, from application pooling to failed request tracing. The ASP.NET schema file builds on and extends the master schema with specific configuration sections for ASP.NET. Within the ASP.NET schema file, you'll find configuration sections for everything from anonymous identification to output cache settings. The FX schema file builds on and extends the ASP.NET schema file. Within the FX schema file, you'll find configuration settings for application settings, connection strings, date-time serialization, and more.

Whereas configuration sections are also grouped together for easier management, section groups do not have schema definitions. If you want to extend the configuration features and options available in IIS, you can do this by extending the XML schema. You extend the schema by following these basic steps:

1. Specify the desired configuration properties and the necessary section container in an XML schema file.
2. Place the schema file in the *%SystemRoot%*\System32\Inetsrv\Config\Schema directory.
3. Reference the new section in IIS 7.0's global configuration file.

The basic syntax for a schema file is as follows:

```
<!–
The text within this section is a comment. It is standard
practice to provide introductory details in the comments at the
beginning of the schema file.
–->
<configSchema>
    <sectionSchema name="configSection1">
    </sectionSchema>
    <sectionSchema name="configSection2">
    </sectionSchema>
    <sectionSchema name="configSection3">
    </sectionSchema>
</configSchema>
```

As an administrator or developer, you don't necessarily need to be able to read and interpret XML schemas to succeed. However, because having a basic understanding of schemas is helpful, I'll introduce the essentials. Within schema files, configuration settings are organized into sets of related features called *schema sections*. The schema for a configuration section is defined in a <sectionSchema> XML element. For example, the features related to the HTTP listener in IIS are defined with a schema section named system.applicationHost/listenerAdapters. In the IIS_schema.xml file, this section is defined as follows:

```
<sectionSchema name="system.applicationHost/listenerAdapters">
 <collection addElement="add" >
  <attribute name="name" type="string" required="true" isUniqueKey="true" />
  <attribute name="identity" type="string" />
  <attribute name="protocolManagerDll" type="string" />
  <attribute name="protocolManagerDllInitFunction" type="string" />
 </collection>
</sectionSchema>
```

This schema definition states that the system.applicationHost/listenerAdapters element can contain a collection of add elements with the following attributes:

- **name** A unique string that is a required part of the add element.
- **identity** An identity string that is an optional part of the add element.
- **protocolManagerDll** A string that identifies the protocol manager DLL.
- **protocolManagerDllInitFunction** A string that identifies the initialization function for the protocol manager DLL.

An attribute of an element is either optional or required. If the attribute definition states required="true" as with the name attribute, the attribute is required and must be provided when you are using the related element. Otherwise, the attribute is considered

optional and does not need to be provided when you are using the related element. In addition to being required, attributes can have other enforced conditions, including:

- **isUniqueKey** If set to true, the related value must be unique.
- **encrypted** If set to true, the related value is expected to be encrypted.

With some attributes, you'll see default values and possibly an enumerated list of the acceptable string values and their related internal values. In the following example, the identityType attribute has a default value of NetworkService and a list of other possible values:

```
<attribute name="identityType" type="enum" defaultValue="NetworkService">
 <enum name="LocalSystem" value="0"/>
 <enum name="LocalService" value="1"/>
 <enum name="NetworkService" value="2"/>
 <enum name="SpecificUser" value="3"/>
</attribute>
```

The friendly name of a value is provided to make the value easier to work with. The actual value used by IIS is provided in the related value definition. For example, if you set identityType to LocalService, the actual configuration value used internally by IIS is 2.

As a standard rule, you cannot use enumerated values in combination with each other. Because of this, the identityType attribute can have only one possible value. In contrast, attributes can have flags, which can be used together to form combinations of values. In the following example, the logEventOnRecycle attribute uses flags and has a default set of flags that are used in combination with each other:

```
<attribute name="logEventOnRecycle" type="flags" defaultValue="Time,
Memory, PrivateMemory">
 <flag name="Time" value="1"/>
 <flag name="Requests" value="2"/>
 <flag name="Schedule" value="4"/>
 <flag name="Memory" value="8"/>
 <flag name="IsapiUnhealthy" value="16"/>
 <flag name="OnDemand" value="32"/>
 <flag name="ConfigChange" value="64"/>
 <flag name="PrivateMemory" value="128"/>
</attribute>
```

Again, the friendly name is provided to make the value easier to work with. The actual value used by IIS is the sum of the combined flag values. With a setting of "Time, Requests, Schedule," the logEventOnRecycle attribute is set to 7 (1+2+4=7).

Attribute values can also have validation. IIS performs validation of attribute values when parsing the XML and when calling the related API. Table 1-1 provides an overview of the validators you'll see in schemas.

Table 1-1 Summary of Attribute Validation Types in an IIS XML Schema

Validation Type	Validation Parameter	Validation Fails If...
validationType= "applicationPoolName"	validationParameter=""	A validated value contains these characters: |<>&\"
validationType= "integerRange"	validationParameter= "<minimum>, <maximum>[,exclude]"	A validated value is outside [inside] range, in integers.
validationType= "nonEmptyString"	validationParameter=""	A validated value has a string value that is not set.
validationType= "siteName"	validationParameter=""	A validated value contains these characters: /\.?
validationType= "timeSpanRange"	validationParameter= "<minimum>,<maximum>, <granularity> [,exclude]"	A validated value is outside [inside] range, in seconds.
validationType= "requireTrimmedString"	validationParameter=""	A validated value has white space at start or end of value.

IIS 7.0 Global Configuration System

IIS uses a global configuration system that is difficult to understand at first but gets easier and easier to understand once you've worked with it awhile. Because there's no sense trying to ease into this, I'll dive right in. If you'll hang with me for a few pages, I'll get you through the roughest parts and zero in on exactly what you need to know—I promise.

IIS configuration settings are stored in configuration files that together set the running configuration of IIS and related components. One way to think of a configuration file is as a container for the settings you apply and their related values. You can apply multiple configuration files to a single server and the applications it is running. Generally, you manage configuration files at the .NET Framework root level, the server root level, and the various levels of a server's Web content directories. A server's Web content directories include the root directory of the server itself, the root directories of configured Web sites, and any subdirectories within Web sites. The root levels and the various levels of a server's Web content directories can be described as containers for the settings you apply and their values. If you know a bit about object-oriented programming, you might expect the concepts of parent-child relationship and inheritance to apply—and you'd be right.

Through inheritance, a setting applied at a parent level becomes the default for other levels of the configuration hierarchy. Essentially, this means that a setting applied at a parent level is passed down to a child level by default. For example, if you apply a

setting at the server root level, the setting is inherited by all Web sites on the server and by all the content directories within those sites.

The order of inheritance is as follows:

```
.NET Framework root → server root → Web Site root →
top-level directory → subdirectory
```

This means that the settings for the current .NET Framework root are passed down to IIS, the settings for IIS are passed down to Web sites, and the settings for Web sites are passed down to content directories and subdirectories. As you might expect, you can override inheritance. To do this, you specifically assign a setting for a child level that contradicts a setting for a parent. As long as overriding a setting is allowed (that is, overriding isn't blocked), the child level's setting will be applied appropriately. To learn more about overriding and blocking, see Chapter 5.

When working with the configuration files, keep the following in mind:

- The .NET Framework root IIS applies depends on the current running version of ASP.NET and the .NET Framework. The default configuration files for the .NET Framework root are Machine.config and Web.config, which are stored in the *%SystemRoot%*\Microsoft.net\Framework*Version*\Config\Machine.config directory. Machine.config sets the global defaults for the .NET Framework settings in addition to some ASP.NET settings. Web.config sets the rest of the global defaults for ASP.NET. See Chapter 8, "Running IIS Applications," and Chapter 9, "Managing Applications, Application Pools, and Worker Processes," for more information about configuring the .NET Framework and ASP.NET.
- The default configuration file for the server root is ApplicationHost.config, which is stored in the *%SystemRoot%*\System32\Inetsrv\Config directory. This file controls the global configuration of IIS. See Chapter 5 for more information about configuring IIS servers.
- The default configuration file for a Web site root is Web.config. This file is stored in the root directory of the Web site to which it applies and controls the behavior for the Web site. See Chapters 8 and 9 for more information about configuring IIS applications.
- The default configuration file for a top-level content directory or a content subdirectory is Web.config. This file is stored in the content directory to which it applies and controls the behavior of that level of the content hierarchy and downwards. See Chapter 6 for more information about configuring content directories.

In some cases, you may want a .config file to include some other .config file. This can be done by using the configSource attribute to refer to the .config file containing the settings you want to use. Currently, the referenced .config file must reside in the same

directory as the original .config file. Note that this behavior may change to allow .config files in other directories to be used. To see how this works, consider the following example from the ApplicationHost.config file:

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- applicationHost.config -->
<configuration>
 <system.webServer>
  <httpErrors>
   <error statusCode="401" prefixLanguageFilePath="%SystemDrive%\
inetpub\custerr" path="401.htm" />
   <error statusCode="403" prefixLanguageFilePath="%SystemDrive%\
inetpub\custerr" path="403.htm" />
   <error statusCode="404" prefixLanguageFilePath="%SystemDrive%\
inetpub\custerr" path="404.htm" />
   <error statusCode="405" prefixLanguageFilePath="%SystemDrive%\
inetpub\custerr" path="405.htm" />
   <error statusCode="406" prefixLanguageFilePath="%SystemDrive%\
inetpub\custerr" path="406.htm" />
   <error statusCode="412" prefixLanguageFilePath="%SystemDrive%\
inetpub\custerr" path="412.htm" />
   <error statusCode="500" prefixLanguageFilePath="%SystemDrive%\
inetpub\custerr" path="500.htm" />
   <error statusCode="501" prefixLanguageFilePath="%SystemDrive%\
inetpub\custerr" path="501.htm" />
   <error statusCode="502" prefixLanguageFilePath="%SystemDrive%\
inetpub\custerr" path="502.htm" />
  </httpErrors>
 </system.webServer>
</configuration>
```

In this example, error elements specify how certain types of HTTP error status codes should be handled. If you wanted to customize the error handling for a server, you might want to extend or modify the default values in a separate .config file and then reference the .config file in ApplicationHost.config. To do this, you would update the ApplicationHost.config file to point to the additional .config file. An example follows.

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- applicationHost.config -->
<configuration>
 <system.webServer>
  <httpErrors configSource=errorMode.config />
</configuration>
```

You would then create the errorMode.config file and store it in the same directory as the ApplicationHost.config file. The following is an example of the contents of the errorMode.config file:

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- errorMode.config -->
```

```
<configuration>
 <system.webServer>
  <httpErrors>
  <error statusCode="401" prefixLanguageFilePath="%SystemDrive%\inetpub\
custerr" path="401.htm" />
  <error statusCode="403" prefixLanguageFilePath="%SystemDrive%\inetpub\
custerr" path="403.htm" />
  <error statusCode="404" prefixLanguageFilePath="%SystemDrive%\inetpub\
custerr" path="404.htm" />
  <error statusCode="405" prefixLanguageFilePath="%SystemDrive%\inetpub\
custerr" path="405.htm" />
  <error statusCode="406" prefixLanguageFilePath="%SystemDrive%\inetpub\
custerr" path="406.htm" />
  <error statusCode="412" prefixLanguageFilePath="%SystemDrive%\inetpub\
custerr" path="412.htm" />
  <error statusCode="500" prefixLanguageFilePath="%SystemDrive%\inetpub\
custerr" path="500.htm" />
  <error statusCode="501" prefixLanguageFilePath="%SystemDrive%\inetpub\
custerr" path="501.htm" />
  <error statusCode="502" prefixLanguageFilePath="%SystemDrive%\inetpub\
custerr" path="502.htm" />
   </httpErrors>
  </system.webServer>
</configuration>
```

When you make these or other types of changes in configuration files, you don't need to worry about restarting IIS or related services. IIS automatically picks up the changes and uses them. In these examples, you'll note that we're working with the system.webServer section of the configuration file. As per the schema definition files, all settings are defined within specific configuration sections. Although sections cannot be nested, a section can exist within a section group, and that section group can in turn be contained in a parent section group. A section group is simply a container of logically related sections.

In ApplicationHost.config, section groups and individual sections are defined in the configSections element. The configSections element controls the registration of sections. Every section belongs to one section group. By default, ApplicationHost.config contains these section groups:

- **system.applicationHost** Defines the following sections: applicationPools, configHistory, customMetadata, listenerAdapters, log, sites, and webLimits.
- **system.webServer** Defines the following sections: asp, caching, cgi, defaultDocument, directoryBrowse, globalModules, handlers, httpCompression, httpErrors, httpLogging, httpProtocol, httpRedirect, httpTracing, isapiFilters, modules, odbcLogging, serverRuntime, serverSideInclude, staticContent, urlCompression, and validation. Includes the security and tracing subgroups.

- **system.webServer.security** A subgroup of system.webServer that defines the following sections: access, applicationDependencies, authorization, ipSecurity, isapiCgiRestriction, and requestFiltering. Includes the authentication subgroup.
- **system.webServer.security.authentication** A subgroup of system.webServer .security that defines the following sections: anonymousAuthentication, basicAuthentication, clientCertificateMappingAuthentication, digestAuthentication, iisClientCertificateMappingAuthentication, and windowsAuthentication.
- **system.webServer.security.tracing** A subgroup of system.webServer.security that defines the traceFailedRequests and traceProviderDefinitions sections.

In ApplicationHost.config, section groups and individual sections are defined as follows:

```
<configSections>
 <sectionGroup name="system.applicationHost">
  <section name="applicationPools" allowDefinition="AppHostOnly"
overrideModeDefault="Deny" />
  <section name="configHistory" allowDefinition="AppHostOnly"
overrideModeDefault="Deny" />
  <section name="customMetadata" allowDefinition="AppHostOnly"
overrideModeDefault="Deny" />
  <section name="listenerAdapters" allowDefinition="AppHostOnly"
overrideModeDefault="Deny" />
  <section name="log" allowDefinition="AppHostOnly"
overrideModeDefault="Deny" />
  <section name="sites" allowDefinition="AppHostOnly"
overrideModeDefault="Deny" />
  <section name="webLimits" allowDefinition="AppHostOnly"
overrideModeDefault="Deny" />
  </sectionGroup>
 <sectionGroup name="system.webServer">
  …
 </sectionGroup>
</configSections>
```

In Machine.config, you'll also find definitions for section groups and individual sections. These are similar to those used in ApplicationHost.config but are used for configuring the .NET Framework and some ASP.NET settings. When working with either .config file, keep in mind that a section is the basic unit of deployment, locking, searching, and containment for configuration settings. Every section has a name attribute and optional allowDefinition and overrideModeDefault attributes. The name attribute sets the unique section name. The allowDefinition attribute specifies the level at which the section can be set:

- **Everywhere** The section can be set in any configuration file including directories mapped to virtual directories that are not application roots, and their subdirectories.
- **MachineOnly** The section can be set only in ApplicationHost.config or Machine.config. Because this is the default setting, a section that doesn't have an allowDefinition attribute uses this setting automatically.

- **MachineToWebRoot** The section can be set only in the .NET Framework root's Machine.config or Web.config file, or in ApplicationHost.config.
- **MachineToApplication** The section can be set only in the .NET Framework root's Machine.config or Web.config file, in ApplicationHost.config, or in Web.config files for application root directories.
- **AppHostOnly** The section can be set only in Web.config files for application root directories.

The OverrideModeDefault attribute sets the default lockdown state of a section. Essentially, this means that it controls whether a section is locked down to the level in which it is defined or can be overridden by lower levels of the configuration hierarchy. If this attribute is not set, the default value is Allow. With Allow, lower level configuration files can override the settings of the related section. With Deny, lower level configuration files cannot override the settings of the related section. As discussed in Chapter 5, you'll typically use location tags to lock or unlock sections for specific Web sites or applications.

Because the complete configuration settings of a server and its related sites and applications are stored in the configuration files, you easily can back up or duplicate a server's configuration. Backing up a server's configuration is a simple matter of creating a copy of the configuration files. Similarly, duplicating a server's configuration on another server is a simple matter of copying the source configuration files to the correct locations on another server.

IIS 7.0 and Your Hardware

Before you deploy IIS 7.0, you should carefully plan the server architecture. As part of your planning, you need to look closely at pre-installation requirements and the hardware you will use. IIS 7.0 is no longer the simple solution for hosting Web sites that it once was. It now provides the core infrastructure for hosting Web servers, Web applications, and Windows SharePoint Services.

Guidelines for choosing hardware for Internet servers are much different from those for choosing other types of servers. A Web hosting provider might host multiple sites on the same computer and might also have service level agreements that determine the level of availability and performance required. On the other hand, a busy e-commerce site might have a dedicated Web server or even multiple load-balanced servers. Given that Internet servers are used in a wide variety of circumstances and might be either shared or dedicated, here are some guidelines for choosing server hardware:

- **Memory** The amount of random access memory (RAM) that's required depends on many factors, including the requirements of other services, the size of frequently accessed content files, and the RAM requirements of the Web applications. In most installations, I recommend that you use at least 1 gigabyte (GB) of RAM. High-volume servers should have a minimum of 2 to 4 GB of RAM. More

RAM will allow more files to be cached, reducing disk requests. For all IIS installations, the operating system paging file size should at least equal the amount of RAM on the server.

> **Note** Don't forget that as you add physical memory, virtual paging to disk grows as well. With this in mind, you might want to ensure that the Pagefile.sys file is on the appropriate disk drive, one that has adequate space for the page file to grow, along with providing optimal input/output (I/O) performance.

> **More Info** For detailed information on memory management and performance tuning, see Chapter 12, "Performance Tuning, Monitoring, and Tracing."

- **CPU** The CPU processes the instructions received by the computer. The clock speed of the CPU and the size of the data bus determine how quickly information moves among the CPU, RAM, and system buses. Static content, such as HTML and images, place very little burden on the processor, and standard recommended configurations should suffice. Faster clock speeds and multiple processors increase the performance scalability of a Web server, particularly for sites that rely on dynamic content. 32-bit versions of Windows run on Intel x86 or compatible hardware. 64-bit versions of Windows run on the x64 family of processors from AMD and Intel, including AMD64 and Intel Extended Memory 64 Technology (Intel EM64T). IIS provides solid benchmark performance on Intel Xeon, AMD Opteron, and AMD Athlon processors. Any of these CPUs provide good starting points for the typical IIS server. You can achieve significant performance improvements with a large processor cache. Look closely at the L1, L2, and L3 cache options available—a larger cache can yield much better performance overall.
- **SMP** IIS supports symmetric multiprocessors (SMPs) and can use additional processors to improve performance. If the system is running only IIS and doesn't handle dynamic content or encryption, a single processor might suffice. You should always use multiple processors if IIS is running alongside other services, such as Microsoft SQL Server or Microsoft Exchange Server.
- **Disk drives** The amount of data storage capacity you need depends entirely on the size of content files and the number of sites supported. You need enough disk space to store all your data plus workspace, system files, and virtual memory. I/O throughput is just as important as drive capacity. However, disk I/O is rarely a bottleneck for Web sites on the public Internet—generally, bandwidth limits throughput. High-bandwidth sites should consider hardware-based redundant array of independent disks (RAID) solutions using copper or fiber channel–based small computer system interface (SCSI) devices.

- **Data protection** Unless you can tolerate hours of downtime, you should add protection against unexpected drive failures by using RAID. Hardware RAID implementations are always preferred over software RAID implementations. RAID 0 (disk striping without parity) offers optimal read/write performance, but if a drive fails, IIS won't be able to continue operation until the drive is replaced and its contents are restored from backup. Because of this, RAID 0 isn't the recommended choice. RAID 1 (disk mirroring) creates duplicate copies of data on separate physical drives, allowing the server to remain operational when a drive fails, and even while the RAID controller rebuilds a replacement drive in a failed mirror. RAID 5 (disk striping with parity) offers good protection against single-drive failure but has poor write performance. Keep in mind that if you've configured redundant load-balanced servers, you might not need RAID. With load balancing, the additional servers might offer the necessary fault tolerance.
- **UPS** Sudden power loss and power spikes can seriously damage hardware. To prevent this, get an uninterruptible power supply (UPS). A properly configured UPS system allows the operating system to automatically shut down the server gracefully in the event of a power outage, and it's also important in maintaining system integrity when the server uses write-back caching controllers that do not have on-board battery backups. Professional hosting providers often offer UPS systems that can maintain power indefinitely during extended power outages.

If you follow these hardware guidelines, you'll be well on your way to success with IIS.

IIS 7.0 Editions and Windows

IIS 7.0 is available for both desktop and server editions of Windows. On Windows Vista, IIS 7.0 offers Web administrators and Web developers a complete platform for building and testing dynamic Web sites and Web applications. IIS 7.0 running on Windows Vista also enables process activation, process management, and the necessary HTTP infrastructure for creating WCF–based applications.

As discussed further in Chapter 2, the way IIS 7.0 works on Windows Vista depends on the edition of Windows Vista you are using. On Windows Vista Starter and Home Basic editions, IIS 7.0 cannot be used to host Web sites, Web applications, or Windows SharePoint Services. On these editions, a limited set of IIS features are available, such as Windows Activation Service components that are used to enable WCF-based applications. Users who install WCF-based applications will not need to install these components. The necessary components are installed automatically by WCF. With these editions, the simultaneous request execution limit for IIS is three, meaning that an application or a group of running applications could make up to three simultaneous requests for Web content through the installed IIS components.

On Windows Vista Home Premium, most of the IIS 7.0 features required for Web site development are available. The available features should allow most casual or hobbyist administrators and developers to build and test dynamic Web sites and Web applications. Many advanced features are missing, however, including advanced authentication components, advanced logging components, and FTP server components. As with Starter and Home Basic editions of Windows Vista, the simultaneous request execution limit for IIS is three for Windows Vista Home Premium, meaning you or running applications could make up to three simultaneous requests for Web content through the installed IIS components.

For Windows Vista Business, Enterprise, and Ultimate editions, all IIS 7.0 features are available. This means that professional Web administrators and Web developers have everything necessary to design, build, and test Web sites and Web applications. The simultaneous request execution limit is ten for these editions of Windows Vista, meaning you or running applications could make up to ten simultaneous requests for Web content through the installed IIS components.

With server editions of Windows, you can use IIS to host Web servers, Web applications, and Windows SharePoint Services. All features of IIS 7.0 are available on all editions of Windows Server 2008. On Windows Server operating systems, IIS 7.0 has no request execution limit. This means that an unlimited number of simultaneous requests can be made to the IIS 7.0 server core.

Web Administration Tools and Techniques

Web administrators will find that there are many ways to manage Web and application servers. The key administration tools and techniques are covered in the following sections.

Managing Resources by Using Key Administration Tools

Many tools are available for managing Web resources. Key tools you'll use are shown in Table 1-2. Most of these tools are available on the Administrative Tools menu. Click Start and choose All Programs, Administrative Tools, and then the tool you want to use. You can use all the tools listed in the table to manage local and remote resources. For example, if you connect to a new computer in IIS Manager, you can manage all its sites and services remotely from your system.

Table 1-2 Quick Reference for Key Web Administration Tools

Administration Tool	Purpose
Active Directory Users and Computers	Manages domain user, group, and computer accounts.
Computer Management	Manages services, storage, and applications. The Services And Applications node provides quick access to Indexing Service catalogs and IIS sites and servers.

Table 1-2 Quick Reference for Key Web Administration Tools

Administration Tool	Purpose
Data Sources (ODBC)	Configures and manages Open Database Connectivity (ODBC) data sources and drivers. Data sources link Web front ends with database back ends.
DNS	Public Internet sites must have fully qualified domain names (FQDNs) to resolve properly in browsers. Use the Domain Name System (DNS) administrative snap-in to manage the DNS configuration of your Windows DNS servers.
Event Viewer	Allows you to view and manages events and system logs. If you keep track of system events, you'll know when problems occur.
Internet Information Services (IIS) 6.0 Manager	Manages Web and application server resources that were designed for IIS 6. This tool is included for backward compatibility only.
Internet Information Services (IIS) Manager	Manages Web and application server resources that were designed for IIS 7.0.
Web Management Service (WMSVC)	Allows you to use the IIS Manager to manage Web and application server resources on remote servers.
Reliability and Performance Monitor	Tracks system reliability and performance allowing you to pinpoint performance problems.
Services	Views service information, starts and stops system services, and configures service logons and automated recoveries.

When you add services to a server, the tools needed to manage those services are automatically installed. If you want to manage these servers remotely, you might not have these tools installed on your workstation. In that case, you need to install the administration tools on the workstation you're using.

Web Administration Techniques

Web administrators have many options for managing IIS. The key administration tools are:

- IIS Manager (InetMgr.exe)
- IIS Administration objects made available through the IIS 7.0 WMI provider
- IIS command-line administration tool (AppCmd.exe)

IIS Manager provides the standard administration interface for IIS. To start IIS Manager, click Start and choose All Programs, Administrative Tools, and then Internet Information Services (IIS) Manager. When started, IIS Manager displays the Start page

shown in Figure 1-1 and automatically connects to the local IIS installation, if it's available. On the Start page, you have the following options:

- **Connect to localhost** Connects you to the IIS installation on the local computer
- **Connect to a server** Allows you to connect to a remote server
- **Connect to a site** Allows you to connect to a specific Web site on a designated Web server
- **Connect to an application** Allows you to connect to a specific Web application on a designated site and server

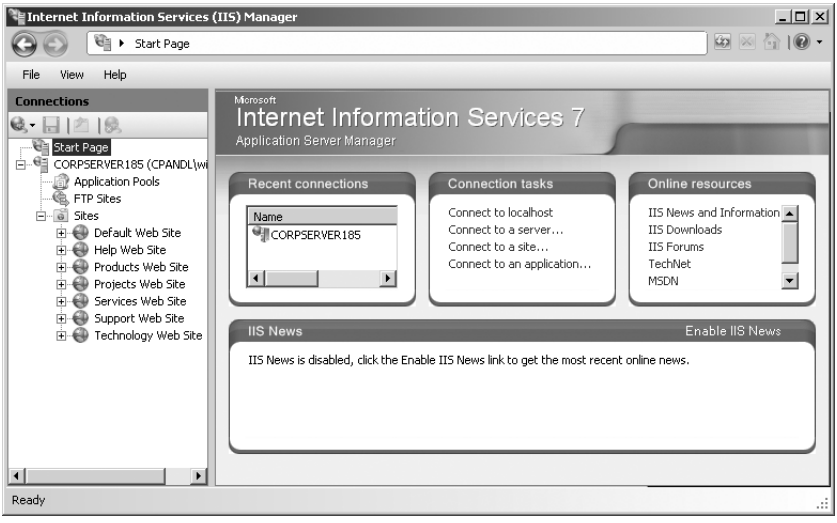

Figure 1-1 You can access servers, sites, and applications by using IIS Manager.

As discussed previously, remote access to an IIS server is controlled by the WMSVC. When you install and start WMSVC on an IIS server, it listens on port 8172 on all unassigned IP addresses and allows remote connections from authorized user accounts. You can connect to a remote server by following these steps:

1. In Internet Information Services (IIS) Manager, click Start Page in the console tree and then click Connect To A Server. This starts the Connect To A Server wizard.
2. Type or select the server name in the Server Name box. For a server on the Internet, type the FQDN of the server, such as www.adatum.com. For a server on the local network, type the computer name, such as WEBSVR87. Port 80 is the default port for connections. As necessary, you can provide the port to which you

want to connect. For example, if you want to connect to the server on port 8080, you would follow the server name by :8080, such as WEBSVR87:8080.

3. After you type the server name (and optionally the port number), click Next. IIS Manager will then try to use your current user credentials to log on to the server. If this fails, you'll need to provide the appropriate credentials on the presented Provide Credentials page before clicking Next to continue. Click Finish to complete the connection.

Tip If IIS Manager displays a connection error stating that the remote server is not accepting connections, you'll need to log on locally or through remote desktop. Once logged on, check to ensure the Management Service is started and configured properly. For more information, see the "Enabling and Configuring Remote Administration" section of Chapter 3.

You can connect to a specific Web site on a designated server by following these steps:

1. In Internet Information Services (IIS) Manager, click Start Page in the console tree and then click Connect To A Site. This starts the Connect To A Site Wizard.
2. Type or select the server name in the Server Name box, such as TESTSVR22. In the Site Name box, type or select the name of the Web site to which you want to connect, such as Default Web Site.
3. Click Next. IIS Manager will then try to use your current user credentials to log on to the server. If this fails, you'll need to provide the appropriate credentials on the presented Provide Credentials page before clicking Next to continue. Click Finish to complete the connection.

You can connect to a specific application on a designated site and server by following these steps:

1. In Internet Information Services (IIS) Manager, click Start Page in the console tree and then click Connect To An Application. This starts the Connect To An Application Wizard.
2. Type or select the server name in the Server Name box, such as TESTSVR22. In the Site Name box, type or select the name of the Web site to which you want to connect, such as Default Web Site.
3. In the Application Name box, type or select the relative path of the Web application to which you want to connect, such as /MyApplication or /Apps/Myapp.
4. Click Next. IIS Manager will then try to use your current user credentials to log on to the server. If this fails, you'll need to provide the appropriate credentials on the presented Provide Credentials page before clicking Next to continue. Click Finish to complete the connection.

As Figure 1-2 shows, IIS Manager has been completely redesigned for IIS 7.0. Instead of being a snap-in for the Microsoft Management Console, IIS Manager is now a standalone application with a browser-like interface. Once you connect to a server, site, or application, IIS Manager automatically connects to these installations upon startup. You can change this behavior by disconnecting from the remote server while in IIS Manager. See Chapter 3 for more information on using IIS Manager.

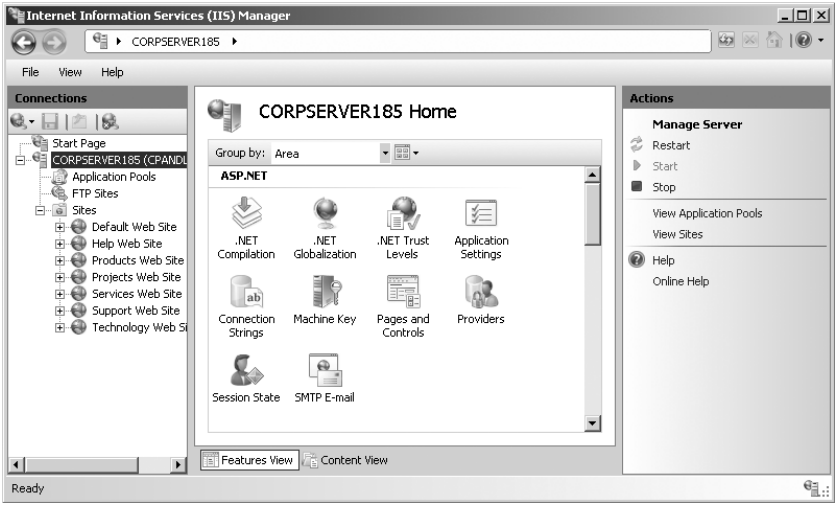

Figure 1-2 IIS Manager has a completely redesigned interface in IIS 7.0.

IIS 7.0 introduces the concept of delegated administration. With *delegated administration*, a machine administrator can delegate administrative control safely and securely. Delegated administration allows different levels of the configuration hierarchy to be managed by other users, such as site administrators or application developers. In a standard configuration, the default delegation state limits write access to most configuration settings to machine administrators only, and you must explicitly modify the delegation settings to grant write access to others. You'll learn more about IIS security and delegation in Chapter 10, "Managing Web Server Security."

IIS Manager and other graphical tools provide just about everything you need to work with IIS 7.0. Still, there are times when you might want to work from the command line, especially if you want to automate installation or administration tasks. To help you with all your command-line needs, IIS 7.0 includes the IIS command-line administration tool (AppCmd.exe). AppCmd.exe is located in the *%SystemRoot%*\System32\Inetsrv directory. By default, this directory is not in your command path. Because of this, you'll need either to add this directory to the default path or change to this directory each time you want to use this tool. Add this directory temporarily to your default path by typing the following at an elevated command prompt:

```
path %PATH%;%SystemRoot%\System32\inetsrv
```

Then add this directory permanently to your default path by typing the following at an elevated command prompt:

```
setx PATH %PATH%;%SystemRoot%\System32\inetsrv
```

Note You use Path to temporarily update the command path for the current window. You use SETX PATH to permanently update the command path for future command windows.

Table 1-3 provides a summary of the core set of administration objects for the IIS command-line administration tool.

Table 1-3 Administration Objects for the IIS Command-Line Administration Tool

Object Type	Description	Related Commands
APP	Allows you to create and manage Web application settings by using related list, set, add, and delete commands	list, set, add, and delete
APPPOOL	Allows you to create and manage application pools by using related list, set, add, delete, start, stop, and recycle commands	list, set, add, delete, start, stop, and recycle
BACKUP	Allows you to create and manage backups of your server configuration by using list, add, delete, and restore commands	list, add, delete, and restore
CONFIG	Allows you to manage general configuration settings by using related list, set, search, lock, unlock, clear, reset, and migrate commands	list, set, search, lock, unlock, clear, reset, and migrate
MODULE	Allows you to manage IIS modules by using related list, set, add, delete, install, and uninstall commands	list, set, add, delete, install, and uninstall
REQUEST	Allows you to list current HTTP requests by using a related list command	list
SITE	Allows you to create and manage virtual sites by using related list, set, add, delete, start, and stop commands	list, set, add, delete, start, and stop
TRACE	Allows you to manage failed request tracing by using related list, configure, and inspect commands	list, configure, and inspect
VDIR	Allows you to create and manage virtual directory settings by using related list, set, add, and delete commands	list, set, add, and delete
WP	Allows you to list running worker processes by using a related list command	list

The basics of working with the IIS command-line administration tool are straightforward. Most administration objects support these basic commands:

- **ADD** Creates a new object with the properties you specify.
- **DELETE** Deletes the object you specify.
- **LIST** Displays a list of related objects. Optionally, you can specify a unique object to list, or you can type one or more parameters to match against object properties.
- **SET** Sets parameters on the object specified.

Some objects support other commands, including:

- **RECYCLE** Recycles the object you specify by deleting it and then re-creating it
- **START** Starts the object you specify if it is stopped
- **STOP** Stops the object you specify if it is started or otherwise active

To type commands, use the following basic syntax:

```
appcmd Command <Object-type>
```

where *Command* is the action to perform, such as list, add, or delete, and Object-type is the object on which you want to perform the action, such as app, site, or vdir. Following this, if you wanted to list the configured sites on a server, you could type the following command at an elevated command prompt:

```
appcmd list site
```

Because the IIS command-line administration tool will also accept plural forms of object names, such as apps, sites, or vdirs, you could also use:

```
appcmd list sites
```

In either case, the resulting output is a list of all configured sites on the server with their related properties, such as:

```
SITE "Default Web Site" (id:1,bindings:http/*:80:,state:Started)
```

You'll find a comprehensive discussion of using the IIS command-line administration tool in Chapter 4, "Managing IIS 7.0 from the Command Line." In addition, you will see examples of using this tool throughout the book.

Chapter 6

Configuring Web Sites and Directories

Tasks for creating and managing Web sites and directories are broken down into several categories. You'll find sections in this chapter on Web site naming and identification, creating Web sites, creating virtual directories, and other topics.

Web Site Naming and Identification

Each Web site deployed in the organization has unique characteristics. Different types of Web sites can have different characteristics. Intranet Web sites typically use computer names that resolve locally and have private Internet Protocol (IP) addresses. Internet Web sites typically use fully qualified domain names (FQDNs) and public IP addresses. Intranet and Internet Web sites can also use host header names, allowing single IP address and port assignments to serve multiple Web sites.

Understanding IP Addresses and Name Resolution

Whether you're configuring an intranet or Internet site, your Web server must be assigned a unique IP address that identifies the computer on the network. An IP address is a numeric identifier for the computer. IP addressing schemes vary depending on how your network is configured, but they're normally assigned from a range of addresses for a particular network segment (also known as a *subnet*). For example, if you're working with a computer on the network segment 192.168.10.0, the address range you have available for computers is usually from 192.168.10.1 to 192.168.10.254.

Although numeric addresses are easy for machines to remember, they aren't easy for human beings to remember. Because of this, computers are assigned text names that are easy for users to remember. Text names have two basic forms:

- Standard computer names, which are used on private networks
- Internet names, which are used on public networks

Private networks are networks that are either indirectly connected to the Internet or completely disconnected from the Internet. Private networks use IP addresses that are reserved for private use and aren't accessible to the public Internet. Private network addresses fall into the following ranges:

- 10.0.0.0–10.255.255.255
- 172.16.0.0–172.31.255.255
- 192.168.0.0–192.168.255.255

Private networks that use Internet technologies are called *intranets*. Information is delivered on intranets by mapping a computer's IP address to its text name, which is the NetBIOS name assigned to the computer. Although Microsoft Windows components use the NetBIOS naming convention for name resolution, Transmission Control Protocol/Internet Protocol (TCP/IP) components use the Domain Name System (DNS). Under Windows, the DNS host name defaults to the same name as the NetBIOS computer name. For example, if you install a server with a computer name of CorpServer, this name is assigned as the NetBIOS computer name and the default DNS host name.

In contrast, public networks are networks that are connected directly to the Internet. Public networks use IP addresses that are purchased or leased for public use. Typically, you'll obtain IP address assignments for your public servers from the provider of your organization's Internet services. Internet service providers (ISPs) obtain blocks of IP addresses from the American Registry for Internet Numbers (ARIN). Other types of organizations also can purchase blocks of IP addresses.

On the Internet, DNS is used to resolve text names to IP addresses. With the DNS name *www.microsoft.com*, *www* identifies a server name and *microsoft.com* identifies a domain name. As with public IP addresses, domain names must be leased or purchased. You purchase domain names from name registrars, such as Internet Network Information Center (InterNIC). When a client computer requests a connection to a site by using a domain name, the request is transmitted to a DNS server. The DNS server returns the IP address that corresponds to the requested host name, and then the client request is routed to the appropriate site.

Don't confuse the public DNS naming system used on the Internet with the private naming system used on intranets. DNS names are configured on DNS servers and resolved to IP addresses before contacting a server. This fact makes it possible for a server to have multiple IP addresses, each with a different DNS name. For example, a server with an internal computer name of WebServer22 could be configured with IP addresses of 207.46.230.210, 207.46.230.211, and 207.46.230.212. If these IP addresses are configured as *www.microsoft.com*, *services.microsoft.com*, and *products.microsoft.com*, respectively, in the DNS server, the server can respond to requests for each of these domain names.

Understanding Web Site Identifiers

Each Web site deployed in your organization has a unique identity it uses to receive and to respond to requests. The identity includes the following:

- A computer or DNS name
- An IP address
- A port number
- An optional host header name

The way these identifiers are combined to identify a Web site depends on whether the host server is on a private or public network. On a private network, a computer called CorpIntranet could have an IP address of 10.0.0.52. If so, the Web site on the server could be accessed in the following ways:

- Using the Universal Naming Convention (UNC) path name: \\CorpIntranet or \\10.0.0.52
- Using a Uniform Resource Locator (URL): *http://CorpIntranet/* or *http://10.0.0.52/*
- Using a URL and port number: *http://CorpIntranet:80/* or *http://10.0.0.52:80/*

On a public network, a computer called Dingo could be registered to use the DNS name *www.microsoft.com* and the IP address of 207.46.230.210. If so, the Web site on the server could be accessed in either of the following ways:

- Using a URL: *http://www.microsoft.com/* or *http://207.46.230.210/*
- Using a URL and port number: *http:// www.microsoft.com:80/* or *http://207.46.230.210:80/*

Hosting Multiple Sites on a Single Server

Using different combinations of IP addresses, port numbers, and host header names, one can host multiple sites on a single computer. Hosting multiple sites on a single server has definite advantages. For example, rather than installing three different Web servers, one could host *www.microsoft.com*, *support.microsoft.com*, and *service.microsoft.com* on the same Web server.

One way to host multiple sites on the same server is to assign multiple IP addresses to the server. Figure 6-1 shows an example of this configuration.

To use this technique, you must follow these steps:

1. Configure the TCP/IP settings on the server so that there is one IP address for each site that you want to host.
2. Configure DNS so that the host names and corresponding IP addresses can be resolved.
3. Configure each Web site so that it uses a specific IP address.

With this technique, users can access the sites individually by typing the unique domain name or IP address in a browser. Following the example shown in Figure 6-1, you can access the Sales intranet by typing **http://SalesIntranet/** or **http://10.0.0.102/**.

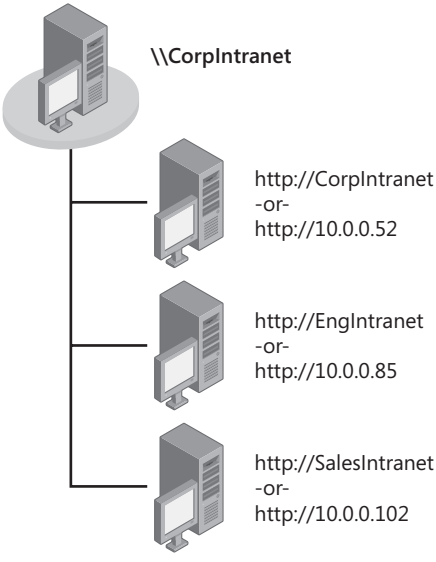

Figure 6-1 You can use multiple IP addresses to host multiple Web sites on a single server.

Another technique you can use to host multiple sites on a single server is to assign each site a unique port number while keeping the same IP address, as shown in Figure 6-2. Users will then be able to do the following:

- Access the main site by typing the DNS server name or IP address in a browser, such as **http://Intranet/** or **http://10.0.0.52/**.
- Access other Web sites by typing the domain name and port assignment or IP address and port assignment, such as **http://Intranet:88/** or **http://10.0.0.52:88/**.

The final method you can use to host multiple sites on a single server is to use host header names. Host headers allow you to host multiple sites on the same IP address and port number. The key to host headers is a DNS name assignment that's configured in DNS and assigned to the site in its configuration.

An example of host header assignment is shown in Figure 6-3. Here, a single server hosts the sites CorpIntranet, EngIntranet, and SalesIntranet. The three sites use the same IP address and port number assignment but have different DNS names.

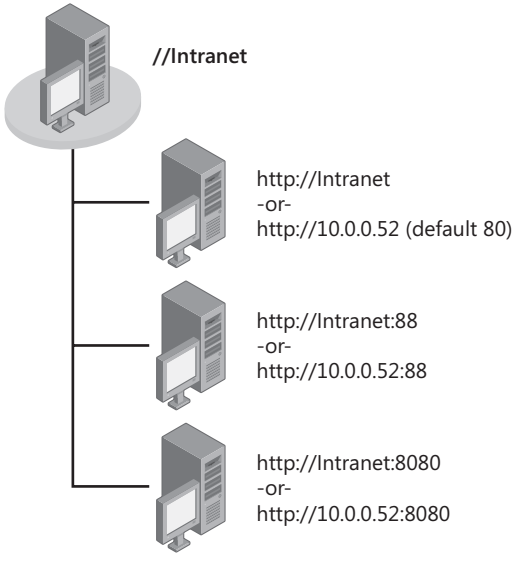

Figure 6-2 Another technique is to use multiple port numbers to host multiple Web sites on a single server.

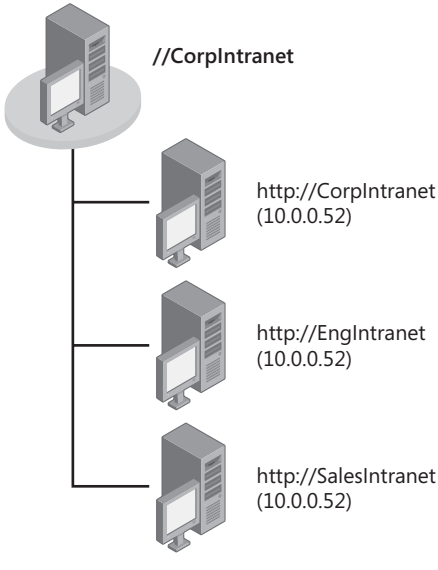

Figure 6-3 You can use host headers to support multiple Web sites on a single server with a single IP address.

To use host headers, you must do the following:

1. Configure DNS so that the host header names and corresponding IP addresses can be resolved.
2. Configure the primary Web site so that it responds to requests on the IP address and port number you've assigned.
3. Configure additional Web sites so that they use the same IP address and port number and also assign a host header name.

Using different IP addresses or different port numbers for each site ensures the widest compatibility because any Web browser can access the related sites without problems. However, as public IP addresses are valuable (and sometimes costly) resources, and non-standard ports require users to type the nonstandard port number, host headers are the most commonly used technique.

Host headers have specific drawbacks. Earlier versions of browsers that don't support Hypertext Transfer Protocol (HTTP) 1.1 are unable to pass host header names back to Internet Information Services (IIS). Although Microsoft Internet Explorer 3, Netscape Navigator 2, and later versions of these browsers support the use of host header names, earlier versions of these browsers don't, and visitors using earlier browsers will reach the default Web site for the IP address. After you configure host headers, you must also register the host header names you've used with DNS to ensure that the names are properly resolved.

Checking the Computer Name and IP Address of Servers

Before you configure Web sites, you should check the server's computer name and IP address. You can view the computer name by completing the following steps:

1. Click Start, and then click Control Panel. In the Control Panel's Classic View, double-click System. In the System console, under Computer Name, Domain, And Workgroup Settings, click Change Settings. Alternatively, you can click Advanced System Settings in the left pane.
2. On the Computer Name tab, you'll see the FQDN of the server and the domain or workgroup membership. The FQDN is the DNS name of the computer.
3. The DNS name is the name that you normally use to access the IIS resources on the server. For example, if the DNS name of the computer is www.microsoft.com and you've configured a Web site on port 80, the URL you use to access the computer from the Internet is *http://www.microsoft.com/*.

You can view the IP address and other TCP/IP settings for the computer by completing the following steps:

1. Click Start, and then click Control Panel. In Control Panel's Classic View, double-click Network And Sharing Center.

2. In the Network And Sharing Center, you'll see a list of tasks in the left pane. Click Manage Network Connections. This opens the Network Connections window.
3. Right-click Local Area Connection, and then select Properties. This opens the Local Area Connection Properties dialog box.
4. Open the Internet Protocol Version 4 (TCP/IPv4) Properties dialog box by double-clicking Internet Protocol Version 4 (TCP/IPv4).
5. The IPv4 Address and other TCP/IP settings for the computer are displayed, as shown in Figure 6-4.

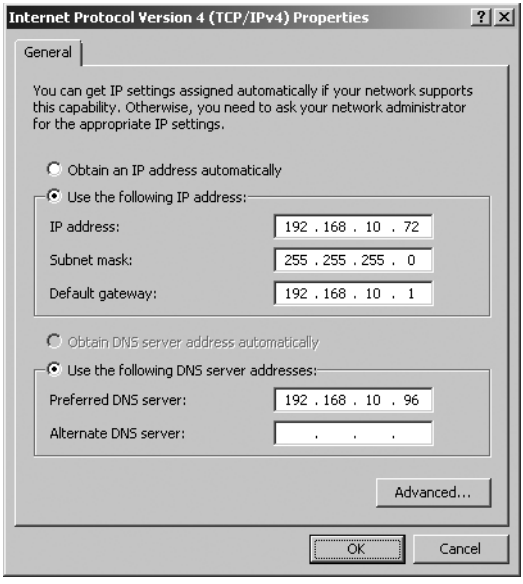

Figure 6-4 Use the Internet Protocol (TCP/IP) Properties dialog box to view and configure TCP/IP settings.

> IIS servers should use static IP addresses. If the computer is obtaining an IP address automatically, you'll need to reconfigure the TCP/IP settings.

Examining Site Configuration

In IIS Manager, you can view a list of the Web sites installed on a server by clicking the node for the computer you want to work with in the left pane and then clicking the Sites node. Sites are listed by name, ID number, Web site status, binding, and path.

By using the IIS Command-line Administration Tool, you can list the existing sites on a server by running the List Site command. Type **appcmd list site** at a command prompt to list all the sites on a server. You can list details about a specific site or the settings of a specific site as shown in these examples:

```
appcmd list site "Default Web Site"

appcmd list site http://localhost/

appcmd list site /serverAutoStart:false
```

You'll then see a summary related to the site configuration, such as:

```
SITE "Shopping Site" (id:6,bindings:https:/*:443:,state:Stopped)
```

These details provided the following information:

- **"Shopping Site"** is the name of the site.
- **id:6** is the identification number of the site.
- **bindings:https:/*:443:** tells you the site uses HTTPS on port 443 and IIS listens for requests on all IP addresses.
- **state:Stopped** tells you that the Web site is stopped and is not active.

You can view the full configuration details for a site by using the /config parameter, such as:

```
appcmd list site "Default Web Site" /config
```

You'll then see a full listing of the configuration details for the site, such as:

```
<site name="Shopping" id="6" state="Starting">
  <bindings>
    <binding protocol="https" bindingInformation="*:443:" />
  </bindings>
  <limits />
  <logFile />
  <traceFailedRequestsLogging />
  <applicationDefaults />
  <virtualDirectoryDefaults />
  <application path="/" applicationPool="Shopping">
    <virtualDirectoryDefaults />
    <virtualDirectory path="/" physicalPath="C:\inetpub\shopping"
    userName="DevTeam" password="RubberChickens" />
  </application>
</site>
```

Note When you are working with sites, applications, and virtual directories, you may need to provide logon credentials for authentication. Any credentials you provide are stored by default as encrypted text in the site, application, or virtual directory configuration. If you view the file with a text editor, you'll see the encrypted text. However, if you view the configuration details at the command prompt by running the List Site command with the /config parameter, you'll see the plaintext password as shown in this listing.

The full details do not include any inherited settings. To view the full configuration details, including inherited values, for a site, you must use the following syntax:

```
appcmd list site "SiteName" /config:*
```

Here is an example:

```
appcmd list site "Shopping Site" /config:*
```

You'll then see a full listing of the configuration details that includes inherited values, such as:

```
<site name="Shopping" id="6" serverAutoStart="true" state="Starting">
  <bindings>
    <binding protocol="https" bindingInformation="*:443:" />
  </bindings>
  <limits maxBandwidth="4294967295" maxConnections="4294967295"
   connectionTimeout="00:02:00" />
  <logFile logExtFileFlags="Date, Time, ClientIP, UserName, ServerIP,
Method,
  UriStem, UriQuery, HttpStatus, Win32Status, ServerPort, UserAgent,
  HttpSubStatus" customLogPluginClsid="" logFormat="W3C"
  directory="F:\inetpub\logs\LogFiles" period="Daily"
  truncateSize="20971520" localTimeRollover="false" enabled="true" />
  <traceFailedRequestsLogging enabled="false"
  directory="F:\inetpub\logs\FailedReqLogFiles" maxLogFiles="50"
  maxLogFileSizeKB="512" customActionsEnabled="false" />
  <applicationDefaults path="" applicationPool="" enabledProtocols="http" />
  <virtualDirectoryDefaults path="" physicalPath="" userName="" password=""
  logonMethod="ClearText" allowSubDirConfig="true" />
  <application path="/" applicationPool="Shopping"
enabledProtocols="http">
    <virtualDirectoryDefaults path="" physicalPath="" userName="" password=""
    logonMethod="ClearText" allowSubDirConfig="true" />
    <virtualDirectory path="/" physicalPath="C:\inetpub\shopping"
    userName="DevTeam" password="RubberChickens" logonMethod="ClearText"
    allowSubDirConfig="true" />
  </application>
</site>
```

Creating Web Sites

With IIS 7.0, you can create both unsecured and secured Web sites. Previous versions of IIS require you to configure a Certificate Authority (CA) to issue a site certificate prior to setting up Secure Sockets Layer (SSL) on a secured Web site, but IIS 7.0 does not require this. IIS 7.0 includes the necessary management features to create and manage SSL certificates. In fact, in most configuration scenarios, a self-signed certificate is created for a server during setup of IIS 7.0. For more information on SSL, see Chapter 11, "Managing Active Directory Certificate Services and SSL."

Creating a Web Site: The Essentials

When you install IIS, the setup process creates a default Web site. In most cases, you aren't required to change any network options to allow users access to the default Web site. You simply tell users the URL path that they need to type into their browser's Address field. For example, if the DNS name for the computer is *www.microsoft.com* and the site is configured for access on port 80, a user can access the Web site by typing **http://www.microsoft.com/** in the browser's Address field.

For name resolution, you must ensure that DNS is updated to include the appropriate records. Specifically, you'll need to ensure that either an A (address) or a CNAME (canonical name) record is created on the appropriate DNS server. An *A* record maps a host name to an IP address. A *CNAME* records sets an alias for a host name. For example, using this record, zeta.microsoft.com can have an alias as www.microsoft.com. If zeta.microsoft.com also hosts service.microsoft.com and sales.microsoft.com, you'd need CNAME records for these also.

On IIS 7.0, all Web Sites run within an application pool context. The settings of the application pool determine the pipeline mode used for requests and the Microsoft .NET Framework version. By default, IIS Manager creates a new application pool for any new site you create. This application pool uses the current .NET Framework version and the default, integrated pipeline mode. When you create a site, you can either accept the new application pool or select an existing application pool to associate with the site. Generally, you'll want to associate a site with a new application pool only when you want a non-standard configuration. For example, if you want a site to run in classic pipeline mode and use an earlier version of the .NET Framework, you could create the required application pool and then create a new Web site that uses this application pool.

The directories and files for the default Web site are created under *%Windir%*\Inetpub\Wwwroot. To help organize additional Web sites into a common directory structure, you might want to create your new site under *%windir%*\Inetpub also. Before you do this, however, you should consider carefully whether the underlying disk structure can support the increased file I/O of the new site. With high-traffic, extremely busy sites, you may need to put each site on a physically separate disk.

By default, IIS uses pass-through authentication for accessing the underlying physical directories used by Web sites and applications. This means that for anonymous access, the Internet user account (IUSR_*ServerName*) is used to access the site's physical directory and that for authenticated access, the actual account name of the authenticated user is used to access the site's physical directory. Thus, permissions for the physical directory must be set accordingly. If you want to map a Web site to a shared folder by using a UNC path, such as \\CentralStorage83\Inetpub\Sales_site, you can do this also. Because the shared folder is on a different server, you might need to set specific user credentials to access the shared folder. IIS Manager allows you to do this.

Creating an Unsecured Web Site

Users access unsecured Web sites by using HTTP. You can create a Web site that uses HTTP by completing the following steps:

1. If you're creating the Web site on a new server, ensure that the World Wide Web Publishing Service has been installed and started on the server.
2. If you want the Web site to use a new IP address, you must configure the IP address on the server before installing the site.
3. In IIS Manager, double-click the icon for the computer you want to work with, and then right-click Sites. On the shortcut menu, choose Add Web Site. This displays the Add Web Site dialog box, shown in Figure 6-5.

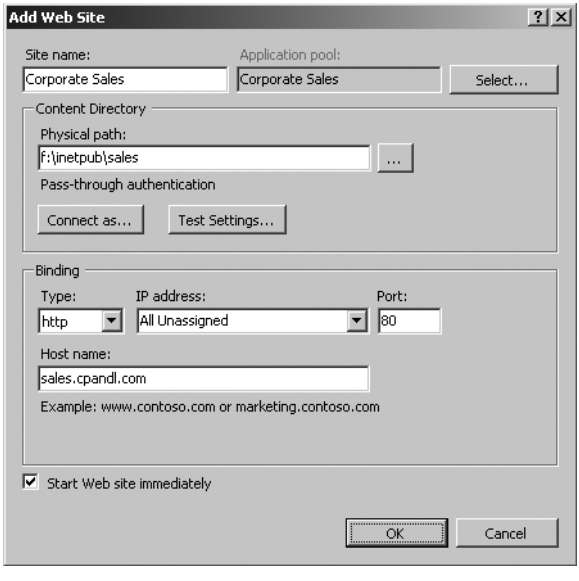

Figure 6-5 Create an unsecured Web site.

4. In the Web Site Name text box, type a descriptive name for the Web site, such as **Corporate Sales**. IIS Manager uses the name you provide to set the name of the new application pool to associate with the site. If you want to use an existing application pool instead of a new application pool, click Select. In the Select Application Pool dialog box, in the Application Pool drop-down list, select the application pool to associate with the site, and then click OK. Note that the .NET Framework version and pipeline mode of a selected application pool are listed on the Properties panel.

5. The Physical Path text box specifies the physical directory that contains the site's content. You can configure the physical path by using a local directory path or a shared folder. Keep the following in mind:
   - To specify a local directory path for the site, click the selection button (...) to the right of the Physical Path text box. In the Browse For Folder dialog box, use the choices provided to select a directory for the Web site. This folder must be created before you can select it. If necessary, click Make New Folder to create the directory.
   - To specify a shared folder for the site, type the desired UNC path in the appropriate text box, such as \\CentralStorage83\inetpub\sales_site. If you need to use alternate credentials to connect to the remote server specified in the UNC path, click Connect As. In the Connect As dialog box, choose Specific User, and then click Set. In the Set Credentials dialog box, type the name of the user account to use for authentication, type and confirm the account password, and then click OK.

   Note If you don't specify a user name and password, the user's Windows credentials are authenticated before allowing access. For an anonymous access site, IIS authenticates the credentials for the IUSR_*ServerName* account, so this account should have access to the shared folder. Otherwise, the network connection to the folder will fail. See the "Working with File and Folder Permissions" section in Chapter 10, "Managing Web Server Security," for more details on access permissions.

6. The Binding settings identify the Web site. To create an unsecured Web site, select HTTP as the type and then use the use the IP Address drop-down list to select an available IP address. Choose (All Unassigned) to allow HTTP to respond on all unassigned IP addresses that are configured on the server. Multiple Web sites can use the same IP address so long as the sites are configured to use different port numbers or host headers.
7. The TCP port for an unsecured Web site is assigned automatically as port 80. If necessary, type a new port number in the Port field. Multiple sites can use the same port as long as the sites are configured to use different IP addresses or host headers.
8. If you plan to use host headers for the site, type the host header name in the field provided. On a private network, the host header can be a computer name, such as EngIntranet. On a public network, the host header must be a DNS name, such as services.microsoft.com. The host header name must be unique within IIS.
9. By default, IIS starts the Web site immediately so long as the bindings you've supplied are unique. If you don't want to start the site immediately, clear the Start Web Site Immediately check box. In most cases, you'll want to finish setting the site's properties before you start the site and make it accessible to users.

By using the IIS Command-line Administration Tool, you can run the Add Site command to add an HTTP site to a server. Sample 6-1 provides the syntax and usage. Technically, bindings and physicalPath are optional, but a site won't work until these parameters are provided. Adding the physical path is what allows IIS to create the root virtual directory and root application for the site.

Sample 6-1 Adding an HTTP Site Syntax and Usage

Syntax

```
appcmd add site /name:Name /id:ID /bindings:http://UrlAndPort
/physicalPath:Path
```

Usage

```
appcmd add site /name:'Sales Site' /id:5 /bindings:
http://sales.adatum.com:80

appcmd add site /name:'Sales Site' /id:5 /bindings:http://*:8080

appcmd add site /name:'Sales Site' /id:5 /bindings:http/*:8080
/physicalPath:'c:\inetpub\mynewsite'
```

Creating a Secured Web Site

Users access secured Web sites by using SSL and HTTPS. Prior to creating a secured Web site, you must ensure that the certificate you want to use is available. You can create certificates as discussed in Chapter 11. You can create a Web site that uses HTTPS by completing the following steps:

1. Follow Steps 1–5 in the section "Creating an Unsecured Web Site," earlier in this chapter.
2. As shown in Figure 6-6, the Binding settings identify the Web site. To create a secured Web site, select HTTPS as the type, and then in the IP Address drop-down list, select an available IP address. Choose (All Unassigned) to allow HTTPS to respond on all unassigned IP addresses that are configured on the server. Multiple Web sites can use the same IP address as long as the sites are configured to use different port numbers or host headers.
3. The TCP port for a secured Web site is assigned automatically as port 443. If necessary, type a new port number in the Port field. Multiple sites can use the same port as long as the sites are configured to use different IP addresses or host headers.
4. Use the SSL Certificate drop-down list to select an available certificate to use for secure communications. After you select a certificate, click View to view details about the certificate.

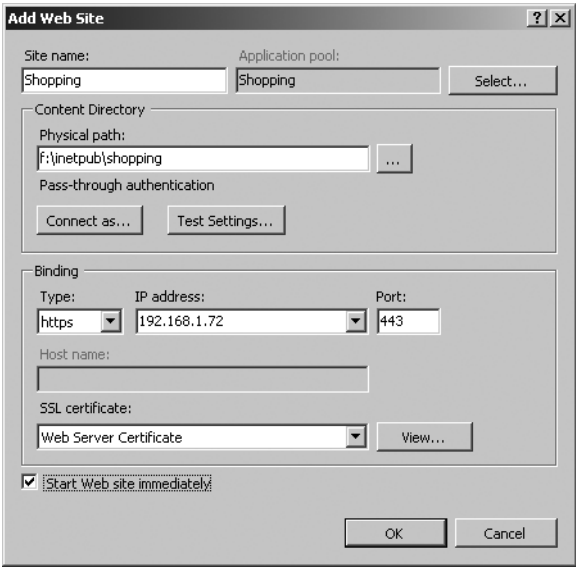

Figure 6-6 Create a secured Web site.

5. By default, IIS starts the Web site immediately as long as the bindings you've supplied are unique. If you don't want to start the site immediately, clear the Start Web Site Immediately check box. In most cases, you'll want to finish setting the site's properties before you start the site and make it accessible to users.

By using the IIS Command-line Administration Tool, you can run the Add Site command to add an HTTPS site to a server. Sample 6-2 provides the syntax and usage. As with unsecured sites, the bindings and physicalPath are optional, but a site won't work until these parameters are provided. Adding the physical path is what allows IIS to create the root virtual directory and root application for the site.

Sample 6-2 Adding an HTTPS Site Syntax and Usage

Syntax

```
appcmd add site /name:Name /id:ID /bindings:https://UrlAndPort
/physicalPath:Path
```

Usage

```
appcmd add site /name:'WWW Shopping Site' /id:6
/bindings:https://store.adatum.com:443

appcmd add site /name:'WWW Shopping Site' /id:6
/bindings:https://*:443

appcmd add site /name:'WWW Shopping Site' /id:6
/bindings:https://*:443 /physicalPath:'c:\inetpub\wwwstore'
```

Managing Web Sites and Their Properties

The sections that follow examine key tasks for managing Web sites and their properties. You configure Web site properties by using IIS Manager and the IIS Command-line Administration tool.

Working with Sites in IIS Manager

When you navigate to the Sites node in IIS Manager and select a site, the Actions pane displays a list of unique actions related to sites as shown in Figure 6-7. You can use the options in the Actions pane as follows:

- **Explore** Opens the site's root directory in Windows Explorer. You can use this option to access the site's Web.config file or to manage the site's physical directories and content files.
- **Edit Permissions** Opens the Properties dialog box for the site's root directory. By using the Properties dialog box, you can configure general settings, sharing, and security.
- **Edit Site** Provides Bindings and Basic Settings options. The Bindings option allows you to view and manage the site's bindings. Basic Settings allows you to view and manage the site's application pool and physical path.
- **Manage Web Site** Provides Start, Stop, and Restart options. These options allow you to manage the site's run state. A stopped site cannot be accessed by users.
- **Browse Web Site** Provides Browse and View options for the site. The Browse options allow you to test the configuration of a specific binding. When you click a Browse link, IIS Manager starts the default browser and connects to the site using the related binding. View Applications displays a page that allows you to view and manage the site's applications. View Virtual Directories displays a page that allows you to view and manage the site's virtual directories.
- **Configure** Provides Failed Request Tracing and Limits options. You can use Failed Request Tracing to trace failed requests through the IIS core. You can use Limits to control incoming connections to the Web site.
- **Help** Displays the IIS Manager help documentation. Because the Help window is displayed on top of the IIS Manager window, you must minimize or close the Help window before you can return to IIS Manager.

Figure 6-7 Working with sites.

Right-clicking a site's node in the left pane displays a shortcut menu with similar, though slightly different, options. The Add Application option allows you to add an application to the site. The Add Virtual Directory option allows you to add a virtual directory to the site. Two additional options that are important are Switch To Content View and Switch To Features View. You can use these options to switch between the following views:

- **Content view** Shows the file contents of the physical directory related to a selected site, application, or virtual directory
- **Features view** Shows the managed features related to a selected site, application, or virtual directory

You can switch between the Content and Features view by right-clicking the site node and then selecting Switch To Content View or Switch To Feature View as appropriate.

You can use the shortcut menu to rename a Web site by right-clicking the site node and then selecting Rename. Next, edit the name of the site as necessary, and then press Enter.

When you right-click the site node and then point to Manage Web Site, you'll see an additional shortcut menu with these options:

- **Restart** Stops and then starts the site. If you suspect that IIS is not processing requests for a site appropriately, restarting the site can in some cases resolve this.

- **Start** Starts a site if it is not running. A site can accept incoming requests only when it is started.
- **Stop** Stops a site if it is running. A site cannot accept or process requests when it is stopped.
- **Browse** Starts the default browser and connects to the site by using the default binding.
- **Advanced Settings** Displays all the settings for a site in a single dialog box, allowing you to manage all settings except the site name and its bindings.

By using the IIS Command-line Administration Tool, you can start or stop a site by running the Start Site and Stop Site commands respectively. Samples 6-3 and 6-4 provide the syntax and usage.

Sample 6-3 Start Site Syntax and Usage

Syntax

```
appcmd start site [/site.name:]SiteNameOrURL
```

Usage

```
appcmd start site "Default Web Site"
```

Sample 6-4 Stop Site Syntax and Usage

Syntax

```
appcmd stop site [/site.name:]SiteNameOrURL
```

Usage

```
appcmd stop site "Default Web Site"
```

Configuring a Site's Application Pool and Home Directory

Each Web site on a server has an application pool and home directory. The application pool determines the request mode and .NET Framework version that IIS loads into the site's worker process. The home directory is the base directory for all documents that the site publishes. It contains a home page that links to other pages in your site. The home directory is mapped to your site's domain name or to the server name. For example, if the site's DNS name is *www.microsoft.com* and the home directory is C:\Inetpub\Wwwroot, browsers use the URL *http://www.microsoft.com/* to access files in the home directory. On an intranet, the server name can be used to access documents in the home directory. For example, if the server name is CorpIntranet, browsers use the URL *http://CorpIntranet/* to access files in the home directory.

You can view or change a site's home directory by completing the following steps:

1. In IIS Manager, navigate to the Sites node by double-clicking icon for the computer you want to work with and then double-clicking Sites.

2. In the left pane, select the node for the site you want to work with.
3. In the Actions pane, click Basic Settings. This displays the Edit Web Site dialog box, as shown in Figure 6-8.

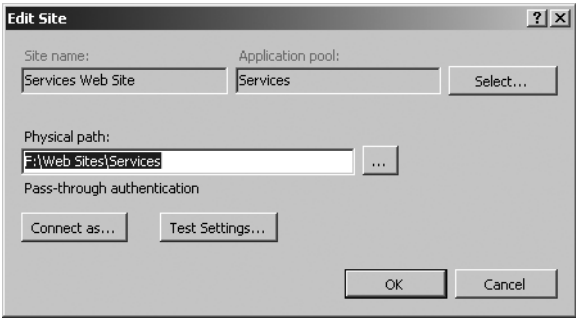

Figure 6-8 You can change a site's home directory at any time.

4. The Application Pool text box lists the application pool currently associated with the site. To choose a different application pool, click Select. In the Select Application Pool dialog box, in the Application Pool drop-down list, select the application pool to associate with the site, and then click OK.
5. If the directory you want to use is on the local computer, type the directory path, such as **C:\Inetpub\Wwwroot**, in the Physical Path text box. To browse for the folder, click the selection button to the right of the Physical Path text box. In the Browse For Folder dialog box, use the settings to select a directory for the Web site. This folder must be created before you can select it. If necessary, click Make New Folder in the Browse For Folder dialog box to create the directory.
6. If the directory you want to use is on another computer and is accessible as a shared folder, type the desired UNC path, such as **\\WebServer22\CorpWWW**, in the Physical Path text box. If you need to use alternate credentials to connect to the remote server specified in the UNC path, click Connect As. In the Connect As dialog box, choose Specific User, and then click Set. In the Set Credentials dialog box, type the name of the user account to use for authentication, type and confirm the account password, and then click OK.

> **Caution** Be careful when setting alternate pass-through credentials. The account you use should not have any additional privileges beyond those required to access content via the Web site. If necessary, you may want to create a new restricted account for this purpose.

7. Click OK to close the Edit Web Site dialog box.

You cannot use the IIS Command-line Administration Tool to configure a site's application pool and home directory in the same way. Whereas IIS Manager maps these changes to the application pool and base virtual directory associated with the site, the IIS Command-line Administration tool does not, and you must edit the application pool and virtual directory settings to make the necessary changes.

Configuring Ports, IP Addresses, and Host Names Used by Web Sites

Throughout this chapter, I've discussed techniques you can use to configure multiple Web sites on a single server. The focus of the discussion has been on configuring unique identities for each site. In some instances, you might want a single Web site to have multiple domain names associated with it. A Web site with multiple domain names publishes the same content for different sets of users. For example, your company might have registered *example.com*, *example.org*, and *example.net* with a domain registrar to protect your company or domain name. Rather than publishing the same content to each of these sites separately, you can publish the content to a single site that accepts requests for each of these identities.

The rules regarding unique combinations of ports, IP addresses, and host names still apply to sites with multiple identities. This means that each identity for a site must be unique. You accomplish this by assigning each identity unique IP address, port, or host header name combinations.

> **Note** When you've installed additional Windows Process Activation Service support components, you may find that IIS allows you to create non-HTTP binding types, including net.tcp, net.pipe, net.msmq, and msmq.formatname. These additional binding types are used to support process activation over Transmission Control Protocol (TCP), named pipes, and Microsoft Message Queuing (MSMQ). These binding types accept a single parameter: the binding information that includes the network address to listen for requests on. See the "Role Services for Application Servers" section of Chapter 2, "Deploying IIS 7.0 in the Enterprise," for more information on non-HTTP process activation.

To change the binding of a Web site, complete the following steps:

1. If you want the Web site to respond to a specific IP address, you must configure the IP address before updating the site.
2. In IIS Manager, navigate to the Sites node by double-clicking the icon for the computer you want to work with and then double-clicking Sites.
3. In the left pane, select the node for the site you want to work with.
4. In the Actions pane, click Bindings. This displays the Site Bindings dialog box, as shown in Figure 6-9.

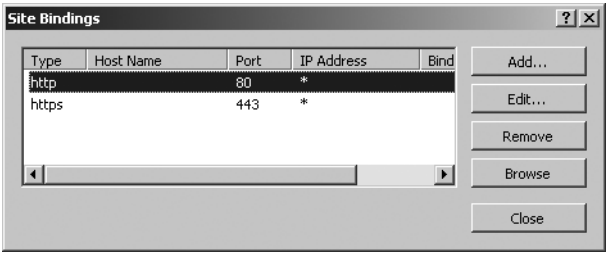

Figure 6-9 You modify a site's identity through the Site Bindings dialog box.

5. Use the Site Bindings dialog box to manage the site's binding by using the following settings:

   - **Add** Adds a new identity. To add a new identity, click Add. In the Add Site Binding dialog box, select the binding type, IP address, and TCP port to use. Optionally, type a host header name or select an SSL certificate as appropriate for the binding type. Click OK when you're finished.
   - **Edit** Allows you to edit the currently selected identity. To edit an identity, click the identity, and then click Edit. In the Edit Site Binding dialog box, select an IP address and TCP port to use. Optionally, type a host header name or select an SSL certificate as appropriate for the binding type. Click OK when you're finished.
   - **Remove** Allows you to remove the currently selected identity. To remove an identity, click the identity, and then click Remove. When prompted to confirm, click Yes.
   - **Browse** Allows you to test an identity. To test an identity, click the identity, and then click Browse. IIS Manager will then open a browser window and connect to the selected binding.

6. When you are finished working with bindings, click Close to close the Site Bindings dialog box.

By using the IIS Command-line Administration Tool, you can add, change or remove bindings by running the Set Site command. Samples 6-5 to 6-7 provide the syntax and usage. When working with the Set Site command, note that you must use the exact syntax shown. Unlike other commands in which you can omit quotes or use double-quotes, you must use single quotes where indicated. Additionally, because you are referencing into the bindings collection, the brackets ([]) in the syntax and usage examples are literal values rather than indicators of optional values. You must use the brackets to indicate that you are referencing into the bindings collection.

Caution Failure to use the exact syntax expected with the bindings collections can result in the Web site becoming unstable. For example, improper use of quotes could cause AppCmd to create the site binding with quotes as part of the binding name. If this happens, the best way to correct the problem is to remove the binding and then add it again. Because you cannot remove the last binding associated with a site, you may need to create another binding and then remove the improperly formatted binding.

Sample 6-5 Adding Site Bindings Syntax and Usage

Syntax

```
appcmd set site /site.name:'Name'
/+bindings.[protocol='ProtocolType',
bindingInformation='IPAddress:Port:HostHeader']
```

Usage

```
appcmd set site /site.name:'WWW Shopping Site'
/+bindings.[protocol='https', bindingInformation='*:443:']
```

Sample 6-6 Changing Site Bindings Syntax and Usage

Syntax

```
appcmd set site /site.name:Name /bindings.[protocol='ProtocolType',
bindingInformation='OldBindingInfo'].bindingInformation:NewBindingInfo
```

Usage

```
appcmd set site /site.name: 'WWW Shopping Site'
/bindings.[protocol='https',
bindingInformation='*:443:'].bindingInformation:*:443:
shopping.cpandl.com
```

Sample 6-7 Removing Site Bindings Syntax and Usage

Syntax

```
appcmd set site /site.name:Name /-bindings.[protocol='ProtocolType',
bindingInformation='BindingInfo']
```

Usage

```
appcmd set site /site.name:'WWW Shopping Site'
/-bindings.[protocol='https', bindingInformation='*:443:']
```

Restricting Incoming Connections and Setting Time-Out Values

You can control incoming connections to a Web site in several ways. You can:

- Set a limit on the amount of traffic allowed to a Web site based on bandwidth usage.

- Set a limit on the number of simultaneous connections.
- Set a connection time-out value to ensure that inactive connections are disconnected.

Normally, Web sites have no bandwidth or connection limits, and this is an optimal setting in most environments. However, high bandwidth usage or a large number of connections can cause the Web site to slow down—sometimes so severely that nobody can access the site. To avoid this situation, you might want to limit the total bandwidth usage, the number of simultaneous connections, or both. When using limits, keep the following in mind:

- Once a bandwidth limit is reached, no additional bandwidth will be available to service new or existing requests. This means that the server would not be able to process new requests for both existing clients and new clients. One reason to set a bandwidth limit is when you have multiple sites sharing the same limited bandwidth connection and these sites are equally important. Keep in mind that most network connections are measured in *bits*, but you set the bandwidth limit in *bytes*.
- Once a connection limit is reached, no other clients are permitted to access the server. New clients must wait until the connection load on the server decreases; however, currently connected users are allowed to continue browsing the site. One reason to set a connection limit is to prevent a single Web site from overloading the resources of an entire server.

The connection time-out value determines when idle user sessions are disconnected. With the default Web site, sessions time out after they've been idle for 120 seconds (2 minutes). This prevents connections from remaining open indefinitely if browsers don't close them correctly.

You can modify a site's limits and time-outs by completing the following steps:

1. In IIS Manager, navigate to the Sites node by double-clicking the icon for the computer you want to work with and then double-clicking Sites.
2. In the left pane, select the node for the site you want to work with.
3. In the Actions pane, click Limits. You'll find Limits under Configure in the lower portion of the Actions pane. Clicking Limits displays the Edit Web Site Limits dialog box, as shown in Figure 6-10.
4. The Limit Bandwidth Usage check box controls bandwidth limits. To remove a bandwidth limit, clear this check box. To set a bandwidth limit, select this check box, and then type a limit in bytes.
5. The Connection Timeout field controls the connection time-out. Type a new value to change the current time-out setting.
6. The Limit Number Of Connections check box controls connection limits. To remove connection limits, clear this check box. To set a connection limit, select this check box, type a limit, and then click OK.

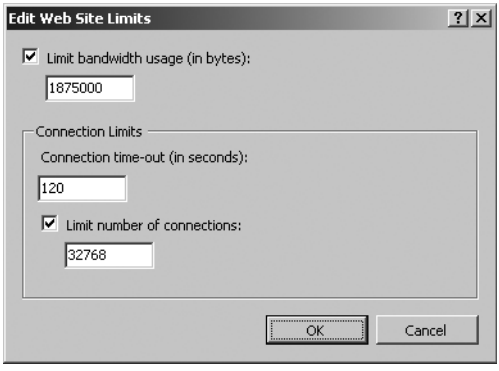

Figure 6-10 You modify a site's limits through the Edit Web Site Limits dialog box.

By using the IIS Command-line Administration Tool, you can run the Set Site command to set and remove limits for a site. Samples 6-8 and 6-9 provide the syntax and usage. Note that time-out values are set in the hh:mm:ss format in which the h position is for hours, the m position is for minutes, and the s position is for seconds. If you remove limits, the default values, such as 120 seconds for connection time-outs, are restored.

Sample 6-8 Setting Site Limits Syntax and Usage

Syntax

```
appcmd set site /site.name:Name [/limits.maxBandwidth:Bandwidth]
[/limits.maxConnections:MaxConnections]
[/limits.connectionTimeout:TimeOut]
```

Usage

```
appcmd set site /site.name:'WWW Shopping Site'
/limits.maxConnections:32768

appcmd set site /site.name:'WWW Shopping Site'
/limits.connectionTimeout:'00:01:30'
```

Sample 6-9 Removing Site Limits Syntax and Usage

Syntax

```
appcmd set site /site.name:Name [/-limits.maxBandwidth]
[/-limits.maxConnections] [/-limits.connectionTimeout]
```

Usage

```
appcmd set site /site.name:'WWW Shopping Site'
/-limits.maxConnections
```

Configuring HTTP Keep-Alives

The original design of HTTP opened a new connection for every file retrieved from a Web server. Because a connection isn't maintained, no system resources are used after the transaction is completed. The drawback to this design is that when the same client requests additional data, the connection must be reestablished, and this means additional traffic and delays.

Consider a standard Web page that contains a main HTML document and 10 images. With standard HTTP, a Web client requests each file through a separate connection. The client connects to the server, requests the document file, gets a response, and then disconnects. The client repeats this process for each image file in the document.

Web servers compliant with HTTP 1.1 support a feature called *HTTP Keep-Alives*. With this feature enabled as per the default configuration in IIS 7.0, clients maintain an open connection with the Web server rather than reopening a connection with each request. HTTP keep-alives are enabled by default when you create a new Web site. In most situations clients will see greatly improved performance with HTTP keep-alives enabled. Keep in mind, however, that maintaining connections requires system resources. The more open connections there are, the more system resources are used. To prevent a busy server from getting bogged down by a large number of open connections, you might want to limit the number of connections, reduce the connection timeout for client sessions, or both. For more information on managing connections, see the "Restricting Incoming Connections and Setting Time-Out Values" section earlier in this chapter.

To enable or disable HTTP keep-alives, follow these steps:

1. In IIS Manager, navigate to the level of the configuration hierarchy you want to manage. You can manage HTTP keep-alives for an entire server at the server level. You can manage HTTP keep-alives for a specific site at the site level.
2. When you group by area, the HTTP Response feature is listed under IIS. Double-click the HTTP Response feature.
3. In the Actions Pane, click Set Common Headers. This displays the Set Common HTTP Response Headers dialog box as shown in Figure 6-11.
4. Select Enable HTTP Keep-Alives to enable HTTP keep-alives. Clear this check box to disable HTTP keep-alives. Then click OK.

By using the IIS Command-line Administration Tool, you can run the Set Config command to enable or disable HTTP keep-alives. Sample 6-10 provides the syntax and usage. If you don't specify a site name, you will enable or disable HTTP keep-alives for the entire server.

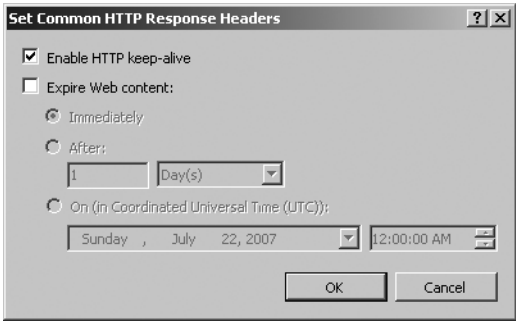

Figure 6-11 Enable or disable HTTP keep-alives.

Sample 6-10 Enabling and Disabling HTTP Keep-Alives Syntax and Usage

Syntax

```
appcmd set config [SiteName] /section:httpProtocol
/allowKeepAlive:[true|false]
```

Usage

```
appcmd set config 'WWW Shopping Site' /section:httpProtocol
/allowKeepAlive:true

appcmd add site /name:'WWW Shopping Site' /id:6 /bindings:
https://*:443

appcmd add site /name:'WWW Shopping Site' /id:6 /bindings:
https://*:443 /physicalPath:'c:\inetpub\wwwstore'
```

Configuring Access Permissions in IIS Manager

In earlier releases of IIS, you configured access permissions for sites and virtual directories. In IIS 7.0, general access permissions are set through the access policy you've configured for the server's managed handlers as discussed in the "Controlling Managed Handlers through the Configuration Files" section of Chapter 5, "Managing Global IIS Configuration." From a perspective of content access, the standard types of access grant the following permissions:

- **Read** Allows users to read documents, such as Hypertext Markup Language (HTML) files
- **Script** Allows users to run scripts, such as ASP files or Perl scripts
- **Execute** Allows users to execute programs, such as ISAPI applications or CGI executable files

You can configure access permissions by completing the following steps:

1. In IIS Manager, navigate to the level of the configuration hierarchy you want to manage. You can manage access permissions for an entire server at the server level. You can manage access permissions for a specific site at the site level.
2. When you group by area, the Handler Mappings feature is listed under IIS. Double-click the Handler Mappings feature.
3. In the Actions Pane, click Edit Feature Permissions.
4. In the Edit Feature Permissions dialog box, shown in Figure 6-12, select or clear permissions as appropriate, and then click OK to apply the settings.

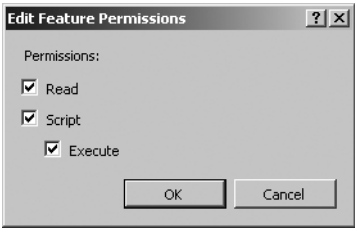

Figure 6-12 Set handler permissions for Web content.

Managing a Site's Numeric Identifier and AutoStart State

Every Web site has an associated numeric identifier and AutoStart state. IIS uses the numeric identifier for internally tracking the site, and you'll find it referenced in log files and trace files. IIS assigns the ID automatically when sites are created. Typically, the default Web site has an ID of 1, the second site created on a server has an ID of 2, and so on.

IIS uses the AutoStart state to determine whether to start the site automatically when the World Wide Web service is started. If the AutoStart state is set to True, IIS starts the site when the World Wide Web service is started. If the AutoStart state is set to False, IIS does not start the site when the World Wide Web service is started, so you must manually start the site.

You can configure a site's ID and AutoStart state by completing the following steps:

1. In IIS Manager, navigate to the Sites node by double-clicking the icon for the computer you want to work with and then double-clicking Sites.
2. In the left pane, select the node for the site you want to work with.
3. In the Actions pane, click Advanced Settings. You'll find Advanced Settings under Browse Web Site in the middle of the Actions pane. Clicking Advanced Settings displays the Advanced Settings dialog box, as shown in Figure 6-13.

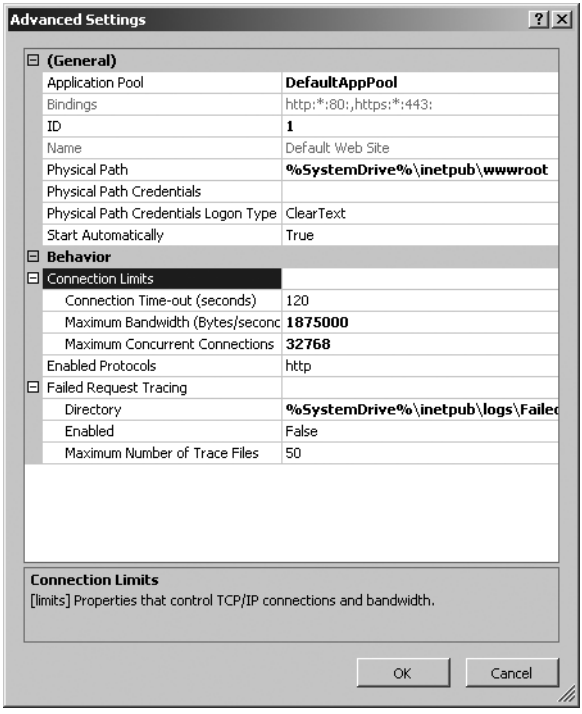

Figure 6-13 You modify a site's ID number and AutoStart state through the Advanced Settings dialog box.

4. ID lists the site's current ID number. To change the ID number, click in the column to the right and then type the desired ID number. The ID number you type cannot be in use already.
5. The Start Automatically item lists the site's current AutoStart state. To change the AutoStart state, click in the column to the right, and then in the selection list that appears, choose either True or False.
6. Click OK to save your settings. Changing the AutoStart state does not change the current run state of the site.

By using the IIS Command-line Administration Tool, you can change a site's ID number and AutoStart state by running the Set Site command. Sample 6-11 provides the syntax and usage. AppCmd will generate an error if you type an ID number that is already in use. In this case, you will need to choose a different ID number.

Sample 6-11 Set Site Syntax and Usage

Syntax

```
appcmd set site [/site.name:]SiteNameOrURL
[/serverAutoStart:true|false] [/id:Number]
```

Usage

```
appcmd set site "Default Web Site" /serverAutoStart:false /id:5
```

Deleting Sites

If you no longer need a site, you can delete the site by using IIS Manager or the IIS Command-line Administration tool. Deleting a site permanently removes the site configuration information from the IIS configuration files. This means that the site's configuration details, including any applications and virtual directories, are removed permanently. Deleting a site does not, however, delete the site's physical directories or content files. If you want to delete the physical directories or content files, you'll need to do this manually by using Windows Explorer.

> **Tip** Rather than permanently deleting a site that you may need in the future, you may want to stop the site and then configure the site's AutoStart state to False as discussed in the "Managing a Site's Numeric Identifier and AutoStart State" section earlier in this chapter. This allows you to use the site in the future if necessary.

You can remove a site permanently by completing the following steps:

1. In IIS Manager, navigate to the Sites node by double-clicking the icon for the computer you want to work with and then double-clicking Sites.
2. In the left pane, right-click the node for the site you want to delete, and then select Remove.
3. When prompted to confirm the action, click Yes.

By using the IIS Command-line Administration Tool, you can remove a site by running the Delete Site command. Sample 6-12 provides the syntax and usage.

Sample 6-12 Delete Site Syntax and Usage

Syntax

```
appcmd delete [/site.name:]site SiteNameOrURL
```

Usage

```
appcmd delete site "Default Web Site"
```

Creating Directories

The directory structure of IIS is based primarily on the Windows Server file system, but it also provides additional functionality and flexibility. Understanding these complexities is critical to successfully managing IIS Web sites.

Understanding Physical and Virtual Directory Structures

Earlier in this chapter, I discussed home directories and how they were used. Beyond home directories, Microsoft Web sites also use the following:

- Physical directories
- Virtual directories

The difference between physical and virtual directories is important. A *physical* directory is part of the file system, and to be available through IIS, it must exist as a subdirectory within the home directory. A *virtual* directory is a directory that isn't necessarily contained in the home directory but is available to clients through an alias. Physical directories and virtual directories are configured and managed through the IIS Manager, but they're displayed differently. Physical directories are indicated with a standard folder icon. Virtual directories are indicated by a folder icon with a globe in the corner.

Both physical and virtual directories have permissions and properties that you can set at the operating system level and the IIS level. You set operating system permissions and properties in Windows Explorer–related dialog boxes. You set IIS permissions and properties in IIS Manager.

You create physical directories by creating subdirectories within the home directory by using Windows Explorer. You access these subdirectories by appending the directory name to the DNS name for the Web site. For example, you create a Web site with the DNS name *products.microsoft.com*. Users are able to access the Web site by using the URL *http://www.microsoft.com/*. You then create a subdirectory within the home directory called "search." Users are able to access the subdirectory by using the URL path *http://www.microsoft.com/search/*.

Even though locating your content files and directories within the home directory makes it easier to manage a Web site, you can also use virtual directories. Virtual directories act as pointers to directories that aren't located in the home directory. You access virtual directories by appending the directory alias to the DNS name for the site. If, for example, your home directory is D:\Inetpub\Wwwroot, and you store Microsoft Office Word documents in E:\Worddocs, you would need to create a virtual directory that points to the actual directory location. If the alias is *docs* for the E:\Worddocs directory, visitors to the *www.microsoft.com* Web site could access the directory by using the URL path *http://www.microsoft.com/docs/*.

Examining Virtual Directory Configuration

All virtual directories are associated with either a site's root application or a specific application. In IIS Manager, you can view a list of the virtual directories associated with a site's root application by selecting the site in the left pane and then under Actions, clicking View Virtual Directories. In IIS Manager, you can view a list of the virtual directories associated with a specific application by selecting the application in the left pane and then under Actions, clicking View Virtual Directories.

By using the IIS Command-line Administration Tool, you can list the existing virtual directories for an application by running the List Vdir command. Type **appcmd list vdir** at a command prompt to list all the virtual directories configured for any and all applications on a server. This listing will include the root virtual directories of all sites and applications configured on the server because these are created as virtual directories. The names of root virtual directories for sites and applications end in a slash. The names of virtual directories that are not mapped to sites and applications do not end in a slash.

You can list details about virtual directories according to the applications with which they are associated, as shown in these examples:

```
appcmd list vdir "Default Web Site/"

appcmd list vdir http://localhost/Sales

appcmd list vdir /app.name:"Default Web Site/Sales"
```

You'll then see a summary entry related to the virtual directory configuration, such as:

```
VDIR "Default Web Site/" (physicalPath:%SystemDrive%\inetpub\wwwroot)
```

You can also list details about virtual directories according to their virtual paths, as shown in this example:

```
appcmd list vdir /path:/Store
```

You'll then see a summary entry related to the virtual directory configuration, such as:

```
VDIR "Default Web Site/Store" (physicalPath:C:\store)
```

These details include the name of the virtual directory and the physical path of the virtual directory.

You can view the full configuration details for a virtual directory by using the /config parameter, such as:

```
appcmd list vdir "Default Web Site/" /config
```

You'll then see a full listing of the configuration details for the virtual directory, such as:

```
<virtualDirectory path="/" physicalPath="C:\inetpub\shopping"
userName="DevTeam" password="RubberChickens" />
```

The full details do not include any inherited settings. To view the full configuration details for a site, including inherited values, you must use the following syntax:

```
appcmd list vdir "VdirName" /config:*
```

Here is an example:

```
appcmd list vdir "Default Web Site/" /config:*
```

You'll then see a full listing of the configuration details that includes inherited values, such as:

```
<virtualDirectory path="/" physicalPath="C:\inetpub\shopping"
userName="DevTeam" password="RubberChickens" logonMethod="ClearText"
allowSubDirConfig="true" />
```

Creating Physical Directories

Within the home directory, you can create subdirectories to help organize your site's documents. You can create subdirectories within the home directory by completing the following steps:

1. In Windows Explorer, navigate to the home directory for the Web site.
2. In the Contents pane, right-click a blank area and then, on the shortcut menu, select New and then select Folder. A new folder is added to the Contents pane. The default name, New Folder, appears in the folder name area and is selected for editing.
3. Edit the name of the folder, and then press Enter. The best directory names are short but descriptive, such as Images, WordDocs, or Downloads.

   Tip If possible, avoid using spaces as part of IIS directory names. Officially, spaces are illegal characters in URLs and must be replaced with an escape code. The escape code for a space is %20. Although most current browsers will replace spaces with %20 for you, earlier versions of browsers might not, so those versions won't be able to access the page.

4. The new folder inherits the default file permissions of the home directory and the default IIS permissions of the Web site. For details on viewing or changing permissions, see Chapter 10.

Tip IIS Manager doesn't display new folders automatically. You might need to click the Refresh button on the toolbar (or press F5) to display the folder.

Creating Virtual Directories

As stated previously, a virtual directory is a directory available to Internet users through an alias for an actual physical directory. In previous versions of IIS, you had to create the physical directory prior to assigning the virtual directory alias. In IIS 7.0, you can create the physical directory if one is needed when you create the virtual directory.

To create a virtual directory, follow these steps:

1. In IIS Manager, navigate to the level of the configuration hierarchy where you want to create the virtual directory. You can add a virtual directory to the site's root application by selecting the site's node. You can add a virtual directory to another application by selecting the application's node.

2. In the Actions pane, click View Virtual Directories. In the main pane, you'll see a list of the site's existing virtual directories (if any).
3. In the Actions pane, click Add Virtual Directory. This displays the Add Virtual Directory dialog box, shown in Figure 6-14.

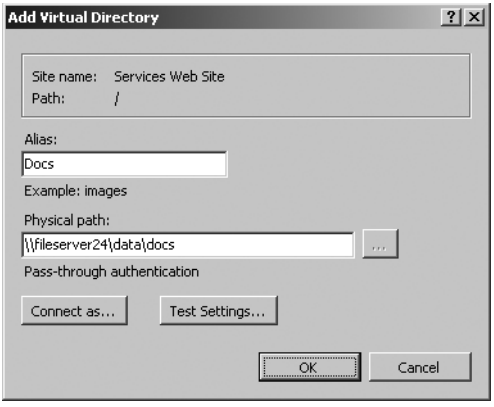

Figure 6-14 Create a virtual directory.

4. In the Alias text box, type the name you want to use to access the virtual directory. As with directory names, the best alias names are short but descriptive.
5. In the Physical Path text box, type the path to the physical directory where your content is stored, or click the selection button to the right of the Physical Path text box to search for a directory. The directory must be created before you can select it. If necessary, click Make New Folder in the Browse For Folder dialog box to create the directory before you select it. However, don't forget about checking and setting permissions at the operating system level as discussed in Chapter 10.
6. If you need to use alternate credentials to connect to the remote server specified in a UNC path, click Connect As. In the Connect As dialog box, choose Specific User, and then click Set. In the Set Credentials dialog box, type the name of the user account to use for authentication, type and confirm the account password, and then click OK.
7. Click OK to create the virtual directory.

Tip When you set logon credentials for a virtual directory, the account name you provide must exist. By default, IIS Manager sets the logon type to ClearText. This means that IIS will use clear text when acquiring the user token necessary to access the physical path. Because IIS passes the logon user call over the back end on an internal network, using a clear-text call typically is sufficient. By editing a virtual directory's properties, you also have the option to set the logon type to Interactive, Batch, or Network. See the "Changing Virtual Directory Paths, Logon Methods, and More" section later in this chapter for more information.

By using the IIS Command-line Administration Tool, you can create virtual directories by running the Add Vdir command. Sample 6-13 provides the syntax and usage. Remember that the physical directory you point to must already exist.

Sample 6-13 Add Vdir Syntax and Usage

Syntax

```
appcmd add vdir /app.name:"ParentAppName" /path: "VirtualPath"
[/physicalPath: "Path"] [/logonMethod:Method] [/userName:User]
[/password:Password]
```

Usage

```
appcmd add vdir /app.name:"Default Web Site/" /path:"/Support" /
physicalPath:"c:\support"

appcmd add vdir /app.name:"Sales Site/" /path:"/Invoices"
/physicalPath:"c:\salesroot\invoices" /logonMethod:ClearText
/userName:SupportUser /password:RainyDayz
```

Managing Directories and Their Properties

When you navigate to a site node in IIS Manager and select a directory, the Actions pane displays a list of unique actions related to directories. With physical directories, denoted by a folder icon, the options allow you to explore the directory in Windows Explorer and edit permissions through the directory's Properties dialog box. You can also browse the folder in the default browser to test the configuration of a specific binding with regard to the selected physical directory. With virtual directories, denoted by a shortcut folder icon, you have additional options for editing a directory's basic and advanced settings. Basic settings allow you to view and manage a directory's physical path and connection credentials. Advanced settings allow you to view and manage a directory's physical path, connection credentials, and logon type.

Enabling or Disabling Directory Browsing

Unlike IIS 6, IIS 7.0 does not have a specific Browse policy that allows users to view a list of files if they enter the name of a valid directory that doesn't have a default file. Instead, you control whether directory browsing is allowed by using the Directory Browsing module. If you want users to be able to browse site directories, you must install, enable, and then configure the Directory Browsing module. Because you typically don't want users to be able to browse every directory on every site hosted on a server, you must be careful when using the Directory Browsing module. Specifically, you'll want to ensure that you enable this module only where necessary and appropriate. For example, if you want users to be able to browse a specific virtual directory, you can enable the module for this virtual directory but disable it elsewhere.

Note Keep in mind that these access permissions act as a layer on top of the server's file access permissions. You set file access permissions at the operating system level as discussed in the "Working with File and Folder Permissions" section of Chapter 10.

Once you've installed the Directory Browsing module, you can enable and configure directory browsing by completing these steps:

1. In IIS Manager, navigate to the level of the configuration hierarchy you want to manage. You can manage directory browsing for an entire server at the server level. You can manage directory browsing for a specific site at the site level.
2. When you group by area, the Directory Browsing feature is listed under IIS. Double-click the Directory Browsing feature.
3. If directory browsing is disabled, you can enable this feature by clicking Enable in the Actions pane.
4. Once directory browsing is enabled, you can use the check boxes to specify the information that IIS displays in a directory listing. The available check boxes are:
   - **Time**. Lists the last modified time for each file
   - **Size**. Lists the size of each file
   - **Extension**. Lists the file extension along with the file name
   - **Date**. Lists the last modified date for each file
   - **Long Date**. Lists the last modified date for each file in extended format
5. Click Apply to save and apply your changes.

You can disable directory browsing by completing these steps:

1. In IIS Manager, navigate to the level of the configuration hierarchy you want to manage. You can manage directory browsing for an entire server at the server level. You can manage directory browsing for a specific site at the site level.
2. When you group by area, the Directory Browsing feature is listed under IIS. Double-click the Directory Browsing feature.
3. If directory browsing is enabled, you can disable this feature by clicking Disable in the Actions pane.

By using the IIS Command-line Administration Tool, you can run the Set Config command to enable or disable directory browsing. Sample 6-14 provides the syntax and usage. If you don't specify a virtual directory name, you will enable or disable directory browsing for the entire server. By including the /showFlags parameter, you can enter the flags in the form of a comma-separated list. The acceptable values are: Date, LongDate, Time, Size, and Extension.

Sample 6-14 Enabling and Disabling Directory Browsing Syntax and Usage

Syntax

```
appcmd set config [VdirName] /section:directoryBrowse
[/enabled:[true|false]] [/showFlags=Flags]
```

Usage

```
appcmd set config "WWW Shopping Site/Sales/" /section:directoryBrowse
/enabled:false /showFlags="Time, Size, Date, LongDate"
```

Modifying Directory Properties

You can modify the settings for a physical or virtual directory at any time. In Windows Explorer, you can set directory permissions and general directory properties by right-clicking the directory name and selecting Properties. In IIS Manager, you can display the same properties dialog box by selecting the physical or virtual directory in the left pane and then clicking Edit Permissions in the Actions pane.

You can configure IIS permissions by completing the following steps:

1. In IIS Manager, in the left pane, select the physical or virtual directory.
2. Double-click the Handler Mappings feature.
3. In the Actions Pane, click Edit Feature Permissions.
4. In the Edit Feature Permissions dialog box, select or clear permissions as appropriate, and then click OK to apply the settings.

Renaming Directories

You can rename physical and virtual directories in IIS Manager. When you rename a physical directory, the actual folder name of the directory is changed. When you rename a virtual directory, the alias to the directory is changed. The name of the related physical directory isn't changed.

To rename a physical directory, follow these steps:

1. In IIS Manager, in the left pane, select the physical directory you want to rename. The directory icon should show a folder. If the directory icon appears as a folder shortcut or a globe with pages in front of it, you've incorrectly selected a virtual directory or application. Do not use this technique with virtual directories or applications.
2. In the Actions pane, click Edit Permissions. This displays the Properties dialog box for the directory.
3. On the General tab, type the new name for the directory in the text box, and then click OK.

Caution Browsers store file and directory paths in bookmarks. When you change a directory name, you invalidate any URL that references the directory in its path string. Because of this, renaming a directory might cause a return visitor to experience the 404 File Or Directory Not Found error. To resolve this problem, you might want to redirect browser requests to the new location by using the technique discussed in the "Redirecting Browser Requests" section of Chapter 7, "Customizing Web Server Content."

In IIS 7.0, you cannot rename virtual directories or applications through IIS Manager. The reason for this is that renaming a virtual directory or application would require several instance changes in the running IIS configuration. To rename a virtual directory, you could delete the existing virtual directory and then create a new one with the desired name. This won't preserve the original directory settings, however.

Changing Virtual Directory Paths, Logon Methods, and More

When you use virtual directories to access shared folders on remote servers, you can set the UNC path to use, logon credentials, and logon type. The logon credentials identify the user that should be impersonated when accessing the physical path for the virtual directory. The logon type specifies the type of logon operation to perform when acquiring the user token necessary to access the physical path. The logon types you can use are as follows:

- **ClearText** IIS uses a clear-text logon to acquire the user token. Because IIS passes the logon user call over the back end on an internal network, using a clear-text call is typically sufficient. This is the default logon type.
- **Interactive** IIS uses an interactive logon to acquire the user token. This gives the related account the Interactive identity for the logon session and makes it appear that the user is logged on locally.
- **Batch** IIS uses a batch logon to acquire the user token. This gives the related account the Batch identity for the logon session and makes it appear that the user is accessing the remote server as a batch job.
- **Network** IIS uses a network logon to acquire the user token. This gives the related account the Network identity for the logon session and makes it appear that the user is accessing the remote server over the network.

In IIS Manager, you can change a virtual directory's physical path, logon credentials, and logon type by completing the following steps:

When you navigate to a site node in IIS Manager and select a directory, the Actions pane displays a list of unique actions related to directories.

1. In IIS Manager, in the left pane, select the virtual directory, and then, in the Actions pane, click Advanced Settings. This displays the Advanced Settings dialog box.

2. Physical Path lists the current physical path for the virtual directory. To change the physical path, click in the column to the right, and then type the desired path. Alternately, click in the column to the right, and then click the selection button to display the Browse For Folder dialog box. Then use this dialog box to select the folder to use.
3. Physical Path Credentials lists the current logon credentials for the virtual directory. In most cases, only UNC paths require logon credentials. To change the logon credentials, click in the column to the right, and then click the selection button to display the Connect As dialog box. In the Connect As dialog box, choose Specific User, and then click Set. In the Set Credentials dialog box, type the name of the user account to use for authentication, type and confirm the account password, and then click OK.
4. Physical Path Credentials Logon Type lists the current logon type for the virtual directory. You need to set the logon type only when you've also set logon credentials. To change the logon type, click in the column to the right, and then in the drop-down list, select the desired logon type. Click OK to save your settings.

By using the IIS Command-line Administration Tool, you can configure a virtual directories path and logon details by running the Set Vdir command. Sample 6-15 provides the syntax and usage.

Sample 6-15 Set Vdir Syntax and Usage

Syntax

```
appcmd set vdir [[/vdir.name:]"VdirNameOrUrl"]
[/physicalPath:Path] [/logonMethod:Method] [/userName:User]
[/password:Password]
```

Usage

```
appcmd set vdir "Default Web Site/Invoices" /logonMethod:Network

appcmd set vdir /vdir.name:"Sales Site/Invoices"
/physicalPath:"c:\salesroot\invoices" /logonMethod:ClearText
/userName:SupportUser /password:RainyDayz
```

Deleting Directories

You can delete physical directories by using Windows Explorer. When you delete a physical directory, the directory and its contents are removed. When you delete local directories and files, Windows moves them to the Recycle Bin by default, but you can bypass the Recycle Bin by holding down the Shift key when deleting. You also can configure servers to bypass the Recycle Bin automatically when deleting (though this is not a recommended best practice).

You can delete virtual directories by using IIS Manager. When you delete a virtual directory, only the alias to the directory is removed. The actual contents of the related physical directory aren't changed.

To delete a virtual directory by using IIS Manager, follow these steps:

1. In the IIS Manager, right-click the virtual directory you want to delete, and on the shortcut menu, select Remove.
2. When asked to confirm the action, click Yes.

By using the IIS Command-line Administration Tool, you can delete a virtual directory by running the Delete Vdir command. Sample 6-16 provides the syntax and usage.

Sample 6-16 Delete Vdir Syntax and Usage

Syntax

```
appcmd delete vdir [[/vdir.name:]"VdirNameOrUrl"]
```

Usage

```
appcmd delete vdir "Default Web Site/Support"
```