

# IIS 10

## Essentials for Administration

William R. Stanek  
*Author & Series Editor*



William R. Stanek, Jr.  
*Author*

# IT Pro Solutions

## Notes from the Road



I've been a writer for 30 years, having finished my first novel in 1986, and I'm no stranger to the ups and downs that come with the business. Heck, even though my work won awards in the interim, I didn't truly break into publishing until 1995 when my first full-length work of nonfiction catapulted its way onto bestseller lists. That work was followed by a dozen other topsellers, nearly all of which were published by Macmillan and distributed to the world by Simon & Schuster, that is until the business turned and I found myself at a crossroads.

The year was 1998 and I turned away from full-time writing for a short while to work for a Seattle-based startup. Around the same time, I jumped ship from Macmillan and an opportunity to write for Microsoft arose. Microsoft was working on a new series of books called Pocket Consultants. They needed a writer who could write fast, clearly, concisely, authoritatively and just as important meet crazy timelines not just once or a few times but always. Surprisingly, always hitting timelines isn't something many writers can do while delivering quality work.

I went over to Microsoft without any hesitation. They loved my writing so much that my style became the Pocket Consultant style and soon I had not just one contract with them but three, then four. I not only hit my timelines while writing clearly, concisely and authoritatively, I consistently walloped them.

Microsoft loved this. Soon the Pocket Consultants and I were synonymous. In the years that followed, I wrote dozens. Not only were the books read in print by millions (thank you, readers!), articles and extracts from the books were posted on Microsoft websites and read by millions more.

To all the readers out there who miss my Pocket Consultants, great things are still happening. In my IT Pro Solutions series. In my Tech Artisans series. In my Administrator's Reference series. Hang in there with me as I blaze new trails with Stanek & Associates and we'll get to visit new places together.

—William R. Stanek



## Notes from the Road



I began my tech career as an intern at Stanek & Associates in 2007 and am currently working as a development and engineering lead for the company. While most were out enjoying spring break and summer vacations, I was working as an assistant on the publishing side of the business, before moving over to the technical side of the business during my college years at University of Washington.

I'm thankful to my father for sharing his knowledge with me for the past ten years. I've learned so much and am now honing my advanced skills with SQL Server, Exchange Server, Windows Server, IIS, PowerShell and more. Although I've made contributions to over a dozen other books I worked on with my father, this is the second book I have co-author credit for.

—William R. Stanek Jr.

# **IIS 10 Essentials for Administration**

Covers Internet Information Services (IIS) versions 10, 8, and 7 for  
Windows Server 2016, 2012 R2, and 2012  
Windows 10, 8.1, and 7

**IT Pro Solutions**

**William R. Stanek**  
Author & Series Editor

**William R. Stanek, Jr.**  
Author



# IIS 10 Essentials for Administration

## IT Pro Solutions

Published by Stanek & Associates  
PO Box 362, East Olympia, WA, 98540-0362  
[www.williamrstanek.com](http://www.williamrstanek.com)

Copyright © 2017 William R. Stanek. Seattle, Washington,  
All rights reserved.

No part of this book may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, except as permitted by Sections 107 or 108 of the 1976 United States Copyright Act, without the prior written permission of the publisher. Requests to the publisher for permission should be sent to the address listed previously.

Stanek & Associates is a trademark of Stanek & Associates and/or its affiliates. All other marks are the property of their respective owners. No association with any real company, organization, person or other named element is intended or should be inferred through use of company names, website addresses or screens.

This book expresses the views and opinions of the author. The information contained in this book is provided without any express, statutory or implied warranties.

**LIMIT OF LIABILITY/DISCLAIMER OF WARRANTY: THE PUBLISHER AND THE AUTHOR MAKE NO REPRESENTATIONS OR WARRANTIES WITH RESPECT TO THE ACCURACY OR COMPLETENESS OF THE CONTENTS OF THIS WORK AND SPECIFICALLY DISCLAIM ALL WARRANTIES, INCLUDING WITHOUT LIMITATION WARRANTIES OF FITNESS FOR A PARTICULAR PURPOSE. NO WARRANTY MAY BE CREATED OR EXTENDED BY SALES OR PROMOTIONAL MATERIALS. THE ADVICE AND DISCUSSION IN THIS BOOK MAY NOT BE SUITABLE FOR EVERY SITUATION. THIS WORK IS SOLD WITH THE UNDERSTANDING THAT THE PUBLISHER IS NOT ENGAGED IN RENDERING PROFESSIONAL SERVICES AND THAT SHOULD PROFESSIONAL ASSISTANCE BE REQUIRED THE SERVICES OF A COMPETENT PROFESSIONAL SHOULD BE SOUGHT. NEITHER THE PUBLISHERS, AUTHORS, RESELLERS NOR DISTRIBUTORS SHALL BE HELD LIABLE FOR ANY DAMAGES CAUSED OR ALLEGED TO BE CAUSED EITHER DIRECTLY OR INDIRECTLY HEREFROM. THE REFERENCE OF AN ORGANIZATION OR WEBSITE AS A SOURCE OF FURTHER INFORMATION DOES NOT MEAN THAT THE PUBLISHER OR THE AUTHOR ENDORSES THE INFORMATION THE ORGANIZATION OR WEBSITE MAY**



PROVIDE OR THE RECOMMENDATIONS IT MAY MAKE. FURTHER, READERS SHOULD BE AWARE THAT WEBSITES LISTED IN THIS BOOK MAY NOT BE AVAILABLE OR MAY HAVE CHANGED SINCE THIS WORK WAS WRITTEN.

Stanek & Associates publishes in a variety of formats, including print, electronic and by print-on-demand. Some materials included with standard print editions may not be included in electronic or print-on-demand editions or vice versa.

**Country of First Publication:** United States of America.

**Cover Design:** Creative Designs Ltd.

**Editorial Development:** Andover Publishing Solutions

**Technical Review:** L & L Technical Content Services

You can provide feedback related to this book by emailing the author at [williamstanek@aol.com](mailto:williamstanek@aol.com). Please use the name of the book as the subject line.

Version: 1.0.3.4a

<p><b>Note</b> I may periodically update this text and the version number shown above will let you know which version you are working with. If there's a specific feature you'd like me to write about in an update, message me on Facebook (<a href="http://facebook.com/williamstanekauthor">http://facebook.com/williamstanekauthor</a>). Please keep in mind readership of this book determines how much time I can dedicate to it.</p>
---

**Thank you for buying this book...**

Find out about special offers, free book giveaways, amazing deals, and exclusive content. Plus get updates on favorite books and more when you join William Stanek on Facebook at <http://facebook.com/William.Stanek.Author>. William's on twitter at <http://twitter.com/williamstanek>.

Connect with Will by visiting him on LinkedIn @ <http://linkedin.com/in/will-stanek/>.



# Table of Contents

## **Notes from the Road**

## **How to Use This Guide**

[Print Readers](#)

[Digital Book Readers](#)

[Support Information](#)

[Conventions & Features](#)

[Share & Stay in Touch](#)

## **Part 1. Deploying IIS 10**

### **Chapter 1. IIS 10 Running Start**

#### **Introducing IIS 10**

[IIS Versions and Installable Components](#)

[IIS Architecture and Features](#)

[Web Management Service](#)

[IIS and PowerShell](#)

#### **Sizing Your Hardware**

#### **IIS Administration Essentials**

[Managing IIS Resources](#)

[Connecting for Administration](#)

[Connecting to Remote Servers](#)

[Connecting to Remote websites](#)

[Connecting to Azure Sites](#)

[Connecting to Web Apps](#)

### **Chapter 2. Planning for IIS 10**

#### **Navigating IIS Protocols**

[HTTP and SSL](#)

[FTP](#)

[SMTP](#)

#### **Understanding IIS Roles**

#### **Navigating the Installation Options**

[Using Application Servers](#)

[Using Web Servers](#)

## **[Chapter 3. Setting Up IIS](#)**

[Installing Application Servers](#)

[Installing Web Servers](#)

[Installing IIS on Windows Desktops](#)

[Managing Installed Roles and Role Services](#)

[Viewing Configured Roles and Role Services](#)

[Adding or Removing Roles on Servers](#)

[Viewing and Modifying Role Services](#)

## **[Part 2. Core IIS 10 Administration](#)**

### **[Chapter 4. Navigating IIS 10 Architecture](#)**

[Working with IIS and URLs](#)

[Understanding the Core Architecture](#)

[Working with websites](#)

[Working with Web Applications and Virtual Directories](#)

[Controlling Access to Servers, Sites, and Apps](#)

[Understanding Services and Processing](#)

[Essential IIS Services and Processes](#)

[IIS Worker Process Isolation Mode](#)

[Understanding and Using IIS Applications](#)

[Understanding and Using ASP.NET Applications](#)

### **[Chapter 5. Managing IIS Servers & Services](#)**

[IIS Management Essentials](#)

[Using IIS Manager](#)

[Navigating Nodes](#)

[Grouping Tasks](#)

[Expanding Nodes](#)

[Configuring Remote Administration](#)

[Adding the Management Service](#)

[Enabling Remote Administration](#)

[Restarting the Management Service](#)

## Managing IIS Services

Starting, Stopping, and Pausing IIS Services

Configuring Service Startup

Configuring Service Recovery

## Other Management Options for IIS Services

Going Old School

Using IIS Manager

Rebooting IIS Servers

## **Chapter 6. Managing IIS 10 from the Prompt**

Command Line Administration

## Using Windows PowerShell

Introducing Windows PowerShell

Running and Using Windows PowerShell

Running and Using Cmdlets

Running and Using Other Commands and Utilities

## Working with Cmdlets

Using Windows PowerShell Cmdlets

Using Cmdlet Parameters

Understanding Cmdlet Errors

Using Cmdlet Aliases

Using Cmdlets with IIS

## **Chapter 7. Using Management Objects for Administration**

## Getting Started with AppCmd

Navigating the .NET Framework Objects

Using AppCmd

## Configuring IIS using Management Objects

Configuration Management Commands

Using Module Management Commands

Site Management Commands

Application Pool Management Commands

Application Management Commands

Virtual Directory Management Commands

[Utility Commands](#)

## **[Chapter 8. Digging into IIS Schema](#)**

[Working with the IIS Configuration System](#)

[Extending XML Schema](#)

[Navigating XML Schema](#)

[IIS Global Configuration](#)

[Configuration Settings](#)

[Configuration Hierarchy](#)

[Including Configuration Files](#)

[Using Sections Groups](#)

[Using Sections](#)

## **[Chapter 9. Managing Global IIS Configuration](#)**

[Understanding Configuration Levels](#)

[Managing Configuration Sections](#)

[Working with Configuration Sections](#)

[Determining Settings for a Configuration Section](#)

[Modifying Settings for a Configuration Section](#)

[Locking and Unlocking Configuration Sections](#)

[Clearing and Resetting Configuration Sections](#)

[Extending IIS with Modules](#)

[Controlling Native Modules through the Configuration Files](#)

[Controlling Managed Modules through the Configuration Files](#)

[Controlling Managed Handlers through the Configuration Files](#)

[Using the Configuration and Schema Files to Install Non-Standard Extension Modules](#)

[Managing Modules](#)

[Viewing Installed Native and Managed Modules](#)

[Installing Native Modules](#)

[Enabling Native Modules](#)

[Enabling Managed Modules](#)

[Editing Native and Managed Module Configurations](#)

[Disabling Native and Managed Modules](#)

[Uninstalling Native Modules](#)

[Sharing Global Configuration](#)

[Working with Shared Configurations](#)

[Exporting and Sharing Global Configuration](#)

### **[Part 3. Creating Customized IIS Solutions](#)**

#### **[Chapter 10. Building Dynamic Websites](#)**

[Configuring IP Addresses and Name Resolution](#)

[Working with Private and Public Networks](#)

[Understanding website Identifiers](#)

[Hosting Multiple Sites on a Single Server](#)

[Checking the Computer Name and IP Address of Servers](#)

[Examining Site Configuration](#)

[Creating Websites](#)

[Creating a Website: The Essentials](#)

[Creating an Unsecured Website](#)

[Creating a Secured Website](#)

[Managing Websites and Their Properties](#)

[Working with Sites in IIS Manager](#)

[Configuring an Application Pool and Home Directory](#)

[Configuring Ports, IP Addresses, and Host Names](#)

[Restricting Incoming Connections and Setting Time-Outs](#)

[Configuring HTTP Keep-Alives](#)

[Configuring Access Permissions in IIS Manager](#)

[Managing the Numeric Identifier and AutoStart State](#)

[Deleting Sites](#)

#### **[Chapter 11. Configuring Directories for Websites](#)**

[Working with Physical and Virtual Directories](#)

[Examining Virtual Directory Configuration](#)

[Creating Physical Directories](#)

[Creating Virtual Directories](#)

[Managing Directories and Their Properties](#)

[Enabling or Disabling Directory Browsing](#)



[Modifying Directory Properties](#)

[Renaming Directories](#)

[Changing Virtual Directory Paths and Logon Methods](#)

[Deleting Directories](#)

## **[Chapter 12. Customizing Web Server Content](#)**

[Managing Web Content](#)

[Opening and Browsing Files](#)

[Modifying the IIS Properties of Files](#)

[Renaming Files](#)

[Deleting Files](#)

[Redirecting Browser Requests](#)

[Redirecting Requests to Other Directories or websites](#)

[Redirecting All Requests to Another website](#)

[Redirecting Requests to Applications](#)

[Customizing Browser Redirection](#)

[Customizing Website Content and HTTP Headers](#)

[Configuring Default Documents](#)

[Configuring Document Footers](#)

[Configuring Included Files](#)

[Using Content Expiration and Preventing Browser Caching](#)

[Enabling Content Expiration](#)

[Disabling Content Expiration](#)

[Using Custom HTTP Headers](#)

[Using Content Ratings and Privacy Policies](#)

[Improving Performance with Compression](#)

[Configuring Content Compression for an Entire Server](#)

[Enabling or Disabling Content Compression for Sites and Directories](#)

[Customizing Web Server Error Messages](#)

[Understanding Status Codes and Error Messages](#)

[Managing Custom Error Settings](#)

[Viewing and Configuring Custom Error Settings](#)

[Adding, Changing, and Removing Custom Error Responses](#)

[Using MIME and Configuring Custom File Types](#)

[Understanding MIME](#)

[Viewing and Configuring MIME Types](#)

[Additional Customization Tips](#)

[Using Update Sites to Manage Outages](#)

[Using Jump Pages for Advertising](#)

[Handling 404 Errors and Preventing Dead Ends](#)

## **[Chapter 13. Web Stats 101](#)**

[Logging Access Stats](#)

[Navigating File Formats](#)

[Choosing Your Logging Configuration](#)

[Working with the NCSA Common Log File Format](#)

[Host Field](#)

[Identification Field](#)

[User Authentication Field](#)

[Time Stamp Field](#)

[HTTP Request Field](#)

[Status Code Field](#)

[Transfer Volume Field](#)

[Working with the Microsoft IIS Log File Format](#)

[Working with the W3C Extended Log File Format](#)

[Working with ODBC Logging](#)

[Working with Centralized Binary Logging](#)

[Understanding Logging](#)

## **[Chapter 14. Configuring Logging](#)**

[Logging Essentials](#)

[Configuring the NCSA Common Logs](#)

[Configuring Microsoft IIS Logs](#)

[Configuring W3C Extended Logs](#)

[Configuring ODBC Logging](#)

[Creating a Logging Database and Table](#)

[Creating a DSN for SQL Server](#)

[Configuring ODBC Logging in IIS](#)

[Configuring Centralized Binary Logging](#)

[Disabling Logging](#)





## How to Use This Guide

This book is designed to provide the tools and guidance you need to get the most out of Internet Information Services (IIS). The first chapter, IIS 10 Running Start, takes you through the essentials for working with the technology. Following this are chapters that will take an in-depth look at specific tasks and aspects of IIS.

Not only has William Stanek been developing expert solutions and writing professionally about IIS for many years, he's also written a number of bestselling books on IIS, web publishing and web servers. In this book, William shares his extensive knowledge, delivering ready answers for day-to-day usage while zeroing in on core commands and techniques.

As with all books in the IT Pro Solutions series, this book is written especially for IT professionals working with, supporting, and managing specific versions of Microsoft products. Here, this means the IIS versions that shipped with various Windows operating systems, including:

- IIS 10, which is part of Windows 10 and Windows Server 2016.
- IIS 8, which is part of Windows 8.1, Windows Server 2012 and Windows Server 2012 R2.
- IIS 7, which is part of early Windows and Windows Server operating systems, including Windows 7, Windows Server 2008, and Windows Server 2008 R2.

Odds are, if you are using any of these operating systems, these are the versions of IIS available. Keep in mind, however, that Microsoft releases IIS versions with different major and minor version numbers. While the first major release of the product has the .0 minor version, the second major release typically has the .5 minor version. This is why you'll see that IIS 7.0 and IIS 7.5 are both available, as are IIS 8.0 and IIS 8.5, IIS 10.0, etc.

Don't worry if you are working with a dot revision version of IIS. All versions of IIS available for all current versions of Windows and Windows Server are substantially similar. You can hone your skills with any base version and use these skills with any higher version.

## Print Readers

Print editions of this book include an index and some other elements not available in the digital edition. Updates to this book are available online. Visit <http://www.williamrstanek.com/iis/> to get any updates. This content is available to all readers.





## Digital Book Readers

Digital editions of this book are available at all major retailers, at libraries upon request and with many subscription services. If you have a digital edition of this book that you downloaded elsewhere, such as a file sharing site, you should know that the author doesn't receive any royalties or income from such downloads.

## Support Information

Every effort has been made to ensure the accuracy of the contents of this book. As corrections are received or changes are made, they will be added to the online page for the book available at:

<http://www.williamrstanek.com/iis/>

If you have comments, questions, or ideas regarding the book, or questions that are not answered by visiting the site above, send them via e-mail to:

[williamstanek@aol.com](mailto:williamstanek@aol.com)

It's important to keep in mind that Microsoft software product support is not offered. If you have questions about Microsoft software or need product support, please contact Microsoft.

Microsoft also offers software product support through the Microsoft Knowledge Base at:

<http://support.microsoft.com/>

# Conventions & Features

This book uses a variety of elements to help keep the text clear and easy to follow. You'll find code terms and listings in monospace, except when you are told to actually enter or type a command. In that case, the command appears in **bold**. When new terms are introduced and defined, they are put in italics.

The first letters of the names of menus, dialog boxes, user interface elements, and commands are capitalized. Example: the Add Roles And Features Wizard. This book also has notes, tips and other sidebar elements that provide additional details on points that need emphasis.

Keep in mind that throughout this book, where click, right-click or double-click is used, you also can use touch equivalents: tap, press and hold, or double tap. Also, when using a device without a physical keyboard, you are able to enter text by using the onscreen keyboard. If a device has no physical keyboard, simply touch an input area on the screen to display the onscreen keyboard.

## Share & Stay in Touch



The marketplace for technology books has changed substantially over the past few years. In addition to becoming increasingly specialized and segmented, the market has been shrinking rapidly, making it extremely difficult for books to find success. To ensure the books you need for your career remain available, raise your voice and support this work.

Without support from you, the reader, future books will not be possible. Your voice matters. If you found the book to be useful, informative or otherwise helpful, please take the time to let others know by sharing about the book online.

To stay in touch, use Facebook or Twitter. Your messages and comments about the book, especially suggestions for improvements and additions, are always welcome. If there is a topic you think should be covered in the book, write to the email address provided.

**IMPORTANT** The focus of this book is on the core features of IIS. As this book is designed to be clear and concise but not necessarily exhaustive, you may find that you need additional discussion or materials to supplement this text. With this in mind, note that a hands-on companion text is available, called IIS 10: Web Apps, Security & Maintenance. IIS 10: Web Apps, Security & Maintenance picks up right where this book leaves off and puts IIS to work.

**MORE INFO** As with all books William writes, the success and sales of this book determine how much time he can dedicate to updates, revisions and extras. Your suggestions are always welcome. If you have suggestions for additions or

changes to this book, please write. Be sure to reference the full title and edition of the book in your initial correspondence.

# Part 1. Deploying IIS 10

Chapter 1. IIS 10 Running Start

Chapter 2. Planning for IIS

Chapter 3. Setting Up IIS

Chances are that if you work with Windows or Windows Server, you've heard of IIS. You may have even read other books about IIS and put IIS to work. However, you probably still have many questions or you simply may be curious about options available. This book aims to answer your questions about IIS and the options available.

Part 1 focuses on the essentials. You'll learn how to deploy and work with IIS whether for on-premises, hybrid or cloud support. For skilled technologists and engineers, IIS has become increasingly indispensable. You use the technology for hosting internal websites (intranets), external websites accessible to people inside your company (extranets), hosted websites for consumers and much more.

Knowing how to use IIS properly can save you time and effort and can mean the difference between smooth-running operations and frequent problems. Moreover, if you're responsible for websites or web apps, whether they're running on virtual or physical machines in the cloud or on premises, learning the timesaving strategies discussed in this book is not just important, it's essential for daily operations.

**REAL WORLD** Although this book zeroes in on IIS 10, you can use the techniques that you learn in this book with any version of IIS with which you are working, including IIS 8 and 7. The functional core of IIS is the same whether you are working with version 10, 8 or 7. This means you typically can use skills and tools developed for an earlier IIS version with a more current version. However, when you are working across operating systems or versions, you should always test your sites and apps in a development or test environment, where the computers with which you are working are isolated from the rest of the network, before using them in live production environments.



# Chapter 1. IIS 10 Running Start

IIS 10 is the latest release version of Microsoft's web server. IIS provides a suite of services for hosting web applications, streaming media and much more. If you've worked with earlier releases of IIS, you probably want to know how IIS 10 is different. You also may want to know what the heck happened to version 9. Don't worry, you'll learn the answers to these questions and more in this chapter.



## Introducing IIS 10

In today's connected world, especially when you are working remotely, much of your operations are likely running via IIS, whether you realize it or not. IIS even provides the gateway that enables remote administration with PowerShell. With so much of what you do relying on IIS, it's critical that you know how IIS works.

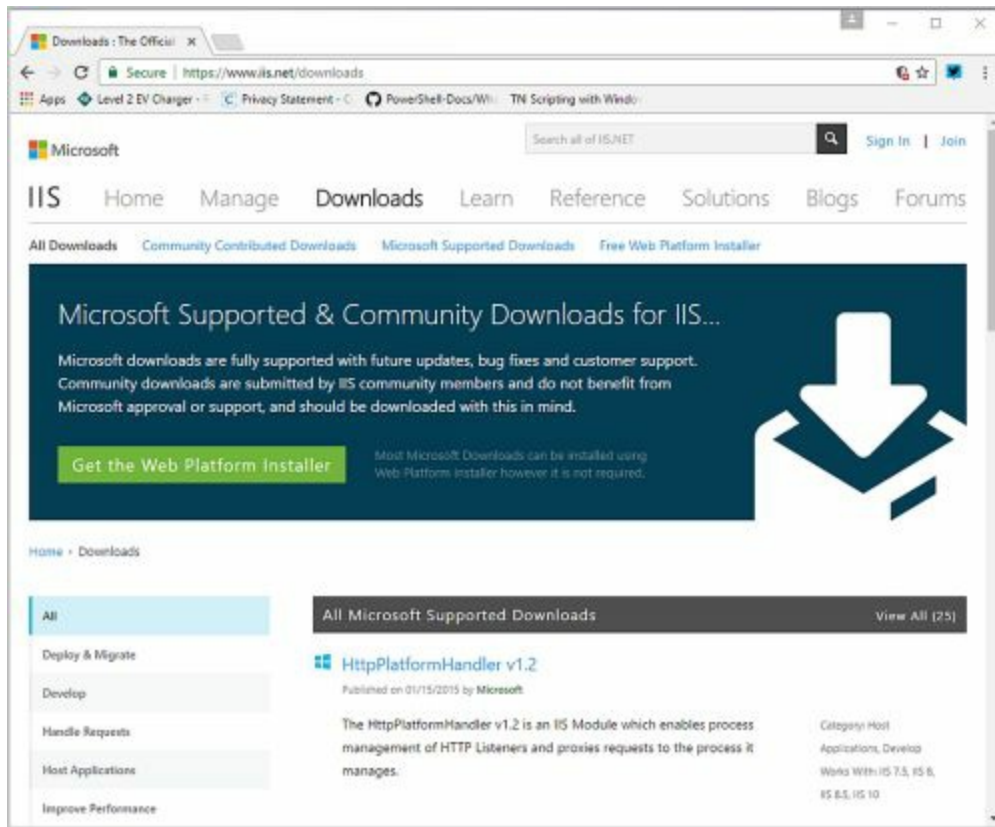
# IIS Versions and Installable Components

Like Windows itself, Microsoft has been developing IIS for many years. Generally, this is good news for IT professionals as it means the product is constantly evolving, but it also means that you need to keep up with the changes to stay current. As you get started with IIS, it's important to note that more so than any other release, IIS 10 is an evolving product. Microsoft has and will continue to make updates to IIS 10 available whether as part of the Windows operating system or as downloads from the Microsoft website. As examples, the Anniversary and Creator updates for Windows 10 and Windows Server 2016 include incremental updates for IIS 10.

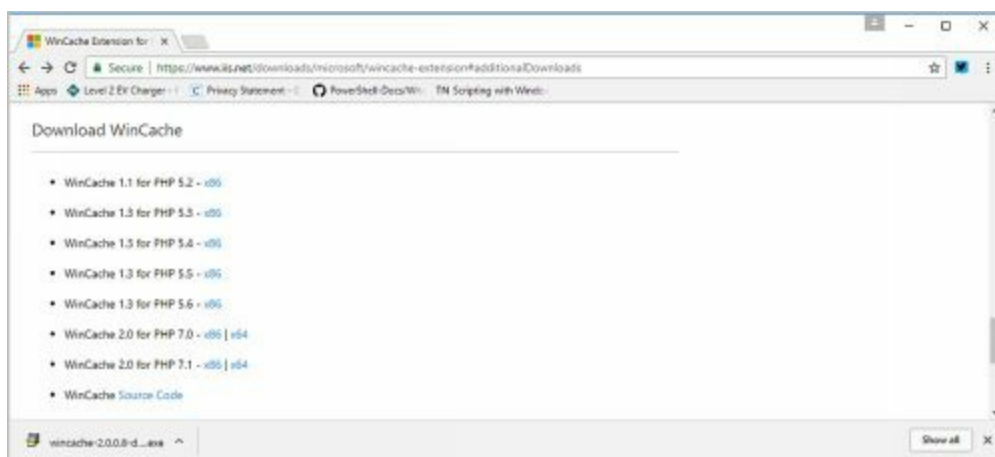
Typically, Microsoft releases dot revisions of the product as well. As examples, both IIS 7 and IIS 8 have minor version updates. IIS 7.0 is part of Windows Vista and Windows Server 2008. IIS 7.5 is part of Windows 7 and Windows Server 2008 R2. IIS 8.0 is part of Windows 8 and Windows Server 2012. IIS 8.5 is part of Windows 8.1 and Windows Server 2012 R2.

Because there are dot release versions of IIS, I try to use a general reference to versions whenever possible, such as saying version 10 instead of version 10.0. This ensures any generally applicable references to version are accurate. As necessary, however, I will refer to specific dot release versions, say version 8.5 as opposed to version 8.0.

Although standard core components are always included in the base installation of IIS, additional components are available for download via the IIS.Net website at [iis.net/downloads](http://iis.net/downloads). These components include community contributed components and Microsoft supported components.



Most of the additional components are available from third party contributors rather than Microsoft. On the IIS.Net website, components are listed by category and the IIS versions they work with. Before downloading any additional components, ensure that the version of IIS you are working with is supported. To download a component, access the component page, scroll down to the Download area and then click the link for the appropriate CPU architecture version, either x86 or x64.



After you download the self-extracting executable or other installation file, transfer the download to your server and install it. For a self-extracting executable, simply double-clicking it will start the installation process. For archived files, such as .zip files, you need to extract the files to a folder and then look for the installer.

# IIS Architecture and Features

IIS 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. Modules are either IIS native modules that use a Win32 DLL or IIS 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.

IIS uses 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 includes additional 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 has many other features, but few are as important as the administration tools, including related graphical and non-graphical administration tools. The graphical administration tool uses a browser-like interface and supports delegated administration, remote administration over Secure HTTP (HTTPS), and extensibility through custom user interface components. The non-graphical administration tools allow you to manage IIS from a prompt, whether locally or remotely.



IIS Manager is the graphical user interface (GUI) for managing both local and remote installations of IIS. 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.

IIS uses 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.

# Web Management Service

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.

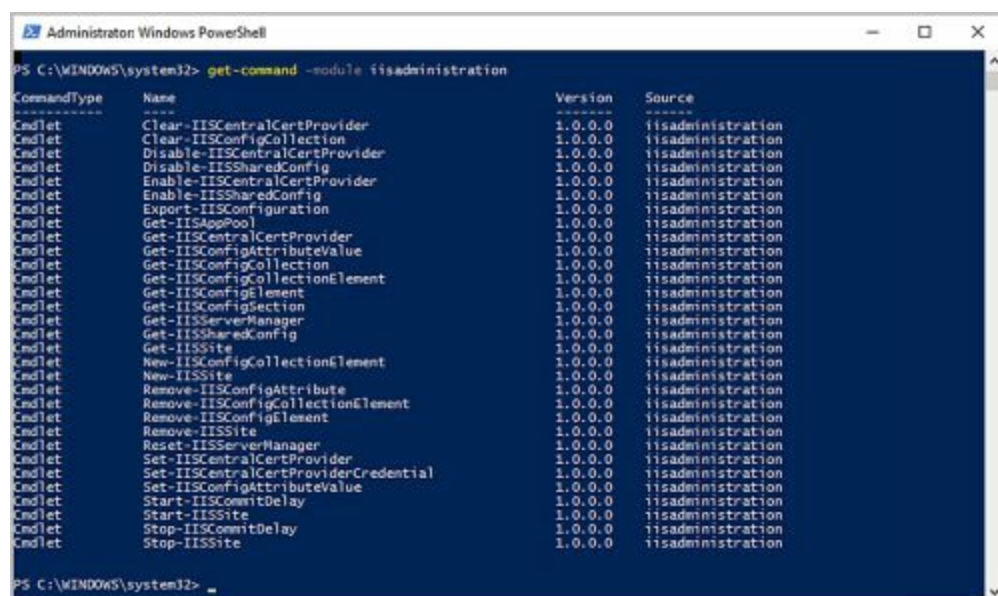
The Web Management Service stores a limited set of editable configuration values in the registry. Each time the service is started, the Web configuration files are regenerated in the following directory: %SystemRoot%\ServiceProfiles\LocalService\AppData\Local\Temp\WMSvc. To enhance security, WMSVC requires 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 “Configuring Remote Administration” in Chapter 5, “Managing IIS Servers & Services,” for more information.

<p><b>NOTE</b> %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%.</p>
---

# IIS and PowerShell

If you worked with IIS 8 or 7, you know IIS Manager and the base architecture haven't changed much as far as general administration goes. What has changed considerably are your management options, especially at the prompt. Although you can continue to use old administration standbys, IIS 10 introduces a new PowerShell module: the IISAdministration module.

The IISAdministration module simplifies administration while providing new functionality and enhancements. Rather than completely replacing the WebAdministration module for administration, Microsoft decided to allow organizations to transition to the new module. This means you can continue using the WebAdministration module for administration while you are familiarizing yourself with the options available with the IISAdministration module.



```
PS C:\WINDOWS\system32> get-command -module iisadministration
```

CommandType	Name	Version	Source
Cmdlet	Clear-IISCentralCertProvider	1.0.0.0	iisadministration
Cmdlet	Clear-IISConfigCollection	1.0.0.0	iisadministration
Cmdlet	Disable-IISCentralCertProvider	1.0.0.0	iisadministration
Cmdlet	Disable-IISSharedConfig	1.0.0.0	iisadministration
Cmdlet	Enable-IISCentralCertProvider	1.0.0.0	iisadministration
Cmdlet	Enable-IISSharedConfig	1.0.0.0	iisadministration
Cmdlet	Export-IISConfiguration	1.0.0.0	iisadministration
Cmdlet	Get-IISAppPool	1.0.0.0	iisadministration
Cmdlet	Get-IISCentralCertProvider	1.0.0.0	iisadministration
Cmdlet	Get-IISConfigAttribute	1.0.0.0	iisadministration
Cmdlet	Get-IISConfigCollection	1.0.0.0	iisadministration
Cmdlet	Get-IISConfigCollectionElement	1.0.0.0	iisadministration
Cmdlet	Get-IISConfigElement	1.0.0.0	iisadministration
Cmdlet	Get-IISConfigSection	1.0.0.0	iisadministration
Cmdlet	Get-IISServerManager	1.0.0.0	iisadministration
Cmdlet	Get-IISSharedConfig	1.0.0.0	iisadministration
Cmdlet	Get-IISSite	1.0.0.0	iisadministration
Cmdlet	New-IISConfigCollectionElement	1.0.0.0	iisadministration
Cmdlet	New-IISSite	1.0.0.0	iisadministration
Cmdlet	Remove-IISConfigAttribute	1.0.0.0	iisadministration
Cmdlet	Remove-IISConfigCollectionElement	1.0.0.0	iisadministration
Cmdlet	Remove-IISConfigElement	1.0.0.0	iisadministration
Cmdlet	Remove-IISSite	1.0.0.0	iisadministration
Cmdlet	Reset-IISServerManager	1.0.0.0	iisadministration
Cmdlet	Set-IISCentralCertProvider	1.0.0.0	iisadministration
Cmdlet	Set-IISCentralCertProviderCredential	1.0.0.0	iisadministration
Cmdlet	Set-IISConfigAttribute	1.0.0.0	iisadministration
Cmdlet	Set-IISConfigAttribute	1.0.0.0	iisadministration
Cmdlet	Start-IISCommitDelay	1.0.0.0	iisadministration
Cmdlet	Start-IISSite	1.0.0.0	iisadministration
Cmdlet	Stop-IISCommitDelay	1.0.0.0	iisadministration
Cmdlet	Stop-IISSite	1.0.0.0	iisadministration

**NOTE** In a hurry to learn more right now? Type the following command at a PowerShell prompt: **get-command -module IISAdministration**. You'll then get a list of all related cmdlets. To learn more about a cmdlet, enter **get-help cmdlet-name -detailed** where cmdlet-name is the name of the cmdlet you want to learn more about, such as **get-help Get-IISAppPool -detailed**.

# Sizing Your Hardware

Before you deploy IIS, 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.

Guidelines for choosing hardware for web 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 8 gigabyte (GB) of RAM. High-volume servers should have a minimum of 16 to 32 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.

- **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. 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 websites 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. 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. Keep in mind IIS 10 running on Windows 10 is primarily meant for testing and development. Because of this, you shouldn’t use IIS 10 running on desktop editions for hosting live, production websites and web applications.

## IIS Administration Essentials

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 IIS Resources

Many tools are available for managing Web resources. Key tools you'll use are shown in Table 1-1. Most of these tools are available on the Tools menu in Server Manager. In Server Manager, click Tools, and then choose 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 1-1** Quick Reference for Key Web Administration Tools

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.
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 10.0.
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.

# Connecting for Administration

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 WMI provider
- IIS command-line administration tool (AppCmd.exe)
- IIS PowerShell cmdlets

IIS Manager provides the standard administration interface for IIS. To start IIS Manager, click the Tools menu in Server Manager, and then click Internet Information Services (IIS) Manager. When started, IIS Manager displays the Start page 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 website on a designated Web server
- **Connect to an application** Allows you to connect to a specific web application on a designated site and server



## Connecting to Remote Servers

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.imaginedlands.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 "Configuring Remote Administration" section of Chapter 5.

After you connect to a remote server, IIS Manager automatically connects to the server upon startup in the future. You can change this behavior by disconnecting from the server while in IIS Manager. See "Using IIS Manager" in Chapter 5 for more information.

## Connecting to Remote websites

You can connect to a specific website 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 website to which you want to connect, such as `Default website`.
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.

After you connect to a remote site, IIS Manager automatically connects to the site upon startup in the future. You can change this behavior by disconnecting from the server or

site while in IIS Manager.

## Connecting to Azure Sites

Using the Connect To A Site Wizard, you can connect to sites running on Microsoft Azure and then remotely manage the sites. The feature that allows you to do this is the Site Control Management (SCM) extension, which is automatically configured for Microsoft Azure sites. You can connect to a Microsoft Azure 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. In the Server Name box, type the SCM URL for the Azure website to which you want to connect. The SCM URL has .SCM added after the site name for the URL. For example, if your site is ImaginedLands and your site URL is `imaginedlands.azurewebsites.net`, your SCM URL is `imaginedlands.scm.azurewebsites.net`. As you must connect via SSL, you also must specify the SSL port in the URL. For example, if the ImaginedLands site uses SSL port 443, you'd enter **`imaginedlands.scm.azurewebsites.net:443`** as the server name.
3. In the Site Name box, type or select the name of the website to which you want to connect, such as ImaginedLands.
4. Click Next. Enter your Azure deployment credentials to log on to the server. If you have changed the automatically generated password for Microsoft Azure, you can reset or retrieve the current password from the Microsoft Azure portal. Click Next and then click Finish to complete the connection.

**REAL WORLD** With Microsoft Azure, the default user name for deployment is the site name with a dollar sign (\$) prepended. For example, if the site name is ImaginedLands, the default user name is \$imaginedlands. In the Microsoft Azure portal, go to the site's dashboard to manage the password. Select Reset Your Deployment Credentials to change the password. Select Download The Publish Profile to download an XML file containing the current password.

After you connect to an Azure site, IIS Manager automatically connects to the site upon startup in the future. You can change this behavior by disconnecting from the site while in IIS Manager.

## Connecting to Web Apps

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 website to which you want to connect, such as Default website.
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.

After you connect to a web app, IIS Manager automatically connects to the app upon startup in the future. You can change this behavior by disconnecting while in IIS Manager.



## Chapter 2. Planning for IIS 10

Before you deploy Internet Information Services (IIS), you should carefully plan the machine and administration architecture. As part of your planning, you need to look closely at the protocols and roles IIS will use and modify both server hardware and technology infrastructure accordingly to meet the requirements of these roles on a per-machine basis. Your early success with IIS will largely depend on your understanding of the ways you can use the software and in your ability to deploy it to support these roles.

# Navigating IIS Protocols

TCP/IP is a protocol suite consisting of Transmission Control Protocol (TCP) and Internet Protocol (IP). TCP/IP is required for internetwork communications and for accessing the Internet. Whereas TCP operates at the transport layer and is a connection-oriented protocol designed for reliable end-to-end communications, IP operates at the network layer and is an internetworking protocol used to route packets of data over a network.

IIS uses protocols that build on TCP/IP, including:

- [Hypertext Transfer Protocol \(HTTP\)](#)
- [Secure Sockets Layer \(SSL\)](#)
- [File Transfer Protocol \(FTP\)](#)
- [Simple Mail Transfer Protocol \(SMTP\)](#)

# HTTP and SSL

As you probably already know, HTTP is an application-layer protocol that makes it possible to publish static and dynamic content on a server so that it can be viewed in client applications, such as Internet Explorer. Publishing a Web document is a simple matter of making the document available in the appropriate directory on an HTTP server and assigning the appropriate permissions so that an HTTP client application can access the document. An HTTP session works like this:

1. The HTTP client application uses TCP to establish a connection to the HTTP server. The default (well-known) port used for HTTP connections is TCP port 80. You can configure servers to use other ports as well. For example, TCP port 8080 is a popular alternative to TCP port 80 for sites that are meant to have limited access.
2. After connecting to the server, the HTTP client application requests a Web page or other resource from the server. In the client application, users specify the pages or resources they want to access by using a Web address, otherwise known as a Uniform Resource Locator (URL).
3. The server responds to the request by sending the client the request resource and any other related files, such as images, that you've inserted into the requested resource. If you've enabled the HTTP Keep-Alive feature on the server, the TCP connection between the client and server remains open to speed up the transfer process for subsequent client requests. Otherwise, the TCP connection between the client and server is closed and the client must establish a new connection for subsequent transfer requests.

That in a nutshell is essentially how HTTP works. The protocol is meant to be simple yet dynamic, and it is the basis upon which the World Wide Web is built.

With HTTP, you can configure access to documents so that anyone can access a document or so that documents can be accessed only by authorized individuals. To allow anyone to access a document, you configure the document security so that clients can use Anonymous authentication. With Anonymous authentication, the HTTP server logs on the user automatically using a guest account, such as IUSR. To require authorization to access a document, configure the document security to require authentication using one of the available authentication mechanisms, such as Basic authentication, which requires a user to type a user name and password.

You can use Secure Sockets Layer (SSL) to enable Hypertext Transfer Protocol Secure (HTTPS) transfers. SSL is an Internet protocol used to encrypt authentication information and data transfers passed between HTTP clients and HTTP servers. With

SSL, HTTP clients connect to Web pages using URLs that begin with https://. The https prefix tells the HTTP client to try to establish a connection using SSL. The default port used with secure connections is TCP port 443 rather than TCP port 80.

All current versions of IIS support HTTP 1.1. IIS 10 also supports HTTP 2.0, which includes many new features and enhancements, including:

- Multiplexed connections, which allow one connection to be used for multiple requests from a client.
- HTTP header compression in addition to standard HTTP compression to reduce the amount of data sent to clients.
- Server push to improve performance by predicting requests clients are likely to make and pushing the necessary data to clients before it is requested.

**NOTE** Currently, HTTP 2.0 is used only when clients are using TLS with secure HTTP (HTTPS) connections. If a client isn't in a secure HTTP session or the browser running on the client doesn't support HTTP 2.0, the client will use HTTP 1.1, which is the standard default HTTP version.

**MORE INFO** Currently, HTTP 2.0 as implemented doesn't support Windows authentication with NTLM or Kerberos. As a result, when Windows authentication is used, clients will use HTTP 1.1 instead of HTTP 2.0. Additionally, it's important to note that HTTP 2.0 doesn't support HTTP inlining, domain sharding or bandwidth throttling, which are optimizations specific to HTTP 1.1.

As an administrator, you don't need to do anything special to ensure HTTP 2.0 is available. HTTP 2.0 is automatically enabled when you install IIS 10 and will be used whenever clients with compatible browsers establish HTTPS connections with TLS. Most current browsers support HTTP 2.0 and you can confirm whether HTTP 2.0 is being used by examining the protocol logs, as discussed in Chapters 13 and 14.

# FTP

FTP is an application-layer protocol that makes it possible for client applications to retrieve files from or transfer files to remote servers. FTP predates HTTP, and its usage is in decline as compared to HTTP. With FTP, you can publish a file so that a client can download it by making the file available in the appropriate directory on an FTP server and assigning the appropriate permissions so that an FTP client application can access the document. To upload a file to an FTP server, you must grant an FTP client application permission to log on to the server and access directories used for uploading files.

An FTP session works like this:

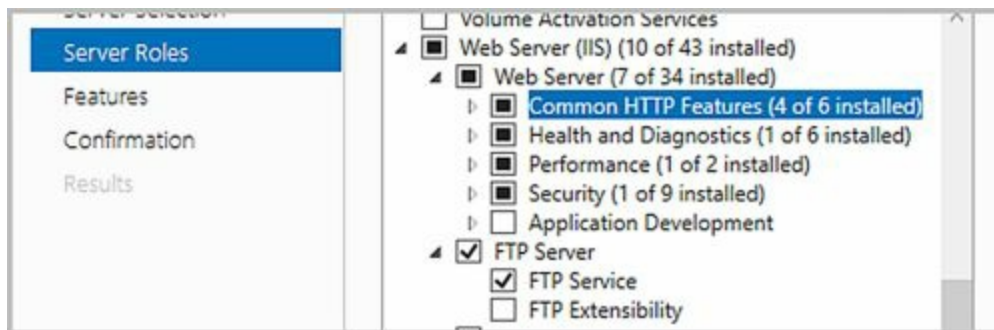
1. The FTP client application uses TCP to establish a connection to the FTP server. The default (well-known) port used for FTP connections is TCP port 21. FTP servers listen on this port for client connection requests. After the client and server establish a connection, the server randomly assigns the client a TCP port number above 1023. This initial TCP connection (with port 21 for the server and a random port for the client) is then used for transmission of FTP control information, such as commands sent from the client to the server and response codes returned by the server to the client.
2. The client then issues an FTP command to the server on TCP port 21. Standard FTP commands include GET for downloading a file, CD for changing directories, PUT for uploading files, and BIN for switching to binary mode.
3. When the client initiates a data transfer with the server, the server opens a second TCP connection with the client for the data transfer. This connection uses TCP port 20 on the server and a randomly assigned TCP port above 1023 on the client. After the data transfer is complete, the second connection goes in a wait state until the client initiates another data transfer or the connection times out.

That in a nutshell is how FTP works. As you can see, FTP is a bit clunkier than HTTP, but it is still fairly simple.

<p><b>REAL WORLD</b> What sets FTP and HTTP apart is primarily the way you transfer files. FTP transfers files as either standard text or encoded binaries. HTTP has the capability to communicate the file format to the client, and this capability allows the client to determine how to handle the file. If the client can handle the file format directly, it renders the file for display. If the client has a configured helper application, such as with PDF documents, the client can call the helper application and let it render the file for display within the client window. The component that makes it possible for HTTP clients and servers to determine</p>
--

file format is their support for the Multipurpose Internet Mail Extensions (MIME) protocol. Using the MIME protocol, an HTTP server identifies each file with its corresponding MIME type. For example, an HTML document has the MIME type text/html, and a GIF image has the MIME type image/gif.

To make FTP available on a Web server, add the FTP Service role. If you need to configure custom providers, ASP.NET users or IIS Manager users as well, add the FTP Extensibility role.



With FTP, you can allow anonymous downloads and uploads in addition to restricted downloads and uploads. To allow anyone to access a file, configure directory security so that clients can use Anonymous authentication. With Anonymous authentication, the FTP server logs the user on automatically using a guest account and allows the anonymous user to download or upload files as appropriate. To require authorization to log on and access a directory, configure directory security to require authentication using one of the available authentication mechanisms, such as Basic authentication, which requires a user to type a user name and password prior to logging on and downloading or uploading files.

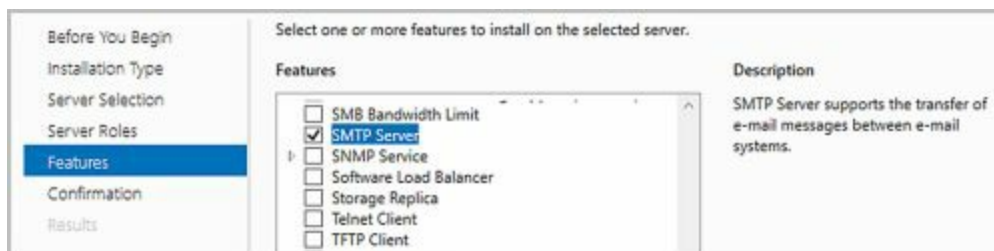
# SMTP

SMTP is an application-layer protocol that makes it possible for client applications to send e-mail messages to servers and for servers to send e-mail messages to other servers. A related protocol for retrieving messages from a server is Post Office Protocol version 3 (POP3). IIS 10 does not include SMTP or POP3 services.

With IIS 10, a web application can send e-mail on behalf of a user by using the SMTP E-mail component of Microsoft ASP.NET. An SMTP session initiated by a web application works like this:

1. The web application generates an e-mail message in response to something a user has done.
2. The System.Net.Mail API (a component of ASP.NET) delivers the email to an online SMTP server or stores the message on disk where it is stored for later delivery.
3. When sending mail to an SMTP server, the IIS server uses TCP port 25 to establish the connection. SMTP can be running on the local machine or on a different machine.

That is essentially how SMTP is used by web applications. Microsoft doesn't provide other e-mail features as a part of IIS. However, a separate SMTP Server component is included as an optional feature that you can install on a computer running a Windows Server operating system.



# Understanding IIS Roles

You can deploy IIS on both desktop and server platforms. On desktop platforms, you can use IIS for designing, building, and testing dynamic websites and web applications. On server platforms, IIS can have several different roles, including:

- **Application server** Application servers host distributed applications built using ASP.NET, Enterprise Services Network Support, and Microsoft .NET Framework. You can deploy application servers with or without Web Server (IIS) support. When you deploy an application server without Web Server (IIS) support, you configure application services through the application server core APIs and by adding or removing role services.
- **Web server** Web servers use the services bundled in IIS to host websites and web applications. Websites hosted on a Web server can have both static content and dynamic content. You can build web applications hosted on a Web server by using ASP.NET and .NET Framework. When you deploy a Web Server, you can manage the server configuration by using IIS modules and administration tools.

When configuring application servers and Web servers, it is important to understand exactly what comprises the .NET Framework. The Microsoft .NET Framework is a managed code programming model for Windows that includes:

- **Windows CardSpace (WCS)** A suite of .NET technologies for managing digital identities. Windows CardSpace supports any digital identity system and gives users consistent control of their digital identities. A digital identity can be as simple as an e-mail address and password used to log on to a website, or it can include a user's full contact and logon information. Client applications display each digital identity as an information card. Each card contains information about a particular digital identity, including what provider to contact to acquire a security token for the identity. By selecting a card and sending it to a provider such as Amazon or Yahoo!, users can validate their identity and log on to the service offered by the site.
- **Windows Communication Foundation (WCF)** A suite of .NET technologies for building and running connected systems. WCF supports a broad array of distributed systems capabilities to provide secure, reliable, and transacted messaging along with interoperability. Servers establish distributed communications through service endpoints. Service endpoints have an endpoint address, a binding that specifies how the endpoint can communicate, and a contract description that details what an endpoint communicates.
- **Windows Presentation Foundation (WPF)** A suite of .NET technologies for building applications with attractive and effective user interfaces. WPF supports



tight integration of application user interfaces, documents, and media content, allowing developers to create a unified interface for all types of documents and media. This means that applications can use the same interface for displaying forms, controls, fixed-format documents, on-screen documents, 2D images, 3D images, video, and audio.

- **Windows Workflow Foundation (WF)** A suite of .NET technologies for building workflow-enabled applications on Windows. WF provides a rules engine that allows for the declarative modeling of units of application logic within the scope of an overall business process. What this means is that developers can use WF to model and implement the necessary programming logic for a business process from start to finish.

To support applications written for IIS 6, you can deploy IIS 10 with IIS 6 compatibility enabled. If you have existing IIS 6 server installations, you can also install the IIS 6 Management Compatibility tools to support remote administration of these server installations.



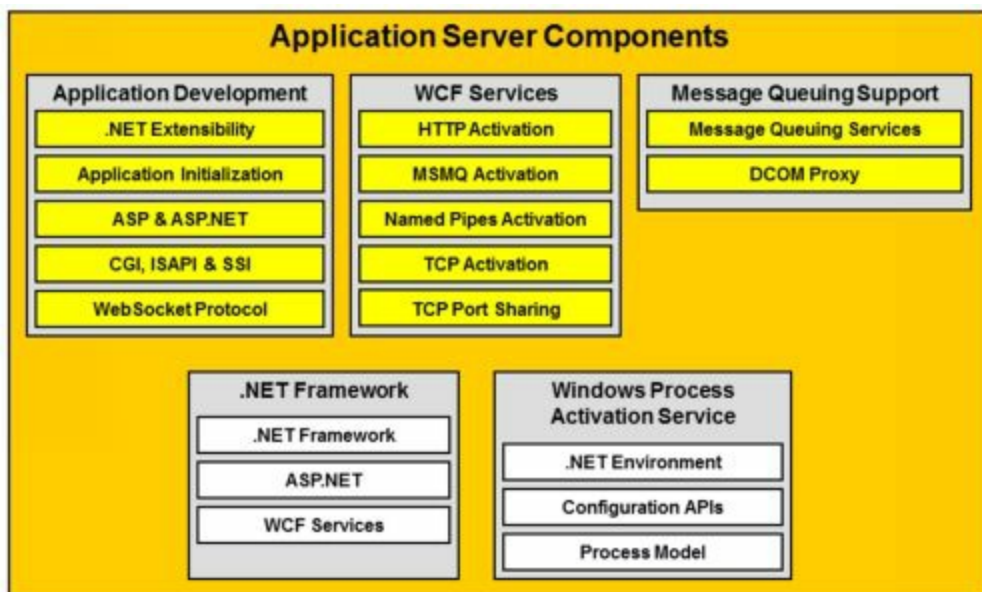
You also can deploy IIS 10 to support remote administration. You can use both desktop and server platforms for remote administration of other IIS servers in addition to the sites and applications configured on these servers. For remote administration of an IIS server, you must enable the Management Service (WMSVC) on the server you want to manage remotely. Then install the Web management tools on the machine you want to use for remote administration.

## Navigating the Installation Options

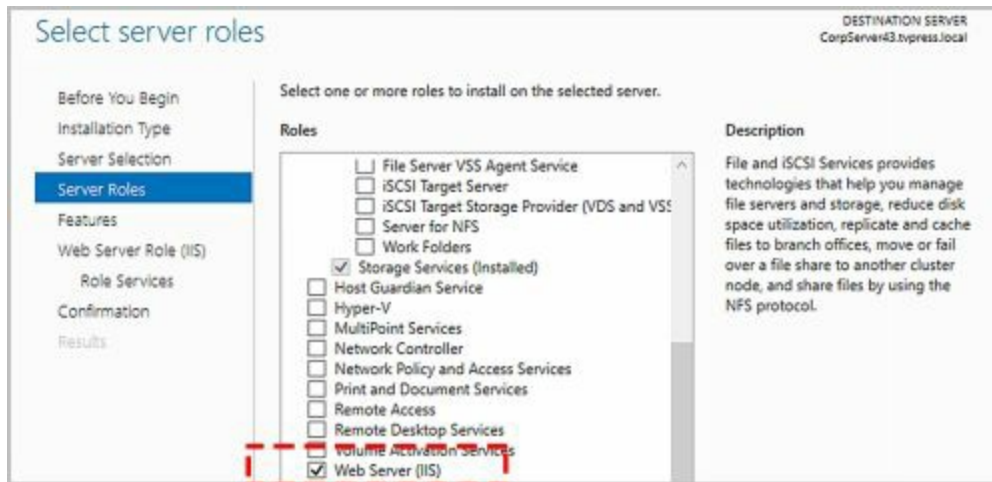
As discussed previously, you can deploy IIS on computers running Windows Server to support two specific roles: application server and Web server. You can deploy IIS running on a Windows desktop to support designing, building, and testing sites and applications. The components used to support these roles are referred to as either role services or features, depending on which user interface you are working with. In the sections that follow, I discuss each of the server roles and the related role services.

# Using Application Servers

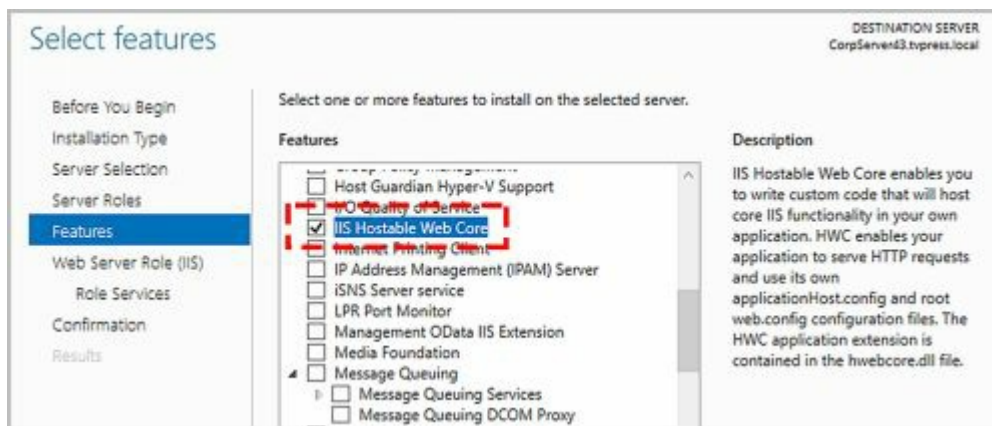
You use application servers running Windows Server to host distributed applications integrated with IIS. When you install an application server, you don't necessarily need a full installation of IIS. You only need the components necessary for application services, including either the standard core features or the hostable web core, the application development components, the WCF Services components, the .NET Framework components and the Windows Activation Service components. All other components are optional and should be installed based on the specific requirements of the distributed applications you are hosting.



If your application server is tightly integrated with IIS, you can install the standard core features by adding the Web Server (IIS) role to the application server. The Web Server (IIS) role allows the application server to host websites with both static and dynamic content. The websites support the standard IIS server extensions and allow you to create Web pages containing dynamic content. This allows an application server to host an internal or external website or provide an environment for developers to create web applications.



Not all application servers need a full installation of IIS, however. If you will host core IIS functionality in your own application, you may only need to install the IIS Hostable Web Core feature. The hostable web core allows your applications to service HTTP requests while maintaining control over the configuration.

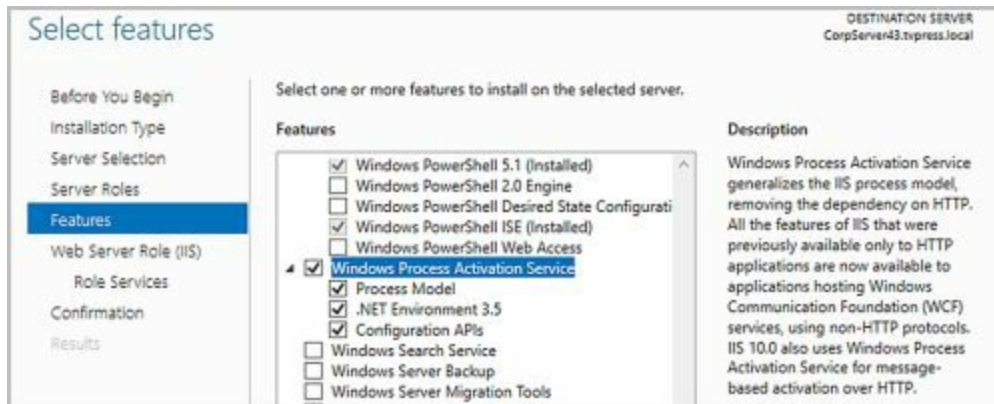


Most application servers use .NET to provide the necessary framework and APIs for applications. When you are working with Windows 10 and Windows Server 2016, two versions of the .NET framework are available version 3.5 and version 4.6. These technologies allow you to deliver managed-code applications that model business processes.

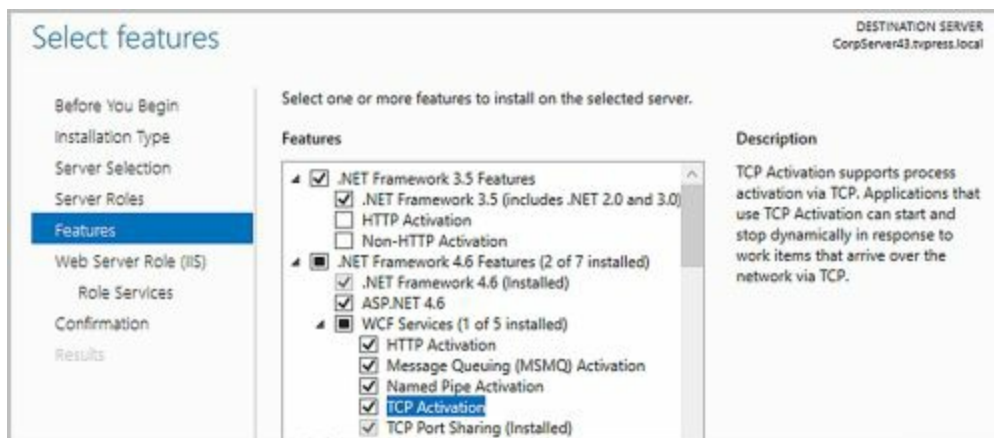
The Windows Process Activation Service supports distributed Web-based applications that use different protocols to transfer information. You can use the following related components:

- **.NET Environment** Installs the .NET Environment for use with managed code activation.
- **Configuration APIs** Installs the managed code APIs that allow you to configure the process model.
- **Process Model** Installs a process model for developing and running

applications.



Windows Process Activation Service enables the application server to invoke applications remotely over a network by using protocols such as HTTP, Microsoft Message Queuing (MSMQ), named pipes, and TCP. This allows applications to start and stop dynamically in response to incoming requests, resulting in improved performance and enhanced manageability. To specify which protocols an application server can use with Windows Process Activation, you install the related WCF services components.



The WCF Services components are used as follows:

- **HTTP Activation** Supports process activation over HTTP. This is the standard activation method used by most web applications. Applications that support HTTP Activation can start and stop dynamically in response to requests that arrive via HTTP. With HTTP, the application and the computers with which it communicates need to be online to pass active communications back and forth without the need for queuing requests.
- **Message Queuing Activation** Supports process activation over Microsoft Message Queue (MSMQ). This activation method is used when the application server runs distributed messaging applications. Applications that support MSMQ Activation and message queuing can start and stop dynamically in response to

requests that arrive via MSMQ. With message queuing, source applications send messages to queues, where they are stored temporarily until target applications retrieve them. This queuing technique allows applications to communicate across different types of networks and with computers that may be offline.

- **Named Pipe Activation** Supports process activation over named pipes. Applications that support Named Pipe Activation can start and stop dynamically in response to requests that arrive via named pipes. You use this activation method when web applications communicate with older versions of the Windows operating system. A named pipe is a portion of memory that one process can use to pass information to another process such that the output from one process is the input of the other process. Named pipes have standard network addresses such as `\\.\Pipe\Sql\Query`, which a process can reference on a local machine or a remote machine. The Named Pipes protocol is used primarily for local or remote connections by applications written for legacy versions of Windows.
- **TCP Activation** Supports process activation over TCP. Applications that support TCP Activation can start and stop dynamically in response to requests that arrive via TCP. With TCP, the application and the computers with which it communicates need to be online so they can pass active communications back and forth without the need for queuing requests.
- **TCP Port Sharing** Allows multiple applications to share a single TCP port. By using this feature, many web applications can coexist on the same server in separate, isolated processes while sharing the network infrastructure required for sending and receiving data over TCP ports.

When using Windows Process Activation Support, these additional features may be required:

- **Non-HTTP Activation** Provides non-HTTP activation support using any of the following: MSMQ, named pipes, and TCP. IIS installs this feature as a WCF Activation component. (This is a .NET Framework 3.5 component.)
- **Message Queuing Services** Provides the necessary server functions for message queuing. IIS installs this feature as a Message Queuing/Message Queuing Services component.

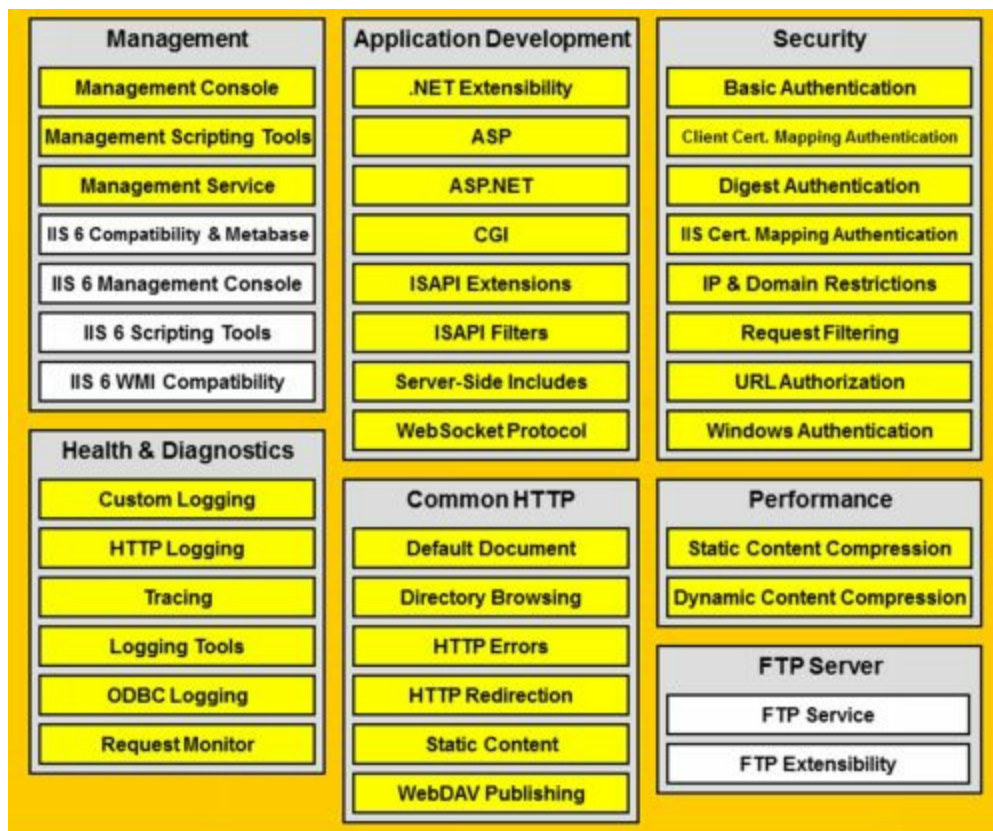
In your deployment planning, there is a distinct advantage to deploying an application server with the Web Server (IIS) role. When you deploy an application server with Web Server support, you can configure application services using the APIs provided by ASP.NET and the .NET Framework. Because the server includes IIS configuration and administration components, you'll have all of the common IIS features available and will be able to configure the server by using the IIS modules and the IIS administration tools.



# Using Web Servers

Web servers running on Windows desktop editions of Windows Server can host websites and web applications. When you install a Web server, several configuration features are installed automatically as part of the server core, and other features are installed by default (if applicable for the operating system version you are using).

These features represent core internal components in addition to the recommended minimum and required components for managing a Web server and publishing a website. In most installations of IIS, you will want to install additional features based on the specific requirements of the websites and web applications the server is hosting.



Many different features are available with Web servers. The IIS Server Core features provide the foundation functions for IIS. You can use these features as follows:

- **Anonymous Authentication** Supports anonymous access to a server. With anonymous access, any user can access content without having to provide credentials. Each server has to have at least one authentication mechanism configured, and this is the default mechanism.
- **Configuration Validation** Validates the configuration of a server and its applications. If someone improperly configures a server or application, IIS generates errors that can help detect and diagnose the problem.

- **HTTP Cache** Improves performance by returning a processed copy of a requested Web page from cache, resulting in reduced overhead on the server and faster response times. IIS supports several levels of caching including output caching in user mode and output caching in kernel mode. When you enable kernel-mode caching, cached responses are served from the kernel rather than from IIS user mode, giving IIS an extra boost in performance and increasing the number of requests IIS can process.
- **Protocol Support** Provides support for common protocols used by Web servers, including HTTP keep-alives, custom headers, and redirect headers. HTTP keep-alives allows clients to maintain open connections with servers, which speeds up the request process once a client has established a connection with a server. Custom headers and redirect headers allow you to optimize the way IIS works to support advanced features of the HTTP 1.1 specification.

The Common HTTP features install the common services required for serving Web content. You can use these features as follows:

- **Default Document** Supports displaying of default documents. When you've enabled this feature and a user enters a request with a trailing '/', such as `http://www.imaginedlands.com/`, IIS can redirect the request to the default document for the Web server or directory. For best performance, you should list the default document you use the most first and reduce the overall list of default documents to only those necessary.
- **Directory Browsing** Supports directory browsing functionality. When you've enabled default documents but there is no current default document, IIS can use this feature to generate a listing of the contents of the specified directory. If you haven't enabled the default document or directory browsing features, and a client requests a directory-level URL, IIS returns an empty response.
- **HTTP Errors** Supports custom error and detailed error notification. When you enable this feature and the server encounters an error, the server can return a customer error page to all clients regardless of location, a detailed error message to all clients regardless of location, or a detailed error for local clients and a custom error page for remote clients. IIS displays a custom error page based on the type of HTTP error that occurred.
- **HTTP Redirection** Supports redirection of HTTP requests to send users from an old site to a new site. In the default configuration for redirection, all requests for files in the old location are mapped automatically to files in the new location you specify. You can customize this behavior in several ways.
- **Static Content** Supports static Web content, such as HTML documents and GIF or JPEG images. The `staticContent/mimeMap` configuration collection in the



applicationHost.config file determines the list of file extensions supported.

- **WebDAV Publishing** Supports distributed authoring and publishing, enabling users to publish and retrieve files using HTTP.

**NOTE** Each of these common features has a related IIS native module that Setup installs and activates when you select the feature. For the exact mapping of common features to their corresponding native modules, see the appendix. You'll learn more about working with these features in Chapter 9, "Managing Global IIS Configuration."

The Application Development features install the features required for developing and hosting web applications. You can use these features as follows:

- **.NET Extensibility** Enables a Web server to host .NET Framework applications and provides the necessary functionality for IIS integration with ASP.NET and the .NET Framework. When you are working with managed modules, you must also enable the Managed Engine. The Managed Engine is the actual server component that performs the integration functions.
- **ASP** Enables a Web server to host classic Active Server Pages (ASP) applications. Web pages that use ASP are considered to be dynamic because IIS generates them at request time. To use ASP, you must also use ISAPI Extensions.
- **ASP.NET** Enables a Web server to host ASP.NET applications. Web pages that use ASP.NET are considered to be dynamic because they are generated at request time. To use ASP.NET, you must also use .NET Extensibility, ISAPI Extensions and ISAPI Filters.
- **CGI** Enables a Web server to host Common Gateway Interface (CGI) executables. CGI describes how executables specified in Web addresses, also known as gateway scripts, pass information to Web servers. By default, IIS handles all files with the .exe extension as CGI scripts.
- **ISAPI Extensions** Allows ISAPI Extensions to handle client requests. In the IIS server core, several components rely on handlers that are based on ISAPI Extensions, including ASP and ASP.NET. By default, IIS handles all files with the .dll extension as ISAPI Extensions.
- **ISAPI Filters** Allows ISAPI Filters to modify Web server behavior. IIS uses ISAPI Filters to provide additional functionality. When you select ASP.NET as part of the initial setup, Setup configures an ASP.NET filter to provide this functionality. In applicationHost.config, each version of ASP.NET installed on the Web server must have a filter definition that identifies the version and path to the related filter.
- **Server-Side Includes** Allows a Web server to parse files with Server-Side Includes (SSI). SSI is a technology that allows IIS to insert data into a document

when a client requests it. When this feature is enabled, files with the .stm, .shtm, and .shtml extension are parsed to see if they have includes that should be substituted for actual values. If this feature is disabled, IIS handles .stm, .shtm, and .shtml files as static content, resulting in the actual include command being returned in the request.

- **Web Socket Protocol** Allows a Web server to use applications that communicate over the WebSocket protocol. (IIS 8.5 and later only)

Health and Diagnostics features enable you to monitor your servers, sites, and applications and to diagnose problems if they occur. You can use these features as follows:

- **Custom Logging** Enables support for custom logging. Typically, custom logging uses the ILogPlugin interface of the Component Object Model (COM). Rather than using this feature, Microsoft recommends that you create a managed module and subscribe to the RQ\_LOG\_REQUEST notification.
- **HTTP Logging** Enables support for logging website activity. You can configure IIS to use one log file per server or one log file per site. Use per-server logging when you want all websites running on a server to write log data to a single log file. Use per-site logging when you want to track access separately for each site on a server.
- **Logging Tools** Allows you to manage server activity logs and automate common logging tasks using scripts.
- **ODBC Logging** Enables support for logging website activity to ODBC-compliant databases. In IIS, ODBC logging is implemented as a type of custom logging.
- **Request Monitor** Allows you to view details on currently executing requests, the run state of a website or the currently executing application domains, and more.
- **Tracing** Supports tracing of failed requests. Another type of tracing that you can enable after configuration is HTTP tracing, which allows you to trace events and warnings to their sources through the IIS server core.

Security features make it possible to control access to a server and its content. You can use these features as follows:

- **Basic Authentication** Requires a user to provide a valid user name and password to access content. All browsers support this authentication mechanism, but they transmit the password without encryption, making it possible for a malicious individual to intercept the password as the browser is transmitting it. If you want to require Basic Authentication for a site or directory, you should disable Anonymous Authentication for the site or directory.

- **Client Certificate Mapping Authentication** Maps client certificates to Active Directory accounts for the purposes of authentication. When you enable certificate mapping, this feature performs the necessary Active Directory certificate mapping for authentication of authorized clients.
- **Digest Authentication** Uses a Windows domain controller to authenticate user requests for content. Digest Authentication can be used through firewalls and proxies.
- **IIS Client Certificate Mapping Authentication** Maps SSL client certificates to a Windows account for authentication. With this method of authentication, user credentials and mapping rules are stored within the IIS configuration store.
- **IP and Domain Restrictions** Allows you to grant or deny access to a server by IP address, network ID, or domain. Granting access allows a computer to make requests for resources but doesn't necessarily allow users to work with resources. If you require authentication, users still need to authenticate themselves. Denying access to resources prevents a computer from accessing those resources, meaning that denied users can't access resources even if they could have authenticated themselves.
- **Request Filtering** Allows you to reject suspicious requests by scanning URLs sent to a server and filtering out unwanted requests. By default, IIS blocks requests for file extensions that could be misused and also blocks browsing of critical code segments.
- **URL Authorization** Supports authorization based on configuration rules. This allows you to require logon and to allow or deny access to specific URLs based on user names, .NET roles, and HTTP request method.
- **Windows Authentication** Supports Windows-based authentication using NTLM, Kerberos, or both. You'll use Windows Authentication primarily in internal networks.

What all of the previously listed security features have in common is a focus on client authentication and authorization. Support for centralized SSL server certificates also can be added as an option. Here, you publish server certificates to a file share to simplify management of your server certificates.

For enhancing performance, IIS supports both static compression and dynamic compression. With static compression, IIS performs an in-memory compression of static content upon first request and then saves the compressed results to disk for subsequent use. With dynamic content, IIS performs in-memory compression every time a client requests dynamic content. IIS must compress dynamic content every time it is requested because dynamic content changes.

When you are trying to improve server performance and interoperability, don't overlook

the value of these extended features:

- **File Cache** Caches file handles for files opened by the server engine and related server modules. If IIS does not cache file handles, IIS has to open the files for every request, which can result in performance loss.
- **Managed Engine** Enables IIS integration with the ASP.NET runtime engine. When you do not configure this feature, ASP.NET integration also is disabled, and no managed modules or ASP.NET handlers will be called when pooled applications run in Integrated mode.
- **Token Cache** Caches Windows security tokens for password based authentication schemes, including Anonymous Authentication, Basic Authentication, and Digest Authentication. Once IIS has cached a user's security token, IIS can use the cached security token for subsequent requests by that user. If you disable or remove this feature, a user must be logged on for every request, which can result in multiple logon user calls that could substantially reduce overall performance.
- **HTTP Trace** Supports request tracing for whenever a client requests one of the traced URLs. The way IIS handles tracing for a particular file is determined by the trace rules that you create.
- **URI Cache** Caches the Uniform Resource Identifier (URI)-specific server state, such as configuration details. When you enable this feature, the server will read configuration information only for the first request for a particular URI. For subsequent requests, the server will use the cached information if the configuration does not change.

You use Web management tools for administration and can divide the available tools into two general categories: those required for managing IIS 10 and those required for backward compatibility with IIS 6. You can use the related setup features as follows:

- **IIS Management Console** Installs the Internet Information Services (IIS) Manager, the primary management tool for working with IIS.
- **IIS Management Scripts and Tools** Installs the IIS command line administration tool and related features for managing Web servers from the command prompt.
- **IIS Management Service** Installs the Web Management Service (WMSVC), which provides a hostable Web core that acts as a standalone Web server for remote administration.
- **IIS Metabase Compatibility** Provides the necessary functionality for backward compatibility with servers running IIS 6 websites by installing a component that translates IIS 6 metabase changes to the IIS 10 and configuration stores.

- **IIS 6 WMI Compatibility** Provides the necessary functionality for scripting servers running IIS 6 websites by installing the IIS 6 Windows Management Instrumentation (WMI) scripting interfaces.
- **IIS 6 Scripting Tools** Provides the necessary functionality for scripting servers running IIS 6 websites by installing the IIS 6 Scripting Tools.
- **IIS 6 Management Console** Installs the Internet Information Services (IIS) 6.0 Manager, which is required to remotely manage servers running IIS 6 sites and to manage FTP servers for IIS 6.

## Chapter 3. Setting Up IIS

The way you set up IIS depends on the operating system you are using and whether you are using IIS for application services or Web services. You can also configure IIS as part of a desktop installation. I discuss deploying IIS in each of these situations in this chapter.

# Installing Application Servers

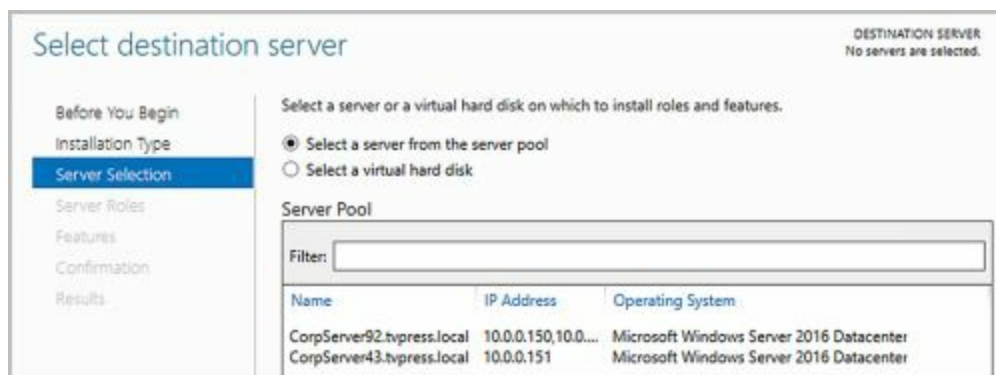
Unlike earlier releases of the operating system, Windows Server 2016 doesn't have a specific option for deploying an application server. That doesn't mean you can't deploy application servers, it just means that you need to work a little harder and know a little more about the options beforehand. If you've read through Chapters 1 and 2 (or at least skimmed them), then you are ready.

You can install an application server with or without Web server support by following these steps:

1. Start Server Manager on a server running Windows Server 2016. In Server Manager, click Manage, and then click Add Roles And Features. This starts the Add Roles And Features Wizard. If the wizard displays the Before You Begin page, read the Before You Begin page, and then click Next. You can avoid seeing the Before You Begin page the next time you start this wizard by selecting the Skip This Page By Default check box before clicking Next.
2. On the Installation Type page, Role-Based Or Feature-Based Installation is selected by default. Click Next.



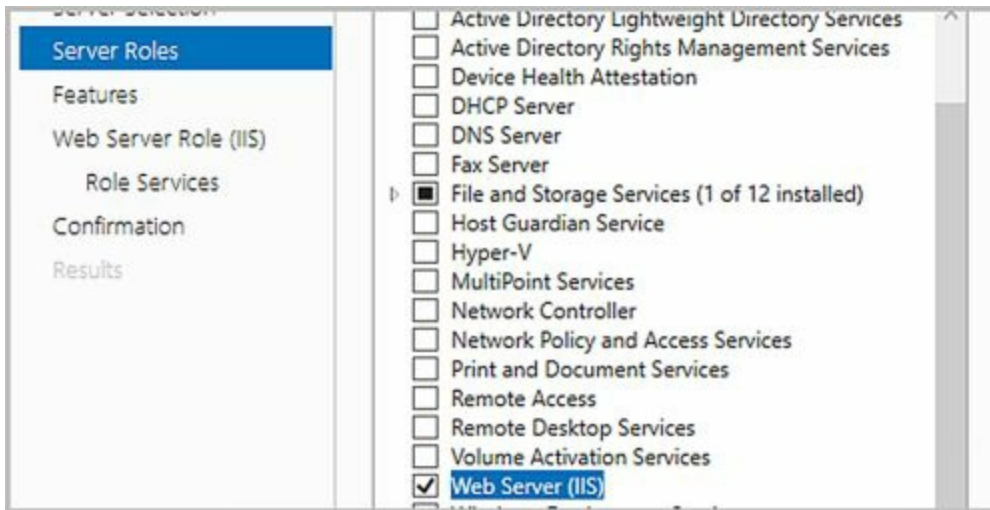
3. On the Server Selection page, you can choose to install roles and features on running servers or virtual hard disks. Either select a server from the server pool or select a server from the server pool on which to mount a virtual hard disk (VHD). If you are adding roles and features to a VHD, click Browse and then use the Browse For Virtual Hard Disks dialog box to locate the VHD. When you are ready to continue, click Next.



**NOTE** Only servers that have been added for management in Server Manager

are listed. If you need to add a server for management, click Cancel, add the server using the Manage / Add Servers option and then restart the wizard.

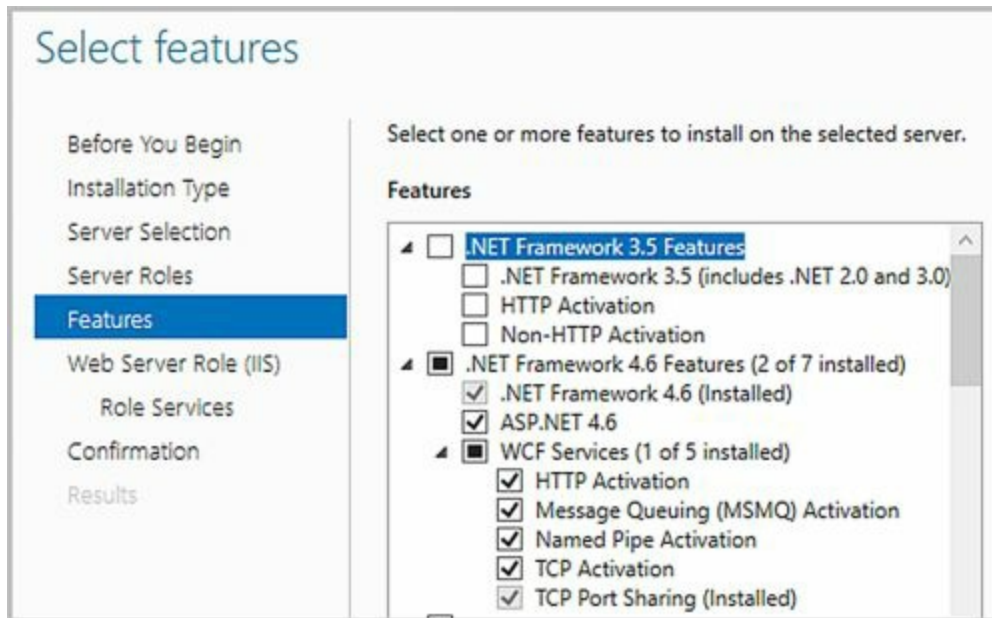
4. To deploy an application server with Web Server (IIS) support, select Web Server (IIS) on the Select Server Roles page.



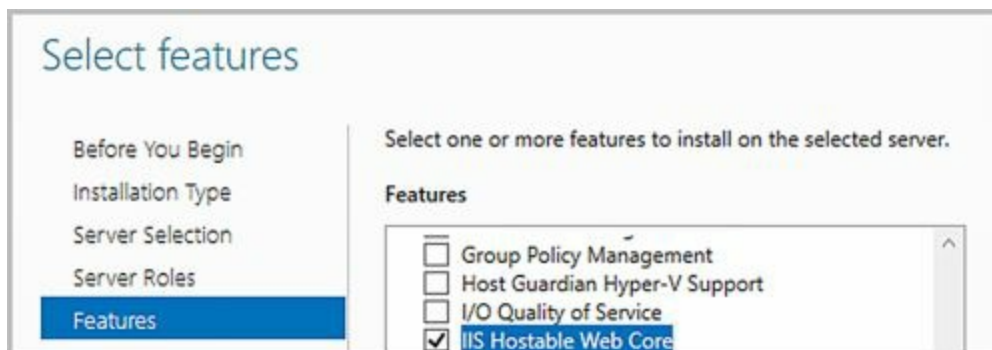
**NOTE** We recommend selecting Web Server (IIS) Support if the application server will host websites, web applications integrated with IIS or other Web services integrated with IIS. This will ensure that Windows selects the required Web Common features by default, and this will be helpful later in the setup process.

5. Click Next to continue and display the Features page. Choose the .NET Framework components to install. Most application servers will need .NET Framework 4.6, ASP.NET 4.6 and a subset of the WCF Services. If your application requires .NET Framework 3.5, select .NET Framework 3.5 Features and the appropriate subfeatures.

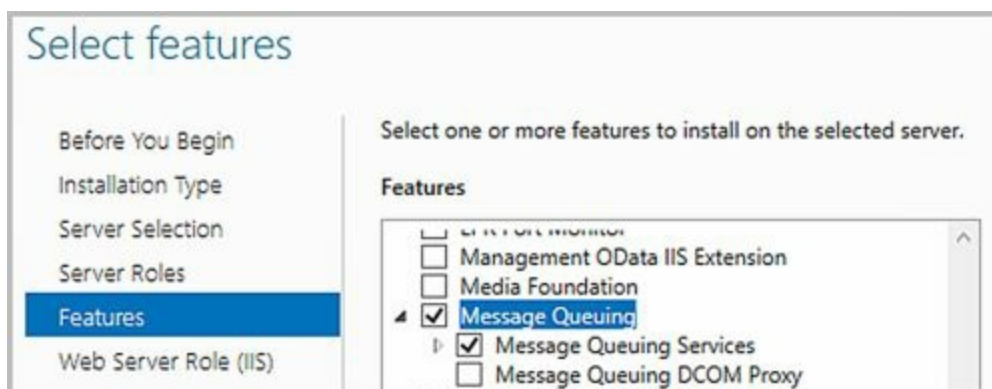




6. If you are installing an application server without IIS support, select the IIS Hostable Web Core option on the Features page. Keep in mind that if you are using HTTP Activation, the core IIS features will be installed as well, as they are required.

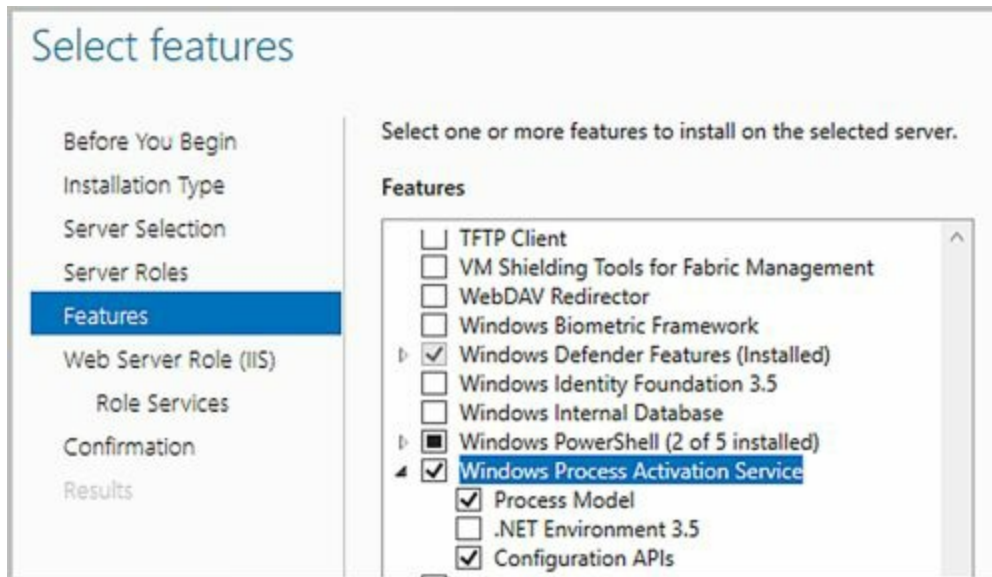


7. If the application server will use Message Queuing Activation, Message Queuing and Message Queuing Services are installed automatically as well. However, if the application will use distributed transactions and queuing, you may also need the Message Queuing DCOM Proxy.

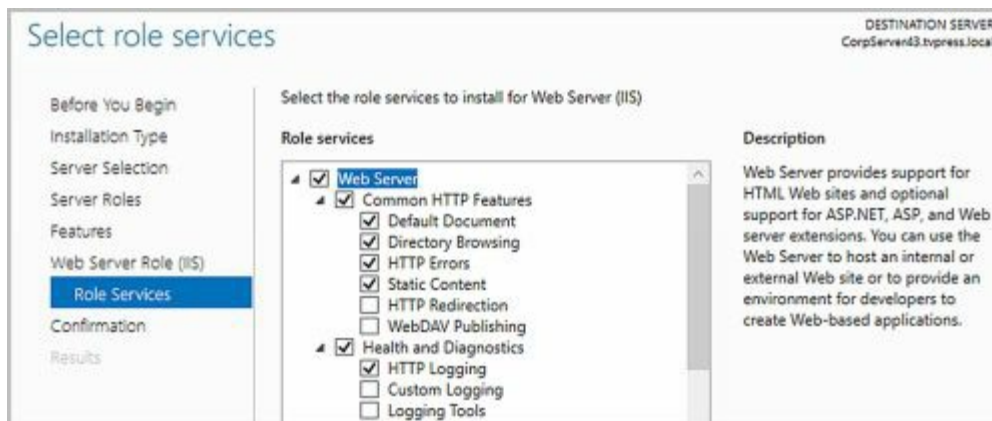


8. If the application server will use any of the WCF activation services, the standard

Windows Process Activation Service components are selected for installation automatically. The additional .NET 3.5 components are only required if your application will use .NET Framework 3.5 features.



9. If you are installing an application server with IIS support, you'll need to click Next twice to display the role services page for IIS. You can elect to accept the default common Web features that are already selected for you or you can configure the exact features you'd like to use.



10. In most cases, you'll want to select additional features rather than trying to remove features. When selecting or clearing role services, keep the following in mind before you click Next to continue:
- If you select a role service with additional required features, you'll see a dialog box listing the additional required roles. After you review the required roles, click Add Features to accept the additions and close the dialog box. If you click Cancel instead, the wizard clears the feature you previously selected.
  - If you try to remove a role service that is required based on a previous selection, you'll see a warning prompt about dependent services that the wizard must also remove. In most cases, you'll want to click Cancel to preserve the previous

selection. If you click Remove Features, the wizard will remove the previously selected dependent services, which could cause the Web server to not function as expected.

11. On the Confirmation page, click the Export Configuration Settings link to generate an installation report that can be displayed in Internet Explorer. If the server on which you want to install roles or features doesn't have all the required binary source files, the server gets the files via Windows Update by default or from a location specified in Group Policy.



**MORE INFO** You can specify an alternate path for the source files. To do this, click the Specify An Alternate Source Path link, type that alternate path in the box provided, and then click OK. For example, if you mounted a Windows image and made it available on the local server, you could enter the alternate path, such as `c:\mountdir\windows\winsxs`. For network shares, enter the UNC path to the share, such as `\\Server14\WinServer2016\`. For mounted Windows images, enter the WIM path prefixed with `WIM:` and including the index of the image to use, such as `WIM:\\Server14\WinServer2016\install.wim:4`.

12. After you review the installation options and save them as necessary, click Install to begin the installation process. The Installation Progress page tracks the progress of the installation. If you close the wizard, click the Notifications icon in Server Manager and then click the link provided to reopen the wizard.

When the wizard finishes installing the server with the roles and features you selected, the Installation Progress page will be updated to reflect this. Review the installation details to ensure that all phases of the installation were completed successfully.

Note any additional actions that might be required to complete the installation, such as restarting the server or performing additional installation tasks. If any portion of the installation failed, note the reason for the failure. Review the Server Manager entries for installation problems and take corrective actions as appropriate.

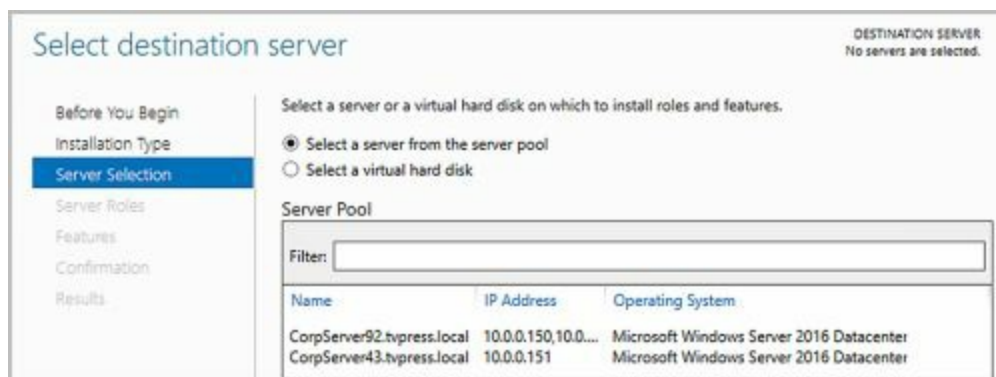
# Installing Web Servers

You can install a Web server by following these steps:

1. In Server Manager, click Manage, and then click Add Roles And Features. This starts the Add Roles And Features Wizard. If the wizard displays the Before You Begin page, read the Before You Begin page, and then click Next. You can avoid seeing the Before You Begin page the next time you start this wizard by selecting the Skip This Page By Default check box before clicking Next.
2. On the Installation Type page, Role-Based Or Feature-Based Installation is selected by default. Click Next.



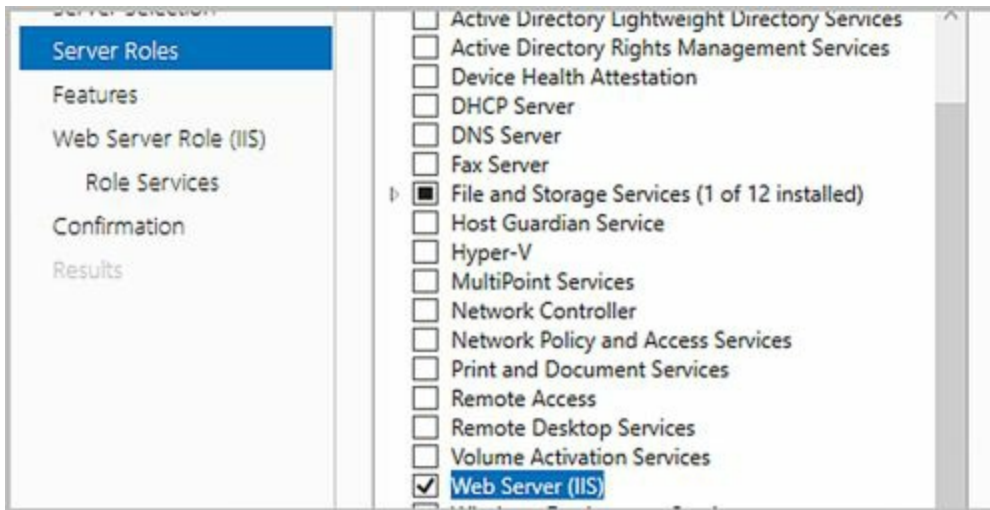
3. On the Server Selection page, you can choose to install roles and features on running servers or virtual hard disks. Either select a server from the server pool or select a server from the server pool on which to mount a virtual hard disk (VHD). If you are adding roles and features to a VHD, click Browse and then use the Browse For Virtual Hard Disks dialog box to locate the VHD. When you are ready to continue, click Next.



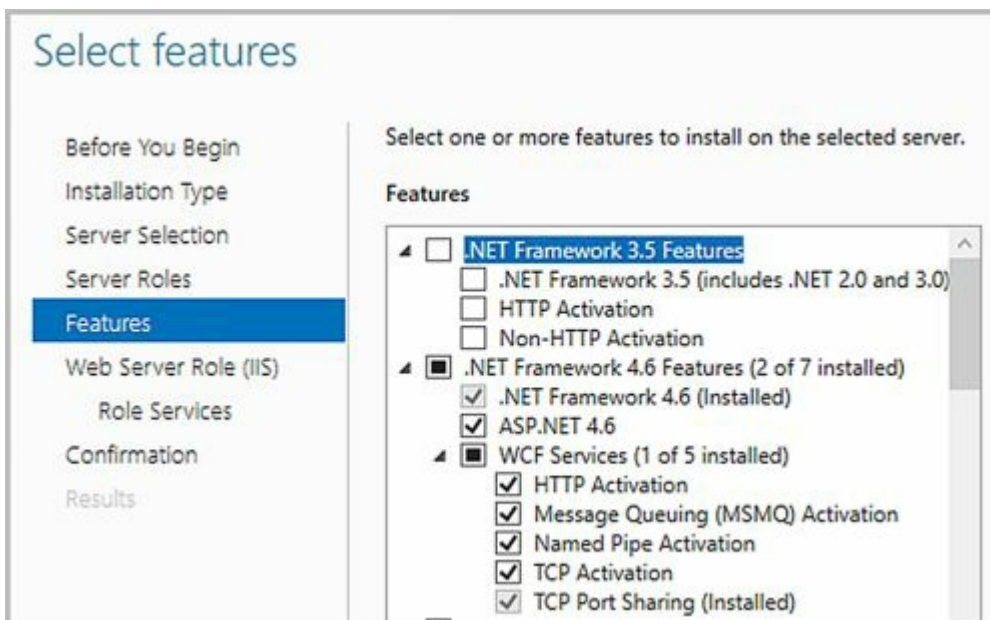
**NOTE** Only servers that have been added for management in Server Manager are listed.

4. On the Select Server Roles page, select the Web Server (IIS) role and then when prompted to install management tools, click Add Features to install the management tools or Cancel to skip installation of the management tools. If you don't install the management tools, you will need to manage IIS from a computer where the tools are installed. Click Next to continue.



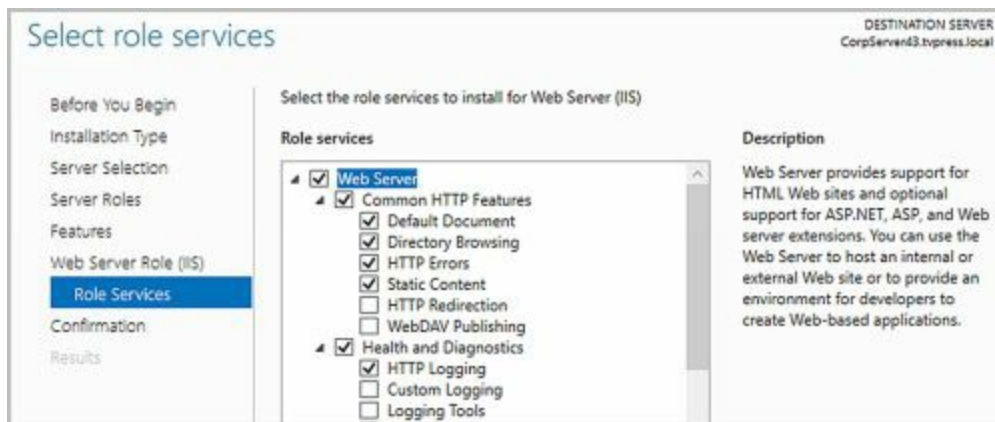


5. On the Features page, select any optional features that you want to install, such as activation features for .NET Framework 4.6 and WCF. If additional features are required to install a feature you select, you'll see an additional dialog box. Click Add Features to close the dialog box and add the required features. When you are ready to continue, click Next twice.

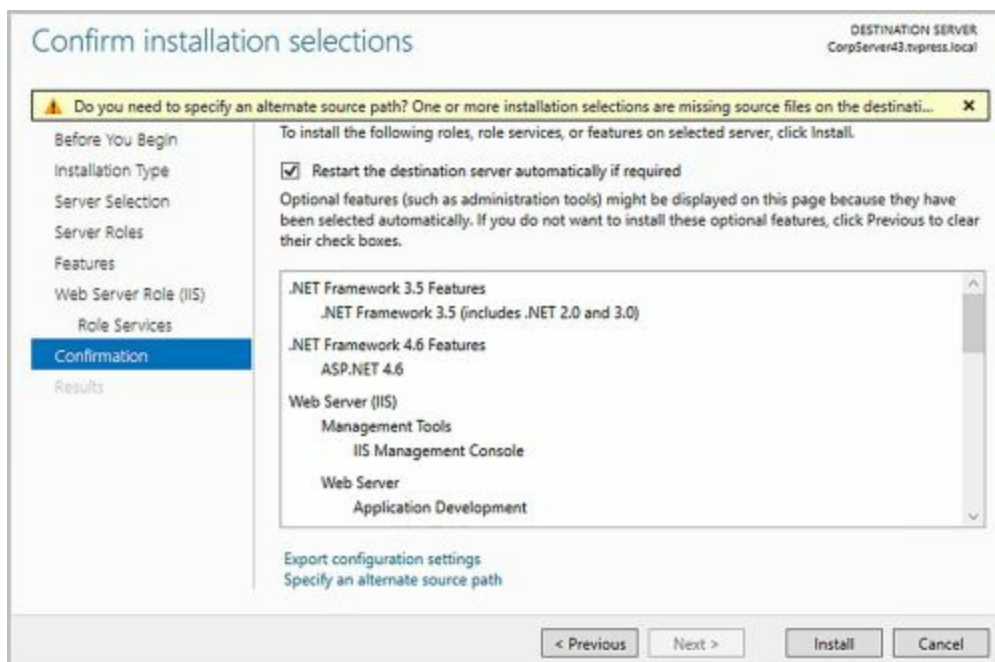


6. On the Select Role Services page, the wizard selects the core set of standard features by default. When selecting or clearing role services, keep the following in mind before you click Next to continue:
  - If you select a role service with additional required features, you'll see a dialog box listing the additional required roles. After you review the required roles, click Add Features to accept the additions and close the dialog box. If you click Cancel instead, the wizard will clear the feature you previously selected.
  - If you try to remove a role service that is required based on a previous selection, you'll see a warning prompt about dependent services that the wizard must also remove. In most cases, you'll want to click Cancel to preserve the previous

selection. If you click Remove Features, the wizard will also remove the previously selected dependent services, which could cause the Web server to not function as expected.



7. On the Confirmation page, review the installation options and save them as necessary using the Export option. When you are ready to continue, click Install to begin the installation process. The Installation Progress page tracks the progress of the installation. If you close the wizard, click the Notifications icon in Server Manager and then click the link provided to reopen the wizard.



**NOTE** If the server on which you want to install roles or features doesn't have all the required binary source files, the server gets the files via Windows Update by default or from a location specified in Group Policy. You also can specify an alternate path for the source files. To do this, click the Specify An Alternate Source Path link, type that alternate path in the box provided, and then click OK. For example, if you mounted a Windows image and made it available on the local server, you could enter the alternate path, such as c:\mountdir\windows\winsxs.

For network shares, enter the UNC path to the share, such as \\Server14\WinServer2016\. For mounted Windows images, enter the WIM path prefixed with WIM: and including the index of the image to use, such as WIM:\\Server14\WinServer2016\install.wim:4.

When the wizard finishes installing the server with the roles and features you selected, the Installation Progress page will be updated to reflect this. Review the installation details to ensure that all phases of the installation were completed successfully.

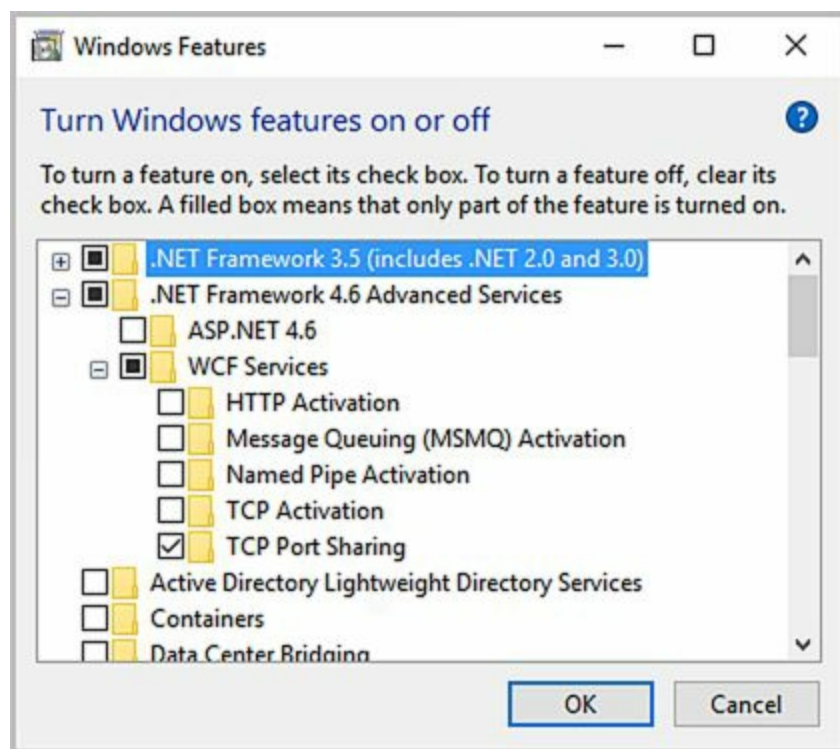
Note any additional actions that might be required to complete the installation, such as restarting the server or performing additional installation tasks. If any portion of the installation failed, note the reason for the failure. Review the Server Manager entries for installation problems and take corrective actions as appropriate.



# Installing IIS on Windows Desktops

In Windows 10, you configure operating system components as Windows features that you can turn on or off rather than add or remove. Install and configure Web server features by completing these steps:

1. Type Windows Features in the Search box. Then in the Search results, click Turn Windows Features On Or Off.



2. This displays the Windows Features dialog box. You'll find Windows features for Web servers under the following nodes:
  - **.NET Framework 3.5** Includes the HTTP Activation and Non-HTTP Activation components for WCF
  - **.NET Framework 4.6** Includes the ASP.NET 4.6 and components for WCF
  - **Internet Information Services/FTP Server** Includes the FTP Extensibility and the FTP Server components
  - **Internet Information Services/Web Management Tools** Includes the IIS 6 Management and IIS Management components
  - **Internet Information Services/World Wide Web Services** Includes the Application Development, Common HTTP, Health and Diagnostics, Performance, and Security components
  - **Internet Information Services Hostable Web Core** Includes the hostable web core components
  - **Microsoft Message Queue (MSMQ) Server** Includes the MSMQ Core

server components in addition to support and integration components for message queuing

- **Windows Process Activation Service** Includes the .NET Environment, Configuration APIs, and Process Model

To turn features on, select feature check boxes. To turn features off, clear feature check boxes. As you select features, Windows selects any required related features automatically without a warning prompt.

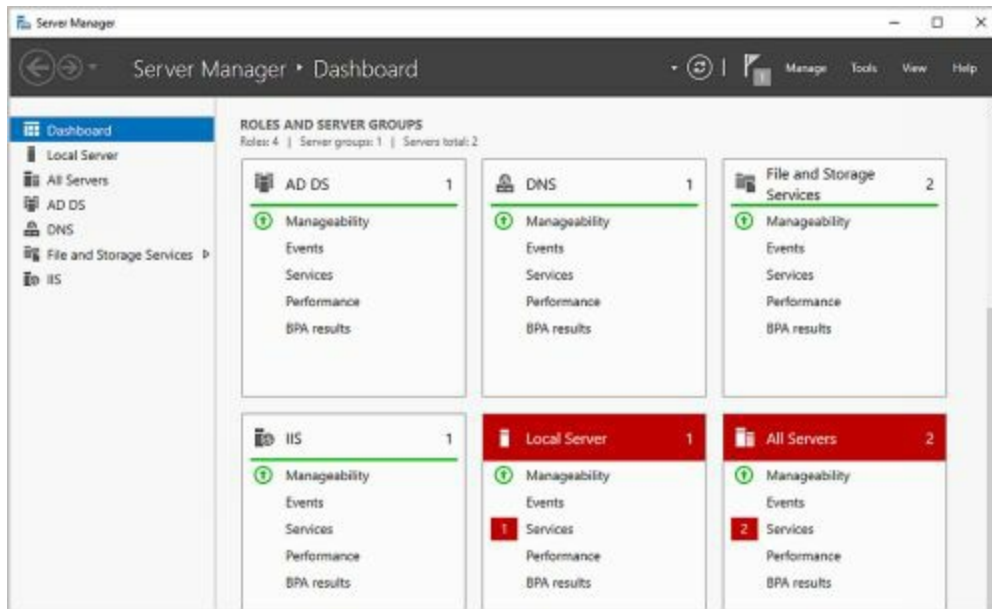
3. When you click OK, Windows reconfigures components as appropriate for any changes you've made. You may need your original installation media.

## Managing Installed Roles and Role Services

When you are working with Web and application servers, Server Manager is the primary tool you'll use to manage roles and role services. Not only can you use Server Manager to add or remove roles and role services, you can also use Server Manager to view the configuration details and status for roles and roles services.

# Viewing Configured Roles and Role Services

On Windows Server, Server Manager lists roles you've installed on the details pane of the Dashboard. The main view of the Roles and Server Groups node lists the names of the roles installed, the number of managed servers with those roles, and the general status of the roles.



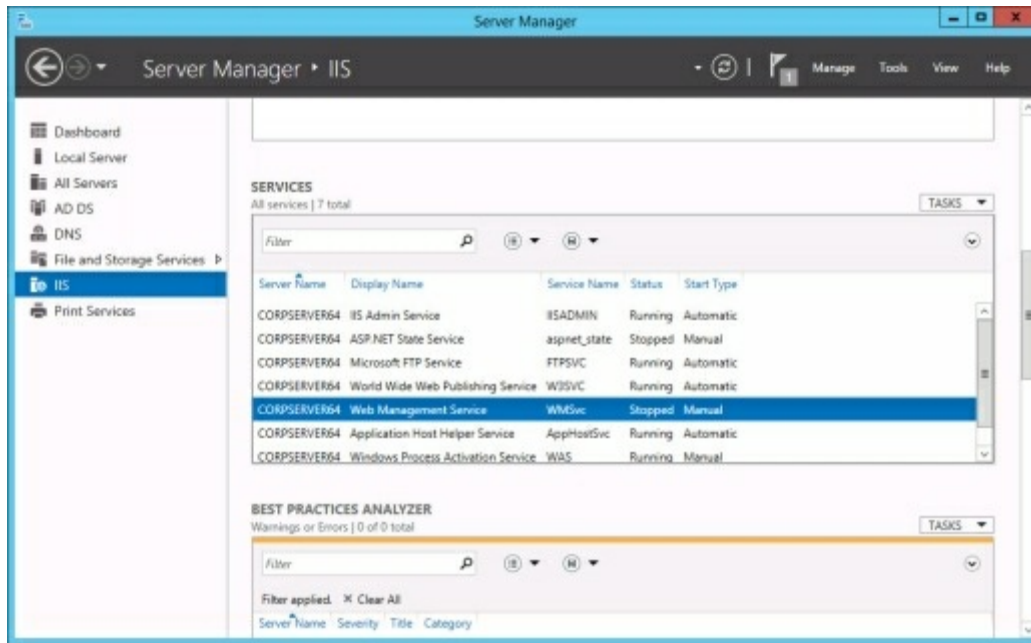
When servers with a role are operating normally, the status is green. When one or more servers with a role have an abnormal status or errors, the status is red.

**TIP** When working with Server Manager, don't overlook the back, forward and other navigation options on the ribbon. By default, Server Manager refreshes the details once every ten minutes. You can refresh the details manually by selecting the Refresh button.



If you want to set a different default refresh interval, click Manage and then click Server Manager Properties. Specify the desired data refresh period using the options provided, and then click OK.

In Server Manager's main window, if you click a role, Server Manager provides information about all managed servers that have that role installed. You can then select a server in the list to get expanded summary details on the events and services for the related role. If you click an event and then click View Event Properties, you can get detailed information about the event. Additionally, Server Manager provides details regarding the system services used by the role and their status.



You can manage a service by clicking it and then clicking the related Stop, Start, or Restart links provided. In many cases, if a service isn't running as you think it should, you can click Restart to resolve the issue by stopping and then starting the service.

# Adding or Removing Roles on Servers

When you select the IIS node in Server Manager and then select the server you want to manage on the Servers pane, the Roles And Features pane provides details on the current roles that are installed. On the Roles And Features pane, under Tasks, you'll find options for adding and removing roles. You can add a role as discussed previously in this chapter.

To remove a server role, complete the following steps:

1. In Server Manager, select Manage, and then click Remove Roles And Features. This starts the Remove Roles And Features Wizard. If the wizard displays the Before You Begin page, read the Welcome message and then click Next twice. You can avoid seeing the Before Your Begin page the next time you start this wizard by selecting the Skip This Page By Default check box before clicking Next.
2. On the Remove Server Roles page, clear the check box for the role you want to remove, and then click Next twice. If you try to remove a role that another role depends on, you'll see a warning prompt stating that you cannot remove the role unless you also remove the other role as well. If you click Remove Dependent Role Services, the wizard will remove both roles.
3. On the Confirm Removal Selections page, review the related role services that the wizard will remove based on your previous selections, and then click Remove.
4. When the wizard finishes modifying the server configuration, you'll see the Removal Results page. Review the modification details to ensure that all phases of the removal process completed successfully. If any portion of the removal process failed, note the reason for the failure and then use the previously discussed troubleshooting techniques to help resolve the problem.

# Viewing and Modifying Role Services

In Server Manager, you can view the role services configured for a role by completing the following steps:

1. Select the role in the left pane and then in the main pane select the server in the Servers pane.
2. Scroll down to the Roles And Features section where you'll find a list of role services that you are installed.
3. Manage role services by using the Add Role Services and Remove Role Services functions provided for the related role details entry.

You can add role services by completing the following steps:

1. In Server Manager, select Manage and then select Add Roles And Features. This starts the Add Role And Features Wizard. If the wizard displays the Before You Begin page, read the Before You Begin page, and then click Next. You can avoid seeing the Before You Begin page the next time you start this wizard by selecting the Skip This Page By Default check box before clicking Next.
2. On the Installation Type page, Role-Based Or Feature-Based Installation is selected by default. Click Next.
3. On the Server Selection page, you can choose to install roles and features on running servers or virtual hard disks. Either select a server from the server pool or select a server from the server pool on which to mount a virtual hard disk (VHD). If you are adding roles and features to a VHD, click Browse and then use the Browse For Virtual Hard Disks dialog box to locate the VHD. When you are ready to continue, click Next.

<p><b>NOTE</b> Only servers that have been added for management in Server Manager are listed. These servers must be running Windows Server 2012 or later.</p>
---

4. On the Select Server Roles page, the wizard makes the currently selected roles and role services unavailable so that you cannot select them. To add a role or role service, select it in the Roles list. When you are finished selecting roles to add, continue through the installation process by following the prompts and clicking Next until you see the Confirmation page. Click Install to install the selected roles and role services.
5. When the wizard finishes installing the server with the roles and role services you selected, the Installation Progress page will be updated to reflect this. Review the installation details to ensure that all phases of the installation were completed successfully.

6. Note any additional actions that might be required to complete the installation, such as restarting the server or performing additional installation tasks.
7. If any portion of the installation failed, note the reason for the failure. Review the Server Manager entries for installation problems and take corrective actions as appropriate.

You can remove role services by completing the following steps:

1. In Server Manager, select Manage and then select Remove Roles And Features. This starts the Remove Roles And Features Wizard. Click Next twice.
2. On the Server Selection page, you can choose to install roles and features on running servers or virtual hard disks. Either select a server from the server pool or select a server from the server pool on which to mount a virtual hard disk (VHD). If you are adding roles and features to a VHD, click Browse and then use the Browse For Virtual Hard Disks dialog box to locate the VHD. When you are ready to continue, click Next.
3. On the Remove Server Roles page, the wizard selects the currently installed roles and role services. To remove a role or role service, clear the related check box. If any other installed components require the role or service you want to remove, you'll see a prompt stating you must also remove dependent features. Click Remove Features to also remove the dependent features or click Cancel to specify that you don't want to remove the previously select role or service.
4. When you are finished selecting roles and role services to remove, click Next, and then click Remove.



## Part 2. Core IIS 10 Administration

Chapter 4. Navigating IIS 10 Architecture

Chapter 5. Managing IIS Servers & Services

Chapter 6. Managing IIS 10 from the Prompt

Chapter 7. Using Management Objects for Administration

Chapter 8. Digging into IIS Schema

Chapter 9. Managing Global IIS Configuration



## Chapter 4. Navigating IIS 10 Architecture

IIS provides the necessary foundation for many different types of services that require remote access and web hosting functionality. Because of this, whether you are planning to host web sites, web apps or simply set up remote access gateways, you should have a deep understanding of web protocols, the core IIS architecture and basic IIS operations.

## Working with IIS and URLs

To retrieve files from IIS servers, clients must know three things: the server's address, where on the server the file is located, and which protocol to use to access and retrieve the file. Normally, this information is specified as a Uniform Resource Locator (URL). URLs provide a uniform way of identifying resources that are available. The basic mechanism that makes URLs so versatile is their standard naming scheme.

URL schemes name the protocol the client will use to access and transfer the file. Clients use the name of the protocol to determine the format for the information that follows the protocol name. The protocol name is generally followed by a colon and two forward slashes. The information after the double slash marks follows a format that depends on the protocol type referenced in the URL. Here are two general formats:

`protocol://hostname:port/path_to_resource`

`protocol://username:password@hostname:port/path_to_resource`

Host name information used in URLs identifies the address to a host and is broken down into two or more parts separated by periods. The periods are used to separate domain information from the host name. Common domain names for Web servers begin with `www`, such as `www.imaginedlands.com`, which identifies the Imagined Lands WWW server in the commercial domain. Domains you can specify in your URLs include:

- **com** Commercial sites
- **edu** Education sites
- **gov** Nonmilitary government sites
- **mil** Military sites
- **net** Network sites
- **org** Organizational sites

Port information used in URLs identifies the port number to be used for the connection. Generally, you don't have to specify port numbers in your URLs unless the connection will be made to a port other than the default. Port 80 is the default port for HTTP. If you request a URL on a server using the URL `http://www.imaginedlands.com/books.htm`, port 80 is assumed to be the default port value. On the other hand, if you wanted to make a connection to port 8080, you'd need to type in the port value, such as `http://www.imaginedlands.com:8080/books.htm`.

Port values that fall between zero and 1023, referred to as well-known ports, are reserved for specific data type uses on the Internet. Port values between 1024 and 49151 are considered registered ports, and those between 49152 and 65535 are

considered dynamic ports.

The final part of a URL is the path to the resource. This path generally follows the directory structure from the server’s home directory to the resource specified in the URL.

URLs for FTP can also contain a user name and password. User name and password information allows users to log in to an FTP server using a specific user account. For example, the following URL establishes a connection to the Microsoft FTP server and logs on using a named account, such as  
`ftp://sysadmin:rad$4@ftp.williamrstanek.com/public/download.doc`.

In this instance, the account logon is sysadmin, the password is rad\$4, the server is ftp.williamrstanek.com, and the requested resource is public/download.doc.

If a connection is made to an FTP server without specifying the user name and password, you can configure the server to assume that the user wants to establish an anonymous session. In this case the following default values are assumed: anonymous for user name and the user’s e-mail address as the password.

URLs can use uppercase and lowercase letters, the numerals 0–9, and a few special characters, including:

- Asterisks (\*)
- Dollar signs (\$)
- Exclamation points (!)
- Hyphens (-)
- Parentheses (left and right)
- Periods (.)
- Plus signs (+)
- Single quotation marks (‘)
- Underscores ( \_ )

You’re limited to these characters because other characters used in URLs have specific meanings, as shown in Table 4-1.

**TABLE 4-1** Reserved Characters in URLs

:	The colon is a separator that separates the protocol from the rest of the URL scheme; separates the host name from the port number; and separates the user name from the password.
//	The double slash marks indicate that the protocol uses the format defined by the Common Internet Scheme Syntax (see RFC 1738 for more information).
/	

	The slash is a separator and is used to separate the path from the host name and port. The slash is also used to denote the directory path to the resource named in the URL.
~	The tilde is generally used at the beginning of the path to indicate that the resource is in the specified user's public Hypertext Markup Language (HTML) directory.
%	Identifies an escape code. Escape codes are used to specify special characters in URLs that otherwise have a special meaning or aren't allowed.
@	The at symbol is used to separate user name and/or password information from the host name in the URL.
?	The question mark is used in the URL path to specify the beginning of a query string. Query strings are passed to Common Gateway Interface (CGI) scripts. All the information following the question mark is data the user submitted and isn't interpreted as part of the file path.
+	The plus sign is used in query strings as a placeholder between words. Instead of using spaces to separate words that the user has entered in the query, the browser substitutes the plus sign.
=	The equal sign is used in query strings to separate the key assigned by the publisher from the value entered by the user.
&	The ampersand is used in query strings to separate multiple sets of keys and values.
^	The caret is reserved for future use.
{ }	Braces are reserved for future use.
[ ]	Brackets are reserved for future use.

To make URLs even more versatile, you can use escape codes to specify characters in URLs that are either reserved or otherwise not allowed. Escape codes have two components: a percent sign and a numeric value. The percent sign identifies the start of an escape code. The number following the percent sign identifies the character being escaped. The escape code for a space is a percent sign followed by the number 20 (%20). To refer to a file called "my party hat.htm," for example, you could use this escape code in a URL such as this one:

<http://www.williamrstanek.com/docs/my%20party%20hat.htm>

## Understanding the Core Architecture

You can think of IIS as a layer over the operating system where, in most cases, you might need to perform an operating system–level task before you perform an IIS task. Websites, web applications, and virtual directories are the core building blocks of IIS servers. Every IIS server installation has these core building blocks. As you set out to work with IIS servers and these basic building blocks, you'll also want to consider what access and administrative controls are available.

## Working with websites

You can use a single IIS server to host multiple websites. Websites are containers that have their own configuration information, which includes one or more unique bindings. A website binding is a combination of an Internet Protocol (IP) address, port number, and optional host headers on which HTTP.sys listens for requests. Many websites have two bindings: one for standard requests and one for secure requests. For example, you could configure a website to listen for standard HTTP requests on IP address 192.168.10.52 and TCP port 80. If you've also configured the server for Secure Sockets Layer (SSL), you also could configure a website to listen for Secure HTTP (HTTPS) requests on IP address 192.168.10.52 and TCP port 443.

When you install IIS on a server, the installation process creates a default website and configures the bindings for this site so that HTTP.sys listens for requests on TCP port 80 for all IP addresses you've configured on the server. Thus if the server has multiple IP addresses, HTTP.sys would accept requests from any of these IP addresses, provided that the requests are made on TCP port 80. Increasingly, modern websites use host headers. Host headers allow you to assign multiple host names to the same IP address and TCP port combination. Here, IIS uses the host name passed in the HTTP header to determine the site that a client is requesting. For example, a single server could use host headers to host [catalog.imaginedlands.com](http://catalog.imaginedlands.com), [sales.imaginedlands.com](http://sales.imaginedlands.com), and [www.imaginedlands.com](http://www.imaginedlands.com) on IP address 192.168.15.68 and TCP port 80.



# Working with Web Applications and Virtual Directories

IIS handles every incoming request to a website within the context of a web application. A web application is a software program that delivers Web content to users over HTTP or HTTPS. Each website has a default web application and one or more additional web applications associated with it. The default web application handles incoming requests that you haven't assigned to other web applications. Additional web applications handle incoming requests that specifically reference the application.

Each web application must have a root virtual directory associated with it. The root virtual directory sets the application name and maps the application to the physical directory that contains the application's content. Typically, the default web application is associated with the root virtual directory of the website and any additional virtual directories you've created but haven't mapped to other applications. Following this, in the default configuration, the default applications handles an incoming request for the / directory of a website in addition to other named virtual directories, such as /images or /data. IIS maps references to /, /images, /data, or other virtual directories to the physical directory that contains the related content. For the / directory of the default website, the default physical directory is %SystemRoot%/Inetpub/Wwwroot.

When you create a web application, the application's name sets the name of the root virtual directory. Therefore, if you create a web application called Sales, the related root virtual directory is called Sales, and this virtual directory in turn maps to the physical directory that contains the application's content, such as %SystemRoot%/Inetpub/Wwwroot/Sales.

# Controlling Access to Servers, Sites, and Apps

By default, IIS is configured to allow anyone to anonymously access the websites and applications configured on an IIS server. You can control access to websites and web applications by requiring users to authenticate themselves. IIS supports a number of authentication methods for websites, including Basic authentication, Digest authentication, Client Certificate authentication, and Windows authentication. When working with Microsoft ASP.NET and web applications, you also can use ASP.NET impersonation and Forms authentication.

Regardless of the authentication techniques you use, however, Windows Server permissions ultimately determine if users can access files and directories. Before users can access files and directories, you must ensure that the appropriate users and groups have access at the operating system level. After you set operating system-level permissions, you must set IIS-specific security permissions. For more information on IIS security, see IIS 10 Web Applications, Security & Maintenance.

As an administrator, you can manage the configuration of IIS from the command prompt or within IIS Manager. For administration of Web servers, websites, and web applications using the command line, Windows Management Instrumentation (WMI), or direct editing of the configuration files, you must have write permissions on the target configuration files. For administration of Web servers, websites, and web applications using IIS Manager, IIS specifies three administrative roles:

- **Web server administrator** A Web server administrator is a top-level administrator who has complete control over an IIS server and can delegate administration of features to website administrators and web application administrators. A Web server administrator is a member of the Administrators group on the local server or a domain administrator group in the domain of which the server is a member.
- **website administrator** A website administrator is an administrator who has been delegated control of a specific website and any applications related to that website. A website administrator can delegate control of a web application to a web application administrator.
- **web application administrator** A web application administrator is an administrator who has been delegated control of a specific web application. A website administrator can delegate control of a web application to a web application administrator.

The settings that administrators can configure depend on their administrative role on a particular server. Web server administrators have server-level permissions and as such:

- Have full access to all websites on a server.
- Have full access to all applications on websites.
- Have full access to all virtual directories used by sites and applications.
- Have full access to all physical directories used by sites and applications.
- Have full access to files in virtual and physical directories.
- Can designate website and web application administrators.

website administrators have site-level permissions and as such:

- Don't have full access to all websites on a server.
- Have full access to applications used by delegated sites.
- Have full access to virtual directories used by delegated sites.
- Have full access to physical directories used by delegated sites.
- Have full access to files in virtual and physical directories for sites delegated.
- Can designate web application administrators within delegated sites.

web application administrators have application-level permissions and as such:

- Don't have full access to all websites on a server.
- Don't have full access to all applications on a server.
- Have full access to virtual directories used by delegated applications.
- Have full access to physical directories used by delegated applications.
- Have full access to files in virtual and physical directories for sites delegated.
- Can designate web application administrators for delegated applications.

# Understanding Services and Processing

A strong understanding of the services and processing architecture used by IIS are essential to your success as an administrator. Although this section provides an initial discussion of applications and application pools, IIS applications and application pools are discussed in detail in IIS Web Applications, Security & Maintenance: The Personal Trainer.

# Essential IIS Services and Processes

Windows services and processes are other areas where Windows and IIS are tightly integrated. Table 4-2 provides a summary of key services that IIS uses or depends on. Note that the services available on a particular IIS server depend on its configuration. Still, this is the core set of services that you'll find on most IIS servers.

**TABLE 4-2** Essential IIS Services

Application Host Helper Service (Apphostsvc)	Provides configuration history services and app pool account mapping for file and directory access locking. Logs on as Local System and is configured to restart automatically as part of service recovery.
ASP.NET State Service (aspnet_state)	Provides support for out-of-process session state management for ASP.NET. Out-of-process state management ensures that the session state is preserved when an application's worker process is recycled. IIS uses this service only when you set the Session State mode to State Server or SQL Server. Otherwise, IIS does not use this service. Logs on as Network Service and is configured to restart automatically as part of service recovery.
FTP Publishing Service (MSFTPSVC)	Provides services for transferring files by using FTP. If the server isn't configured as an FTP server, the server doesn't need to run this service. In a standard installation of IIS, when you install FTP, this service is, by default, configured for manual startup only. Logs on as Local System but is not configured to restart automatically as part of service recovery. This service is dependent on the IIS Admin Service.
IIS Admin Service (IISADMIN)	Allows administration of IIS 6.0–related features, including the metabase. If the server doesn't need backward compatibility, the server doesn't need to run this service. Logs on as Local System and is configured to run iisreset.exe as part of service recovery. This service is dependent on the HTTP, RPC, and Security Accounts Manager services.
Web Management Service (WMSVC)	Enables remote and delegated management of IIS using IIS Manager. If a server is locked down so it can be accessed locally only, the server doesn't need to run this service. Logs on as Local Service and is configured to restart automatically as part of service recovery. This service is dependent on the HTTP service.
Windows Process Activation Service (WAS)	Provides essential features for messaging applications and the Microsoft .NET Framework, including process activation, resource management, and health management. This service is essential for IIS. Logs on as Local System and is configured to run iisreset.exe as part of service recovery. This service is dependent on the RPC service.
World Wide Web Publishing Service (W3SVC)	Provides essential services for transferring files by using HTTP and administration through the IIS Manager. This service is essential for IIS. Logs on as Local System but is not configured to restart automatically as part of service recovery. This service is dependent on the HTTP service and the Windows Process Activation Service.

As the table shows, the World Wide Web Publishing Service and Windows Process Activation Service provide the essential services for IIS. Management of the Web service and web applications is internalized. The Web Administration Service component of the Web Service Host is used to manage the service itself. Worker

processes are used to control applications, and no Internet Server Application Programming Interface (ISAPI) applications run within the IIS process context.

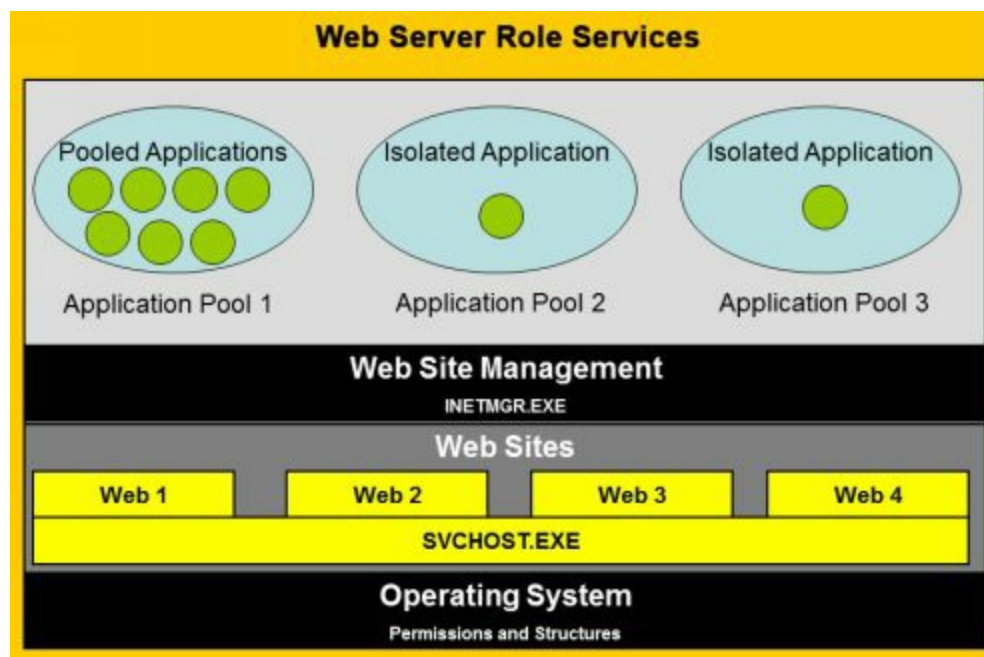
# IIS Worker Process Isolation Mode

Worker Process isolation mode is the standard processing mode for websites and web applications. This mode allows sites and applications to:

- Recycle worker threads
- Monitor process health
- Use advanced application pooling configurations
- Take advantage of other IIS features

The World Wide Web Publishing Service and Windows Process Activation Service provide the essential services for IIS. From a high level, the standard IIS operating mode works as depicted in the figure.

Service Host processes control all Web resources running on a server. Starting, pausing, or stopping the World Wide Web Publishing Service affects all websites on the server. It doesn't directly affect the Service Host. Instead, Windows Server uses an intermediary to control the Service Host for you. For non-Web services, this intermediary is the Inetmgr.exe process. A single instance of Inetmgr.exe is used to manage websites and web applications.



Management of the Web service and web applications is internalized. The Web Administration Service component of the Web Service Host is used to manage the service itself. Worker processes are used to control applications, and no ISAPI applications run within the IIS process context.

Worker processes are used in several ways:

- **Single worker process—single application** Here a single worker process running in its own context (isolated) handles requests for a single application as well as instances of any ISAPI extensions and filters the application's need. The application is the only one assigned to the related application pool.
- **Single worker process—multiple applications** Here, a single worker process running in its own context (isolated) handles requests for multiple applications assigned to the same application pool as well as instances of any ISAPI extensions and filters the applications' needs.
- **Multiple worker processes—single application** Here, multiple worker processes running in their own context (isolated) share responsibility for handling requests for a single application as well as instances of any ISAPI extensions and filters the application's needs. The application is the only one in the related application pool.
- **Multiple worker processes—multiple applications** Here, multiple worker processes running in their own context (isolated) share responsibility for handling requests for multiple applications assigned to the same application pool as well as instances of any ISAPI extensions and filters the applications' needs.

The standard operating mode ensures that all sites run within an application context and have an associated application pool. The default application pool is DefaultAppPool. You can also assign sites and applications to custom application pools.

Each application or site in an application pool can have one or more worker processes associated with it. The worker processes handle requests for the site or application.

You can configure application pools to manage worker processes in many ways. You can configure automatic recycling of worker threads based on a set of criteria such as when the process has been running for a certain amount of time or uses a specific amount of memory. You can also have IIS monitor the health of worker threads and take actions to recover automatically from failure. These features might eliminate or reduce your dependence on third-party monitoring tools or services.

You can also create a Web garden in which you configure multiple worker processes to handle the workload. Applications configured using this technique are more responsive, more scalable, and less prone to failure. Why? A Hypertext Transfer Protocol (HTTP) listener, called Http.sys, listens for incoming requests and places them in the appropriate application pool request queue. When a request is placed in the queue, an available worker process assigned to the application can take the request and begin processing it. Idle worker processes handle requests in first-in, first-out (FIFO) order.



Worker processes can also be started on demand. If there are unallocated worker processes and no current idle worker processes, IIS can start a new worker process to handle the request. In this way, resources aren't allocated until they're needed, and IIS can handle many more sites than it could if all processes were allocated on startup.

# Understanding and Using IIS Applications

You can configure websites to run several different types of applications, including:

- [Common Gateway Interface \(CGI\) programs.](#)
- [Internet Server Application Programming Interface \(ISAPI\) applications.](#)
- [ASP.NET applications using managed code.](#)

CGI describes how programs specified in Web addresses, also known as gateway scripts, pass information to Web servers. Gateway scripts pass information to servers through environment variables that capture user input in forms in addition to information about users submitting information. In IIS 10, standard CGI is implemented through the CgiModule and multi-threaded CGI is implemented through the FastCgiModule. The CgiModule has a managed handler that specifies that all files with the .exe extension are to be handled as CGI programs.

The way CGI programs are handled is determined by the way you've configured the CGI feature within IIS. By default, CGI is disabled. When you enable CGI, the CgiModule is the default handler for .exe programs. You can modify the handler configuration for .exe programs to use the FastCgiModule. This configuration is useful if you've installed the PHP Hypertext Preprocessor (PHP) on your IIS server and want to use it. Once you've configured the server to use FastCgi for .exe programs, you should add handler mappings for PHP-related file extensions and configure these mappings so that they use the PHP executable, such as Php-cgi.exe. For example, you could add mappings for \*.php and \*.php5. Your IIS server would then process files with the .PHP and .PHP5 extensions through Php-cgi.exe.

In IIS 10, ISAPI is implemented using two modules, IsapiModule and IsapiFilterModule. The IsapiModule makes it possible to use ISAPI applications and ISAPI extensions. In the IIS server core, several components rely on handlers that are based on ISAPI extensions, including ASP and ASP.NET. The IsapiModule has a managed handler that specifies that all files with the .dll extension are to be handled as ISAPI extensions. If you remove this module, ISAPI extensions mapped as handlers or explicitly called as ISAPI extensions won't work anymore.

IIS uses ISAPI filters to provide additional functionality. If you selected ASP.NET during initial configuration, an ASP.NET filter is configured to provide classic functionality through aspnet\_filter.dll, an ISAPI filter. For classic ASP.NET functionality, each version of ASP.NET installed on a Web server must have a filter definition that identifies the version and path to the related filter. After you install new versions of ASP.NET, you can add definitions for the related filter.

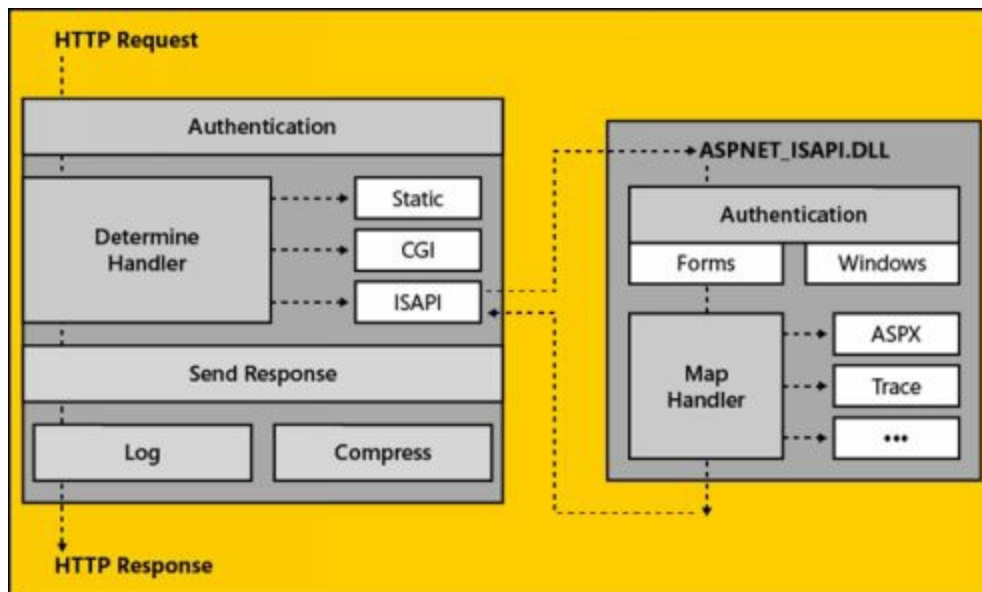
ISAPI and CGI restrictions control the allowed ISAPI and CGI functionality on a server. When you want to use an ISAPI or CGI application, you must specifically allow the related DLL or EXE to run.

# Understanding and Using ASP.NET Applications

When you are working with ASP.NET, it is important to consider the managed pipeline mode you will use. IIS supports two modes for processing requests to ASP.NET applications:

- Classic
- Integrated

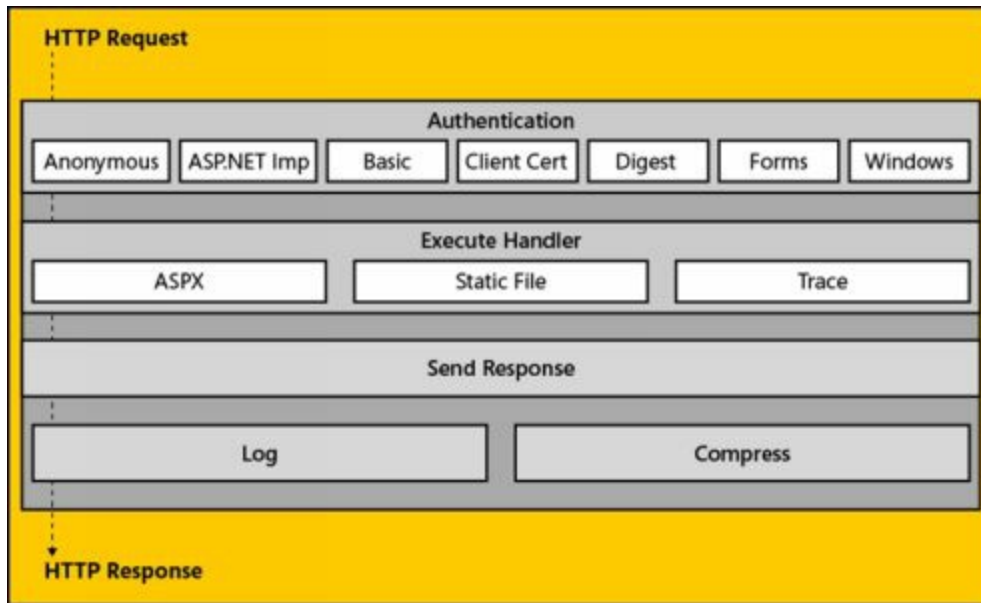
Classic pipeline mode, depicted in the figure, is the standard processing mode used with IIS 6.0.



If a managed web application runs in an application pool with classic mode, IIS processes the requests in an application pool by using separate processing pipelines for IIS and ISAPI. This means that requests for ASP.NET applications are processed in multiple stages like this:

1. The incoming HTTP request is received through the IIS core.
2. The request is processed through ISAPI.
3. The request is processed through ASP.NET.
4. The request passes back through ISAPI.
5. The request passes back through the IIS core where the HTTP response finally is delivered.

Integrated pipeline mode, depicted in the next figure, is a dynamic processing mode that can be used with current releases of IIS. If a managed web application runs in an application pool with integrated mode, IIS processes the requests in an application pool by using an integrated processing pipeline for IIS and ASP.NET.



This means that requests for ASP.NET applications are processed directly like this:

1. The incoming HTTP request is received through the IIS core and ASP.NET.
2. The appropriate handler executes the request and delivers the HTTP response.

From an administrator perspective, applications running in classic pipeline mode can appear to be less responsive than their integrated counterparts. From an application developer perspective, classic pipeline mode has two key limitations. First, services provided by ASP.NET modules and applications are not available to non-ASP.NET requests. Second, ASP.NET modules are unable to affect certain parts of IIS request processing that occurred before and after the ASP.NET execution path.

With an integrated pipeline, all native IIS modules and managed modules can process incoming requests at any stage. This enables services provided by managed modules to be used for requests to pages created using static content, ASP.NET, PHP, and more. Direct integration makes it possible for developers to write custom authentication modules, to create modules that modify request headers before other components process the request, and more.

When working with the integrated pipeline mode, it is important to keep in mind that in this mode ASP.NET does not rely on the ISAPI or ISAPI Extension modules. Because of this, the running of an integrated ASP.NET application is not affected by the ISAPI CGI restriction list. The ISAPI CGI restriction list applies only to ISAPI and CGI applications (which includes ASP.NET classic applications). For integrated mode to work properly, you must specify handler mappings for all custom file types.

Further, many applications written for classic pipeline mode will need to be migrated to run properly in integrated pipeline mode. The good news is that the Configuration

Validation module, included as a part of the server core, can automatically detect an application that requires migration and return an error message stating that the application must be migrated. You can migrate applications by using Appcmd.exe (general- purpose IIS command-line administration tool). Any migration error reported by IIS typically contains the necessary command for migrating the application. To use this command to migrate an application automatically, right-click the command-prompt icon and choose Run As Administrator. You then can migrate an application manually by running the following command at the elevated command prompt:

```
appcmd migrate config AppPath
```

where AppPath is the virtual path of the application. The virtual path contains the name of the associated website and application. For example, if an application named SalesApp was configured on the Default website and needed to be migrated, you could do this by running the following command:

```
appcmd migrate config "Default website/SalesApp"
```

When AppCmd finishes migrating the application, the application will run in both classic and integrated modes.

**REAL WORLD** Although IIS notifies you initially about applications that you need to migrate, IIS will not notify you about migration problems if you subsequently change the application code so that it uses a configuration that is not compatible with integrated mode. In this case, you may find that the application doesn't run or doesn't work as expected, and you'll need to migrate the application manually from a command prompt. If you don't want to see migration error messages, modify the validation element in the application's Web.config file so that its validateIntegratedModeConfiguration attribute is set to false, such as:

```
<system.webServer>
```

```
<validation validateIntegratedModeConfiguration="false" />
```

```
</system.webServer>
```

## Chapter 5. Managing IIS Servers & Services

Core Internet Information Services (IIS) administration tasks revolve around connecting to servers, managing services, and configuring remote administration. In IIS, you connect to individual servers and manage their IIS components through the IIS Manager whether you are working with a local server or a remote server. To perform most administration tasks with sites and servers, you'll need to log in to the IIS server using an account that has administrator privileges.

## IIS Management Essentials

When you installed IIS, you had the opportunity to install the IIS management tools. The standard administration tool for IIS 10 is Internet Information Server (IIS) Manager. The standard administration tool for backwards compatibility with IIS 6.0 websites and services is Internet Information Services (IIS) 6.0 Manager.



# Using IIS Manager

You can access Internet Information Services (IIS) Manager by clicking the Tools menu in Server Manager, and then clicking Internet Information Services (IIS) Manager. IIS Manager automatically connects to the local IIS installation (if available).



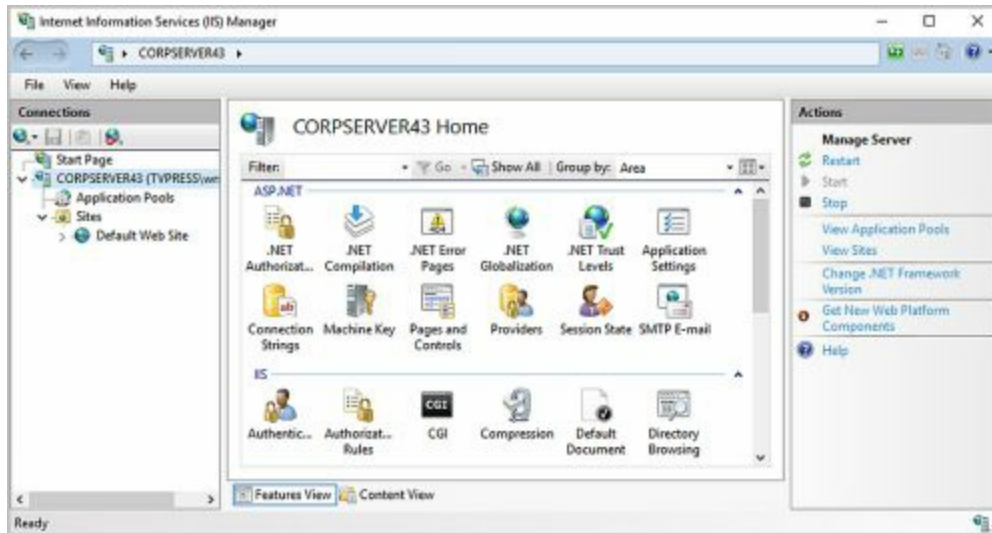
Using the choices available when you select the Start Page node, you can connect to one or more remote servers, sites, and applications as discussed in “Connecting for Administration” in Chapter 1, “IIS 10 Running Start.” Each additional computer, site, or application to which you connect will have a separate node that you can use to manage its resources.

**REAL WORLD** Firewalls and proxy servers might affect your ability to connect to systems at remote locations. If you need to connect regularly to servers through firewalls or proxies, you’ll need to consider the administration techniques you might want to use and then consult your company’s network or security administrator to determine what steps need to be taken to allow those administration techniques. Typically, the network/security administrator will have to open TCP or UDP ports to allow remote communication between your computer or network and the remote computer or network. Each type of tool you want to use might require you to open different ports. By default, the Web Management Service (WMSVC) running on an IIS server listens on TCP port 8172. Because any administrator can easily change the default listen port, you may need to check the current configuration by logging on locally or checking your organization’s configuration policy documentation. Be sure to provide the connection port when setting the server name. For example, to connect to [www.imaginedlands.com](http://www.imaginedlands.com) on TCP port 8175, you’d type the server name as **[www.imaginedlands.com:8175](http://www.imaginedlands.com:8175)**.

## Navigating Nodes

The node level you select determines what IIS Manager displays in the right pane. When you select a server node in the left pane, the right pane displays the core administration tasks. By default, IIS Manager groups the tasks into three areas:

- **ASP.NET** Includes tasks related to managing ASP.NET and the .NET Framework
- **IIS** Includes tasks related to managing sites and applications
- **Management** Includes tasks related to configuring administrative roles, delegation, and remote administration



## Grouping Tasks

Using the Group By drop-down list, you also can select Category to group by category or No Grouping to list the tasks in alphabetical order. The categories are similar to the ones used during set up and include Application Development, Health And Diagnostics, HTTP Features, Performance, Security, and Server Components. The Views button, to the right of the Group By drop-down list, allows you to control how the tasks are listed. The views available are:

- **Details** Lists tasks with a small icon, task name, and summary description
- **Icon** Lists tasks with the task name under a large icon
- **Tiles** Lists tasks with a large icon to the left of the task name
- **List** Lists tasks with a small icon to the left of the task name

## Expanding Nodes

When you expand a server node by double-clicking it, you'll see the following additional nodes as well:

**Application Pools** Allows you to view and manage the application pools on the

server. When you select the Application Pools node, you'll see a list of application pools by name, status, and other key statistics.

**Sites** Allows you to view and manage the websites on the server. When you select the Sites node, you'll see a list of websites on the server organized by name, ID, status, binding, and local directory path. When you expand the Sites node by double-clicking it, you'll see the sites on the server.

**NOTE** You may also see a node for FTP sites. The availability of this node and the way this node works depends on whether you are using classic. I'll refer to FTP as originally implemented in IIS 6 as "classic FTP." Classic FTP runs within the context of IIS 6. This means classic FTP uses IIS 6 compatibility mode and requires IIS 6 compatibility features, such as the IIS Manager console for IIS 6 and the IIS 6 metabase. Otherwise, IIS uses the standard FTP server.

When you select the node for a specific site, you'll see a list of the site's top-level applications and virtual directories. Selecting the node for an application or virtual directory allows you to manage the configuration at that level.

# Configuring Remote Administration

The Web Management Service (WMSVC) enables remote and delegated management of IIS through IIS Manager. You must install, start and configure this service before you can remotely manage a server and before delegated users can perform administration tasks.

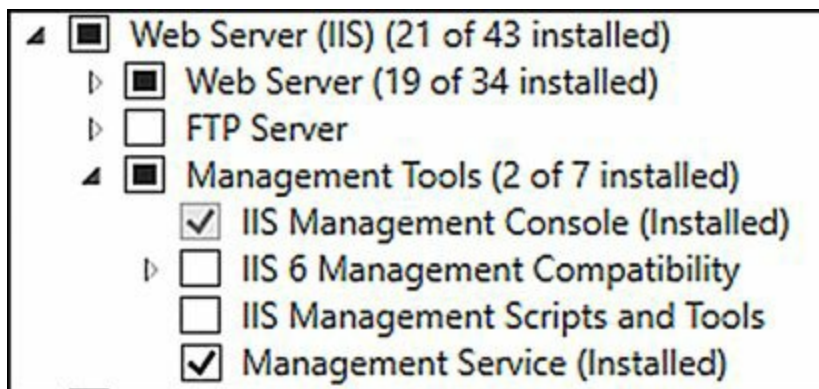
## Adding the Management Service

You can add the Web Management Service to an IIS server by completing these steps:

1. In Server Manager, select Manage and then select Add Roles And Features. This starts the Add Role And Features Wizard. If the wizard displays the Before You Begin page, read the Before You Begin page, and then click Next. You can avoid seeing the Before You Begin page the next time you start this wizard by selecting the Skip This Page By Default check box before clicking Next.
2. On the Installation Type page, Role-Based Or Feature-Based Installation is selected by default. Click Next.
3. On the Server Selection page, you can choose to install roles and features on running servers or virtual hard disks. Either select a server from the server pool or select a server from the server pool on which to mount a virtual hard disk (VHD). If you are adding roles and features to a VHD, click Browse and then use the Browse For Virtual Hard Disks dialog box to locate the VHD. When you are ready to continue, click Next.

**NOTE** Only servers that have been added for management in Server Manager are listed. These servers must be running Windows Server 2012 or later.

4. On the Select Server Roles page, scroll down until you see the Web Server (IIS) role. Under Management Tools, select Management Service.



5. Click Next twice. On the Confirmation page, review the selected options and then click Install when you are ready to proceed.
6. When the wizard finishes installing the Management Service, the Installation

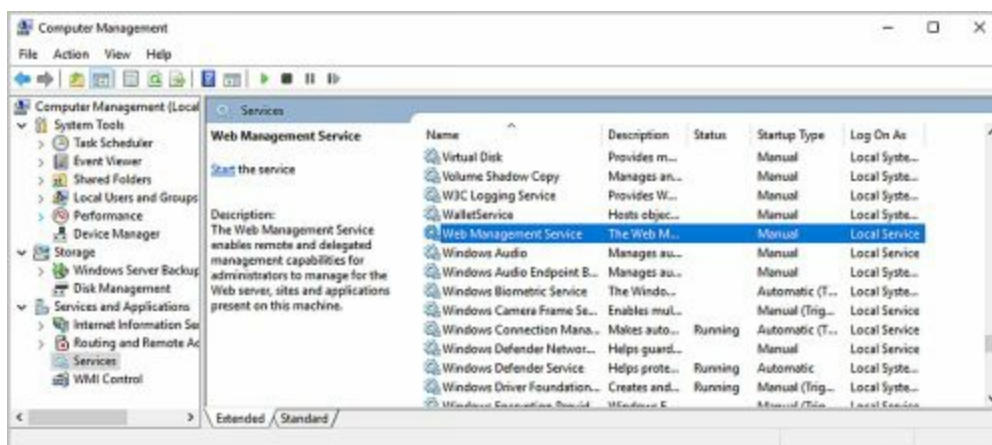
Progress page will be updated to reflect this. Review the installation details to ensure that all phases of the installation were completed successfully.

After you install the Web Management Service, you should restart the Windows Server operating system to ensure the Management Service module can be loaded into the IIS stack. The previously unavailable Manager Service option will then be available in IIS Manager.



The Management Service is configured for manual start by default. If the server will be remotely managed by running Server Manager on other computers, such as your management workstation, you should configure the service to start automatically. To do this, follow these steps:

1. In Computer Management, choose Services under Services And Applications.
2. In the right-hand pane, right-click Web Management Service, and then choose Properties.

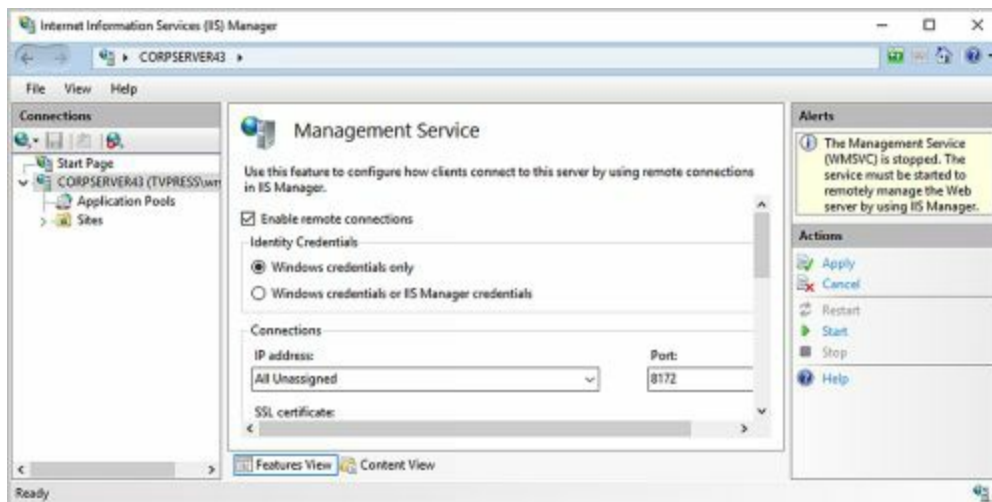


3. On the General tab, choose a startup type in the Startup Type drop-down list. Select Automatic to start the service when the system boots up.
4. Click OK.

## Enabling Remote Administration

You can configure the Web Management Service by completing these steps:

1. Start IIS Manager. In the left pane, select the icon for the computer you want to work with. If the computer isn't shown, connect to it as discussed previously, and then select it.
2. When you group by Area, the Management Service feature is listed under Management. Select the Management Service feature and then in the Actions pane, click Open Feature. This displays the Management Service pane.



3. If the Web Management Service is started, you must stop it before you can configure its properties. Click Stop.
4. If you want to allow local management and local delegated administration only, clear the Enable Remote Connections check box. Otherwise, select this check box to allow remote administration.
5. Under Identity Credentials, use one of the following options to determine the permitted credentials:
  - **Windows Credentials Only** Choose this option to restrict remote access for administration to those individuals with Windows administrator accounts.
  - **Windows Credentials Or IIS Manager Credentials** Choose this option to allow remote access for administration to those individuals with Windows administrator accounts or IIS Manager accounts.
6. Under Connections, use the IP Address drop-down list to select the IP addresses on which the server will listen for remote connections. You can select a specific IP address to allow connections on that IP address only or All Unassigned to allow connections on any configured IP address.
7. Under Connections, in the Port box, type the TCP port number on which the server should listen for remote administrator connections. The default port is TCP port

8172.

8. All remote administration activities are encrypted automatically using SSL. Under Connections, in the SSL Certificate drop-down list, select the certificate the server should use for encryption.
9. All remote administration activities are logged automatically to the %SystemDrive%\Inetpub\logs\WMSvc directory on the IIS server. To use a different directory, click Browse, and then use the Browse For Folder dialog box to select the new logging location. To disable remote administration logging, clear the Log Requests To check box.
10. By default, any client with an IPv4 address can connect to the Web server. To restrict access to clients with specific IP addresses, set Access For Unspecified Clients to Deny and then add allowed clients using the Allow option.
11. Click Start to run the Web Management Service with the updated configuration.

## Restarting the Management Service

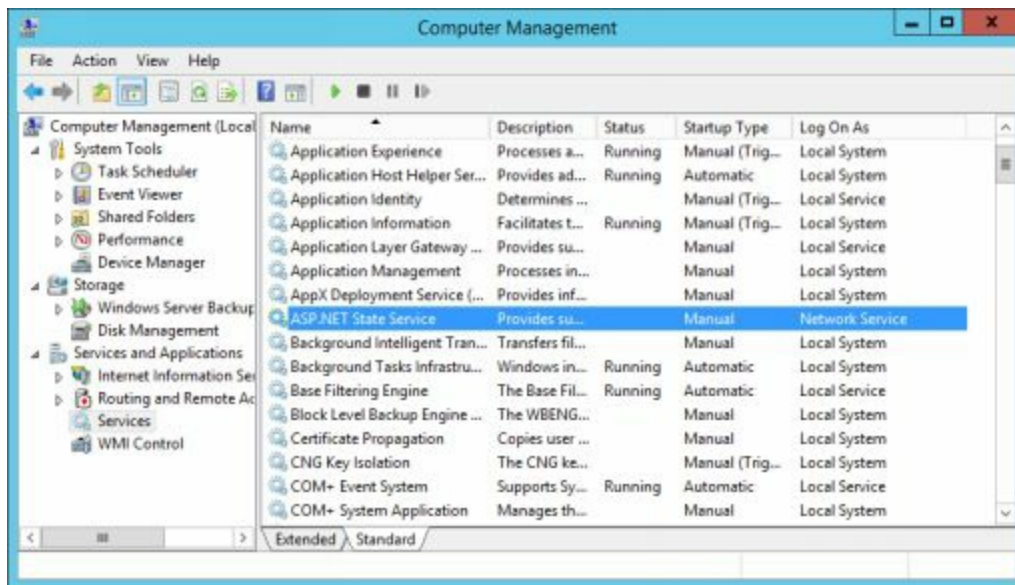
You can start, stop, or restart the Web Management Service by completing these steps:

1. Start the IIS Manager. In the left pane, select the icon for the computer you want to work with. If the computer isn't shown, connect to it as discussed previously, and then select it.
2. When you group by Area, the Management Service feature is listed under Management. Select the Management Service feature and then in the Actions pane, click Open Feature.
3. In the Actions pane, you can do one of the following:
  - Select Start to start the Web Management Service.
  - Select Stop to stop the Web Management Service.
  - Select Restart to stop and then start the Web Management Service as necessary to ensure that the service and all related processes are recycled for troubleshooting.



# Managing IIS Services

Each IIS server in the organization relies on a set of services for publishing pages, transferring files, and more. To manage IIS services, you can use the Services node in either the Server Manager or the Computer Management console. With Server Manager you can manage only local server installations but have additional options for working with server features and roles. With Computer Management, you can work with both local and remote servers.



You can start Computer Management by doing the following:

1. Click the Tools menu in Server Manager, and then Computer Management.
2. If you want to connect to a remote computer, right-click Computer Management in the console tree and on the shortcut menu, select Connect To Another Computer. You can now choose the IIS server whose services you want to manage.
3. Expand the Services And Applications node by clicking the triangle-shaped node icon next to it, and then choose Services.

The key fields of this dialog box are used as follows:

- **Name** The name of the service.
- **Description** A short description of the service and its purpose.
- **Status** The status of the service as Started, Paused, or Stopped. (Stopped is indicated by a blank space.)
- **Startup Type** The startup setting for the service.

<p><b>NOTE</b> Automatic services are started when the system boots up. Manual services are started by users or other services. Disabled services are turned off</p>
--



and can't be started.

- **Log On As** The account the service logs on as. The default in most cases is the local system account.

# Starting, Stopping, and Pausing IIS Services

As an administrator, you'll often have to start, stop, or pause IIS services. You manage IIS services through the Computer Management console or through the Services console. When you manage IIS services at this level, you're controlling all sites or virtual servers that use the service. For example, if a computer publishes three websites and you stop the World Wide Web Publishing Service, all three websites are stopped and are inaccessible.

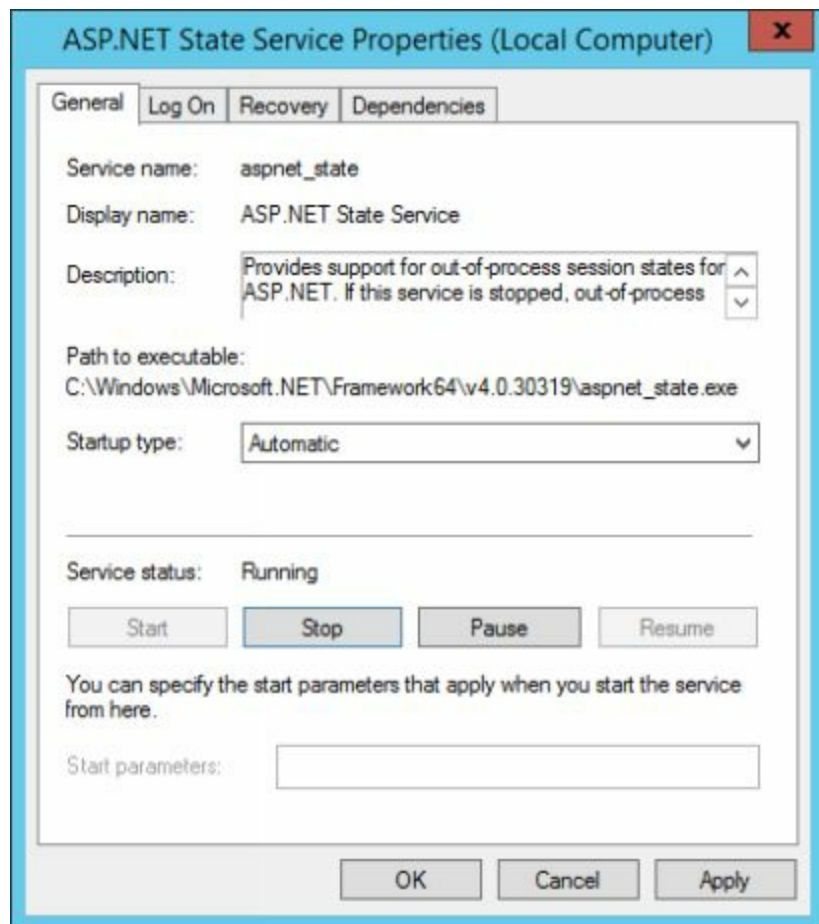
To start, stop, or pause services in the Computer Management console, follow these steps:

1. In the left pane, right-click Computer Management in the console tree and on the shortcut menu, select **Connect to Another Computer**. You can now choose the IIS server whose services you want to manage.
2. Expand the **Services And Applications** node by clicking the triangle-shaped node icon next to it, and then choose **Services**.
3. In the right pane, right-click the service you want to manipulate, and then select **Start**, **Stop**, or **Pause** as appropriate. You can also choose **Restart** to have Windows stop and then start the service after a brief pause. In addition, if you pause a service, you can select **Resume** to resume normal operation.

**TIP** When services that are set to start automatically fail to do so, the status area is blank, and you'll usually receive notification in a dialog box. Service failures can also be logged to the system's event logs. In Windows Server, you can configure actions to handle service failures automatically. For example, you could have Windows Server attempt to restart the service for you. For details, see the section of this chapter titled "Configuring Service Recovery."

# Configuring Service Startup

Most IIS services are configured to start automatically, and normally they shouldn't be configured with another startup setting. That said, if you're troubleshooting a problem, you might want a service to start manually. You might also want to disable a service so that its related virtual servers don't start. For example, if you move an FTP server to a new server, you might want to disable the FTP Publishing service on the original IIS server. In this way the FTP Publishing service isn't used, but you could turn it on if you need to (without your having to reinstall FTP support).



You configure service startup as follows:

1. In the left pane of the Computer Management console, connect to the IIS server whose services you want to manage.
2. Expand the Services And Applications node by clicking the the triangle-shaped node icon next to it, and then choose Services.
3. In the right-hand pane, right-click the service you want to configure, and then choose Properties.
4. On the General tab, choose a startup type in the Startup Type drop-down list.

Select Automatic to start the service when the system boots up. Select Automatic (Delayed Start) to delay the start until other automatic services are started. Select Manual to allow the service to be started manually. Select Disabled to turn off the service.

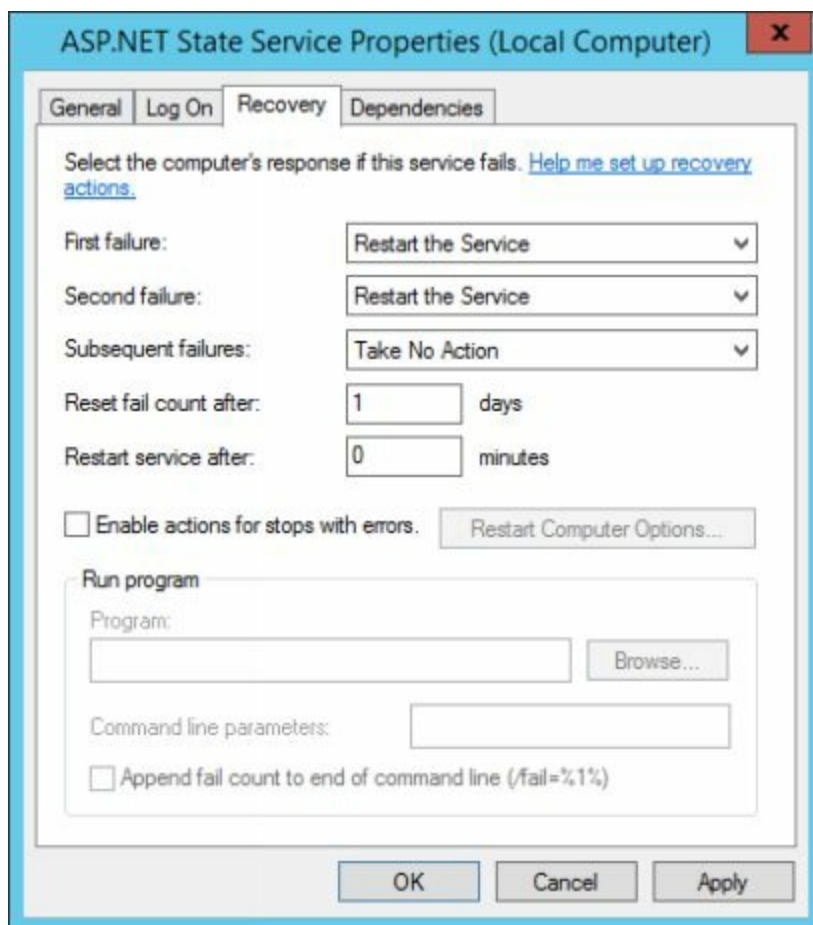
5. Click OK.

# Configuring Service Recovery

You can configure Windows services to take specific actions when a service fails. For example, you could attempt to restart the service or reboot the server.

To configure recovery options for a service, follow these steps:

1. In the left pane of the Computer Management console, connect to the computer whose services you want to manage.
2. Expand the Services And Applications node by clicking the triangle-shaped node icon next to it, and then choose Services.
3. In the right pane, right-click the service you want to configure, and then choose Properties.
4. Select the Recovery tab. You can now configure recovery choices for the first, second, and subsequent recovery attempts. The available choices are:
  - Take No Action
  - Restart The Service
  - Run A Program
  - Restart The Computer



The screenshot shows the 'ASP.NET State Service Properties (Local Computer)' dialog box with the 'Recovery' tab selected. The dialog has four tabs: 'General', 'Log On', 'Recovery', and 'Dependencies'. The 'Recovery' tab contains the following settings:

- Select the computer's response if this service fails.** [Help me set up recovery actions.](#)
- First failure:** Restart the Service (dropdown)
- Second failure:** Restart the Service (dropdown)
- Subsequent failures:** Take No Action (dropdown)
- Reset fail count after:** 1 days
- Restart service after:** 0 minutes
- ☐ **Enable actions for stops with errors.** [Restart Computer Options...](#)
- Run program**
  - Program:** [text box] [Browse...](#)
  - Command line parameters:** [text box]
  - ☐ **Append fail count to end of command line (/fail=%1%)**

At the bottom of the dialog are three buttons: 'OK', 'Cancel', and 'Apply'.

5. Configure other settings based on your previously selected recovery settings. If you elected to restart the service, you'll need to specify the restart delay. After stopping the service, Windows waits for the specified delay before trying to start the service. In most cases a delay of 1–2 minutes should be sufficient.
6. Click OK.

When you configure recovery options for critical services, you might want Windows to try to restart the service on the first and second attempts and then reboot the server on the third attempt.

## Other Management Options for IIS Services

IIS includes a few other options for managing services. These include a command-line tool, called `Iisreset`, and options in IIS Manager.

# Going Old School

You can use the `iisreset.exe` command-line utility to start, stop, and restart IIS services. With a standard installation of IIS, `iisreset` only controls the World Wide Web Publishing Service and the Windows Process Activation Service. If you've also installed IIS 6 or FTP, the IIS Admin Service and FTP Publishing Service also are controlled with `iisreset`.

**TIP** By default, FTP Publishing Service is configured for manual startup only. Because of this, if you use IIS 6.0 Manager or `iisreset` to start or restart Internet services, FTP Publishing Service will not be started. To ensure that FTP Publishing Service is started or restarted, you must set the startup type to Automatic.

To start services that are stopped on the local computer, type the following command:

```
iisreset /start
```

To stop IIS services that are running, paused, or in an unknown state on the local computer, type the following command:

```
iisreset /stop
```

To stop and then restart IIS services on the local computer, type the following command:

```
iisreset /restart
```

You can also control IIS services on remote computers. To do this, use the following syntax:

```
iisreset computername command
```

such as:

```
iisreset engsvr01 /restart
```

With the Restart command, the sequence of tasks is important to understand. This command performs the following tasks:

1. Stops Internet Information Services running on the computer.
2. Attempts to resolve potential problems with runaway processes or hung applications by stopping all related processes.
3. Starts IIS services and then starts DLL Hosts as necessary.

Table 5-1 provides a listing of all switches for the `iisreset.exe` command-line utility. Rebooting computers is covered in the section of this chapter titled “Rebooting IIS



Servers.”

**TABLE 5-1 IISRESET Switch Functions**

/DISABLE	Disables restarting of IIS services on the local system.
/ENABLE	Enables restarting of IIS services on the local system.
/NOFORCE	Doesn't forcefully terminate IIS services if attempting to stop them gracefully fails.
/REBOOT	Reboots the local or designated remote computer.
/REBOOTONERROR	Reboots the computer if an error occurs when starting, stopping, or restarting IIS services.
/RESTART	Stops and then restart all IIS services. Attempts to resolve potential problems with runaway processes or hung applications.
/START	Starts all IIS services that are stopped.
/STATUS	Displays the status of all IIS services.
/STOP	Stops all IIS services that are running, paused, or in an unknown state.
/TIMEOUT:val	Specifies the time-out value (in seconds) to wait for a successful stop of IIS services. On expiration of this time-out, the computer can be rebooted if the /REBOOTONERROR parameter is specified. With /STOP and /RESTART, an error is issued. The default value is 20 seconds for restart, 60 seconds for stop, and 0 seconds for reboot.

# Using IIS Manager

Sites and virtual servers that use the same IIS services can be controlled individually or as a group. You can control individual sites and virtual servers much as you do other server resources. For example, if you're changing the configuration of a site or performing other maintenance tasks, you might need to stop the site, make the changes, and then restart it. When a site is stopped, the site doesn't accept connections from users and can't be used.

In IIS Manager, you can start, stop, or restart all websites published on a server by following these steps:

1. Start IIS Manager.
2. In the left pane, select the icon for the computer you want to work with. If the computer isn't shown, connect to it as discussed previously, and then select it.
3. In the Actions pane, you can do one of the following:
  - Select Start to start the World Wide Web Publishing Service and make all websites on the server available.
  - Select Stop to stop the World Wide Web Publishing Service and make all websites on the server unavailable.
  - Select Restart to stop and then start the World Wide Web Publishing Service as necessary to ensure that the service and all related processes are recycled for troubleshooting.

In IIS Manager, you can start, stop, or restart an individual website by following these steps:

1. Start IIS Manager.
2. In the left pane, expand the node for the computer you want to work with. If the computer isn't shown, connect to it as discussed previously, and then expand the computer node.
3. With the Sites node selected in the left pane, in the Name list, click the website you want to work with.
4. In the Actions pane under Manage website, select Start, Stop, or Restart to start, stop, or restart the selected website.

# Rebooting IIS Servers

Using the `iisreset.exe` utility, you can reboot local and remote computers. To use this feature, you must have installed IIS on the computer and you must be a member of a group that has the appropriate user rights. To reboot a local system, you must have the right to shut down the system. To reboot a remote system, you must have the right to force shutdown from a remote system. You should reboot an IIS server only if the Restart IIS procedure fails.

To reboot a computer by using `iisreset.exe`, type the following command:

```
iisreset computername /reboot
```

such as in the following example:

```
iisreset engsvr01 /reboot
```

If users are working on files or performing other tasks that need to be exited gracefully, you should set a time-out value for services and processes to be stopped. By default, the time-out is zero seconds, which forces immediate shutdown and tells Windows Server not to wait for services to be shut down gracefully. You could set a time-out value of 60 seconds when rebooting `engsvr01` as follows:

```
iisreset engsvr01 /reboot /timeout:60
```

## Chapter 6. Managing IIS 10 from the Prompt

IIS uses the IIS Command-line Administration Tool (AppCmd.exe) to complement the expanding role of IIS administrators and developers. AppCmd provides an extensible command-line environment for IIS that builds on the existing framework provided by Microsoft .NET Framework. When you install IIS, the installation process installs AppCmd when you select the IIS Management Scripts and Tools role service. Another command-line environment you can use to work with IIS is Windows PowerShell. Once you install Windows PowerShell, you can use its capabilities to configure and manage IIS.

# Command Line Administration

IIS Manager and other graphical tools provide just about everything you need to work with IIS. 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 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. Alternatively, you can modify the server's path environment variable using the Advanced System Settings in Control panel.

Table 6-1 provides a summary of the core set of management objects for the IIS command-line administration tool.

**TABLE 6-1** Administration Objects for the IIS Command-Line Administration Tool

APP	Allows you to create and manage web application settings by using related list, set, add, and delete commands
APPPool	Allows you to create and manage application pools by using related list, set, add, delete, start, stop, and recycle commands
BACKUP	Allows you to create and manage backups of your server configuration by using list, add, delete, and restore commands
CONFIG	Allows you to manage general configuration settings by using related list, set, search, lock, unlock, clear, reset, and migrate commands
MODULE	Allows you to manage IIS modules by using related list, set, add, delete, install, and uninstall commands
REQUEST	Allows you to list current HTTP requests by using a related list command
SITE	Allows you to create and manage virtual sites by using related list, set, add, delete, start, and stop commands

TRACE	Allows you to manage failed request tracing by using related list, configure, and inspect commands
VDIR	Allows you to create and manage virtual directory settings by using related list, set, add, and delete commands
WP	Allows you to list running worker processes by using a related list command

The basics of working with the IIS command-line administration tool are straightforward. Most management 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 website" (id:1,bindings:http/*:80;,state:Started)
```

You'll find a comprehensive discussion of using management objects for administration in Chapter 7, "Using Management Objects for Administration." In addition, you will see

examples of using this tool throughout the book.

## Using Windows PowerShell

Anyone with a UNIX background is probably familiar with the concept of a command shell. Most UNIX-based operating systems have several full-featured command shells available, including Korn shell (KSH), C shell (CSH), and Bourne Shell (SH).

Although Microsoft Windows operating systems have always had a command-line environment, they've lacked a full-featured command shell, and this is where Windows PowerShell comes into the picture.



# Introducing Windows PowerShell

Not unlike the less-sophisticated Windows command prompt, the UNIX command shells operate by executing built-in commands, external commands, and command-line utilities and then returning the results in an output stream as text. The output stream can be manipulated in various ways, including redirecting the output stream so that it can be used as input for another command. This process of redirecting one command's output to another command's input is called piping, and it is a widely-used technique for shell scripting.

The C Shell is one of the more sophisticated UNIX shells. In many respects, C Shell is a marriage of some of the best features of the C programming language and a full-featured UNIX shell environment. Windows PowerShell takes the idea of a full-featured command shell built on a programming language a step further. It does this by implementing a scripting language based on Microsoft C# and an object model based on the .NET Framework.

Basing the scripting language for Windows PowerShell on C# ensures that the scripting language can be easily understood by current C# developers and also allows new developers to advance to C#. Using an object model based on the .NET Framework allows Windows PowerShell to pass complete objects and all their properties as output from one command to another. The ability to redirect objects is extremely powerful and allows for a much more dynamic manipulation of a result set. For example, not only can you get the name of a particular user, but you also can get the entire related user object. You can then manipulate the properties of this user object as necessary by referring to the properties you want to work with by name.

# Running and Using Windows PowerShell

Windows PowerShell is built into current versions of Windows and Windows Server. On these operating systems, you can start the PowerShell console by using the Search box. Type powershell in the Search box, and then press Enter. Or you can click the Windows logo, click the Apps button, and then choose Windows PowerShell.

Although Windows PowerShell runs by default with standard user privileges, you'll often need administrator privileges when working with PowerShell. You can run PowerShell in elevated, administrator mode by using the Search box. Type powershell in the Search box, right-click Windows PowerShell in the search results and then select Run As Administrator.

By default with Windows 10 and Windows Server 2016, Command Prompt and Command Prompt (Admin) are options on the shortcut menu that is displayed when you right-click in the lower left corner or press Windows key + X. The alternative is for the Windows PowerShell prompt and the Windows PowerShell (Admin) prompt to be displayed on this menu. To configure which options are available, on the desktop, right-click the taskbar and then select Properties. In the Taskbar And Navigation Properties dialog box, on the Navigation tab, select or clear the Replace Command Prompt With Windows PowerShell... checkbox as appropriate.

Usually, when the shell starts, you will see a message similar to the following:

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
```

You can disable this message by starting the shell with the `-nologo` parameter, like so:

**powershell -nologo**

For a complete list of PowerShell parameters, type **powershell -?**.

When the shell starts, user and system profiles are run to set up the environment. The following is a list and description of the profile files run, in the order of their execution:

1. [%AllUsersProfile%\My Documents\WindowsPowerShell\Microsoft.PowerShell\\_profile.ps1](#)

A system-wide profile executed for all users. This profile is used by the system administrator to configure common settings for the Windows PowerShell.

2. [%UserProfile%\My Documents\WindowsPowerShell\Microsoft.PowerShell\\_profile.ps1](#)

A user-specific profile for the logged on user. This profile is used by the current

user to configure specific user settings for the Windows PowerShell.

To start Windows PowerShell without loading profiles, use the `-nopprofile` parameter, like so:

```
powershell -nopprofile
```

The first time you start Windows PowerShell, you typically see a message indicating that scripts are disabled and that none of the listed profiles is executed. This is the default secure configuration for the Windows PowerShell. To enable scripts for execution, type the following command at the PowerShell prompt:

```
set-executionpolicy allsigned
```

This command sets the execution policy to require all scripts to have a trusted signature to execute. For a less restrictive environment, you can run the following command:

```
set-executionpolicy remotesigned
```

This command sets the execution policy so that scripts downloaded from the Web execute only if they are signed by a trusted source. To work in an unrestricted environment, you can run the following command:

```
set-executionpolicy unrestricted
```

This command sets the execution policy to run scripts regardless of whether they have a digital signature.

# Running and Using Cmdlets

Windows PowerShell introduces the concept of a cmdlet (pronounced “command let”). A cmdlet is the smallest unit of functionality in the Windows PowerShell. You can think of a cmdlet as a built-in command. Rather than being highly complex, most cmdlets are quite simple and have a small set of associated properties.

You use cmdlets the same way you use any other commands and utilities. Cmdlet names are not case-sensitive. This means that you can use a combination of both uppercase and lowercase characters. After starting the Windows PowerShell, you can type the name of the cmdlet at the prompt and it will run in much the same way as a command-line command.

For ease of reference, cmdlets are named using verb-noun pairs. As Table 6-2 shows, the verb tells you what the cmdlet does in general. The noun tells you what specifically the cmdlet works with. For example, the get-variable cmdlet gets a named Windows PowerShell environment variable and returns its value. If you don’t specify which variable to get as a parameter, get-variable returns a list of all Windows PowerShell environment variables and their values.

**TABLE 6-2** Common Verbs Associated with Cmdlets and Their Meanings

New-	Creates a new instance of an item or object
Remove-	Removes an instance of an item or object
Set-	Modifies specific settings of an object
Get-	Queries a specific object or a subset of a type of object

You can work with cmdlets in two ways:

- Executing commands directly at the shell prompt
- Running commands from within scripts

You can enter any command or cmdlet you can run at the Windows PowerShell command prompt into a script by copying the related command text to a file and saving the file with the .ps1 extension. You can then run the script in the same way you would any other command or cmdlet.

**NOTE** Windows PowerShell also includes a rich scripting language and allows the use of standard language constructs for looping, conditional execution, flow control, and variable assignment. Discussion of these features is beyond the scope of this book.

From the Windows command-line environment or a batch script, you can execute Windows PowerShell cmdlets with the `-command` parameter. Typically when you do this, you will also want to suppress the Windows PowerShell logo and stop execution of profiles. After doing this, you could type the following command at a command prompt or insert it into a `.bat` script:

```
powershell -nologo -nopprofile -command get-service
```

Finally, when you are working with Windows PowerShell, it is important to remember that the current directory may not be part of the environment path. Because of this, you may need to use “`.\`” when you run a script in the current directory, such as:

```
.\runtasks
```

# Running and Using Other Commands and Utilities

Because Windows PowerShell runs within the context of the Windows command prompt, you can run all Windows command-line commands, utilities, and graphical applications from within the Windows PowerShell. However, it is important to remember that the Windows PowerShell interpreter parses all commands before passing off the command to the command prompt environment. If the Windows PowerShell has a like-named command or a like-named alias for a command, this command is executed rather than the expected Windows command. (See the “Using Cmdlet Aliases” section later in this chapter for more information on aliases.)

PowerShell commands and programs not used in Windows must reside in a directory that is part of the PATH environment variable. If the item is found in the path, it is run. The PATH variable also controls where the Windows PowerShell looks for applications, utilities, and scripts. In Windows PowerShell, you can work with Windows environment variables by using `$env`. If you want to view the current settings for the PATH environment variable, type **`$env:path`**. If you want to add a directory to this variable, you can use the following syntax:

```
$env:path += ";DirectoryPathToAdd"
```

where `DirectoryPathToAdd` is the directory path you want to add to the path, such as:

```
$env:path += ";C:\Scripts"
```

To have this directory added to the path every time you start the Windows PowerShell, you can add the command line as an entry in your profile. Keep in mind that cmdlets are like built-in commands rather than stand-alone executables. Because of this, they are not affected by the PATH environment variable.

## Working with Cmdlets

Cmdlets provide the basic foundation for working with a computer from within the Windows PowerShell. Although there are many different cmdlets for many different uses, cmdlets all have common features. I'll examine these common features in this section.

# Using Windows PowerShell Cmdlets

At the Windows PowerShell prompt, you can get a complete list of available cmdlets by typing **help \***. To get help documentation on a specific cmdlet, type **help** followed by the cmdlet name, such as:

```
help get-variable
```

Table 6-3 provides a list of cmdlets you'll use commonly for administration. Although there are many other available cmdlets, these are the ones you're likely to use the most.

**TABLE 6-3** Cmdlets Commonly Used for Administration

ConvertFrom-SecureString	Exports a secure string to a safe format
ConvertTo-SecureString	Creates a secure string from a normal string
Get-Alias	Returns alias names for cmdlets
Get-AuthenticodeSignature	Gets the signature object associated with a file
Get-Credential	Gets a credential object based on a password
Get-Date	Gets the current date and time
Get-EventLog	Gets the log data from the Windows log files
Get-ExecutionPolicy	Gets the effective execution policy for the current shell
Get-Host	Gets host information
Get-Location	Displays the current location
Get-PSDrive	Gets the drive information for the specified PS drive
Get-Service	Gets a list of services
Import-Alias	Imports an alias list from a file
New-Alias	Creates a new cmdlet-alias pairing
New-Service	Creates a new service
Push-Location	Pushes a location to the stack
Read-Host	Reads a line of input from the host console
Restart-Service	Restarts a stopped service
Resume-Service	Resumes a suspended service
Set-Alias	Maps an alias to a cmdlet
Set-AuthenticodeSignature	Places an Authenticode signature in a script or other file
Set-Date	Sets the system date and time on the host system
Set-ExecutionPolicy	Sets the execution policy for the current shell
Set-Location	Sets the current working location to a specified location
Set-Service	Makes and sets changes to the properties of a service
Start-Service	Starts a stopped service



Start-Sleep	Suspends shell or script activity for the specified period
Stop-Service	Stops a running service
Suspend-Service	Suspends a running service
Write-Output	Writes an object to the pipeline

# Using Cmdlet Parameters

All cmdlet parameters are designated with an initial hyphen (-). To reduce the amount of typing required, some parameters are position-sensitive such that you can sometimes pass parameters in a specific order without having to specify the parameter name. For example, with `get-service`, you aren't required to specify the `-Name` parameter; you can type simply:

```
Get-service ServiceName
```

where `ServiceName` is the name of the service you want to examine, such as:

```
Get-service W3SVC
```

This command line returns the status of the World Wide Web Publishing Service. Because you can use wildcards, such as `*`, with name values, you can also type **`get-service w*`** to return the status of all services that start with `W`, including Web Management Service, Windows Process Activation Service, and World Wide Web Publishing Service.

All cmdlets support the common set of parameters listed in Table 6-4. However, for you to use these parameters, you must run the cmdlet in such a way that these parameters are returned as part of the result set.

**TABLE 6-4** Common Cmdlet Parameters

-Confirm	Pauses processes and requires the user to acknowledge the action before continuing. Remove- and Disable- cmdlets have this parameter.
-Debug	Provides programming-level debugging information about the operation.
-ErrorAction	Controls the command behavior when an error occurs.
-ErrorVariable	Sets the name of the variable (in addition to the standard error) in which to place objects for which an error has occurred.
-OutBuffer	Sets the output buffer for the cmdlet.
-OutVariable	Sets the name of the variable in which to place output objects.
-Verbose	Provides detailed information about the operation.
-WhatIf	Allows the user to view what would happen if a cmdlet were run with a specific set of parameters. Remove- and Disable- cmdlets have this parameter.

# Understanding Cmdlet Errors

When you work with cmdlets, you'll encounter two standard types of errors:

- **Terminating errors** Errors that halt execution
- **Nonterminating errors** Errors that cause error output to be returned but do not halt execution

With both types of errors, you'll typically see error text that can help you resolve the problem that caused it. For example, an expected file might be missing or you may not have sufficient permissions to perform a specified task.

# Using Cmdlet Aliases

For ease of use, Windows PowerShell lets you create aliases for cmdlets. An alias is an abbreviation for a cmdlet that acts as a shortcut for executing the cmdlet. For example, you can use the alias **gsv** instead of the cmdlet name **get-service**.

Table 6-5 provides a list of commonly used default aliases. Although there are many other aliases, these are the ones you'll use most frequently.

**TABLE 6-5** Commonly Used Cmdlet Aliases

clear, cls	Clear-Host
Diff	Compare-Object
cp, copy	Copy-Item
Epal	Export-Alias
Epcsv	Export-Csv
Foreach	ForEach-Object
Fl	Format-List
Ft	Format-Table
Fw	Format-Wide
Gal	Get-Alias
ls, dir	Get-ChildItem
gcm	Get-Command
cat, type	Get-Content
h, history	Get-History
gl, pwd	Get-Location
gps, ps	Get-Process
gsv	Get-Service
gv	Get-Variable
group	Group-Object
ipal	Import-Alias
ipcsv	Import-Csv
r	Invoke-History
ni	New-Item
mount	New-PSDrive

nv	New-Variable
rd, rm, rmdir, del, erase	Remove-Item
rv	Remove-Variable
sal	Set-Alias
sl, cd, chdir	Set-Location
sv, set	Set-Variable
sort	Sort-Object
sasv	Start-Service
sleep	Start-Sleep
spps, kill	Stop-Process
spsv	Stop-Service
write, echo	Write-Output

You can create additional aliases using the Set-Alias cmdlet. The syntax is:

```
Set-alias aliasName cmdletName
```

where aliasName is the alias you want to use and cmdletName is the cmdlet for which you are creating an alias. The following example creates a “go” alias for the get-process cmdlet:

```
Set-alias go get-process
```

To use your custom aliases whenever you work with Windows PowerShell, type the related command line in your profile.

# Using Cmdlets with IIS

Using cmdlets with IIS is a simple matter of running cmdlets in a way that affects the IIS installation. For example, you can use the Get-Service, Start-Service, Pause-Service, Resume-Service, and Stop-Service cmdlets to manage the services used by IIS. New Windows PowerShell cmdlets that are specific to IIS also will become available periodically for your use in managing IIS servers. As these cmdlets become available, you'll be able to install them through server updates or by downloading and installation an installer program.

Although the shared configuration feature of IIS is much more efficient, Windows PowerShell could be used to help you deploy the same IIS configuration to multiple servers. Listing 6-1 provides the source code and examples for doing this.

## LISTING 6-1 Deploying IIS with Windows PowerShell

### Listing for ServerList.txt

```
WebServer84
WebServer92
WebServer76
WebServer15
```

### Listing for FileList.txt

```
C:\windows\microsoft.net\framework\v4.0.30319\config\machine.config
C:\windows\microsoft.net\framework\v4.0.30319\config\web.config
C:\windows\System32\inetsrv\config\applicationHost.config
C:\inetpub\wwwroot\web.config
C:\inetpub\wwwroot\SalesApp\web.config
C:\inetpub\wwwroot\SupportApp\web.config
```

### DeployServers.ps1

```
#####
# sourceComputer sets the source computer for the deployment.
# This scripts looks for two text files in the same directory:
# ServerList.txt sets the list of servers for the deployment.
# FileList.txt sets the list of files to copy.
#####

$sourceComputer = "WebServer95"
$serverList = get-content '.\ServerList.txt'
$filesToCopy = get-content '.\FileList.txt'

foreach ($targetComputer in $serverList)
{
    foreach ($file in $filesToCopy)
    {
        $sourcePath = "\\" + (join-path $sourceComputer $file)
        $targetPath = "\\" + (join-path $targetComputer $file)
    }
}
```

```

write-host -for this "$targetComputer : Copying files from $sourcePath"

copy-item $sourcePath $targetPath -recurse -force
}
}

```

This listing uses three source files that you've placed in the same directory:

- **ServerList.txt** contains a list of servers to which you want to copy configuration files and for which you have full administrator privileges.
- **FileList.txt** contains a list of configuration files you want to copy from the source according to the full file paths listed.
- **DeployServers.ps1** contains the script that you will execute to deploy the configuration to the previously listed servers. In this script, the `$sourceComputer` variable sets the name of the source server for the deployment.

Before you use these files you should update them and test them in a development environment. If you update these files for your environment and then execute `DeployServers.ps1` from a Windows PowerShell command line, you will copy the configuration files from your source server to the designated target servers. The script uses the `get-content` cmdlet to read computer names from the `ServerList.txt` file and configuration files from the `FileList.txt` file. The outer `foreach` loop iterates through each computer name stored in the server list, storing each name into the `$targetComputer` variable in turn. The inner loop iterates through each file name in the file list, storing each name in the `$file` variable in turn. Next, the `join-path` cmdlet is used to concatenate strings to produce complete source and destination paths. Finally, the `copy-item` cmdlet is used to perform the copy actions, whereas the `-recurse` parameter will copy all subdirectories (if necessary) and the `-force` parameter causes existing files to be overwritten.

## Chapter 7. Using Management Objects for Administration

The IIS command line administration tool (AppCmd.exe) is a command-line management interface built on the .NET Framework. You use AppCmd to manage most aspects of IIS configuration that you would otherwise manage in IIS Manager. This means that you can typically use either tool to configure IIS.



## Getting Started with AppCmd

After you've installed the IIS Management Scripts and Tools role service as discussed in Chapter 3, "Setting Up IIS," you can use AppCmd to manage the configuration of an IIS server from the command line. AppCmd is located in the %SystemRoot%\System32\Inetsrv directory. Because this directory is not in your command path by default, you should add this directory to your command path. Once you've done so, you can run AppCmd by completing the following steps:

1. By default with Windows 10 and Windows Server 2016, Command Prompt and Command Prompt (Admin) are options on the shortcut menu that is displayed when you right-click in the lower left corner or press Windows key + X. If your server has PowerShell configured as options instead, type cmd in the Search box then right-click Command Prompt and select Run As Administrator.
2. In the Command Prompt window, type the necessary command text, or run a script that invokes AppCmd.

# Navigating the .NET Framework Objects

Because AppCmd is an extension of .NET Framework, a specific set of management objects are available for your use. As Table 7-1 shows, these objects are identified by .NET Framework class IDs. Each object class has an instance name and an alias. For example, DefaultSiteObjectClass, which is used to configure the properties of websites, is assigned the instance name site and the alias sites. This allows you to reference DefaultSiteObjectClass using either site or sites in the command text. The actions (commands) you can perform on an object are defined as a list of verb names that are passed through to the object. For DefaultSiteObject, the related actions you can perform are List, Set, Add, Delete, Start, and Stop.

**TABLE 7-1** IIS Management Objects

OBJECT NAME	OBJECT CLASS ID	OBJECT ALIAS
App	DefaultAppObject	Apps
Apppool	DefaultAppPoolObject	Apppools
Backup	DefaultBackupObject	Backups
Config	DefaultConfigObject	Configs
Vdir	DefaultDirObject	Vdirs
Module	DefaultModuleObject	Modules
Request	DefaultRequestObject	Requests
Site	DefaultSiteObject	Sites
Trace	DefaultTraceObject	Traces
Wp	DefaultWorkerProcessObject	Wps

AppCmd has a helper file called Appcmd.xml. This file, written in XML, helps the command-line tool when you are working with .NET Framework objects, actions, and aliases. As the following example shows, entries in Appcmd.xml are organized according to management objects and the related actions:

```
<appcmd>
  <object name="site" alias="sites" classId="DefaultSiteObject" >
    <verb name="list" classId="DefaultSiteObject" />
    <verb name="set" classId="DefaultSiteObject" />
    <verb name="add" classId="DefaultSiteObject" />
    <verb name="delete" classId="DefaultSiteObject" />
    <verb name="start" classId="DefaultSiteObject" />
    <verb name="stop" classId="DefaultSiteObject" />
  </object>
  . . .
  <object name="trace" alias="traces" classId="DefaultTraceObject" >
    <verb name="list" classId="DefaultTraceObject" />
```

```
<verb name="configure" classId="DefaultTraceObject" />
<verb name="inspect" classId="DefaultTraceObject" />
</object>
</appcmd>
```

Although this is an important file for AppCmd's internal use, it is not one you can or should modify. In fact, if you modify this file, AppCmd may not be able to initialize objects properly according to their class IDs, which will cause AppCmd to fail to run.

# Using AppCmd

When you are working with AppCmd, you can get help information on available commands:

- To view a list of management objects and general parameters, type **appcmd** at the command prompt.
- To view actions related to a specific management object, type **appcmd ObjectName /?** where ObjectName is the name of the management object you want to examine, such as **appcmd trace /?**.
- To view the syntax for an action used with a particular object, type **appcmd Action ObjectName /?** where Action is the action to perform and ObjectName is the name of the management object on which you want to perform the action, such as **appcmd configure trace /?**.

When you work with AppCmd, you'll find that just about every command accepts parameters and specific parameter values that qualify what you want to work with. These parameters and parameter values correspond to attributes and attribute values assigned in IIS configuration files. Further, most commands require that you specify the name of the configuration feature or property you are working with. Typically, you can specify the name in a relative way, such as **"Default website"** when referring to a website or **"Default website/SalesApp"**. You can also specify a name in a literal way by referring to the exact type of name you are working with, such as **/site.name="Default website"** when referring to a website or **/app.name="Default website/SalesApp"**. To see more clearly how this works, consider the following syntax example:

```
appcmd add module [/module.name:] "ModuleName"  
[/app.name: "AppPath"]
```

In this syntax example, the brackets tell you that **/module.name:** is optional. Thus, you could specify the module name using either of the following syntaxes:

```
appcmd add module "CustModule"
```

or

```
appcmd add module /module.name:"CustModule"
```

Typically, a command that accepts a name-related parameter has it as its first parameter, allowing you to specify the name with or without the literal reference. When configuration features or properties have names in addition to aliases, you can specify either value as the identity. Although quotation marks around parameter values are

optional in most cases, you should use them in most instances. This will ensure that you include quotation marks when they are mandatory, such as when a name value contains spaces. For example, you can list properties related to the default website by running the following command:

```
appcmd list site "Default website"
```

But the same code, without the quotation marks but with the same syntax otherwise (that is, `appcmd list site Default website`), generates a syntax error.

With List commands, you can typically return an object set containing all related items simply by omitting the name. For example, if you type **appcmd list site** at the command prompt without specifying an identity, you get a list of all sites on the server.

## Configuring IIS using Management Objects

You can use AppCmd to manage the configuration of your IIS servers. Commands you run work with objects matching a specific set of criteria. The sections that follow provide an overview of the available commands with their most commonly used syntaxes.

# Configuration Management Commands

Several configuration management commands are provided. These commands, along with their syntaxes, follow:

- **AppCmd List Config** Lists configuration sections from the server level by default or at a specified configuration level.

```
appcmd list config ["ConfigPath"] [/section:SectionName]
[/parameter1:value1 ...]
```

- **AppCmd Set Config** Modifies a configuration section at the server level by default or at a specified configuration level.

```
appcmd set config ["ConfigPath"] /section:SectionName
[/parameter1:value1 ...]
```

- **AppCmd Search Config** Searches the configuration file(s) at or below the server level or at by default or below a specified level for definitions of the specified configuration settings.

```
appcmd search config ["ConfigPath"] [/section:SectionName]
[/parameter1:value1 ...]
```

- **AppCmd Lock Config** Locks the specified configuration section at the server level or at a specified level so it cannot be overridden at a lower level.

```
appcmd lock config ["ConfigPath"] /section:SectionName
[/parameter1:value1 ...]
```

- **AppCmd Unlock Config** Unlocks the specified configuration section at the server level by default or at a specified level so it can be overridden at a lower level.

```
appcmd unlock config ["ConfigPath"] /section:SectionName
[/parameter1:value1 ...]
```

- **AppCmd Clear Config** Clears and optionally deletes the specified configuration section at the server level by default or at a specified level.

```
appcmd clear config ["ConfigPath"] /section:SectionName
[/parameter1:value1 ...] [/delete]
```

- **AppCmd Reset Config** Resets the specified configuration section at the server level by default or at a specified level to its default configuration state.

```
appcmd reset config ["ConfigPath"] /section:SectionName
[/parameter1:value1 ...]
```

- **AppCmd Migrate Config** Migrates the configuration features of a legacy server so that the server can use new server features. You can optionally clear the

original configuration after migration and recurse through lower configuration levels to ensure that all lower levels are also migrated.

```
appcmd migrate config ["ConfigPath"] [/section:SectionName]  
[/clear] [/recurse]
```



# Using Module Management Commands

Several module management commands are provided. These commands, along with their syntaxes, follow:

- **AppCmd List Module** Returns a list of modules enabled for a specified application or having specific module attributes.

```
appcmd list module [/module.name:]"ModuleName"  
[/app.name:"AppPath"]
```

- **AppCmd Set Module** Sets the properties of a specified module.

```
appcmd set module [/module.name:]"ModuleName"  
[/app.name:"AppPath"] [/parameter1:value1 ...]
```

- **AppCmd Add Module** Enables a new managed module or an installed native module with the specified settings.

```
appcmd add module /name:"ModuleName" [/app.name:"AppPath"]  
[/parameter1:value1 ...]
```

- **AppCmd Delete Module** Disables a module by removing it from the enabled list.

```
appcmd delete module [/module.name:]"ModuleName"  
[/app.name:"AppPath"]
```

- **AppCmd Install Module** Installs a native module. By default, modules are also added to the enabled list.

```
appcmd install module /name:"ModuleName" /image:PathToDLL  
[/add:true|false]
```

- **AppCmd Uninstall Module** Uninstalls the specified native module. By default modules are also removed from the enabled list.

```
appcmd uninstall module [/module.name:]"ModuleName"  
[/remove:true|false]
```

# Site Management Commands

You can manage websites and their configurations by using the following commands and command-line syntaxes:

- **AppCmd List Site** Lists virtual sites on a server.

```
appcmd list site [/site.name:]SiteNameOrURL  
[/parameter1:value1 ...]
```

- **AppCmd Set Site** Configures a virtual site on a server.

```
appcmd set site [/site.name:]SiteNameOrURL  
[/parameter1:value1 ...]
```

- **AppCmd Add Site** Adds a new virtual site on a server.

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

**NOTE** Technically, bindings and physicalPath are optional, but a site won't work until you provide these parameters. Adding the physical path is what allows IIS to create the root virtual directory and root application for the site.

- **AppCmd Delete Site** Deletes a virtual site on a server.

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

- **AppCmd Start Site** Starts a virtual site on a server.

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

- **AppCmd Stop Site** Stops a virtual site on a server.

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

# Application Pool Management Commands

You can manage application pools and their configurations by using the following commands and command-line syntaxes:

- **AppCmd List Apppool** Lists the application pools on a server.

```
appcmd list apppool [[/apppool.name:]"AppPoolName"]  
[/parameter1:value1 ...]
```

- **AppCmd Set Apppool** Sets the properties of an application pool on a server.

```
appcmd set apppool [[/apppool.name:]"AppPoolName"  
[/managedRuntimeVersion:"Version"  
[/managedPipelineMode: Integrated|Classic]  
[/queueLength:"queueLength"] [/autoStart:true|false]  
[/managedRuntimeLoader "ManagedLoader"]  
[/CLRConfigFile "AppPoolConfigFile"]  
[/startMode "AlwaysRunning" | "OnDemand"]
```

- **AppCmd Add Apppool** Creates an application pool on a server.

```
appcmd add apppool /name:"AppPoolName"  
[/managedRuntimeVersion:"Version"  
[/managedPipelineMode: Integrated|Classic]  
[/queueLength:"queueLength"] [/autoStart:true|false]  
[/managedRuntimeLoader "ManagedLoader"]  
[/CLRConfigFile "AppPoolConfigFile"]  
[/startMode "AlwaysRunning" | "OnDemand"]
```

- **AppCmd Delete Apppool** Deletes an application pool from a server.

```
appcmd delete apppool [[/apppool.name:]"AppPoolName"]
```

- **AppCmd Start Apppool** Starts an application pool on a server.

```
appcmd start apppool [[/apppool.name:]"AppPoolName"] [/wait]  
[/timeout:WaitTimeMilliseconds]
```

- **AppCmd Stop Apppool** Stops an application pool on a server.

```
appcmd stop apppool [[/apppool.name:]"AppPoolName"] [/wait]  
[/timeout:WaitTimeMilliseconds]
```

- **AppCmd Recycle Apppool** Recycles the worker processes of an application pool on a server.

```
appcmd recycle apppool [[/apppool.name:]"AppPoolName"]  
[/parameter1:value1 ...]
```

# Application Management Commands

You can manage applications and their configurations by using the following commands and command-line syntaxes:

- **AppCmd List App** Lists the properties of all applications or a specific application on a server.

```
appcmd list app [/app.name:]AppNameOrURL  
[/site.name:"SiteName"] [/apppool.name:"AppPoolName"]  
[/path:"VirtualPath"] [/parameter1:value1 ...]
```

- **AppCmd Set App** Sets the properties of an application on a server.

```
appcmd set app [/app.name:]AppNameOrURL [/parameter1:value1 ...]
```

- **AppCmd Add App** Creates an application on a server.

```
appcmd add [/app.name:]app /site.name: "ParentSiteName"  
/path: "VirtualPath" /physicalPath: "Path"
```

<p><b>NOTE</b> Technically, physicalPath is optional, but an application won't work until you provide this parameter. Adding the physical path is what allows IIS to create the root virtual directory and map it to the virtual path you provide.</p>
--

- **AppCmd Delete App** Deletes an application on a server.

```
appcmd delete [/app.name:]app AppNameOrURL
```

# Virtual Directory Management Commands

You can manage virtual directories and their configurations by using the following commands and command-line syntaxes:

- **AppCmd List Vdir** Lists the virtual directories or properties of a specific virtual directory on a server.

```
appcmd list vdir [[/vdir.name:]"VdirNameOrUrl"]  
[/app.name:"ParentAppName"] [/path: "VirtualPath"]  
[/parameter1:value1 ...]
```

- **AppCmd Set Vdir** Sets the properties of a specific virtual directory on a server.

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

- **AppCmd Add Vdir** Creates a virtual directory on a server.

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

- **AppCmd Delete Vdir** Deletes a virtual directory on a server.

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

# Utility Commands

Several general-purpose utility commands are provided. These commands, along with their syntaxes, follow:

- **AppCmd List Wp** Lists the worker processes currently running on a server.

```
appcmd list wp [/process.name:]"ProcessID"  
[/wp.name: "ProcessID"] [/apppool.name: "AppPoolName"]
```

- **AppCmd List Request** Lists the requests currently executing on a server. Optionally finds requests that have been executing for longer than a specified time in milliseconds.

```
appcmd list request [/process.name:]"ProcessID"  
[/request.name: "ProcessID"] [/site.name:"SiteName"]  
[/wp.name:"WpName"] [/apppool.name:"AppPoolName"]  
[/elapsed:Milliseconds]
```

- **AppCmd List Backup** Lists the configuration backups or a specified configuration backup on a server.

```
appcmd list backup [/backup.name:]"BackupName"
```

- **AppCmd Add Backup** Creates a configuration backup on a server.

```
appcmd add backup [/name:"BackupName"]
```

- **AppCmd Delete Backup** Deletes a configuration backup on a server.

```
appcmd delete backup [/backup.name:]"BackupName"
```

- **AppCmd Restore Backup** Restores a configuration backup, overwriting the current system state. By default, AppCmd stops the server before performing the restore.

```
appcmd restore backup [/backup.name:]"BackupName"  
[/stop:true|false]
```

- **AppCmd List Trace** Lists the failed requests logs for a site, a worker process, or with the specified log attributes.

```
appcmd list trace [/trace.name:]"TraceName"  
[/site.name:"SiteName"] [/wp.name: "WorkerProcessName"]  
[/verb:Verb] [/statusCode:Status Code]
```

- **AppCmd Configure Trace** Configures failed request tracing for a server or site.

```
appcmd configure trace ["SiteName"] [/enablesite | /disablesite] [/enable | /disable] [/path:Path]  
[/areas:TraceProvider1/Area1, TraceProvider1/Area2,...]  
[/verbosity]
```

[/timeTaken: "**ExecuteTime**"] [/statuscodes: "**code1,code2,...**"]

- **AppCmd Inspect Trace** Displays trace events logged on a server.

appcmd inspect trace [/trace.name:] "**TraceName**"

[/event.name: "**EventName**"]

[/level: **VerbosityLevel**]

## Chapter 8. Digging into IIS Schema

You can use IIS to publish information on intranets, extranets, and the Internet. Because today's websites 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 is how IIS uses the configuration schema and its global configuration system.



## Working with the IIS Configuration System

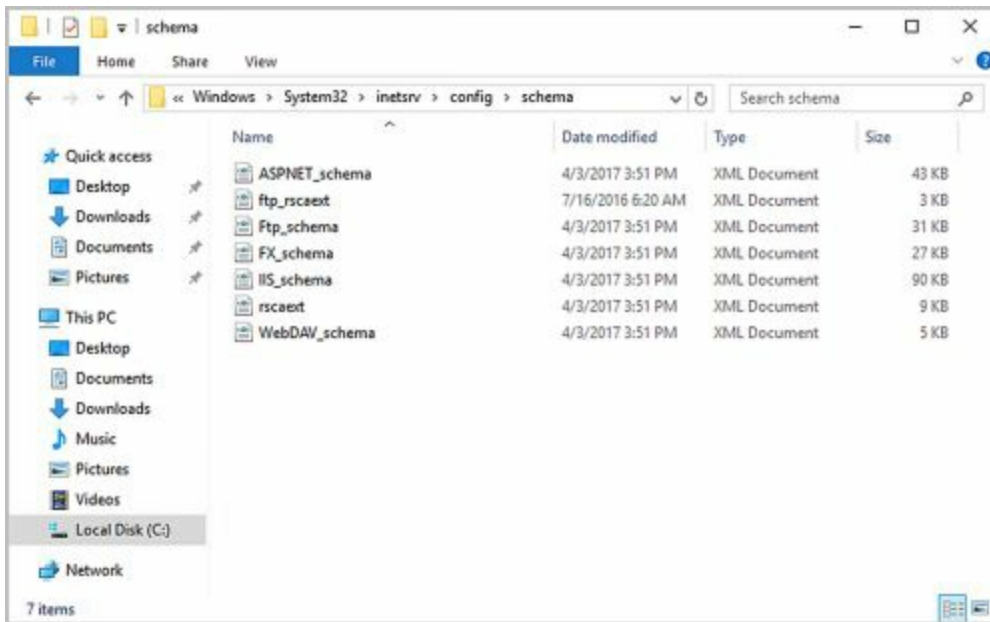
IIS 10 has a unified configuration system for storing server, site, and application settings. You can manage these settings by using an included set of managed code, APIs, and tools. You can also manage these settings by directly editing the configuration files themselves. Direct editing of configuration 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 10 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:

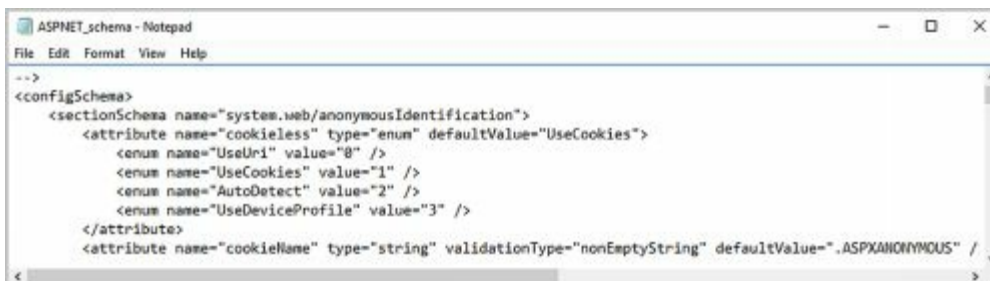
- **IIS\_schema.xml** This file provides the IIS configuration schema.
- **ASPNET\_schema.xml** This file provides the ASP.NET configuration schema.
- **FX\_schema.xml** This file provides the .NET Framework configuration schema (providing features beyond what the ASP.NET schema offers).
- **rscaext.xml** 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.



```
<configSchema>
  <sectionSchema name="appSettings">
    <attribute name="file" type="string" />
    <collection addElement="add" removeElement="remove" clearElement="clear">
      <attribute name="key" isUniqueKey="true" type="string" />
      <attribute name="value" type="string" />
    </collection>
  </sectionSchema>
  <sectionSchema name="configProtectedData">
    <attribute name="defaultProvider" type="string" defaultValue="RsaProtectedConfigurationProvider" />
    <element name="providers">

```

# Extending XML Schema

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 the 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>
```

# Navigating XML Schema

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 8-1 provides an overview of the validators you’ll see in schemas.

**TABLE 8-1** Summary of Attribute Validation Types in an IIS XML Schema

validationType= "applicationPoolName"	Set the value using validationParameter="Value". Validation fails if a validated value contains these characters:  <&"
validationType= "integerRange"	Set the value using validationParameter=" <minimum>, <maximum>[,exclude]". Validation fails if a validated value is outside [inside] range, in integers.
validationType= "nonEmptyString"	

	Set the value using validationParameter="Value". Validation fails if a validated value has a string value that is not set.
validationType="siteName"	Set the value using validationParameter="Value". Validation fails if a validated value contains these characters: ^.?.
validationType="timeSpanRange"	Set the value using validationParameter="<minimum>,<maximum>,<granularity> [,exclude]". Validation fails if a validated value is outside [inside] range, in seconds.
validationType="requireTrimmedString"	Set the value using validationParameter="Value". Validation fails if a validated value has white space at start or end of value.

## IIS Global Configuration

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.



# Configuration Settings

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 websites, and any subdirectories within websites.

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.

# Configuration Hierarchy

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 websites on the server and by all the content directories within those sites.

The order of inheritance is as follows:

.NET Framework root --> server root --> website 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 websites, and the settings for websites 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 "Managing Configuration Sections" in Chapter 9, "Managing Global IIS Configuration."

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 IIS 10 Web Applications, Security & Maintenance 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 9 for more information about configuring IIS servers.
- The default configuration file for a website root is Web.config. This file is stored in the root directory of the website to which it applies and controls the behavior for the website. See IIS 10 Web Applications, Security & Maintenance 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 11, “Configuring Directories for Websites,” for more information about configuring content directories.

## Including Configuration Files

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 />
  </system.webServer>
</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.

# Using Sections Groups

In earlier examples, you may have noted 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.

## Using Sections

When working with `ApplicationHost.config` or `Machine.config`, 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 9, you'll typically use location tags to lock or unlock sections for specific websites 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.



## Chapter 9. Managing Global IIS Configuration

Managing a server's global configuration is a key part of website management and optimization. Website properties identify the site, set its configuration values, and determine where and how documents are accessed. You can manage a server's global configuration at several levels:

- [As global defaults](#)
- [As site defaults](#)
- [As application or directory defaults](#)

You set global defaults at the Web server level, and all websites and applications on the server can inherit them. You set individual defaults at the website level, and they apply only to the selected website. You set application and directory defaults at the directory level, and they apply only to the selected application or directory. Unlike in Internet Information Services (IIS) 6.0, changes you make to the configuration are applied automatically and you do not need to restart servers, sites, or applications to apply configuration changes.

## Understanding Configuration Levels

You use global properties to set default property values for new websites and applications created on a server. Anytime you change global properties, existing websites and applications will also inherit the changes. In most cases, the new settings are applied automatically without having to restart server processes. In other cases, when a configuration is locked or restricted, the changes are inherited only if you unlock or unrestrict the configuration.

Table 9-1 provides a summary of all standard administrative features according to the level at which they can be configured. When you are configuring IIS features, it is important to keep in mind the inheritance hierarchy. As discussed in “Configuration Hierarchy” in Chapter 8, “Digging into IIS Schema,” settings you assign in a higher level of the hierarchy are inherited by the lower levels of the hierarchy. The server node represents the top of the hierarchy, followed by the site node, the top-level application/virtual directory nodes within a site, the virtual directory nodes within applications or other virtual directories, and so on.

**TABLE 9-1** IIS Features According to the Configuration Level

.NET Authorization Rules	Allows you to specify ASP.NET rules for authorizing users to access website and apps. Configure: Server, Site, Application.
.NET Compilation	Allows you to configure batch, behavior, and assembly properties for compiling managed code. Configure: Server, Site, Application.
.NET Error Pages	Allows you to configure HTTP error responses. Configure: Server, Site, Application.
.NET Globalization	Allows you to configure language and encoding properties for managed code. Configure: Server, Site, Application.
.NET Profile	Allows you to configure the information that will be stored on a per-user basis in a .NET Profile. Configure: Site, Application.
.NET Roles	Allows you to configure user groups for use with Membership Users and Forms authentication. Configure: Site, Application.
.NET Trust Levels	Allows you to set the trust level for managed modules, handlers, and applications. Configure: Server, Site, Application.
.NET Users	Allows you to configure users who belong to .NET Roles and who use Forms authentication. Configure: Site, Application.
Application Settings	Allows you to configure name and value pairs for managed code use at run time. Configure: Server, Site, Application.
ASP	Allows you to configure properties for ASP applications. Configure: Server, Site, Application.
Authentication	Allows you to view and manage authentication modes. Configure: Server, Site, Application.

Authorization Rules	Allows you to specify IIS rules for authorizing users to access applications. Configure: Server, Site, Application.
CGI	Allows you to configure properties for CGI programs. Configure: Server, Site, Application.
Compression	Allows you to configure and manage the way static compression and dynamic compression are used. Configure: Server, Site, Application.
Connection Strings	Allows you to configure strings that websites and applications can use to connect to data sources. Configure: Server, Site, Application.
Default Document	Allows you to configure default files to return when clients do not specify a file in a request. Configure: Server, Site, Application.
Directory Browsing	Allows you to configure information to display in a directory listing. Configure: Server, Site, Application.
Error Pages	Allows you to configure custom error pages to return when errors occur. Configure: Server, Site, Application.
Failed Request Tracing	Allows you to configure logging of failed request traces. Configure: Server, Site, Application.
Feature Delegation	Allows you to configure the default delegation state for features in IIS Manager. Configure: Server.
Handler Mappings	Allows you to configure resources that handle responses for specific request types. Configure: Server, Site, Application.
HTTP Redirect	Allows you to configure rules for redirecting incoming requests to another file or URL. Configure: Server, Site, Application.
HTTP Response Headers	Allows you to configure HTTP headers that are added to responses from the Web server. Configure: Server, Site, Application.
IIS Manager Permissions	Allows you to configure permissions for users who can manage websites and applications. Configure: Site, Application.
IIS Manager Users	Allows you to designate and manage website and web application administrators. Configure: Server.
IP and Domain Restrictions	Allows you to restrict or grant access to Web content based on IP addresses or domain names. Configure: Server, Site, Application.
ISAPI and CGI Restrictions	Allows you to restrict or enable specific ISAPI and CGI extensions on the Web server. Configure: Server.
ISAPI Filters	Allows you to configure ISAPI filters that modify IIS functionality. Configure: Server, Site.
Logging	Allows you to configure how IIS logs requests on the Web server. Configure: Server, Site, Application.
Machine Key	Allows you to configure encryption, validation, and decryption settings for managed application services. Configure: Server, Site, Application.

Management Service	Allows you to configure IIS Manager for delegated and remote administration. Configure: Server.
MIME Types	Allows you to configure extensions and associated content types that are served as static files. Configure: Server, Site, Application.
Modules	Allows you to configure native and managed modules that process requests on the Web server. Configure: Server, Site, Application.
Output Caching	Allows you to configure rules for caching served content in the output cache. Configure: Server, Site, Application.
Pages and Controls	Allows you to configure properties for pages and controls in Microsoft ASP.NET applications. Configure: Server, Site, Application.
Providers	Allows you to configure providers for provider-based application services, including those used with .NET Roles, .NET Users, and .NET Profiles. Configure: Server, Site, Application.
Request Filtering	Allows you to configure filtering rules for file name extensions, URLs, HTTP verbs, query strings and more. Configure: Server, Site, Application.
Server Certificates	Allows you to create and manage certificates for websites that use Secure Sockets Layer (SSL). Configure: Server.
Session State	Allows you to configure session state settings and Forms authentication cookie settings. Configure: Server, Site, Application.
Shared Configuration	Allows you to specify whether to use the IIS configuration of the local server or a remote location. Configure: Server.
SMTP E-mail	Allows you to configure e-mail address and delivery options to send e-mail messages from web applications. Configure: Server, Site, Application.
SSL Settings	Allows you to specify requirements for SSL and client certificates. Configure: Site, Application.
Worker Processes	Allows you to view information about worker processes and currently executing requests. Configure: Server.

In IIS Manager, you can access the global Web server configuration level by clicking the node for the computer you want to work with in the left pane and then double-clicking the configuration task you want to work with. Alternately, you can click the task to select it and then click Open Feature in the Actions pane.

In the IIS configuration files, the <configSections> element defines configuration sections by using <section> elements. A configuration section is the basic unit of deployment and locking for a server's configuration properties. You use the allowDefinition attribute of the related <section> element to specify the level or levels where the related properties of the section can be set.

Table 9-2 shows the acceptable values for the allowDefinition attribute. By assigning one of these values to a configuration section, you can specify the number or levels at

which the configuration can be controlled. MachineOnly is the default setting. You can use MachineOnly to specify that a configuration section can be managed using the Microsoft .NET Framework root and server root configuration files.

**TABLE 9-2** Attributes for Controlling Configuration Sections

Everywhere	Can be used at the .NET Framework, server, site, application and virtual directory levels.
MachineOnly	Can be used at the .NET Framework and server levels.
MachineToWebRoot	Can be used at the in .NET Framework, server, and site levels.
MachineToApplication	Can be used at the .NET Framework, server, site, and application levels.
AppHostOnly	Can only be used at the application level.

The OverrideModeDefault attribute controls whether a section is locked down to the level in which it is defined or can be overridden. You can set this attribute to one of two acceptable values—either Allow or Deny—or leave the attribute without a value. If you leave the attribute without a value or set it to Allow, lower-level configuration files can override the settings of the related section. Otherwise, overriding settings is not allowed. For example, you can use MachineOnly to specify that a configuration section can be managed using the .NET Framework root and server root configuration files. If you also set OverrideModeDefault to Allow, any settings configured at the .NET Framework level can be overridden by settings at the server root level. However, if you set OverrideModeDefault to Deny, settings configured at the .NET Framework level cannot be overridden by settings at the server root level.

You manage configuration locking by editing the configuration files or by using the IIS Command -line Administration Tool. In the configuration files, individual section elements are typed like this:

```
<section name="asp" overrideModeDefault="Deny" />
<section name="isapiCgiRestriction" allowDefinition="AppHostOnly"
overrideModeDefault="Deny" />
```

In this example, the asp section uses the default allowDefinition of MachineOnly, and isapiCgiRestriction uses the allowDefinition of AppHostOnly. To change the way these sections are used, you can directly edit the related attribute values, such as shown in the following example:

```
<section name="asp" allowDefinition="MachineToApplication"
overrideModeDefault="Allow" />
<section name="isapiCgiRestriction" allowDefinition="AppHostOnly"
overrideModeDefault="Allow" />
```

NOTE    Because IIS also allows you to use location locking, it is easy to

confuse global configuration locking and location locking. With global configuration locking, you specify the permitted levels at which configuration settings can be managed. With location locking, you lock or unlock a specific configuration section at a specific configuration level. You'll learn more about location locking in "Managing Configuration Sections," which is the next section of this chapter.

## Managing Configuration Sections

You can manage configuration sections and control the way they are used by IIS at any configuration level. This means that you can control the usage of configuration sections for an entire server, individual sites, individual applications, and individual virtual directories. Although you can manage locking by editing the configuration files, the easiest way to view and work with configuration sections is to use the IIS Command-line Administration Tool.

# Working with Configuration Sections

Each configuration section in the applicationHost.config file has an OverrideModeDefault attribute that controls whether a section is locked down to the level in which it is defined or can be overridden at lower levels of the configuration hierarchy. As discussed previously in the chapter in the section “Understanding Configuration Levels,” you can either allow or deny override. If you allow overriding the server level configuration, you can then use location locking to lock or unlock specific configuration sections at specific configuration levels. To understand this concept better, consider the following example:

You want to allow each site on a server to use a different set of default documents but do not want individual applications within sites to be able to use different sets of default documents. With this in mind, you allow the default document settings to be overridden in the applicationHost.config file as shown here:

```
<sectionGroup name="system.webServer">
  <section name="defaultDocument" overrideModeDefault="Allow" />
</sectionGroup>
```

You then lock the default document settings at the site level for each site on the server using location locking. The related entries in the Web.config file for each site are shown here:

```
<configuration>
<location path="" overrideMode="Deny">
  <system.webServer>
    <defaultDocument>
      <files />
    </defaultDocument>
  </system.webServer>
</location>
</configuration>
```

The default document settings are now locked at the site level. Because of this, you can manage the document settings at the site level but cannot manage the document settings for individual applications or virtual directories. In fact, if you access the Default Document feature for an application in IIS Manager, you will find that the Apply and Cancel actions are dimmed, so they cannot be selected. You also won't be able to configure the features by using AppCmd. In both instances, you should also see an error message stating that the configuration section cannot be used at this path because it is locked.

<p><b>NOTE</b> With configuration locking, it is important to remember that locking controls configuration through IIS Manager and AppCmd only. If someone has</p>
--



write access to the site-level directory on the server, he or she could edit the site's Web.config file and remove any restrictions you've enforced.



# Determining Settings for a Configuration Section

By using the IIS Command-line Administration Tool, you can determine the settings for a configuration section in several different ways. You can use the List Config command to determine the exact settings being applied or inherited for any configuration section at any level of the configuration hierarchy. Sample 9-1 provides the syntax and usage. The ConfigPath is the application path for the configuration level you want to examine. The ConfigPath for the default website is “Default website/”.

## SAMPLE 9-1 List Config Syntax and Usage

### Syntax

```
appcmd list config ["ConfigPath"] [/section:SectionName]
[/parameter1:value1 ...]
```

### Usage

```
appcmd list config "Default website/SalesApp"
```

```
appcmd list config /section:defaultDocument
```

### Example Output

```
c:\appcmd list config "Default website/SalesApp"
/section:defaultDocument
```

```
<system.webServer>
  <defaultDocument enabled="true">
    <files>
      <add value="Default.htm" />
      <add value="Default.asp" />
      <add value="index.htm" />
      <add value="index.html" />
      <add value="iisstart.htm" />
      <add value="default.aspx" />
    </files>
  </defaultDocument>
</system.webServer>
```

You can use the Search Config command to search the configuration files to determine exactly where unique settings are being applied on an IIS server. If you type **appcmd search config** without providing any additional parameters, you'll get a list of the server, site, and application paths where configuration files have been created. Other ways to use Search Config are to specify a starting configuration path from which to begin the search or to specify a configuration section to determine the locations where it is uniquely configured. You can also search for configuration sections by name and enabled or disabled state.

Sample 9-2 provides the syntax and usage for Search Config. Based on the example

output, the configuration section was configured in three locations: applicationHost.config, the default website's Web.config, and the Sales application/virtual directory's Web.config.

## SAMPLE 9-2 Search Config Syntax and Usage

### Syntax

```
appcmd search config ["ConfigPath"] [/section:SectionName]
[/parameter1:value1 ...]
```

### Usage

```
appcmd search config
```

```
appcmd search config "Default website/"
```

```
appcmd search config "Default website/" /section:defaultDocument
```

```
appcmd search config "Default website/" /section:defaultDocument
/enabled
```

```
appcmd search config "Default website/" /section:defaultDocument
/enabled:true
```

### Example Output

```
c:\appcmd search config
CONFIGSEARCH "MACHINE/WEBROOT/APPHOST"
CONFIGSEARCH "MACHINE/WEBROOT/APPHOST/Default website"
CONFIGSEARCH "MACHINE/WEBROOT/APPHOST/Default website/SalesApp" CONFIGSEARCH
"MACHINE/WEBROOT/APPHOST/Default website/Sales"
```

```
c:\appcmd search config /section:defaultDocument CONFIGSEARCH "MACHINE/WEBROOT/APPHOST"
CONFIGSEARCH "MACHINE/WEBROOT/APPHOST/Default website"
CONFIGSEARCH "MACHINE/WEBROOT/APPHOST/Default website/Sales"
```



# Modifying Settings for a Configuration Section

By using the IIS Command-line Administration Tool, you can run the Set Config command to modify the settings of a configuration section at the server level by default or at a specified configuration level. Sample 9-3 provides the syntax and usage for Set Config. The sample also provides examples for adding entries to a collection element. Here, entries are added to the files collection associated with the defaultDocument configuration section.

## SAMPLE 9-3 Set Config Syntax and Usage

### Syntax

```
appcmd set config ["ConfigPath"] /section:SectionName  
[/parameter1:value1 ...]
```

### Usage

```
appcmd set config /section:defaultDocument /enabled:true
```

```
appcmd set config "Default website/SalesApp"  
/section:defaultDocument /enabled:true
```

### Usage for Adding an Entry to a Named Collection

```
appcmd set config /section:defaultDocument  
/++files.[value="main.html"]
```

### Usage for Inserting an Entry at a Specific Location

```
appcmd set config /section:defaultDocument  
/++files.[@start,value='main.html']
```

```
appcmd set config /section:defaultDocument  
/++files.[@end,value='main.html']
```

```
appcmd set config /section:defaultDocument  
/++files.[@2,value='main.html']
```

### Usage for Removing an Entry from a Named Collection

```
appcmd set config /section:defaultDocument  
/-files.[value='main.html']
```

```
appcmd set config /section:defaultDocument /-files.[@2]
```

# Locking and Unlocking Configuration Sections

By using the IIS Command-line Administration Tool, you can run the Lock Config command to lock a configuration section. By default, the configuration is locked at the server level, but you can also specify a specific configuration level to lock. Locking the configuration at a specific level prevents the related settings from being overridden at a lower level of the configuration hierarchy.

Sample 9-4 provides the syntax and usage for Lock Config. When you use Lock Config, AppCmd creates the necessary location lock for you so that you don't have to type the related markup manually.

## SAMPLE 9-4 Lock Config Syntax and Usage

### Syntax

```
appcmd lock config ["ConfigPath"] /section:SectionName
```

### Usage

```
appcmd lock config "Default website/" /section:defaultDocument
```

By using the IIS Command-line Administration Tool, you can run the Unlock Config command to unlock a configuration section. By default, the configuration is unlocked at the server level, but you can also specify a specific configuration level to unlock. Unlocking the configuration at a specific level allows the related settings to be overridden at a lower level of the configuration hierarchy.

Sample 9-5 provides the syntax and usage for Unlock Config. When you use Unlock Config, AppCmd removes a previously set location lock so that you don't have to remove the related markup manually.

## SAMPLE 9-5 Unlock Config Syntax and Usage

### Syntax

```
appcmd unlock config ["ConfigPath"] /section:SectionName  
[/parameter1:value1 ...]
```

### Usage

```
appcmd unlock config "Default website/" /section:defaultDocument
```

# Clearing and Resetting Configuration Sections

By using the IIS Command-line Administration Tool, you can run the Clear Config command to clear and optionally delete a specified configuration section. By default, the configuration is cleared at the server level, but you can also specify a specific configuration level to clear. Clearing the configuration at a specific level allows inherited settings from a parent level to be used at that level and at lower levels of the configuration hierarchy. Sample 9-6 provides the syntax and usage for Clear Config.

## SAMPLE 9-6 Clear Config Syntax and Usage

### Syntax

```
appcmd clear config ["ConfigPath"] /section:SectionName [/delete]
```

### Usage

```
appcmd clear config "Default website/" /section:defaultDocument
```

```
appcmd clear config "Default website/" /section:defaultDocument  
/delete
```

By using the IIS Command-line Administration Tool, you can run the Reset Config command to reset a specified configuration section to its default configuration state. By default, the configuration is reset at the server level, but you can also specify a specific configuration level to reset. Sample 9-7 provides the syntax and usage for Reset Config.

## SAMPLE 9-7 Reset Config Syntax and Usage

### Syntax

```
appcmd reset config ["ConfigPath"] /section:SectionName
```

### Usage

```
appcmd reset config /section:defaultDocument
```



## Extending IIS with Modules

IIS supports native modules that use a Win32 DLL and managed modules that use a .NET Framework Class Library contained within an assembly. The configuration tasks available in IIS Manager depend on the role services and modules you've installed and enabled on your IIS server.

With IIS, installing and enabling modules are two separate processes. To use native modules, you must install and enable them. To use managed modules, however, you need only to enable them. As discussed in Chapter 3, "Setting Up IIS," you can add or remove role services, and when you do this, you install or uninstall related modules. Installing a module through a related role service registers the module so that it can be used with IIS. In many but not all cases, this also configures and enables the related IIS module automatically.

By adding or removing role services on a server, you make modules and their related DLLs available for use on a server. After you've added the appropriate role services to a server, you may also want to manage modules through the configuration files and the administration tools. The key reasons to do this are to:

- [Manage the level at which modules are available](#)
- [Manage module-specific properties](#)

In the sections that follow, I discuss how you can control modules and their related handlers through the configuration files. To ensure a better understanding of IIS configuration architecture, you should read and review this section even if you do not plan to edit the configuration files manually.

# Controlling Native Modules through the Configuration Files

You can manage native modules at the server, site, application, and virtual directory level. Because of inheritance, settings you assign at a higher level of the configuration hierarchy are inherited automatically by lower levels of the configuration hierarchy. In the <globalModules> section of the configuration files, native modules you've installed are identified by their name and DLL image, such as:

```
<globalModules>
<add name="DefaultDocumentModule"
image="%windir%\system32\inetsrv\defdoc.dll" />
<add name="DirectoryListingModule"
image="%windir%\system32\inetsrv\dirlist.dll" />
...
</globalModules>
```

In the <modules> section of the configuration files, native modules you've enabled are identified by their name, such as:

```
<modules>
<add name="DefaultDocumentModule" />
<add name="DirectoryListingModule" />
<add name="StaticFileModule" />
...
</modules>
```

You can uninstall native modules by removing the corresponding entry from the <globalModules> and <modules> sections of the appropriate configuration file. For example, if you remove the <globalModules> and <modules> entries for DirectoryListingModule from the applicationHost.config file, you uninstall this module for the server and all lower configuration levels.

Rather than uninstalling native modules, you may want to disable them at a specific level of the configuration hierarchy and then enable them only where they should be used. For example, if you want a module to be used only with designated applications, you can leave the corresponding entry in the <globalModules> section and remove the corresponding entry from the <modules> sections of the applicationHost.config file. Then in the Web.config files for the applications that should be able to use the module, you add an entry for the module in the <modules> section to allow the application to use the module.

# Controlling Managed Modules through the Configuration Files

Like native modules, you can control managed modules at the server, site, application, and virtual directory level. Because of inheritance, settings you assign at a higher level of the configuration hierarchy are inherited automatically by lower levels of the configuration hierarchy.

Unlike native modules, managed modules do not need to be installed before you can enable them for use. In the <modules> section of the configuration files, managed modules you've enabled are identified by their name and associated .NET type, such as:

```
<modules>
<add name="Profile" type="System.Web.Profile.ProfileModule"
preCondition="managedHandler" /> <add name="UrlMappingsModule"
type="System.Web.UrlMappingsModule" preCondition="managedHandler"
/>
</modules>
```

As the example also shows, all managed modules also have a precondition that stipulates that they must use a managed handler by default. Preconditions on managed modules provide conditional logic that controls the way Web content is handled. If you remove the managedHandler precondition from a managed module, IIS will also apply the module to content that is not served by managed handlers. For example, the Forms authentication module has a managedHandler precondition and is therefore called only when ASP.NET content, such as .aspx pages, are requested. If an .html page is requested, the Forms authentication is not called. However, if you want to protect all Web content with Forms authentication, you can do so by removing the managedHandler precondition from the Forms authentication module entry in the configuration files.

You can also enable all managed modules to run for all requests in your applications by setting runAllManagedModulesForAllRequests property in the <modules> section to true as shown in the following example:

```
<modules runAllManagedModulesForAllRequests="true" />
```

When you set the runAllManagedModulesForAllRequests property to true, the managedHandler precondition has no effect and IIS runs all managed modules for all requests.

You can disable managed modules by removing the corresponding entry from the <modules> section of the appropriate configuration file. For example, if you remove the <modules> entry for the OutputCache module from the applicationHost.config file, you disable this module for the server and all lower configuration levels. If you want a

managed module to be used only with designated applications, you can add an entry for the module in the <modules> section of the application's Web.config file.

# Controlling Managed Handlers through the Configuration Files

IIS processes all requests for Web content based on the type of content as determined by the file extension of the requested resource. Specifically, the file extension requested by a user tells IIS which handler to use to process the request. Each type of content has a specific handler that is identified in a handler mapping. This allows IIS to use handler mappings to automatically select the appropriate set of handlers for a particular type of file.

Using preconditions, IIS takes handler mapping a step further than do earlier versions of IIS. Handler preconditions ensure that IIS uses only handlers that are compatible with the runtime version of the .NET Framework and operating mode being used by the application pool processing a request. Handler mappings are configured at the server level during setup, and you'll find handler mappings in the <handlers> section of the applicationHost.config file. Here is an example:

```
<system.webServer>
<handlers accessPolicy="Read, Script">

<add name="AssemblyResourceLoader-Integrated-4.0" path="WebResource.axd" verb="GET,DEBUG"
type="System.Web.Handlers.AssemblyResourceLoader" precondition="integratedMode,runtimeVersionv4.0" />

<add name="PageHandlerFactory-Integrated-4.0" path="*.aspx" verb="GET,HEAD,POST,DEBUG"
type="System.Web.UI.PageHandlerFactory" precondition="integratedMode,runtimeVersionv4.0" />

<add name="SimpleHandlerFactory-Integrated-4.0" path="*.ashx" verb="GET,HEAD,POST,DEBUG"
type="System.Web.UI.SimpleHandlerFactory" precondition="integratedMode,runtimeVersionv4.0" />

</handlers>
</system.webServer>
```

The preconditions assigned to a handler control the way the handler works. The standard types of preconditions are as follows:

- **Mode** Applications running on a web server can use Classic mode or Integrated mode. With the Mode precondition, components that need to run in a particular operating mode can be marked to load only in worker processes that have this operating mode. This is important because setting the operating mode is an application pool property, and IIS worker processes use this property to determine how to process requests. Use classicMode to ensure that the handler is loaded only by the IIS core for application pools that are running in Classic mode. Use integratedMode to ensure that the handler is loaded only by the IIS core for application pools that are running in Integrated mode.

- **Runtime Version** IIS can support different versions of the .NET Framework side by side. However, currently only one version of the .NET Framework can be loaded in a worker process at a time. With the Runtime Version precondition, you can mark components so that they are used only when a worker process loads a particular version of the .NET Framework. This is important because setting the version of the .NET Framework is an application pool property and IIS worker processes use this property to preload the appropriate version of the .NET Framework on startup.
- **Bitness** 64-bit processors are the standard in server computing. To ensure compatibility with older applications, computers running 64-bit versions of Windows Server 2012 and Windows Server 2012 R2 provide a 32-bit execution environment on top of the 64-bit execution environment. IIS takes advantage of this and allows you to run 32-bit and 64-bit worker processes side by side. To ensure that IIS loads DLLs with the right bitness into a worker process, you must set the correct Bitness precondition. Use bitness32 for DLLs that are designed for 32-bit operating systems. Use bitness64 for DLLs that are designed for 64-bit operating systems.

Thus, if a handler has the following precondition assignment:

```
preCondition="classicMode, runtimeVersionv2.0, bitness32"
```

IIS uses the handler only when the application pool processing a request is using Classic operating mode, and a 32-bit execution environment.

In the applicationHost.config file, the <handlers> section also sets the global access policy for managed handlers. Access policy, set using the accessPolicy attribute of the <handlers> section, determines whether handlers that require Read, Script or Execute permission can run. Required access, set using the requireAccess attribute of the add element, specifies the type of access a handler requires.

With accessPolicy, the type of access can be set as:

- **Read** Allows handlers that require Read access to run. With a Read access policy, IIS processes files as static content, which does not allow IIS to process any scripts contained in files.
- **Script** Allows handlers that require Script access to run. With a Script access policy, IIS processes files as dynamic content, which allows IIS to process any scripts contained in files.
- **Execute** Allows handlers that require Execute access to run. With an Execute access policy, IIS allows files being requested to be directly executed, such as would be necessary to execute unmapped ISAPI extensions.

The following example shows how `accessPolicy` and `requireAccess` are used in configuration files:

```
<system.webServer>
<handlers accessPolicy="Read, Script">
  <add name="ISAPI-dll" path="*.dll" verb="*" modules="IsapiModule"
resourceType="File" requireAccess="Execute" />
  . . .
</handlers>
</system.webServer>
```

This means that only handlers that require Read or Script access will execute. IIS will not run any handler that requires Execute access. If IIS receives a request for an unmapped ISAPI extension, IIS will generate a runtime error. If you wanted to allow Execute access, you would need to change the access policy as shown in this example:

```
<handlers accessPolicy="Read, Script, Execute">
  . . .
</handlers>
```

You should change access policy to allow execute permissions only if this is a requirement for your server environment. Otherwise, you'll want to set access policy so that only Script and Read permissions are allowed. If you wanted to prevent IIS from handling dynamic content, you could set Read access policy as shown in the following example:

```
<handlers accessPolicy="Read">
  . . .
</handlers>
```

This means that only handlers that require Read access will execute. IIS will not run any handler that requires Execute or Script access. If IIS receives a request for dynamic content, such as an .aspx page or an unmapped ISAPI extension, IIS will generate a runtime error.

# Using the Configuration and Schema Files to Install Non-Standard Extension Modules

As additional modules become available from Microsoft or other sources, you'll be able to download, install, and enable them to extend the functionality of your IIS servers. You should never install a new module without first thoroughly testing it in a development environment to determine the possible impact. Once you've rigorously tested the module and your servers to ensure that there are no adverse effects, you should also test various configurations and uninstall procedures to ensure that the module can be configured as expected and uninstalled cleanly.

Once they are approved by Microsoft, most modules are made available at the IIS website (<http://www.iis.net>). Once you've downloaded an extension module, you should be able to install and enable it easily. The general steps for installing and enabling a module will be similar to the following:

1. Unzip the compressed files in the download by right-clicking the Zip file and then selecting Extract All.
2. In the Extract Compressed (Zipped) Folders dialog box, click Extract to accept the default folder location for the files. Alternately, click Browse, and then use the Select A Destination dialog box to select a destination folder for the files you are extracting. Then click OK.
3. Review the module's ReadMe file or other documentation and then run the module's Setup program.

The module's Setup program should perform some or all of the following configuration tasks:

1. Stop IIS services, if necessary.
2. Copy any necessary schema files to the %SystemRoot%\System32\Inetsrv\Config\Schema directory. IIS reads in the schema files in this directory automatically during startup of the application pools.
3. Copy the module DLL to the %SystemRoot%\System32\Inetsrv directory. This allows IIS to execute the DLL.
4. Create the appropriate entries in the <moduleProviders> and <modules> sections of the administration schema file so that the module is available for management. The administration schema file (Administration.xml) controls the management interface available in IIS Manager and is located in the %SystemRoot%\System32\Inetsrv\Config\Schema directory.



5. Create the appropriate entries in the <globalModules> and <modules> sections of the applicationHost.config file so that the module is installed and enabled. The entry in the <globalModules> section maps to the DLL in the %SystemRoot\System32\Inetsrv directory.
6. If you later register programs for use with the module, you'll create handler mappings in the <handlers> section of the applicationHost.config file to allow the programs to be used for processing requests based on specific file extensions.
7. Start IIS services, if the services were previously stopped.

If the module you are installing doesn't have a setup program, you must perform similar procedures manually to ensure that the module is properly configured. You can learn more about an extension module by examining its schema file. The named attributes in the schema file represent properties that you can set to optimize the module behavior. Each attribute also should have details on the associated values for each property. You'll need to refer to the module documentation to determine how the module is used.

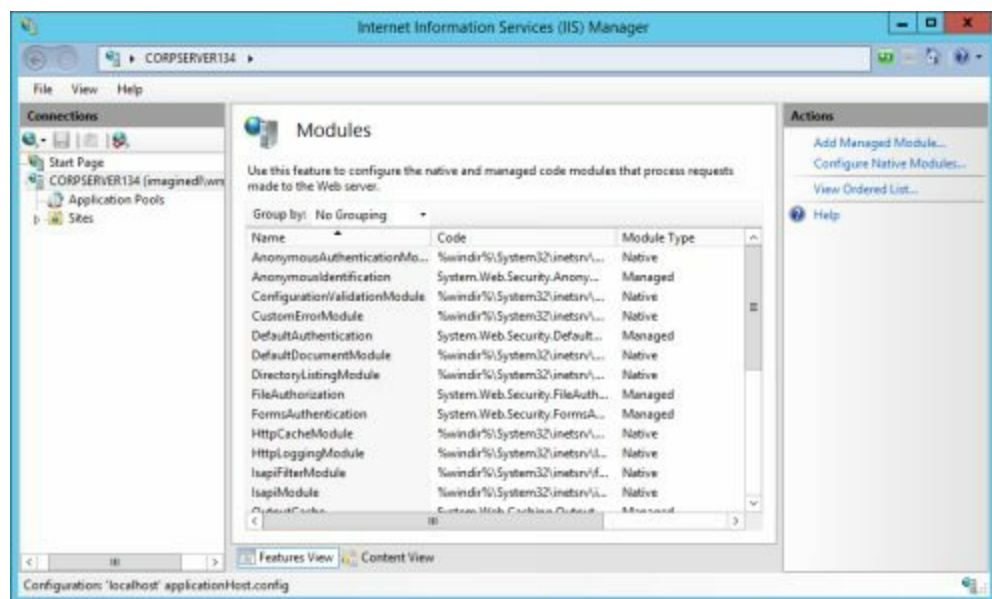
## Managing Modules

You can manage modules and control the way they are used by IIS at any configuration level. This means that you can configure module usage for an entire server, individual sites, individual applications, and individual virtual directories. The easiest way to view and work with IIS modules is to use IIS Manager or the IIS Command-line Administration Tool.

# Viewing Installed Native and Managed Modules

In IIS Manager, you can view the modules that are installed and enabled by completing the following steps:

1. Start IIS Manager. Navigate to the level of the configuration hierarchy you want to manage. To view global configuration details, select the computer name in the left pane. To view site or application details, expand nodes as necessary until you can select the site, application, or virtual directory you want to work with.
2. When you group by area, the Modules feature is listed under IIS. Double-click the Modules feature. This displays the Modules page



3. On the Modules page, you can use the Group By drop-down list to group module entries in several different ways. You can group by:
  - **Entry Type** Groups modules according to whether they have local definitions or inherited definitions. Modules that are enabled at that level are listed under Local. Modules that are enabled at a higher level are listed under Inherited.
  - **Module Type** Groups modules according to whether they are native or managed modules. Native modules are listed under Native Modules. Managed modules are listed under Managed Modules.

By using the IIS Command-line Administration Tool, you can view the local and inherited modules enabled at a specific configuration level by running the List Module command. Sample 9-8 provides the syntax and usage. You can set the /app.name parameter to the name of the application path to examine. If you use the default application path for a site, you can list the modules enabled for the site. If you do not set this parameter, AppCmd returns the server-level configuration details.

## SAMPLE 9-8 List Module Syntax and Usage

### Syntax

```
appcmd list module [[/module.name:]"ModuleName"]  
[/app.name:"AppPath"]
```

### Usage for Listing All Modules on a Server

```
appcmd list module
```

### Usage for Listing All Modules for Applications or Sites

```
appcmd list modules /app.name:"Default website/SalesApp"
```

### Usage for Listing Specific Modules for Applications or Sites

```
appcmd list module "FormsAuthentication" /app.name:"Default website/"
```

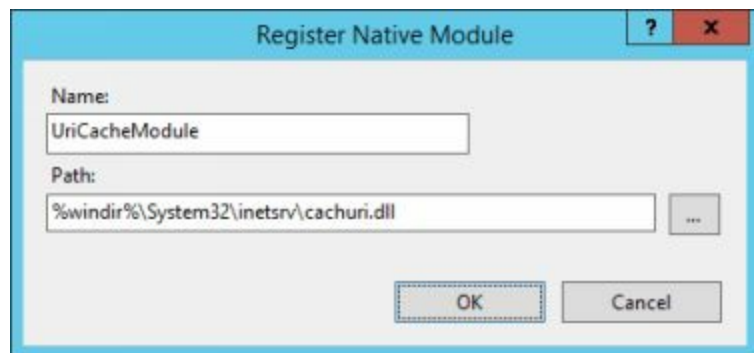
```
appcmd list modules /app.name:"Default website/"  
/type:System.Web.Security.FormsAuthenticationModule
```

```
appcmd list modules /app.name:"Default website/" /preCondition:managedHandler
```

# Installing Native Modules

You manage the installation of native modules at the server level. The best way to install a module for the first time is to add the related role service as discussed in Chapter 3 in the section “Viewing and Modifying Role Services.” Using the appropriate role service to install a module ensures that Setup inserts all necessary and related settings into the configuration files. If you subsequently uninstall a module by editing the configuration files or using IIS Manager, you can reinstall and enable a module at the server level by following these steps:

1. In IIS Manager, select the name of the server you want to work with in the left pane.
2. Access the Modules page by double-clicking the Modules feature. In the Actions pane, click Configure Native Modules.
3. In the Configure Native Modules dialog box, click Register. In the Register Native Module dialog box, register the module you want to install by typing the module name and module executable path as per the appendix, “Comprehensive IIS Module and Schema Reference.”



4. By default, modules you register are enabled automatically for use at the server level and below. If you don't want the module to be enabled in this way, you can disable the module as discussed in the section “Disabling Native and Managed Modules” later in this chapter. Click OK to complete the registration.

Using the IIS Command-line Administration Tool, you can install a native module by running the Install Module command. Sample 9-9 provides the syntax and usage. By default, modules are also added to the enabled list. If you don't want to enable the module you are installing, set the /add parameter to false.

## SAMPLE 9-9 Install Module Syntax and Usage

### Syntax

```
appcmd install module /name:"ModuleName" /image:PathToDLL  
[/add:true|false]
```

## Usage

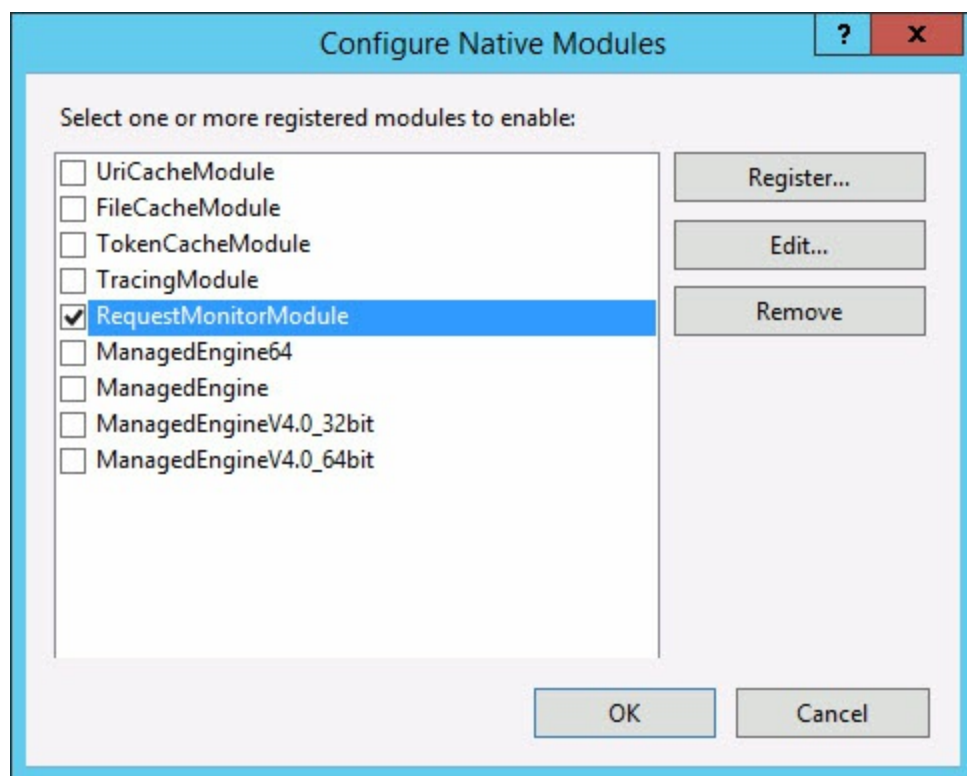
```
appcmd install module /name:"FastCGI"
```

```
/image:%windir%\System32\inetsrv\iisfcgi.dll
```

# Enabling Native Modules

When you install a native module, the module is registered and enabled automatically for use. The registration entry is created in the <globalModules> section of the applicationHost.config file, and the enablement entry is created in the <modules> section of the applicationHost.config file. If you disable a native module at the server level or another level of the configuration hierarchy, you can enable it as necessary by completing the following steps:

1. In IIS Manager, navigate to the level of the configuration hierarchy you want to manage.
2. Access the Modules page by double-clicking the Modules feature. In the Actions pane, click Configure Native Modules.
3. In the Configure Native Modules dialog box, you'll see a list of modules that are registered (installed) but disabled. Select one or more registered modules to install, and then click OK.



In the IIS Command-line Administration Tool, you can enable a native module by using the Add Module command. Sample 9-10 provides the syntax and usage. You can use the /app.name parameter to set the level at which the module should be enabled. If you do not set this parameter, AppCmd enables the module for use at the server level.

**SAMPLE 9-10** Add Module Syntax and Usage for Native Modules

## Syntax

```
appcmd add module /name:"ModuleName" [/app.name:"AppPath"]  
[/parameter1:value1 ...]
```

## Usage

```
appcmd add module /name:"WindowsAuthenticationModule"
```

```
appcmd add module /name:"WindowsAuthenticationModule"  
/app.name:"Default website/"
```

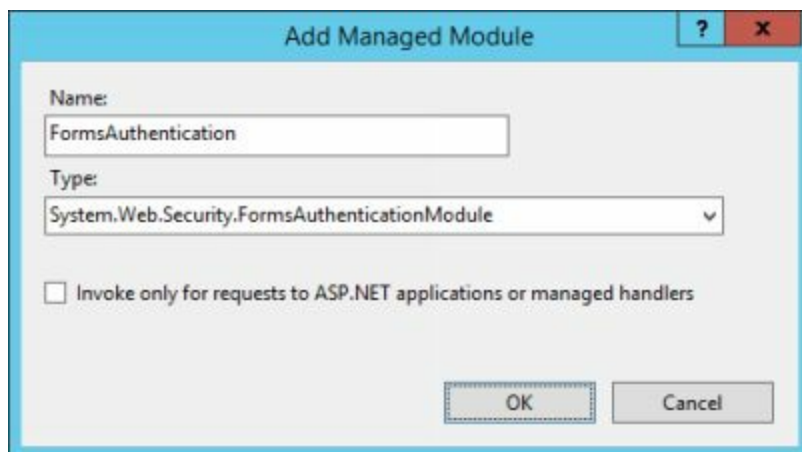


# Enabling Managed Modules

Although managed modules are installed automatically as part of the .NET Framework, they are not enabled for use automatically. This means that you must enable any managed modules that you want to make available. Further, you must install and activate the ManagedEngine module to provide the necessary integration functionality between IIS and the .NET Framework.

You can enable managed modules by completing the following steps:

1. In IIS Manager, navigate to the level of the configuration hierarchy you want to manage.
2. Access the Modules page by double-clicking the Modules feature. In the Actions pane, click Add Managed Module.
3. In the Add Managed Module dialog box, specify the managed module you want to enable by typing the module name and selecting the module's .NET library type as per the appendix.



4. Select the Invoke Only For Requests To... check box if you want the managed module to process only requests made to ASP.NET applications or managed handlers. If you do not select this check box, IIS Manager will run the module for all requests in your applications.

**NOTE** To add a module, the library type must be installed in the Global Assembly Cache (GAC). Selecting the Invoke Only For Requests To check box ensures that the module is added to the <modules> section and has the managedHandler precondition. See the section “Controlling Managed Modules through the Configuration Files” earlier in this chapter for more information.

In the IIS Command-line Administration Tool, you can enable a managed module by using the Add Module command. Sample 9-11 provides the syntax and usage. You can use the /app.name parameter to set the level at which the module should be enabled. If

you do not set this parameter, AppCmd enables the module for use at the server level.

## SAMPLE 9-11 Add Module Syntax and Usage for Managed Modules

### Syntax

```
appcmd add module /name:"ModuleName" /type:ManagedModuleType  
[/app.name:"AppPath"] [/precondition:managedHandler]
```

### Usage

```
appcmd add module /name:"CustModule" /type:CustNamespace.CustModuleClass /app.name:"Default website/"  
/precondition:managedHandler
```

# Editing Native and Managed Module Configurations

The properties you can configure for a module depend on its type. With native modules, you can configure the module name and image (executable path). With managed modules, you can configure the module name and .NET library type. With either type of module, you can also set preconditions, such as the need to use a managedHandler.

You can edit a module's configuration by completing the following steps:

1. In IIS Manager, navigate to the level of the configuration hierarchy you want to manage. Keep in mind that you can manage native module configurations only at the server level.
2. Access the Modules page by double-clicking the Modules feature and then double-clicking the module you want to modify.
3. With native modules, you'll see the Edit Native Module Registration dialog box, which allows you to set the module name and path (image). With managed modules, you'll see the Edit Managed Module dialog box, which allows you to set the module's name, type, and managed handler requirements. After you make the necessary changes, click OK to save your new settings.

**TIP** Managed modules can have different configurations at different levels of the configuration hierarchy. You can use this feature to remove managed handler requirements for a specific website or application. If you modify the configuration of a managed module at a lower level of the configuration hierarchy, you can click the module and then click Revert To Parent to restore the original configuration settings from the parent configuration level.

With the IIS Command-line Administration Tool, you can modify a module's configuration by using the Set Module command. Sample 9-12 provides the syntax and usage.

## SAMPLE 9-12 Set Module Syntax and Usage

### Syntax

```
appcmd set module [[/module.name:]"Module Name"]  
[/app.name:"AppPath"] [/parameter1:value1 ...]
```

### Usage

```
appcmd set module "ManagedEngine"  
/image: "%windir%\Microsoft.NET\Framework\v4.0.30319\webengine.dll"  
/preCondition:"preCondition="integratedMode, runtimeVersionv4.0,  
bitness32"
```

```
appcmd set module "FormsAuthentication" /app.name:"Default website/" /type:
```

"System.Web.Security.FormsAuthenticationModule"  
/preCondition:managedHandler

# Disabling Native and Managed Modules

You can disable a native or managed module at the server level or another level of the configuration hierarchy. When you disable a module at the server level, IIS removes the corresponding module entry from the <modules> section of the applicationHost.config file. When you disable a module at another level, IIS inserts a remove entry into the <modules> section of the related Web.config. In the following example, an administrator has removed HttpCacheModule from a site or application:

```
<modules>  
<remove name="HttpCacheModule">  
</modules>
```

Because disabling a module does not uninstall the module, a module disabled at one level is still available to be used at other levels of the configuration hierarchy. With this in mind, you can disable a module by completing the following steps:

1. In IIS Manager, navigate to the level of the configuration hierarchy you want to manage.
2. Access the Modules page by double-clicking the Modules feature.
3. Select the module you want to remove by clicking it, and then in the Actions pane, click Remove.
4. When prompted to confirm the action, click Yes.

With the IIS Command-line Administration Tool, you can disable a module by running the Delete Module command. Sample 9-13 provides the syntax and usage.

## SAMPLE 9-13 Delete Module Syntax and Usage

### Syntax

```
appcmd delete module [[/module.name:]"Module Name"]  
[/app.name:"App Path"]
```

### Usage

```
appcmd delete module "CustModule"
```

```
appcmd delete module "CustModule" /app.name:"Default website/SalesApp"
```

# Uninstalling Native Modules

You manage the installation of native modules at the server level. When you uninstall a native module, the module is deregistered and disabled. As a result, IIS removes the registration entry from the <globalModules> section of the applicationHost.config file. This prevents the module from being used at other levels of the configuration hierarchy.

Before you can use IIS Manager to uninstall a native module, you must first disable the module at the server level. You can then uninstall the native module by completing the following steps:

1. In IIS Manager, in the left pane, select the name of the server you want to work with.
2. Access the Modules page by double-clicking the Modules feature. In the Actions pane, click Configure Native Modules.
3. In the Configure Native Modules dialog box, select the module you want to uninstall, and then click Remove.
4. When prompted to confirm the action, click Yes.

By using the IIS Command-line Administration Tool, you can uninstall a module by running the Uninstall Module command. Sample 9-14 provides the syntax and usage. By default, the /remove parameter is set to true so that the module is also removed from the enabled list.

## SAMPLE 9-14 Uninstall Module Syntax and Usage

### Syntax

```
appcmd uninstall module [/module.name:]"ModuleName"  
[/remove:true|false]
```

### Usage

```
appcmd uninstall module /name:"CertificateMappingAuthenticationModule" /remove:true
```

## Sharing Global Configuration

A Web server farm is a group of IIS servers working together to provide common services. In large IIS installations like Web server farms, you'll often want all servers that provide the same services to share a common configuration. Don't worry—IIS makes it easy for Web servers to share a common configuration. All you need to do is point the servers to a shared configuration location and then copy the desired configuration to this location.

# Working with Shared Configurations

To facilitate a discussion on global configuration sharing, I'll refer to Web servers that share a global configuration as shared servers. The key to success with shared servers lies in careful management of the global configuration. Once sharing is set up, any configuration change you make to any one of the shared servers is applied to every other shared server. To avoid problems and the potential for multiple administrators to inadvertently change settings on different servers simultaneously, I recommend the following:

- Designating one administrator as the configuration administrator
- Using a nondedicated/dedicated configuration server

The configuration administrator is the central coordinator for changes to the shared server configuration. He or she is the only person who can authorize configuration changes. The fact that a prior request from this person is required to make configuration changes ensures that only one person at a time is making changes to the shared configuration.

With a nondedicated/dedicated configuration server, you make all configuration changes on a specific server. If this server is a part of the Web farm, it is a nondedicated configuration server, and any changes you make to the server are immediately applied to all servers in the Web farm. If this server is not a part of the Web farm and not used for other purposes, it is a dedicated configuration server and any changes you make to the server must be pushed out to the Web farm using the configuration export process. Because you can test changes prior to implementing them, there are definite advantages to using a dedicated configuration server.



# Exporting and Sharing Global Configuration

Once you've configured a Web server with the settings you want to use, you can export its configuration to a central configuration location, such as a NTFS shared folder. Providing that all your Web servers can access this location, you can then enable your servers to access this location to obtain their global configuration settings. To protect the configuration files from unauthorized viewing, you create encryption keys as part of the configuration process.

On the fully configured server for which you want to share global configuration, you can export the configuration by completing the following steps:

1. In IIS Manager, in the left pane, select the name of the server you want to work with. Access the Shared Configuration page by double-clicking the Shared Configuration feature. In the Actions pane, click Export Configuration.
2. In the Export Configuration dialog box, type the folder path to the save location for the configuration files. For shared NTFS folders, this should be in the form of a Universal Naming Convention (UNC) path name, such as \\FileServer23\WebConfig. If you want to select the path location rather than type it, click the options button and then use the Network node in the Browse For Folder dialog box to help you find the save location.
3. If you want to use alternate credentials or need additional permissions to access the save location, click Connect As. Type the user name and password and confirm the password of an account with appropriate permissions. Click OK.

<b>TIP</b> For the export to be successful, the account you use must have Change permissions on the NTFS share and Modify permissions for NTFS.
---

4. In the Export Configuration dialog box, type and then confirm a strong password for the encryption keys that will be used to secure the configuration files. A strong password is at least eight characters long and contains at least three of these four elements: numbers, symbols, uppercase letters, and lowercase letters.
5. Click OK to export the configuration. If the export is successful, you'll see a prompt stating this. Click OK. Otherwise, if the export fails, note the error provided and correct any problems, such as insufficient permissions, and then repeat this procedure.

On servers that you want to use the shared configuration, you can apply the shared configuration by completing the following steps:

1. In IIS Manager, in the left pane, select the name of the server you want to work with. Access the Shared Configuration page by double-clicking the Shared

Configuration feature. In the main pane, select the Enable Shared Configuration check box.

2. In the Physical Path text box, type the folder path to the shared configuration location. For shared NTFS folders, this should be in the form of a UNC path name, such as \\FileServer23\WebConfig. If you want to select the path location rather than type it, click the options button and then use the Network node in the Browse For Folder dialog box to help you find the save location.
3. Type the user name and password and confirm the password of an account with appropriate permissions to access the shared configuration location. The user name must be entered in DOMAIN\username format, such as IMAGINEDL\wrstaneK.
4. In the Actions pane, click Apply. When prompted, enter the encryption password and then click OK.
5. The computer's current IIS encryption keys are backed up and saved in the current configuration directory on the server. To restore these keys later, turn off shared configuration. When prompted about this, click OK again.
6. IIS Manager applies the shared global configuration. When prompted, confirm that the changes were successful before clicking OK. If IIS Manager was unable to apply the changes, ensure the NTFS and Share permissions on the shared location are set appropriately for the account you specified previously and that you entered the correct encryption password.
7. Exit and then restart all instances of IIS Manager. If remote administration is allowed, you must restart the Web Management Service as well. In IIS Manager's left pane, select the server you just configured. Access the Management Service page by double-clicking the Management Service feature. In the Actions pane, click Restart.

If you no longer want a server to use a shared configuration, follow these steps to disable shared configuration:

1. In IIS Manager, in the left pane, select the name of the server you want to work with. Access the Shared Configuration page by double-clicking the Shared Configuration feature.
2. In the main pane, clear the Enable Shared Configuration check box. In the Actions pane, click Apply. When prompted, do one of the following:
  - To restore the Web server's original configuration (that is, the configuration it was using prior to applying the shared configuration), click No. The server's original configuration is restored, along with its original encryption keys.
  - To continue to use the current configuration, as specified in the shared

configuration location, and copy this configuration over the original configuration, click Yes. The shared configuration is copied to the server, along with the shared encryption keys. Because sharing is disabled, any updates made to the shared configuration are not applied to the server.

3. When prompted, confirm that the changes were successful before clicking OK. If there are errors, they are likely due to NTFS and Share permissions on the shared location. Ensure permissions are set appropriately for the account you specified previously.

## Part 3. Creating Customized IIS Solutions

Chapter 10. Building Dynamic Websites

Chapter 11. Configuring Directories for Websites

Chapter 12. Customizing Web Server Content

Chapter 13. Web Stats 101

Chapter 14. Configuring Logging



## Chapter 10. Building Dynamic Websites

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

# Configuring 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

# Working with Private and 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.williamrstanek.com`, `www` identifies a server name and `williamrstanek.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.imaginedlands.com](http://www.imaginedlands.com), [services.imaginedlands.com](http://services.imaginedlands.com), and [products.imaginedlands.com](http://products.imaginedlands.com), respectively, in the DNS server, the server can respond to requests for each of these domain names.

# Understanding website Identifiers

Each website 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 website 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 website 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/

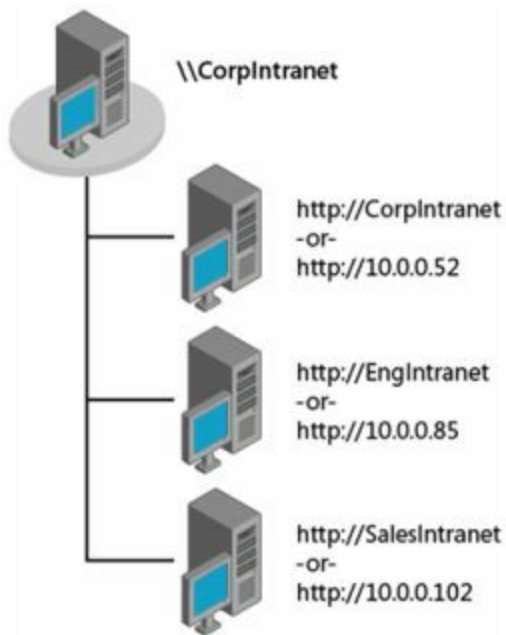
On a public network, a computer called Dingo could be registered to use the DNS name www.imaginedlands.com and the IP address of 207.46.230.210. If so, the website on the server could be accessed in either of the following ways:

- Using a URL: http://www.imaginedlands.com/ or http://207.46.230.210/
- Using a URL and port number: http:// www.imaginedlands.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.imagedlands.com`, `support.imagedlands.com`, and `service.imagedlands.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.



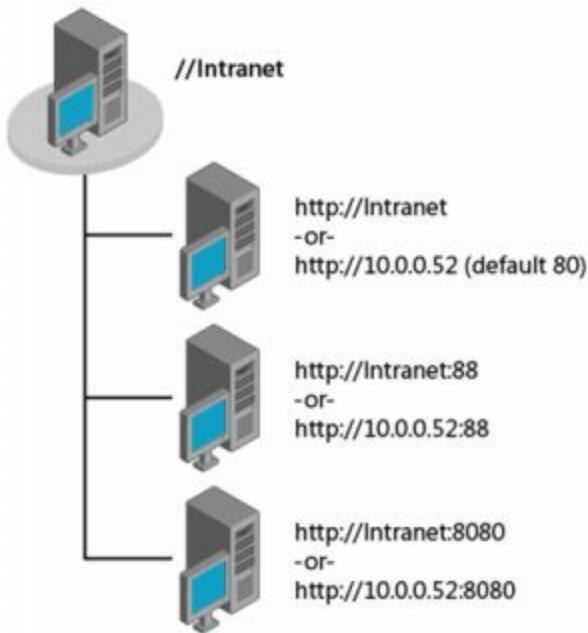
To use this technique, you must follow these steps:

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 website so that it uses a specific IP address.

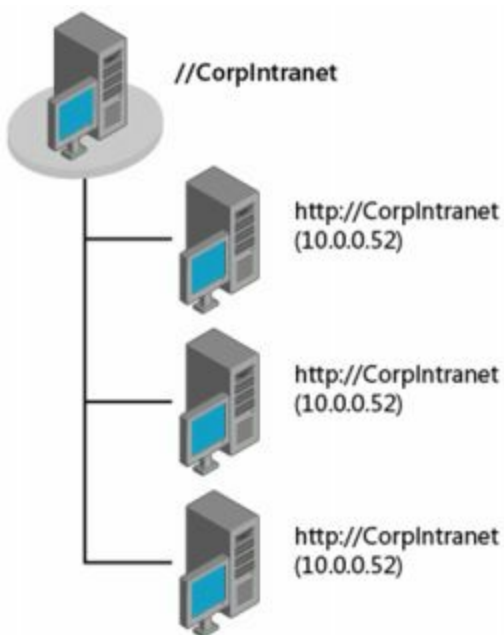
With this technique, users can access the sites individually by typing the unique domain name or IP address in a browser. Following the example shown previously, you can access the Sales intranet by typing **`http://SalesIntranet/`** or **`http://10.0.0.102/`**.

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. 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 websites 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.



With host header assignment, 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.

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 website so that it responds to requests on the IP address and port number you've assigned.
3. Configure additional websites 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.

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

1. Right-click in the lower left corner or press Windows key + X, and then 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.imaginedlands.com` and you've configured a website on port 80, the URL you use to access the computer from the Internet is `http://www.imaginedlands.com/`.

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

1. Right-click in the lower left corner or press Windows key + X, and then click Control Panel. In Control Panel's Category View, click Network and Internet, and then click Network And Sharing Center.
2. In the Network And Sharing Center, you'll see a list of tasks in the left pane. Click Change Adapter Settings. This opens the Network Connections window.
3. Right-click Ethernet, and then select Properties. This opens the Ethernet 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.

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.

## Internet Protocol Version 4 (TCP/IPv4) Properties



### General

You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.

☐ Obtain an IP address automatically

☒ Use the following IP address:

IP address: 192 . 168 . 10 . 72  
Subnet mask: 255 . 255 . 255 . 0  
Default gateway: 192 . 168 . 10 . 1

☐ Obtain DNS server address automatically

☒ Use the following DNS server addresses:

Preferred DNS server: 192 . 168 . 10 . 96  
Alternate DNS server: . . .

☐ Validate settings upon exit

Advanced...

OK

Cancel

# Examining Site Configuration

In IIS Manager, you can view a list of the websites 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, website 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 website"
```

```
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 website 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 website" /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 Websites

With IIS, you can create both unsecured and secured websites. 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 website, but IIS does not require this. IIS 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.

# Creating a Website: The Essentials

When you install IIS, the setup process creates a default website. In most cases, you aren't required to change any network options to allow users access to the default website. 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.imaginedlands.com` and the site is configured for access on port 80, a user can access the website by typing **`http://www.imaginedlands.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.imaginedlands.com` can have an alias as `www.imaginedlands.com`. If `zeta.imaginedlands.com` also hosts `service.imaginedlands.com` and `sales.imaginedlands.com`, you'd need CNAME records for these also.

On IIS, all websites 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 website that uses this application pool.

The directories and files for the default website are created under `%Windir%\Inetpub\Wwwroot`. To help organize additional websites 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 websites 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 website 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 Website

Users access unsecured websites by using HTTP. You can create a website that uses HTTP by completing the following steps:

1. If you're creating the website 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 website 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 website. This displays the Add website dialog box.
4. In the website Name text box, type a descriptive name for the website, 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 website. 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.

6. The Binding settings identify the website. To create an unsecured website, select HTTP as the type and then 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 websites 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 website 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.imaginedlands.com. The host header name must be unique within IIS.

9. By default, IIS starts the website immediately so long as the bindings you've supplied are unique. If you don't want to start the site immediately, clear the Start website 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 10-1 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 10-1 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.imaginedlands.com:80
```

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

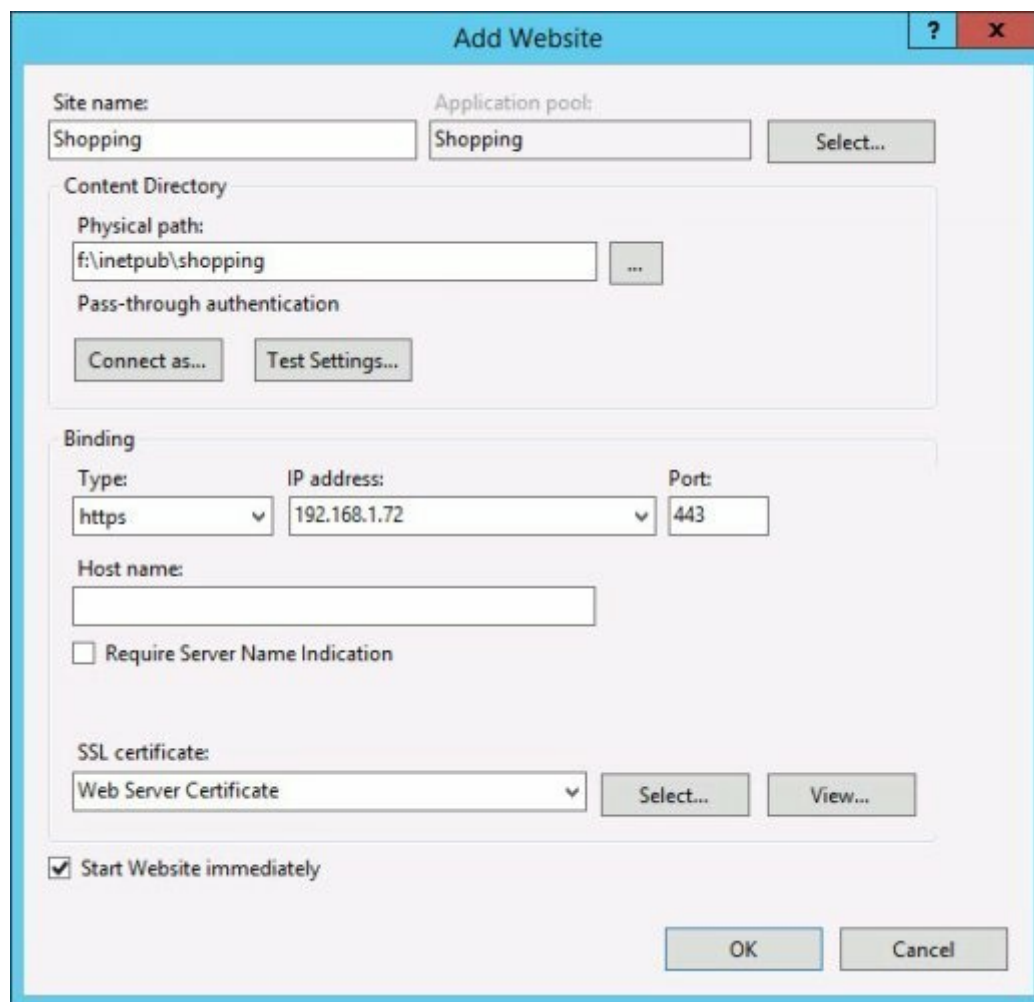
```
appcmd add site /name:'Sales Site' /id:5 /bindings:http://*:8080  
/physicalPath:'c:\inetpub\mynewsite'
```



# Creating a Secured Website

Users access secured websites by using SSL and HTTPS. Prior to creating a secured website, you must ensure that the certificate you want to use is available. You can create a website that uses HTTPS by completing the following steps:

1. Follow Steps 1–5 in the section “Creating an Unsecured website,” earlier in this chapter.
2. The Binding settings identify the website. To create a secured website, 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 websites can use the same IP address as long as the sites are configured to use different port numbers or host headers.



The screenshot shows the 'Add Website' dialog box with the following configuration:

- Site name:** Shopping
- Application pool:** Shopping
- Content Directory:**
  - Physical path:** f:\inetpub\shopping
  - Pass-through authentication:** Connect as... Test Settings...
- Binding:**
  - Type:** https
  - IP address:** 192.168.1.72
  - Port:** 443
  - Host name:** (empty field)
  - ☐ Require Server Name Indication
- SSL certificate:** Web Server Certificate
- ☒ Start Website immediately

Buttons: OK, Cancel

3. The TCP port for a secured website 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.
5. By default, IIS starts the website immediately as long as the bindings you've supplied are unique. If you don't want to start the site immediately, clear the Start website 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 10-2 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.

## SAMPLE 10-2 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.imaginedlands.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 Websites and Their Properties

The sections that follow examine key tasks for managing websites and their properties. You configure website 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.. You can use the options in the Actions pane as follows:

- **Explore** Opens the site's root directory in File 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 website** 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 website** 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 website.
- **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.

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 website 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 website, 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 10-3 and 10-4

provide the syntax and usage.

### SAMPLE 10-3 Start Site Syntax and Usage

#### Syntax

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

#### Usage

```
appcmd start site "Default website"
```

### SAMPLE 10-4 Stop Site Syntax and Usage

#### Syntax

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

#### Usage

```
appcmd stop site "Default website"
```

# Configuring an Application Pool and Home Directory

Each website 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.imaginedlands.com` and the home directory is `C:\Inetpub\Wwwroot`, browsers use the URL `http://www.imaginedlands.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 website dialog box.



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 website. 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 website. If necessary, you may want to create a new restricted account for this purpose.

7. Click OK to close the Edit website 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

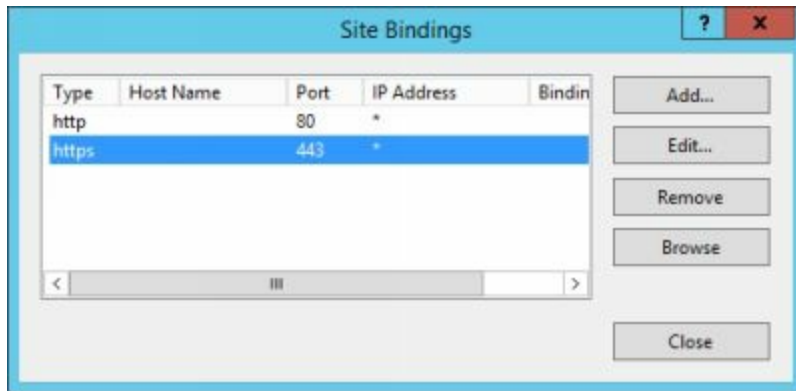
Throughout this chapter, I've discussed techniques you can use to configure multiple websites on a single server. The focus of the discussion has been on configuring unique identities for each site. In some instances, you might want a single website to have multiple domain names associated with it. A website 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 "Installing Application Servers" section of Chapter 3, "Setting Up IIS," for more information on non-HTTP process activation.

To change the binding of a website, complete the following steps:

1. If you want the website 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.



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 10-5 to 10-7 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 website 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 10-5 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 10-6 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.imaginedlands.com
```

## SAMPLE 10-7 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-Outs

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

- Set a limit on the amount of traffic allowed to a website 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, websites 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 website 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 website from overloading the resources of an entire server.

The connection time-out value determines when idle user sessions are disconnected. With the default website, 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 website Limits dialog box.

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.



The screenshot shows the 'Edit Website Limits' dialog box. It has a title bar with a question mark and a close button. The main area contains three sections: 'Limit bandwidth usage (in bytes):' with a checked checkbox and a text box containing '1875000'; 'Connection Limits' with a sub-section 'Connection time-out (in seconds):' containing a text box with '120'; and 'Limit number of connections:' with a checked checkbox and a text box containing '32768'. At the bottom are 'OK' and 'Cancel' buttons.

By using the IIS Command-line Administration Tool, you can run the Set Site command to set and remove limits for a site. Samples 10-8 and 10-9 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 10-8 Setting Site Limits Syntax and Usage

##### Syntax

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

##### 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 10-9 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

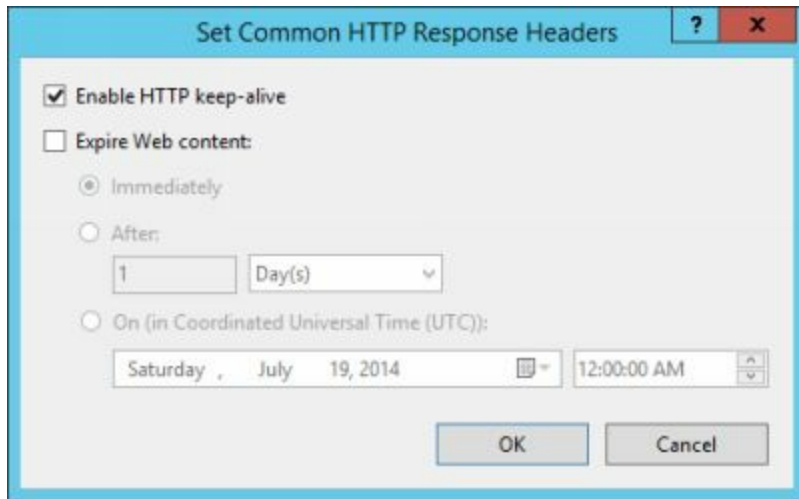
The original design of HTTP opened a new connection for every file retrieved from a Web server. Because a connection isn't maintained, no system resources are used after the transaction is completed. The drawback to this design is that when the same client requests additional data, the connection must 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, 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 website. 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 time-out for client sessions, or both. For more information on managing connections, see the "Restricting Incoming Connections and Setting Time-Outs" 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 Headers feature.
3. In the Actions Pane, click Set Common Headers. This displays the Set Common HTTP Response Headers dialog box.
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 10-10 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.

## SAMPLE 10-10 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'
```



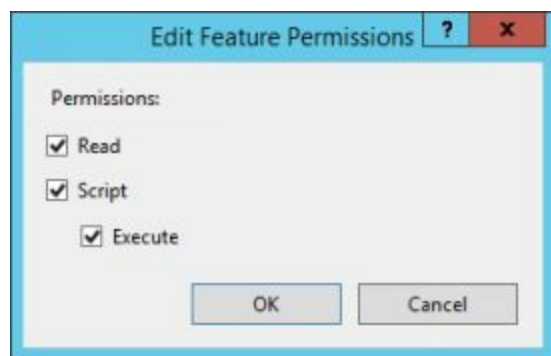
# Configuring Access Permissions in IIS Manager

In earlier releases of IIS, you configured access permissions for sites and virtual directories. In IIS, 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 9, "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, select or clear permissions as appropriate, and then click OK to apply the settings.



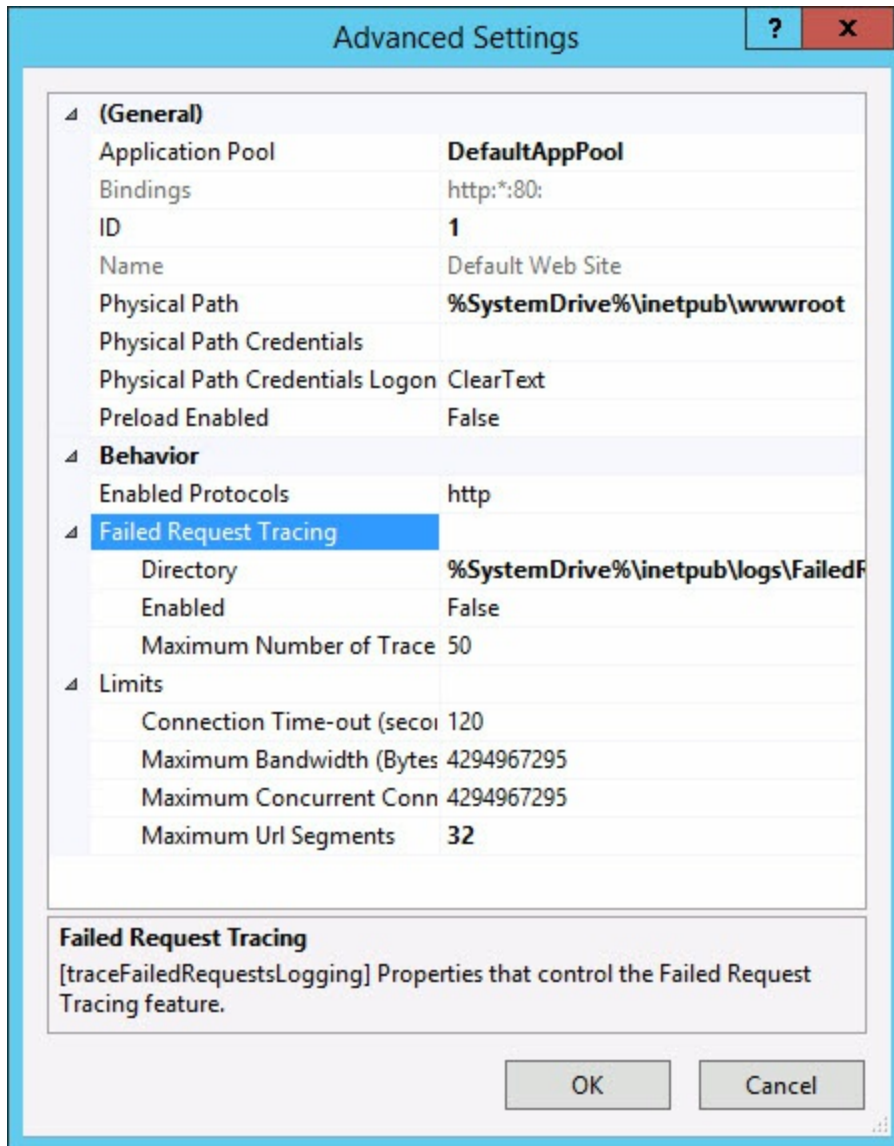
# Managing the Numeric Identifier and AutoStart State

Every website 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 website 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 website in the middle of the Actions pane. Clicking Advanced Settings displays 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 10-11 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 10-11 Set Site Syntax and Usage

## Syntax

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

## Usage

appcmd set site "Default website" /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 File 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 the 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 10-12 provides the syntax and usage.

## SAMPLE 10-12 Delete Site Syntax and Usage

### Syntax

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

### Usage

```
appcmd delete site "Default website"
```

## Chapter 11. Configuring Directories for Websites

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 websites.

## Working with Physical and Virtual Directories

In Chapter 10 “Building Dynamic Websites,” I discussed home directories and how they were used. Beyond home directories, Microsoft websites 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 File 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 File Explorer. You access these subdirectories by appending the directory name to the DNS name for the website. For example, you create a website with the DNS name `products.imagedlands.com`. Users are able to access the website by using the URL `http://www.imagedlands.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.imagedlands.com/search/`.

Even though locating your content files and directories within the home directory makes it easier to manage a website, 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.imagedlands.com` website could access the directory by using the URL path `http://www.imagedlands.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 website/"
```

```
appcmd list vdir http://localhost/Sales
```

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

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

```
VDIR "Default website/" (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 website/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 website/" /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 website/" /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 File Explorer, navigate to the home directory for the website.
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 website. For details on configuring permissions, see [IIS 10: Web Applications, Security & Maintenance](#).

**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, 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.



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.
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 and Logon Methods" 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 11-1 provides the syntax and usage. Remember that the physical directory you point to must already exist.

### SAMPLE 11-1 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 website/" /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 File 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

IIS 10.0 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.

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 11-2 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 11-2 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 File 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.

<p><b>CAUTION</b> 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 12, “Customizing Web Server Content.”</p>
---

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 and Logon Methods

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 11-3 provides the syntax and usage.

### SAMPLE 11-3 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 website/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 File 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 11-4 provides the syntax and usage.

## SAMPLE 11-4 Delete Vdir Syntax and Usage

### Syntax

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

### Usage

```
appcmd delete vdir "Default website/Support"
```

## Chapter 12. Customizing Web Server Content

Most Web administrators don't need to create Web server content. Typically, content creation is the job of Web designers, and content management is the job of Web administrators. Designers and administrators often work closely together to ensure that corporate sites, intranets, and extranets have the exact look and feel that management wants. A large part of this is customizing the way the Web server uses content. You might need to configure the server to redirect browser requests to other directories or websites. You might need to enable compression to improve performance or assign specific types of default documents to be used.

You can customize the content in many other ways, too. Rather than use generic error messages, you might want to create custom error messages that are specific to your company's Web pages. Custom error messages can contain menus, graphics, links, and text that help lost users find their way. If your organization uses unique types of content, you might need to configure servers to use additional content types. To help track advertising, you might want to create jump pages. To better manage outages, you might want to create an update site. These techniques and more are discussed in this chapter.

Don't worry. You don't have to master every technique in this chapter, but the more you know about customizing content and the options available, the better you'll be as an administrator. Before Internet Information Services (IIS) can serve static content, dynamic content, or both, you must enable the appropriate Common HTTP and Application Development role services.

## Managing Web Content

Every website on a server has a home directory. The home directory is the base directory for all documents that the site publishes. Copying files into the home directory, a virtual directory, or any subdirectory of these directories is, in fact, how you publish documents on a website.

Documents inherit the default properties of the site and the default permissions of the Windows folder in which they're placed. You can change these properties and permissions for each individual document or for all documents within a directory.

<p><b>CAUTION</b> Browsers can cache file and directory paths in bookmarks. To prevent errors when renaming or deleting files, you might want to redirect browser requests to the new location using the technique discussed in the “Redirecting Browser Requests” section later in this chapter.</p>
---

# Opening and Browsing Files

You can open Web content files by using either File Explorer or IIS Manager. You can open files in a browser by using File Explorer. To do this, right-click the file, and then on the shortcut menu, select Open. This opens the file by using a directory path, such as `D:\Inetpub\Wwwroot\Default.htm`.

You can display most types of files in the default browser by opening them in this way. However, if the file is an .asp document or other type of dynamic content and the website is running, the file won't be displayed correctly. You must be browsing the file through IIS to view it correctly in Microsoft Internet Explorer.

By using the Content View in IIS Manager, you can browse files through IIS. To do this, navigate to the website node, and then in the main pane, click Content View. Next, right-click the file you want to browse, and then on the shortcut menu, select Browse. This opens the file using a localhost path, such as `http://localhost/Default.htm`, rather than a directory path, and ensures that any type of file—static or dynamic—will appear correctly.

# Modifying the IIS Properties of Files

You can modify the settings for a Web file at any time. You set file permissions and general file properties in the file's Properties dialog box. In File Explorer, right-click the file, and then select Properties to display the Properties dialog box. In IIS Manager, navigate to the website node, and then in the main pane, click Content View. Next, right-click the file you want to work with, and then on the shortcut menu, select Edit Permissions.



# Renaming Files

To rename Web files in IIS Manager, follow these steps:

1. In IIS Manager, navigate to the website node, and then in the main pane, click Content View.
2. Right-click the file you want to work with, and then on the shortcut menu, select Edit Permissions. The file's Properties dialog box appears.
3. On the General tab, in the text box, type the new name for the file, and then click OK.

<p><b>NOTE</b> The name change isn't reflected immediately in IIS Manager. To update the file listings, click Refresh Page. This button is located in the upper right corner of the IIS Manager window.</p>
---

# Deleting Files

Web content files are stored under the root directory path for the website and in the root directory path of any virtual directories associated with the website. You can use File Explorer to easily delete any files that are no longer needed. When you delete a file, File Explorer moves the file to the Recycle Bin by default, and it is deleted permanently when you empty the Recycle Bin.

# Redirecting Browser Requests

Browser redirection is a useful technique to prevent errors when you rename or delete content within a website. When you redirect requests, you tell a browser to take the following actions:

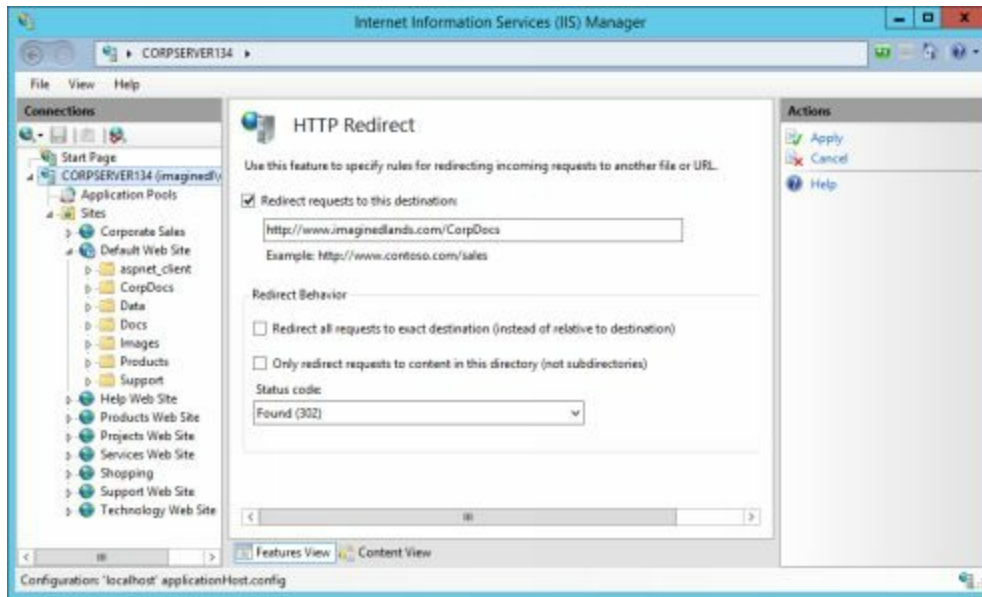
- Look for files in another directory
- Look for files on a different website
- Look for a specific file instead of a set of files
- Run an application instead of accessing the requested files

Each of these redirection techniques is examined in the sections that follow. Tips for creating customized redirection routines are examined in the “Customizing Browser Redirection” section later in this chapter. As discussed in Chapter 2, the HTTP Redirection role service controls the availability of this feature.

# Redirecting Requests to Other Directories or websites

If you rename or delete a directory, you can redirect requests for files in the old directory to the new directory, another directory, or even another website. When a browser requests the file at the original location, the Web server instructs the browser to request the page using the new location. You redirect requests to other directories or websites as follows:

1. In IIS Manager, navigate to the level of the configuration hierarchy you want to manage. You can manage redirection for an entire site at the site level. You can manage directory browsing for a specific directory at the directory level.
2. When you group by area, the HTTP Redirect feature is listed under IIS in the main pane. Select the HTTP Redirect feature, and then in the Actions pane, click Open Feature.
3. On the HTTP Redirect page, select Redirect Requests To This Destination.
4. In the Redirect Requests To This Destination field, type the Uniform Resource Locator (URL) of the destination website and directory. For example, to redirect all requests for <http://www.reagentpress.com/Docs> to <http://www.reagentpress.com/CorpDocs>, type **<http://www.reagentpress.com/CorpDocs>**. To redirect all requests for files located at <http://www.reagentpress.com/Docs> to [techsupport.reagentpress.com/CorpDocs](http://techsupport.reagentpress.com/CorpDocs), type **<http://techsupport.reagentpress.com/CorpDocs>**.
5. Click Apply. Now all requests for files in the old directory are mapped to files in the new directory. For example, if the browser requested <http://www.reagentpress.com/Docs/adminguide.htm>, and you redirected requests to <http://techsupport.reagentpress.com/CorpDocs/>, the browser would request <http://techsupport.reagentpress.com/CorpDocs/adminguide.htm>.



# Redirecting All Requests to Another website

If you stop publishing a website but don't want users to reach a dead end if they visit, you should redirect requests for the old website to a specific page at the new site. You redirect requests to a specific page at another site by completing the following steps:

1. In IIS Manager, navigate to the site you want to manage. In the main pane, the HTTP Redirect feature is listed under IIS when you group by area. Double-click HTTP Redirect to open this feature.
2. On the HTTP Redirect page, select Redirect Requests To This Destination.
3. In the Redirect Requests To This Destination field, type the complete URL path to the page at the new site, such as **<http://support.reagentpress.com/oldsite.html>**.
4. Under Redirect Behavior, select Redirect Requests To Exact Destination, and then click Apply. Now all requests for files at the old site are mapped to a specific page at the new site.

# Redirecting Requests to Applications

If your organization's development team has created a custom application for the website, you can redirect all requests for files in a particular directory (or for the entire site, for that matter) to an application. Parameters passed in the URL can also be passed to the application; the technique you use to do this is as follows:

1. In IIS Manager, navigate to the level of the configuration hierarchy you want to manage. You can manage redirection for an entire site at the site level. You can manage directory browsing for a specific directory at the directory level.
2. In the main pane, the HTTP Redirect feature is listed under IIS when you group by area. Double-click HTTP Redirect to open this feature.
3. On the HTTP Redirect page, select Redirect Requests To This Destination.
4. In the appropriate field, type the application's URL including any variables needed to pass parameters to the program, such as `http://Sales.reagentpress.com/CorpApps/Login.exe?URL=$V+PARAMS=$P`, where \$V and \$P are redirect variables. A complete list of redirect variables is provided in Table 12-1.
5. Under Redirect Behavior, select Redirect Requests To Exact Destination, and then click Apply. Now all requests for files in the directory or site are mapped to the application.

# Customizing Browser Redirection

The previous sections looked at basic redirection techniques. Now it's time to break out the power tools and customize the redirection process. You can customize redirection anytime you select Redirect Requests To This Destination, and choose to redirect a URL.

In all of the previous discussions, when you selected Redirect Requests To This Destination, additional settings appeared in Redirect Behavior section. Without selecting additional check boxes, all requests for files in the old location were mapped automatically to files in the new location. You can change this behavior by changing any of the following settings in the Redirect Behavior section:

- **Redirect All Requests To Exact Destination** Redirects requests to the destination URL without adding any other portions of the original URL. You can use this setting to redirect an entire site or directory to one file. For example, to redirect all requests for the `http://www.reagentpress.com/Downloads` directory to the `http://www.reagentpress.com/Download.htm` file, select this check box for the Downloads directory, and then in the Redirect Requests To This Destination field, type `http://www.reagentpress.com/Download.htm`.
- **Only Redirect Requests To Content In This Directory (Not Subdirectories)** Redirects files in a directory but does not affect files in subdirectories. For example, to redirect files in `http://www.reagentpress.com/products` but not in `http://www.reagentpress.com/products/current` or `http://www.reagentpress.com/products/upcoming`, select this check box, and then in the Redirect Requests To This Destination field, type `http://www.reagentpress.com/products`.
- **Status Code** Sets the HTTP status code for the redirection. Use Found (302) to indicate a standard redirection (HTTP status code 302). Use Temporary (307) to indicate a temporary redirection (HTTP status code 307). Use Permanent (301) to indicate a permanent redirection (HTTP status code 301). Without configuring this setting, redirections are considered non-permanent, and the client browser receives the Standard (302) redirect message. Some browsers can use the Permanent (301) redirect message as the signal to permanently change a URL stored in cache or in a bookmark.

You can also customize redirection by using redirect variables. As Table 12-1 shows, you can use redirect variables to pass portions of the original URL to a destination path or to prevent redirection of a specific file or subdirectory.

TABLE 12-1 Redirect Variable for IIS

--	--



\$S	Passes the matched suffix of the requested URL. The server automatically performs this suffix substitution; you use the \$S variable only in combination with other variables. If /Corpapps is redirected to /Apps, and the original request is for /Corpapps/Login.exe, / Login.exe is the suffix.
\$P	Passes the parameters in the original URL, omitting the question mark used to specify the beginning of a query string. If the original URL is /Scripts / Count.asp?valA=1&valB=2, the string “valA=1&valB=2” is mapped into the destination URL.
\$Q	Passes the full query string to the destination. If the original URL is /Scripts /Count.asp?valA=1&valB=2, the string “?valA=1&valB=2” is mapped into the destination URL.
\$V	Passes the requested path without the server name. If the original URL is //WebServer21 /Apps/Count.asp, the string “/Apps/Count.asp” is mapped into the destination URL.
\$0 through \$9	Passes the portion of the requested URL that matches the indicated wildcard character. If the original URL is //WebServer21/Apps/Data.htm, \$0 would be WebServer21, \$1 would be Apps, and \$2 would be Data.htm.
!	Use this variable to prevent redirecting a subdirectory or an individual file.

By using the IIS Command-line Administration Tool, you can manage redirection by running the Set Config command and the httpRedirection section of the configuration file. Sample 12-1 provides the syntax and usage. The default values for exactDestination and childOnly are false. The default value for httpResponseStatus is Standard.

## SAMPLE 12-1 Configuring Redirection Syntax and Usage

### Syntax

```
appcmd set config ["ConfigPath"] /section:httpRedirect
[/enabled: true|false] [/destination: "DestPath"]
[/exactDestination: true|false] [/childOnly: true|false]
[/httpResponseStatus="Permanent" | "Standard" | "Temporary"]
```

### Usage to Enable a Redirection Rule

```
appcmd set config "Default website/Sales/" /section:httpRedirect
/enabled:true /destination: "http://sales.imaginedlands.com/"
```

### Usage to Disable a Redirection Rule

```
appcmd set config "Default website/Sales/" /section:httpRedirect
/enabled:false
```

## Customizing Website Content and HTTP Headers

IIS sets default values for documents and Hypertext Transfer Protocol (HTTP) headers. You can modify these default values at the site, directory, and file level.

# Configuring Default Documents

Default document settings determine how IIS handles requests that don't specify a document name. If a user makes a request using a directory path that ends in a directory name or forward slash (/) rather than a file name, IIS uses the default document settings to determine how to handle the request. As discussed in Chapter 2, the Default Document role service controls the availability of this feature.

When default document handling is enabled, IIS searches for default documents in the order in which their names appear in the default document list and returns the first document it finds. If a match isn't found, IIS checks to see if directory browsing is enabled, and if so returns a directory listing. Otherwise, IIS returns a 404—File Not Found error.

You can configure default document settings at the server, site, or directory level. This means that individual sites and directories can have default document settings that are different from the server as a whole. Standard default document names include Default.htm, Default.asp, Index.htm, and Index.html. For optimal performance, you should:

- Limit the number of default documents to the essential few
- Order the default documents from the most frequently used to the least frequently used

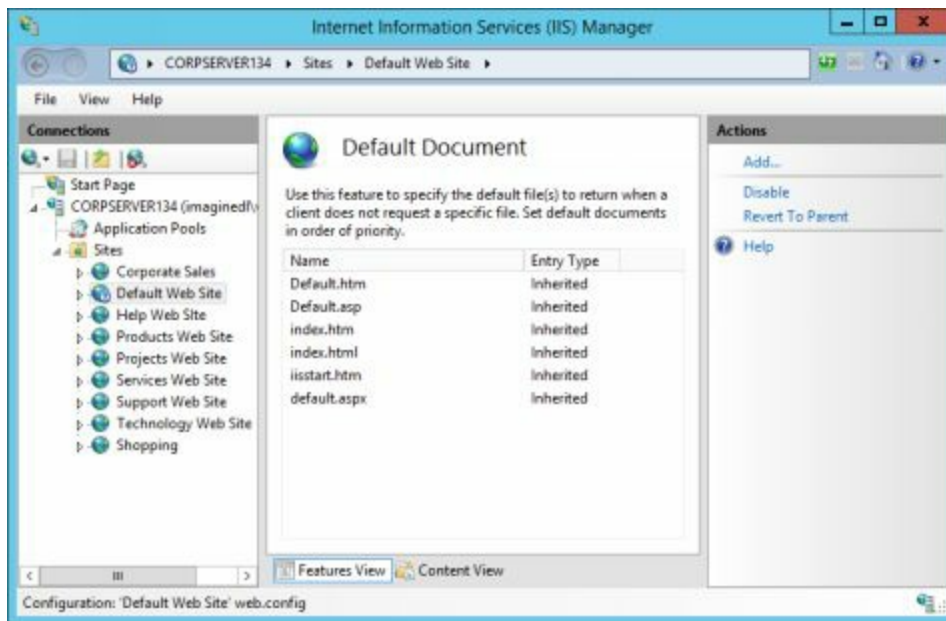
If you do not follow these basic guidelines, you could seriously degrade the performance of IIS.

You can view current default document settings or make changes by following these steps:

1. In IIS Manager, navigate to the server, site, or directory you want to manage. In the main pane, the Default Document feature is listed under IIS when you group by area. Double-click Default Document to open this feature.
2. The settings on the Actions pane determine whether default documents are used. If default document handling is turned off and you want to turn it on, click Enable. If default document handling is turned on and you want to turn it off, click Disable.
3. The current default documents are listed in order of priority. You can use the following techniques to manage default documents:
  - To change the priority order of a default document, select it and then click Move Up or Move Down in the Actions pane.
  - To add a new default document, click Add in the Actions pane, type the name of

the default document, such as **Index.html**, and then click OK.

- To remove a default document, click the default document that you want to remove and then click Remove in the Actions pane. When prompted to confirm, click Yes.



By using the IIS Command-line Administration Tool, you can manage default documents by running the Set Config command and the defaultDocument section of the configuration file. Sample 12-2 provides the syntax and usage for adding, changing, and removing HTTP headers. If any values you are setting include double quotation marks, you must escape the quotation character by enclosing it in double quotation marks. Additionally, because you are referencing into the files 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 files collection.

## SAMPLE 12-2 Configuring Default Documents Syntax and Usage

### Syntax

```
appcmd set config ["ConfigPath"] /section:defaultDocument  
[/enable:true|false] [/files.[value='Value']]
```

### Usage for Enabling Default Documents

```
appcmd set config "Default website" /section:defaultDocument  
/enabled:true
```

### Usage for Disabling Default Documents

```
appcmd set config "Default website" /section:defaultDocument  
/enabled:false
```

### Usage for Adding Default Documents

```
appcmd set config /section:defaultDocument
```

```
/+files.[value="start.htm"]
```

```
appcmd set config /section:defaultDocument  
/+files.[@start,value='start.htm']
```

```
appcmd set config /section:defaultDocument  
/+files.[@end,value='start.htm']
```

```
appcmd set config /section:defaultDocument  
/+files.[@2,value='start.htm']
```

## Usage for Removing Default Documents

```
appcmd set config /section:defaultDocument  
/-files.[value='start.htm']  
appcmd set config /section:defaultDocument /-files.[@2]
```

# Configuring Document Footers

You can configure IIS to automatically insert an HTML-formatted footer document on the bottom of every document it sends. The footer can contain copyright information, logos, or other important information. Although you can enable or disable at the site level, you must specify the default footer to use at the server level. This means that each IIS server can have a default footer that you can elect to use with individual sites hosted on the server.

To configure automatic footers, you need to create an HTML-formatted document and save it to a folder on a Web server's local hard disk drive. The footer document shouldn't be a complete HTML page. Instead, it should include only the HTML tags necessary for content that's to be displayed in the footer. Next, you need to use the IIS Command-line Administration Tool to specify the default document footer for the server. Afterward, you need to enable automatic footers for individual websites. Sample 12-3 provides examples for working with document footers.

## SAMPLE 12-3 Configuring Document Footers Syntax and Usage

### Syntax

```
appcmd set config ["ConfigPath"] /section:staticContent  
[/enableDocFooter:true|false] [/defaultDocFooter:'Value']
```

### Usage for Setting a Document Footer

```
appcmd set config /section:staticContent  
/defaultDocFooter:'footer.htm'
```

### Usage for Enabling Document Footers

```
appcmd set config "Default website" /section:staticContent  
/enableDocFooter:true
```

### Usage for Disabling Document Footers

```
appcmd set config "Default website" /section:staticContent  
/enableDocFooter:false
```

# Configuring Included Files

You can use Server-Side Includes (SSI) directives to insert just about any type of document into a Web content file. SSI is a feature that becomes available when you install the Server-Side Includes role service.

When you install and enable the Server-Side Includes role service, you can use included content with .asp, .aspx, .shtm, and .shtml files. IIS uses the #include directive to insert the contents of a file into a Web page. The #include directive is the only SSI directive that can be used with both .asp and .shtm files. Although you could update the handler mappings for SSI to include .htm and .html files, this could seriously degrade your server's performance. Why? If you enable SSI for .htm and .html files, IIS would need to parse all .htm and .html files to see if they have included content. This additional parsing operation can slow down the overall request handling process on the server.

Included files can have any file name extension that IIS can process. However, a recommended best practice is to use the .inc file extension. IIS processes included files through the interpreter of the original calling page. Thus, if you want to include a .shtm or .shtml page that includes other types of SSI directives, you must call it from a .shtm or .shtml page. If you want to include an .asp or .aspx page that includes dynamic content, you must call it from an .asp or .aspx page.

When you are editing the content for a Web file, the syntax for including files is as follows:

```
<!-- #include file ="FileToInclude" -->
```

where FileToInclude is the name of the file to include and optionally its relative path from the current directory. By default, you can include files in the same directory or in subdirectories only. In the following example, the included file is in the same directory as the calling file:

```
<!-- #include file ="footer.inc" -->
```

To reference a subdirectory of the current directory, you include the subdirectory name as shown in this example:

```
<!-- #include file ="data/footer.inc" -->
```

If you turn on the Enable Parent Paths parameter for the ASP feature, you can include files in parent directories as shown in this example:

```
<!-- #include file ="..\footer.inc" -->
```

Following this, you could insert a custom footer into a set of documents by completing

these steps:

1. Create the custom footer document.
2. Open a document in which you want to include the footer in Notepad or any other text editor.
3. Insert the appropriate include directive into the file.
4. Save the file, and then repeat this procedure for other documents that should include the footer.



# Using Content Expiration and Preventing Browser Caching

Most browsers store documents that users have viewed in cache so that the documents can be displayed later without having to retrieve the entire page from a Web server. You can control browser caching by using content expiration. When content expiration is enabled, IIS includes document expiration information when sending HTTP results to a user. This enables the browser to determine if future requests for the same document need to be retrieved from the server or whether a locally cached copy is still valid.

You can configure content expiration at the server, site, directory, or file level. Server level settings affect all sites on a server. Site level settings affect all pages in the site. Directory-level settings affect all files in the directory and subdirectories of the directory. File-level settings affect the currently selected file only. Three content expiration settings are available:

- **Expire Immediately** Forces cached pages to expire immediately, preventing the browser from displaying the file from cache. Use this setting when you need to ensure that the browser displays the most recent version of a dynamically generated page.
- **Expire After** Sets a specific number of minutes, hours, or days during which the file can be displayed from cache. Use this setting when you want to ensure that the browser will retrieve a file after a certain period.
- **Expire On** Sets a specific expiration date and time. The file can be displayed from cache until the expiration date. Use this setting for time-sensitive material that's no longer valid after a specific date, such as a special offer or event announcement.

<p><b>TIP</b> In ASP pages, you can control content expiration by putting a <code>Response.Expires</code> entry in the HTTP header. Use the value <code>Response.Expires = 0</code> to force immediate expiration. Keep in mind that HTTP headers must be sent to the browser before any page content is sent.</p>
--

## Enabling Content Expiration

You set content expiration at site, directory, and file levels. Keep in mind that individual file and directory settings override site settings. So if you don't get the behavior you expect, check for file or directory settings that might be causing a conflict.

You can configure content expiration for a server, site, directory, or file by completing the following steps:

1. In IIS Manager, navigate to the server, site, directory, or file you want to manage.

In the main pane, double-click HTTP Response Headers.

2. In the Actions pane, click Set Common Headers. This opens the Set Common HTTP Response Headers dialog box.
3. Select the Expire Web Content check box. Do one of the following and then click OK:
  - To force cached pages to expire immediately, select Immediately.
  - To set a specific number of minutes, hours, or days before expiration, select After, and then configure the expiration information in the appropriate fields.
  - To set specific expiration date and time, select On, and then configure the expiration information in the appropriate fields.

By using the IIS Command-line Administration Tool, you can enable content expiration by running the Set Config command and the staticContent section of the configuration file. Sample 12-4 provides the syntax and usage for configuring the various content expiration modes. When you are setting maximum age, you set the age in terms of the maximum number of days, hours, minutes, and seconds for which content is valid. When you are setting content expiration, you set the expiration date in terms of the day, date, and time at which content expires.

#### SAMPLE 12-4 Configuring Content Expiration Syntax and Usage

##### Syntax

```
appcmd set config ["ConfigPath"] /section:staticContent  
/clientCache.cacheControlMode:"NoControl" | "DisableCache" |  
"UseMaxAge" | "UseExpires"  
/clientCache.cacheControlMaxAge:"DD.HH:MM:SS"  
/clientCache.httpExpires:"Day, Date HH:MM:SS"
```

##### Usage for Configuring Immediate Expiration

```
appcmd set config "Default website" /section:staticContent  
/clientCache.cacheControlMode:"DisableCache"
```

##### Usage for Setting Content Expiration

```
appcmd set config "Default website" /section:staticContent  
/clientCache.cacheControlMode:"UseExpires"  
/clientCache.httpExpires:"Mon, 21 Dec 2018 00:00:00"
```

##### Usage for Setting Maximum Age

```
appcmd set config "Default website" /section:staticContent  
/clientCache.cacheControlMode:"UseMaxAge"  
/clientCache.cacheControlMaxAge:"14.00:00:00"
```

## Disabling Content Expiration

You set content expiration at site, directory, and file levels. Keep in mind that individual

file and directory settings override site settings. So if you don't get the behavior you expect, check for file or directory settings that might be causing a conflict.

You can disable content expiration for a server, site, directory, or file by completing the following steps:

1. In IIS Manager, navigate to the server, site, directory, or file you want to manage. In the main pane, double-click HTTP Response Headers.
2. In the Actions pane, click Set Common Headers. This opens the Set Common HTTP Response Headers dialog box.
3. Clear the Expire Web Content check box, and then click OK.

By using the IIS Command-line Administration Tool, you can disable content expiration by running the Set Config command and the staticContent section of the configuration file. Sample 12-5 provides the syntax and usage for disabling content expiration.

#### SAMPLE 12-5 Configuring Content Expiration Syntax and Usage

##### Syntax

```
appcmd set config ["ConfigPath"] /section:staticContent  
/clientCache.cacheControlMode:"NoControl"
```

##### Usage

```
appcmd set config "Default website" /section:staticContent  
/clientCache.cacheControlMode:"NoControl"
```

# Using Custom HTTP Headers

When a browser requests a document on a website handled by IIS, IIS normally passes the document with a response header prepended. Sometimes you might want to modify the standard header or create your own header for special situations. For example, you could take advantage of HTTP headers that are provided for by the HTTP standards but for which IIS provides no interface. Other times you might want to provide information to the client that you couldn't pass using standard HTML elements. To do this, you can use custom HTTP headers.

Custom HTTP headers contain information that you want to include in a document's response header. Entries in a custom header are entered as name value pairs. The Name portion of the entry identifies the value you're referencing. The Value portion of the entry identifies the actual content you're sending.

Custom HTTP headers typically provide instructions for handling the document or supplemental information. For example, the Cache-Control HTTP header field is used to control how proxy servers cache pages. A field value of Public tells the proxy server that caching is allowed. A field value of Private tells the proxy server that caching isn't allowed.

To view or manage custom HTTP headers for a server, site, directory, or file, follow these steps:

1. In IIS Manager, navigate to the server, site, directory, or file you want to manage. In the main pane, double-click HTTP Response Headers. The HTTP Response Headers pane shows currently configured headers in name: value format.
2. Use the following settings to manage existing headers or create new headers:
  - **Add** Adds a custom HTTP header. To add a header, click Add. Type a header name and a header value. Complete the process by clicking OK.
  - **Edit** Edits a custom HTTP header. To edit a header, select it, and then click Edit. In the Properties dialog box that appears, change the header information, and then click OK.
  - **Remove** Removes a custom HTTP header. To remove a header, select it, and then click Remove. When prompted to confirm the action, click Yes.

<p><b>NOTE</b> When you are working with existing HTTP response headers, be sure to note whether the entry type is listed as local or inherited. Local entries are configured at the level you are working with. Inherited entries are configured at a higher level of the configuration hierarchy. If you edit an inherited entry, you will make a local (not global) change to the entry.</p>
---

By using the IIS Command-line Administration Tool, you can manage HTTP headers by running the Set Config command. Samples 12-6 and 12-7 provide the syntax and usage for adding, changing, and removing HTTP headers. If any values you are setting include double quotation marks, you must escape the quotation character by enclosing it in double quotation marks. Additionally, because you are referencing into the customHeaders collection, the brackets ([]) in the syntax and usage examples are literal values rather than indicators of optional values.

## SAMPLE 12-6 Adding an HTTP Header Syntax and Usage

### Syntax

```
appcmd set config ["ConfigPath"] /section:httpProtocol  
/+customHeaders.[name='Name',value='Value']
```

### Usage

```
appcmd set config "Default website" /section:httpProtocol  
/+customHeaders.[name='P3P',value='policyRef=""  
http://www.imaginedlands.com/p3p.xml"""]
```

## SAMPLE 12-7 Removing an HTTP Header Syntax and Usage

### Syntax

```
appcmd set config ["ConfigPath"] /section:httpProtocol  
/-customHeaders.[name='Name',value='Value']
```

### Usage

```
appcmd set config "Default website" /section:httpProtocol  
/-customHeaders.[name='P3P',  
value='policyRef=""http://www.imaginedlands.com/p3p.xml"""]
```

# Using Content Ratings and Privacy Policies

Ratings agencies provide ratings directly to administrators for inclusion in websites when your organization joins or actively participates in a particular rating service. In addition to content ratings, websites also can have privacy policies. Browsers such as Internet Explorer rely on a website's compact privacy policy to determine how the site uses cookies. The World Wide Web Consortium (W3C) has published an official specification regarding Web privacy called the Platform for Privacy Preferences Project (P3P). P3P enables websites to report their privacy practices in a standard format that can be retrieved automatically and interpreted by user agents such as Web browsers. User agents rely on what is reported in the compact privacy policy and generally cannot determine whether cookies are used as reported. Because misuse of a privacy policy can get your organization into hot water, you should ensure that you answer the policy questions appropriately when you generate the policy reference file.

**REAL WORLD** Several times a year, you should review the way your organization uses cookies and update your privacy policy as appropriate. Helping your organization's Web designers understand privacy policy and the reporting requirements can go a long way to ensuring that your site remains in compliance with privacy policy statements. If you educate the Web design team about privacy policy, they can tell you when they've made changes to cookies that affect privacy policy. You then can be proactive rather than reactive in maintaining privacy policies on your organization's websites.

You can learn more about P3P online at <http://www.w3.org/P3P/>. You can obtain details on privacy policy generators online at <http://www.w3.org/P3P/implementations>. Once you've created a P3P reference file for a site, you can copy the file to a directory on the site and reference it in the site's HTTP Response headers by following these steps:

1. Using File Explorer, copy the P3P reference file to the site's root directory or an appropriate subdirectory.
2. In IIS Manager, navigate to the site you want to manage. In the main pane, double-click HTTP Response Headers. The HTTP Response Headers pane shows currently configured headers in name: value format.
3. To add a header to reference the privacy policy, click Add. This opens the Add Custom HTTP Response Header dialog box.
4. In the Name field, type **P3P**.
5. In the Value field, type **policyref="policyURL"** where policyURL is the actual URL to the P3P reference file. Then click OK.

Add Custom HTTP Response Header ? x

Name:  
P3P

Value:  
policyref="http://www.imaginedlands.com/p3p.xml"

OK Cancel

# Improving Performance with Compression

IIS fully supports the HTTP protocol and the compression enhancements it defines. By using HTTP compression, you can compress both static and dynamic results of HTTP queries for transmission to compliant clients.

Using IIS, you can enable and configure compression for both static and dynamic content. The Static Content Compression role service controls the availability of static content compression. The Dynamic Content Compression role service controls the availability of dynamic content compression.

IIS servers can get a big performance boost using static content compression. However, this isn't necessarily the case with dynamic content compression. Before you enable dynamic content compression, you need to look carefully at:

- [The size of the dynamic files](#)
- [The number of different dynamic files](#)
- [The way the dynamic files are being used](#)

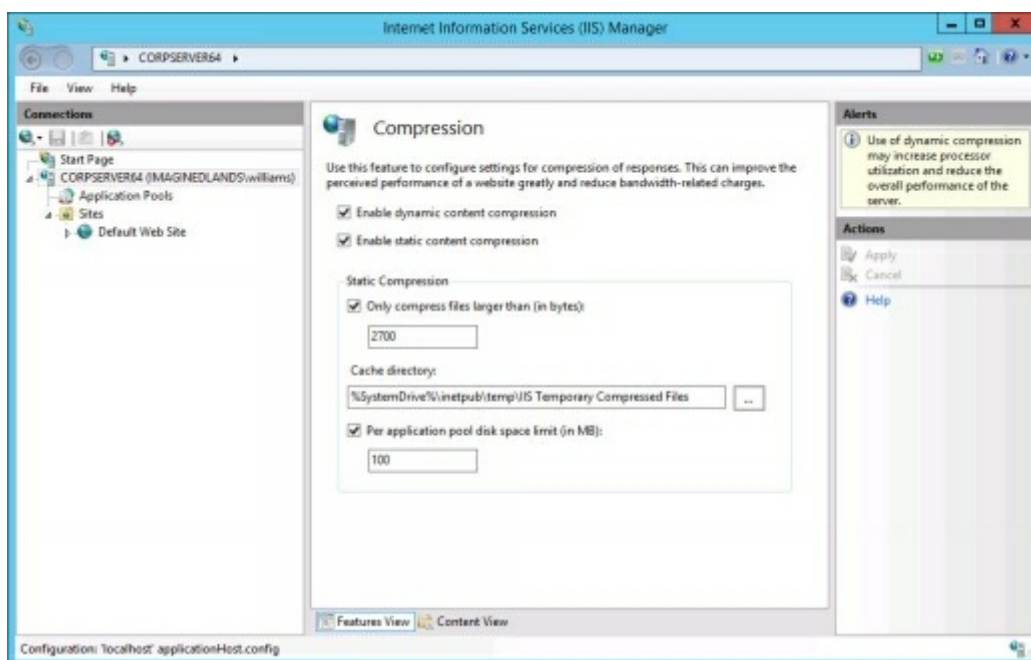
Once you have a firm understanding of how dynamic content is used on a site, you can optimize dynamic content caching to reduce the resource impact of dynamic content compression while reducing files sizes for faster transmission to client browsers. You optimize dynamic content caching by using ASP caching properties and output caching rules as discussed in IIS 10: Web Applications, Security & Maintenance.



# Configuring Content Compression for an Entire Server

You can configure content compression for an entire server and all of the related sites by following these steps:

1. In IIS Manager, select the server you want to manage. In the main pane, double-click Compression. The main pane shows the current state of compression for the server.
2. To enable or disable static content compression, select or clear the Enable Static Content Compression check box as appropriate.
3. To enable or disable dynamic content compression, select or clear the Enable Dynamic Content Compression check box as appropriate.



4. If you've enabled static content compression, use the following check boxes to optimize compression:
  - **Only Compress Files Larger Than (In Bytes)** Use the Only Compress Files Larger Than settings to set the minimum file size in bytes that you want to compress. By default, IIS compresses only files larger than 2700 bytes. If you want to compress all files regardless of size, clear the related check box. Otherwise, select the related check box and specify the minimum file size to compress. On a busy server with many dozens or hundreds of small files, you might want to reduce the number of small files that IIS caches and must retrieve from the cache directory. This will reduce the resource drain from having to compress and retrieve small compressed files while allowing the server to focus on compressing larger files from which the biggest user-perceived performance improvement can be achieved. For example, in this scenario, you might want to compress files only when they are

larger than 4096 bytes.

- **Cache Directory** Use the Cache Directory text box to specify the location where IIS stores static files after they are compressed. IIS stores static files until they expire or the content changes. The directory you use must be on an NTFS-formatted partition. The directory should not be compressed or encrypted.
- **Per Application Pool Disk Space Limit (In MB)** Use the Per Application Pool Disk Space Limit options to set the maximum amount of disk space in megabytes that you want IIS to use when compressing static content. By default, IIS stores up to 100 MB of compressed files for each application pool configured on the server. When the limit is reached, IIS automatically cleans up the temporary directory

5. Click Apply to save your settings.

**NOTE** You can set the dynamic compression buffer limit to control how much data IIS buffers before flushing the buffer to a client. The default is 65536 bytes.

By using the IIS Command-line Administration Tool, you can enable or disable content compression by running the Set Config command and the urlCompression section of the configuration file. Sample 12-8 provides the syntax and usage for enabling and disabling compression. The dynamicCompressionBeforeCache parameter controls whether IIS performs per-URL compression before caching the file. The default is false, which means that IIS caches a file (as appropriate per the current configuration) and then performs compression.

#### SAMPLE 12-8 Enabling or Disabling Content Compression Syntax and Usage

##### Syntax

```
appcmd set config ["ConfigPath"] /section:urlCompression  
[/doStaticCompression:true|false] [/doDynamicCompression:true|false]  
[/dynamicCompressionBeforeCache:true|false]
```

##### Usage

```
appcmd set config "Default website" /section:urlCompression  
/doStaticCompression:true
```

By using the IIS Command-line Administration Tool, you can configure content compression by running the Set Config command and the httpCompression section of the configuration file. Sample 12-9 provides the syntax and usage for configuring compression.

#### SAMPLE 12-9 Configuring Content Compression Syntax and Usage

##### Syntax

```
appcmd set config ["ConfigPath"] /section:httpCompression
```

[/cacheControlHeader:**Cache Time InSeconds**]  
[/doDiskSpaceLimiting:true|false]  
[/dynamicCompressionDisableCpuUsage:true|false]  
[/dynamicCompressionEnableCpuUsage:true|false]  
[/dynamicCompressionLevel:**Level**]  
[/maxDiskSpaceUsage:**MaxSize InMB**]  
[/minFileSizeForComp:**MinSize InBytes**]  
[/noCompressionForHttp10:true|false]  
[/noCompressionForProxies:true|false]  
[/noCompressionForRange:true|false]  
[/sendCacheHeaders:true|false]  
[/staticCompressionDisableCpuUsage:true|false]  
[/staticCompressionEnableCpuUsage:true|false]  
[/staticCompressionLevel:**Level**]

## Usage

apccmd set config "Default website" /section:httpCompression  
/doDiskSpaceLimiting:true

# Enabling or Disabling Content Compression for Sites and Directories

You can enable or disable static content compression for sites and directories by following these steps:

1. In IIS Manager, navigate to the site or directory you want to manage. In the main pane, double-click Compression. The main pane shows the current state of compression for the selected level.
2. To enable or disable static content compression, select or clear the Enable Static Content Compression check box as appropriate.
3. To enable or disable dynamic content compression, select or clear the Enable Dynamic Content Compression check box as appropriate.
4. Click Apply to save your settings.

# Customizing Web Server Error Messages

IIS generates HTTP error messages when Web server errors occur. These errors typically pertain to bad client requests, authentication problems, or internal server errors. As the administrator, you have complete control over how error messages are sent back to clients. When you add the HTTP Custom Errors role service to a server, you can configure IIS to send generic HTTP errors or default custom error files, or you can create your own custom error files.

# Understanding Status Codes and Error Messages

Status codes and error messages go hand in hand. Every time a user requests a file on a server, the server generates a status code. The status code indicates the status of the user’s request. If the request succeeds, the status code indicates this, and the requested file is returned to the browser. If the request fails, the status code indicates why, and the server generates an appropriate error message based on this error code. This error message is returned to the browser in place of the requested file.

A status code is a three-digit number that might include a numeric suffix. The first digit of the status code indicates the code’s class. The next two digits indicate the error category, and the suffix (if used) indicates the specific error that occurred. For example, the status code 403 indicates a forbidden-access problem, and within this access category a number of specific errors can occur: 403.1 indicates that execute access is denied, 403.2 indicates that read access is denied, and 403.3 indicates that write access is denied.

If you examine the Web server logs or receive an error code while trying to troubleshoot a problem, you’ll see status codes. Table 12-2 shows the general classes for status codes. As you can see from the table, the first digit of the status code provides the key indicator as to what has actually happened. Status codes beginning with 1, 2, or 3 are common and generally don’t indicate a problem. Status codes beginning with 4 or 5 indicate an error and a potential problem that you need to resolve.

**TABLE 12-2** General Classes of Status Codes

1XX	Continue/protocol change
2XX	Success
3XX	Redirection
4XX	Client error/failure
5XX	Server error

Knowing the general problem is helpful when you’re searching through log files or compiling statistics. When you’re troubleshooting or debugging, you need to know the exact error that occurred. For this reason, IIS provides detailed error information that includes the HTTP status code and substatus code.

**TIP** Because of security concerns about providing complete details on errors, the HTTP substatus code is no longer passed to clients (in most instances). Instead, clients should see a general status code, such as 401 or 402. If you’re trying to troubleshoot a problem, you might want to configure access logging so

that the substatus codes are recorded in the server logs temporarily. That way you can view the logs to get detailed information on any errors.

**NOTE** In some cases, Internet Explorer might replace custom errors with its own HTTP error message. Typically, this is done when the error message is considered too small to be useful to the user. Internet Explorer attempts to determine message usefulness based on message size. When 403, 405, or 410 error messages are smaller than 256 bytes, or when 400, 404, 406, 500, 500.12, 500.13, 500.15, or 501 error messages are smaller than 512 bytes, the custom error message sent by IIS is replaced by a message generated by Internet Explorer.

# Managing Custom Error Settings

When you add the HTTP Custom Errors role service to an IIS server, you can configure the way IIS handles error messages globally for the entire server and for individual sites and directories. By default, IIS displays detailed errors for local clients and terse errors for remote clients. IIS considers a local client to be any client with an IP address originating on the same network as the IIS server and a remote client to be a client with an IP address on any other network.

Detailed errors include the HTTP status code of the error in addition to the following information:

- **Description** A plain-language description of the error that occurred.
- **Error Code** An internal error code for IIS.
- **Module** The module in which the error occurred while processing the request. If the error occurred in the IIS server core, the module is listed as IIS Web Core.
- **Notification** The component that notified IIS about the error, such as the map request handler.
- **Requested URL** The URL requested including any related port used automatically, such as port 80 with HTTP requests and port 443 with HTTPS requests.
- **Physical Path** The local file path to which the request was mapped. If a user requests a file that does not exist, the file path will not be valid.
- **Logon User** The user account used to access the IIS server. With anonymous access to a server, the user is listed as Anonymous.
- **Logon Method** The authentication method used to log on to the IIS server. With anonymous access to a server, the logon method is listed as Anonymous.
- **Handler** The content handler that was processing the request when the error occurred, such as the StaticFile handler, which is used with standard HTML pages.
- **Most Likely Causes** A list of the most likely causes of the error.
- **Things You Can Try** A list of possible resolutions for the error.
- **Links And More Information** Provides additional information about the error in addition to a link to more information on the Microsoft website.

Terse errors, on the other hand, include only the HTTP status code of the error, the error description, and the server version information. The reason remote clients receive terse errors by default is to protect the integrity of your server and your network by ensuring that information that could be used maliciously is not passed to remote users.

Other security enhancements for IIS ensure that IIS generates HTTP error responses only for a specific, limited subset of the HTTP status codes. By default, the error responses



configured are: 401, 403, 404, 405, 406, 412, 500, 501, and 502. If IIS generated any related error codes or sub-error codes, IIS would provide an error response. These error responses are handled by files located in the %SystemRoot%\Inetpub\Custerr\<LANGUAGE-TAG> directory, where LANGUAGE-TAG is a placeholder for the default language of the client browser, such as en-us. If additional language packs are installed, this setting allows IIS to direct users to pages with the default language of their client browser. However, if no additional language packs are installed, IIS directs client browsers to pages with the default language of the server.

In the %SystemRoot%\Inetpub\Custerr\<LANGUAGE-TAG> directory, you'll find additional custom error files that you can configure for use in HTTP error responses. Because these files contain static content that is inserted into the error response, you can edit the files and optimize them for your environment. Further, because subcode errors are not configured by default, IIS returns general error responses, such as a 404 error rather than a 404.7 or 404.8 error. If you want IIS to be more specific about the exact error that occurred, you can add the appropriate custom error pages. However, don't do this without first considering the possible security implications of doing so. Always ask yourself if the additional information could be used maliciously, and err on the side of caution.

When you configure error responses, it doesn't have to be a choice between custom error pages and detailed errors. Instead of using either type of error response, you can configure a global default error page for an entire server, site, or directory. This error page can be a file with static content to insert into the error response, a URL to execute for dynamic content, or a redirection to a new URL.

You can tailor individual error responses in a similar way. This means that individual error responses can either be a file with static content to insert into the error response, a URL to execute for dynamic content, or a redirection to a new URL.

**REAL WORLD** When you use an .asp or .aspx file to handle custom errors, the error code and the original URL are passed to the dynamic page as query parameters. You must configure the dynamic page to read the parameters from the URL and set the status code appropriately. For example, if Notfound.asp is designed to handle 404 errors and the user accesses a page using the URL <http://www.reagentpress.com/data.htm>, the dynamic page is invoked using the URL <http://www.reagentpress.com/NotFound.asp?404;http://www.reagentpress.com/data.htm>, and your dynamic page must extract the 404 and <http://www.reagentpress.com/data.htm> parameters from the URL.

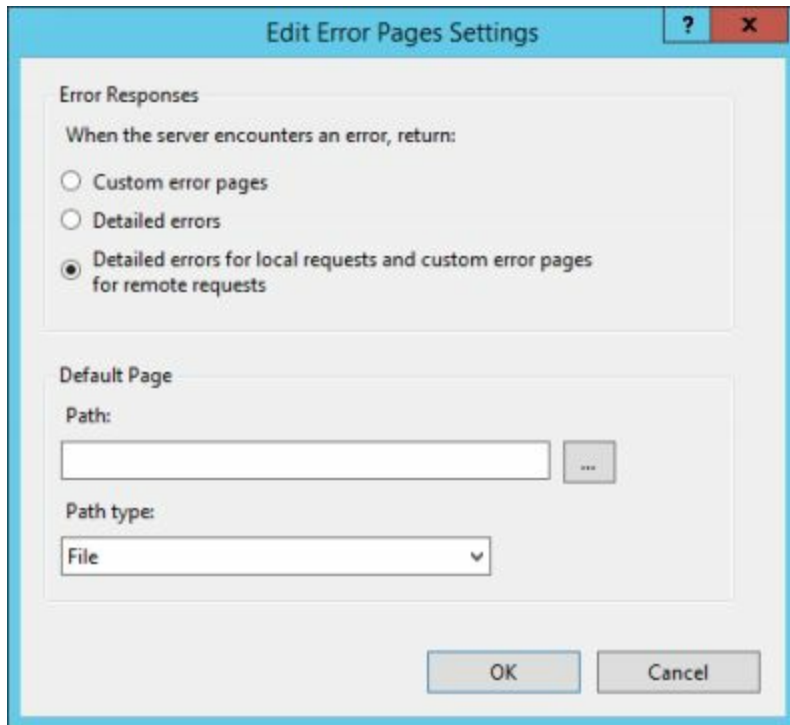
## Viewing and Configuring Custom Error Settings

You can view and configure custom error settings by performing these steps:

1. In IIS Manager, navigate to the server, site, or directory you want to manage. In the main pane, the Error Pages feature is listed under IIS when you group by area. Double-click Error Pages to open this feature.
2. You should now see a list of the configured HTTP error responses and how they're handled. Entries are organized by the following categories:
  - **Status Code** The HTTP status code for the error, which might include a suffix
  - **Path** The file path or URL path associated with the error response
  - **Type** The method used to handle the error (file, URL, or redirection)
  - **Entry Type** The type of entry as either local or inherited

<p><b>NOTE</b> With IIS, the <code>allowAbsolutePathsWhenDelegated</code> property controls whether absolute paths are allowed for custom error pages. By default, this property is set to false and only path that are relative to the site root are allowed.</p>
--

3. When the server encounters an error, IIS can return customer error pages, detailed error pages, or a combination of the two depending on where the client is located. To view and configure how error responses are configured, click Edit Feature Settings.
4. In the Edit Error Pages Settings dialog box, the options on the Error Responses panel control how the server handles error responses. By default, local clients see detailed error responses, and remote clients see custom error pages without additional details. To enhance security you might want all clients to see custom error pages without additional details; in this case, select Custom Error Pages. You'll rarely want to select Detailed Errors, because this setting provides detailed error responses to both local and remote clients.
5. Instead of providing clients with custom error responses, you can specify a default error page that IIS will display for all error responses. To set a default error page and override all other error response settings configured at this level, type the path to the default error page to use and specify the type of path as a file with static content to insert into the error response, a URL to execute for dynamic content, or a redirection to a new URL.
6. Click OK to save your settings.



By using the IIS Command-line Administration Tool, you can configure the error mode and default error page by running the Set Config command and the httpErrors section of the configuration file. Sample 12-10 provides the syntax and usage for configuring the error mode. Sample 12-11 provides the syntax and usage for configuring the default error page.

#### SAMPLE 12-10 Configuring the Error Mode Syntax and Usage

##### Syntax

```
appcmd set config ["ConfigPath"] /section:httpErrors  
/errorMode: "DetailedLocalOnly"|"Custom"|"Detailed"]
```

##### Usage

```
appcmd set config "Default website" /section:httpErrors  
/errorMode: "DetailedLocalOnly"
```

#### SAMPLE 12-11 Configuring a Default Error Page Syntax and Usage

##### Syntax

```
appcmd set config ["ConfigPath"] /section:httpErrors  
[/defaultResponseMode:"File"|"ExecuteURL"|"Redirect"]  
[/defaultPath:"Path"]
```

##### Usage

```
appcmd set config "Default website" /section:httpErrors  
/defaultResponseMode:"ExecuteURL"  
/defaultPath:"C:\inetpub\errors\error.aspx"
```

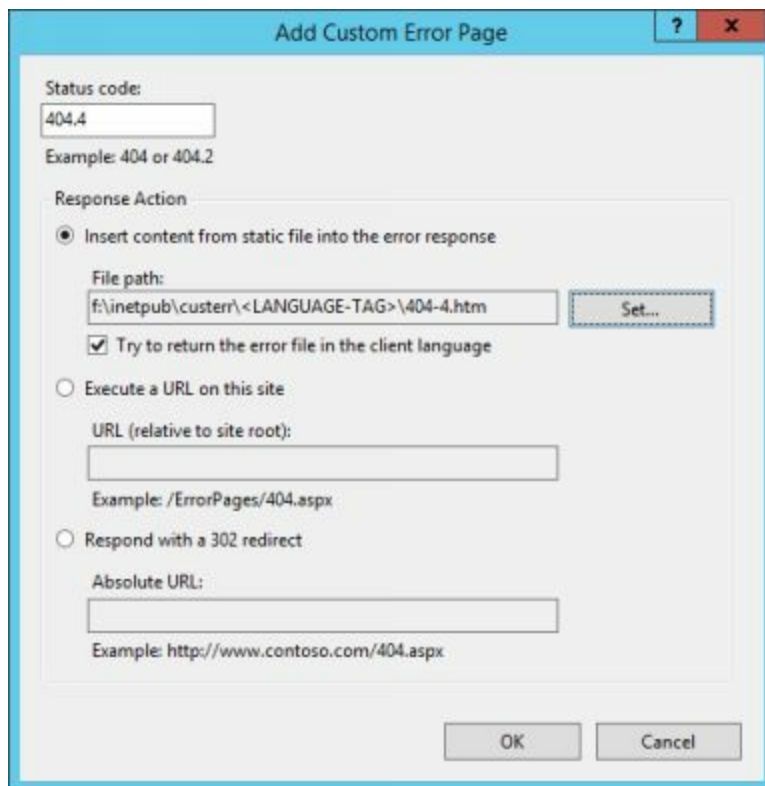
## Adding, Changing, and Removing Custom Error Responses

When you've configured IIS to return individual error responses rather than a default error page, you can manage the way IIS handles each error response. By default, IIS uses the settings you've configured for top-level status codes for any substatus codes you haven't configured. As with the default error page, you can configure error responses to be a file with static content to insert into the error response, a URL to execute for dynamic content, or a redirection to a new URL. When working with static content, you also can configure IIS to try to direct users to error responses for the default language for their client browser.

## Adding Localized Custom Error Responses

You can add a localized custom error response by completing the following steps:

1. In IIS Manager, navigate to the server, site, or directory you want to manage. In the main pane, the Error Pages feature is listed under IIS when you group by area. Double-click Error Pages to open this feature.
2. You should now see a list of the configured error responses. Error responses listed as Local under Entry Type are configured at the current configuration level. To add a custom error response, in the Actions pane, click Add.
3. In the Add Custom Error Page dialog box, type the status code you are configuring, such as 404.4 or 500.100.

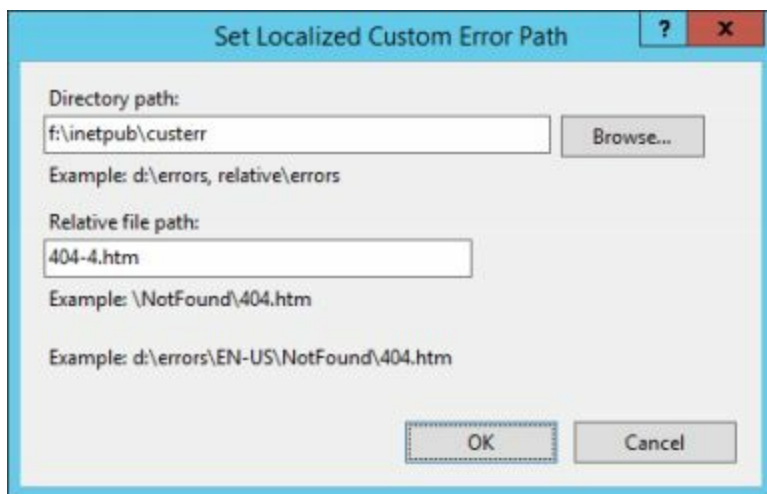


The screenshot shows the 'Add Custom Error Page' dialog box. The 'Status code' field is set to '404.4', with an example '404 or 404.2' provided below it. Under the 'Response Action' section, the first option, 'Insert content from static file into the error response', is selected with a radio button. This option includes a 'File path' field containing 'f:\inetpub\custerr\<LANGUAGE-TAG>\404-4.htm' and a 'Set...' button to its right. Below this, the checkbox 'Try to return the error file in the client language' is checked. The other two options, 'Execute a URL on this site' and 'Respond with a 302 redirect', are unselected. The 'Execute a URL on this site' option has a 'URL (relative to site root)' field with an example '/ErrorPages/404.aspx'. The 'Respond with a 302 redirect' option has an 'Absolute URL' field with an example 'http://www.contoso.com/404.aspx'. At the bottom of the dialog are 'OK' and 'Cancel' buttons.

4. Because you want IIS to try to direct users to the default language for their client browser, select the Try To Return The Error File In The Client Language check

box, and then click Set. This displays the Set Localized Custom Error Path dialog box.

5. Using the Root Directory Path settings, type or select the root directory path for the custom error page. The default root directory path is %SystemRoot%\Inetpub\Custerr.
6. In the Relative File Path text box, type the name of the error file. When working with the standard custom error pages, the file name is the name of the status code with a dash instead of a dot separating any applicable substatus and .htm as the file extension, such as 404-4.htm. The only exception in the standard custom error pages is for 500.100 error responses, which are handled by the 500-100.asp page.
7. Click OK twice to add the custom error page.



**TIP** When you click OK to close the Set Localized Custom Error Path dialog box, IIS Manager sets the full URL path to the custom error response for you. This includes the <LANGUAGE-TAG> placeholder variable. This placeholder is replaced at run time with the client's default language if available and with the server's default language otherwise.

## Adding Non-localized Custom Error Responses

You can add a non-localized custom error response by completing the following steps:

1. In IIS Manager, navigate to the server, site, or directory you want to manage. In the main pane, the Error Pages feature is listed under IIS when you group by area. Double-click Error Pages to open this feature.
2. In the Actions pane, click Add. In the Add Custom Error Page dialog box, type the status code you are configuring, such as 404.4 or 500.100.
3. Since you do not want IIS to try to direct users to the default language for their client browser, you have the following other options:

- **Insert Content From Static File Into The Error Response** Select this option to insert a static file into the error response, and then type the file path for the custom error page. Alternately, click Browse to use the Open dialog box to select the file to use.
- **Execute A URL On This Site** Select this option to execute a URL on this site, and then type a URL relative to the site root into the text box provided. The site root is located with the relative URL of /. Any subdirectories of the site root are below /. Following this, you could reference the 404-4.htm file in the ErrorPages subdirectory of a site by typing **/ErrorPages/404-4.htm**.
- **Respond With A 302 Redirect** Select this option to return the exact URL specified to the client along with a redirection status code. Type an absolute URL path, such as **http://www.reagentpress.com/help/**.

4. Click OK.

## Changing or Removing Custom Error Responses

To change or remove a custom error response in IIS Manager, navigate to the server, site, or directory you want to manage. When you double-click Error Pages in the main pane, you should see a list of the configured error responses. Error responses listed as Local under Entry Type are configured at the current configuration level. You can now perform one of the following actions:

- **Change a response action** To change the response action for a custom error response, click the related entry in the main pane, and then click Edit. In the Edit Custom Error Page, select a new response action, configure related settings as necessary, and then click OK.
- **Change a status code** To change the status code for a custom error response, click the related entry in the main pane, and then click Change Status Code. Type the new status code, and then press Enter.
- **Remove a custom error response** To remove a custom error response, click the related entry in the main pane, and then click Remove. When prompted to confirm, click Yes.

## Using MIME and Configuring Custom File Types

Every static file that's transferred between IIS and a client browser has a data type designator, which is expressed as a Multipurpose Internet Mail Extensions (MIME) type. In the IIS configuration files, MIME type mappings allow you to configure extensions and associated content types that are served as static files.

**NOTE** Dynamic file types do not have MIME type mappings in the IIS configuration files. Instead, dynamic file types have handler mappings. Handler mappings, which also apply to certain types of static content, specify how a file should be processed. For more information on handlers and how they are used with IIS modules, see the “Extending IIS with Modules” section in Chapter 8, “Digging into IIS Schema.”

# Understanding MIME

To understand MIME, you need to know how servers use HTTP to transfer files. HTTP is a multipurpose protocol that you can use to transfer many types of files, including full-motion video sequences, stereo sound tracks, high-resolution images, and other types of media. The transfer of media files wouldn't be possible without the MIME standard. Web servers use MIME to identify the type of object being transferred. Object types are identified in an HTTP header field that comes before the actual data, and this allows a Web client to handle the object file appropriately.

Web servers set the MIME type by using the Content\_Type directive, which is part of the HTTP header sent to client browsers. MIME types are broken down into categories, with each category having a primary subtype associated with it. Basic MIME types are summarized in Table 12-3.

**TABLE 12-3** Basic MIMI Types

Application	Binary data that can be executed or used with another application
Audio	A sound file that requires an output device to be broadcast
Image	A picture that requires an output device to view
Message	An encapsulated mail message
Multipart	Data consisting of multiple parts and possibly many data types
Text	Textual data that can be represented in any character set or formatting language
Video	A video file that requires an output device to preview

MIME subtypes are defined in three categories:

- **Primary** Primary type of data adopted for use as a MIME content type
- **Additional** Additional subtypes that have been officially adopted as MIME content types
- **Extended** Experimental subtypes that haven't been officially adopted as MIME content types

You can easily identify extended subtypes because they begin with the letter x followed by a hyphen. Table 12-4 lists common MIME types and their descriptions.

**TABLE 12-4** Common MIME Types

Application/ mac-binhex40	Macintosh binary-formatted data
Application/msword	Microsoft Office Word document



Application/octet-stream	Binary data that can be executed or used with another application
Application/pdf	Adobe Acrobat Portable Document Format (PDF) document
Application/postscript	Postscript-formatted data
Application/rtf	Rich Text Format (RTF) document
Application/x-compress	Data that has been compressed using UNIX compress
Application/x-gzip	Data that has been compressed using UNIX gzip
Application/x-tar	Data that has been archived using UNIX tar
Application/x-zip-compressed	Data that has been compressed using PKZip or WinZip (or equivalent)
Audio/basic	Audio in a nondescript format
Audio/x-aiff	Audio in Apple Audio Interchange File Format (AIFF)
Audio/x-wav	Audio in Microsoft WAV format
Image/gif	Image in Graphics Interchange Format (GIF)
Image/jpeg	Image in Joint Photographic Experts Group (JPEG) format
Image/tiff	Image in Tagged Image File Format (TIFF)
Text/html	HTML-formatted text
Text/plain	Plain text with no HTML formatting
Video/mpeg	Video in the Moving Picture Experts Group (MPEG) format
Video/quicktime	Video in the Apple QuickTime format
Video/x-msvideo	Video in the Microsoft Audio Video Interleaved (AVI) format

Hundreds of MIME types are configured using file-extension-to-file-type mappings. These mappings allow IIS to support just about any type of file that applications or utilities on the destination computer might expect. If a file doesn't end with a known extension, the file is sent as the default MIME type, which indicates that the file contains application data. In most cases use of the default MIME type means that the client is unable to handle the file or to trigger other utilities that handle the file. If you expect the client to handle a new file type appropriately, you must create a file-extension-to-file-type mapping.

MIME type mappings set at the server configuration level apply to all websites on the server. At the server configuration level, you can edit existing MIME types, configure additional MIME types, or delete unwanted MIME types. You can also create and manage additional MIME type mappings for individual sites and directories. When you

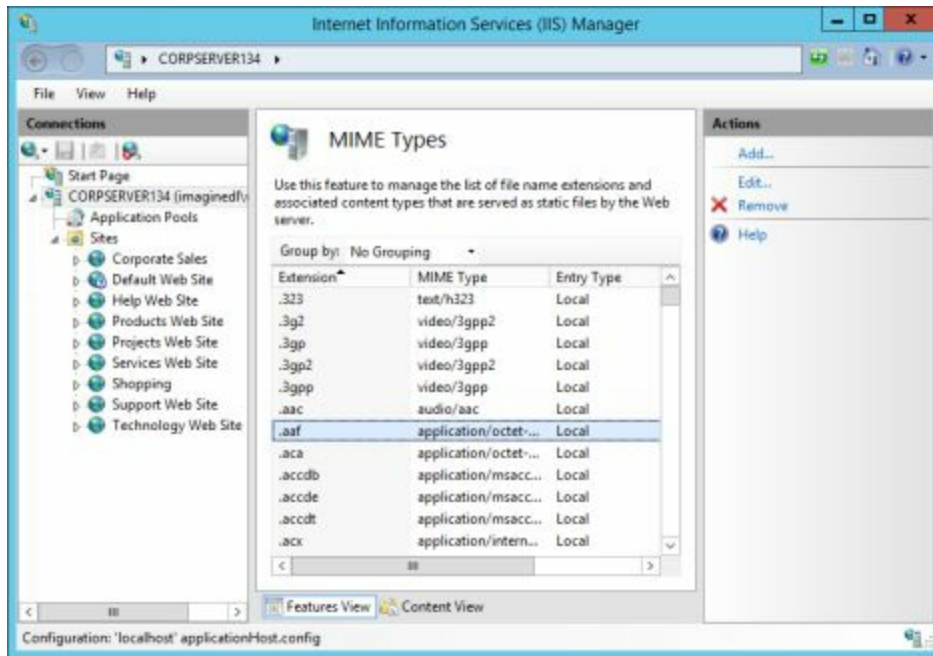
do this, the MIME type mappings are available only in the site or directory in which they're configured.

# Viewing and Configuring MIME Types

You can view and configure MIME types by completing the following steps:

1. In IIS Manager, navigate to the server, site, or directory you want to manage. You can view the MIME types for all websites on a server by selecting the server node. You can view the MIME types for sites and directories by selecting the appropriate nodes.
2. In the main pane, the MIME Types feature is listed under IIS when you group by area. Double-click MIME Types to open this feature. You should now see a list of configured MIME types by file extension and associated MIME type.
3. In IIS Manager, double-click the computer node for the IIS server you want to work with, and then select Properties.
4. Double-click MIME Types. You should see a list of the MIME types. Computer MIME types are active for all websites on the server.
5. Use the following settings to configure MIME types:
  - **Add** Adds a new MIME type. To add a MIME type, click Add. In the File Name Extension field, type a file extension such as .html, and then in the MIME Type field, type a MIME type such as text/html. Complete the process by clicking OK.
  - **Edit** Edits a MIME type mapping. To edit a MIME type, select it, and then click Edit. In the Edit MIME Type dialog box that appears, change the file extension and the content MIME type.
  - **Remove** Removes a MIME type mapping. To remove a MIME type, select it and then click Remove.

By using the IIS Command-line Administration Tool, you can manage MIME types by using the Set Config command and the staticContent section of the configuration file.



Sample 12-12 provides the syntax and usage for adding and removing MIME types. Because you are referencing into the fileExtension 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 fileExtension collection.

## SAMPLE 12-12 Configuring MIME Types Syntax and Usage

### Syntax

```
appcmd set config ["ConfigPath"] /section:staticContent
[ /+["fileExtension='Extension',mimeType='MIMETYPE'"] |
/-["fileExtension='Extension',mimeType='MIMETYPE'"] ]
```

### Usage for Adding MIME Types

```
appcmd set config ["ConfigPath"] /section:staticContent
/+"[fileExtension='.htm',mimeType='text/html']"
```

### Usage for Removing MIME Types

```
appcmd set config ["ConfigPath"] /section:staticContent
/-"[fileExtension='.htm',mimeType='text/html']"
```

## Additional Customization Tips

Update sites, jump pages, and error forwarding are three additional techniques you can use to customize your IIS websites. Each of these techniques is discussed in the sections that follow.

# Using Update Sites to Manage Outages

An update site allows you to handle outages in a way that's customer-friendly. Use the update sites function to display alternate content when your primary sites are offline. So rather than seeing an error message where the user expects to find content, the user sees a message that provides information regarding the outage plus additional helpful information.

Each website you publish should have an update site. You create update sites by completing the following steps:

1. Create or arrange for someone else to create a Web page that can be displayed during outages. The page should explain that you're performing maintenance on the website and that the site will be back online shortly. The page can also provide links to other sites that your organization publishes so that the user has somewhere else to visit during the maintenance.
2. Use File Explorer to create a directory for the update site. The best location for this directory is on the Web server's local drive. Afterward, copy the content files created by the Web development team into this directory.

**TIP** I recommend that you create a top-level directory for storing the home directories and then create subdirectories for each update site. For example, you can create a top-level directory called D:\UpdateSites and then use subdirectories called WWWUpdate, ServicesUpdate, and ProductsUpdate to store the files for [www.reagentpress.com](http://www.reagentpress.com), [services.reagentpress.com](http://services.reagentpress.com), and [products.reagentpress.com](http://products.reagentpress.com), respectively.

3. In IIS Manager, select the main node for the computer you want to work with. If the computer isn't shown, connect to it.
4. Click the Sites node. You should now see a list of websites already configured on the server. You should write down the host header, IP address, and port configuration of the primary site you want to mimic during outages. To view this information, right-click the desired website, and then choose Bindings.
5. Create a new site using the configuration settings you just noted. Name the site so that it clearly identifies the site as an update site. For example, if you are creating an update site for the corporate website, name the update sites "Update Site for the Corporate Web." Ensure that the new site doesn't start by clearing the Start website Immediately check box before you click OK to add the website.
6. Perform the following tasks:
  - Enable default content pages.
  - Remove the existing default documents.

- Add a default document and set the document name to the name of the outage page just created.
- 7. By using the site's Custom Errors feature, edit the properties for 401, 403, 404, 405, 406, 412, 500, 501, and 502 errors. These errors should have the Message Type set to File and have a File path that points to your new outage page.
- 8. Update other site features as necessary.

Once you create the update site, you can activate it as follows:

1. Use IIS Manager to stop the primary site prior to performing maintenance, and then start the related update site.
2. Confirm that the update site is running by visiting the website in your browser. If the site is properly configured, you should be able to append any file name to the URL and be directed to the outage page.
3. Perform the necessary maintenance on the primary site. When you're finished, stop the update site and then start the primary site.
4. Confirm that the primary site is running by visiting the website in your browser.

# Using Jump Pages for Advertising

A jump page is an intermediate page that redirects a user to another location. You can use jump pages to track click-throughs on banner advertisements or inbound requests from advertising done by the company.

With banner ads, jump pages ensure that users visit a page within your site before moving off to a page at an advertiser's site. This allows you to track the success of advertising on your site. Here's how it works:

1. A page in your site has a banner ad that is linked to a jump page on your site.
2. A user clicks on the ad and is directed to the jump page. The Web server tracks the page access and records it in the log file.
3. The jump page is configured to redirect the user to a page on the advertiser's website.

With corporate advertising, jump pages ensure that you can track the source of a visit to advertising done by the company. This allows you to track the success of your company's advertising efforts. Here's how it works:

1. The marketing department develops a piece of advertising collateral—for instance, a product brochure. Somewhere in the brochure, there's a reference to a URL on your site. This is the URL for the jump page you've configured.
2. A user types in the URL to the jump page as it was listed in the ad. The Web server tracks the page access and records it in the log file.
3. The jump page is configured to redirect the user to a page on your website where the advertised product or service is covered.

Each jump page you create should be unique, or you should create a dynamic page that reads an embedded code within the URL and then redirects the user. For example, you can create a page called `Jump.asp` that reads the first parameter passed to the script as the advertising code. Then you can create a link in the banner ad that specifies the URL and the code, such as `Jump.asp?4408`.



# Handling 404 Errors and Preventing Dead Ends

Users hate dead ends, and that's just what a 404 error represents. Rather than having the browser display an apparently meaningless 404—File Or Directory Not Found error, you should throw the user a lifeline by doing one of two things:

- [Replacing the default error file with a file that provides helpful information and links](#)
- [Redirecting all 404 errors to your site's home page](#)

Either technique makes your website a better place to visit. This feature could be the one thing that separates your website from the pack.

## Chapter 13. Web Stats 101

One of your primary responsibilities as a Web administrator may be to log access to your company's Internet servers. Gathering the correct access information and recording this information in the proper format so that it can be read and analyzed can be a challenge. Thankfully, there is software that you can use to analyze access logs. It's called tracking software. You'll find many different types of tracking software. Most commercial tracking software produces detailed reports that include tables and graphs that summarize activity for specific periods. For example, you could compile tracking reports daily, weekly, or monthly.

## Logging Access Stats

Access logs are created when you enable logging for an IIS server. Every time someone requests a file from your website, an entry goes into the access log, making the access log a running history of every successful and unsuccessful attempt to retrieve information from your site. Because each entry has its own line, entries in the access log can be easily extracted and compiled into reports. From these reports, you can learn many things about those who visit your site. You can do the following:

- Determine the busiest times of the day and week
- Determine which browsers and platforms are used by people who visit your site
- Discover popular and unpopular resources
- Discover sites that refer users to your site
- Learn more about the effectiveness of your advertising
- Learn more about the people who visit your site
- Obtain information about search engine usage and keywords
- Obtain information about the amount of time users spend at the site

# Navigating File Formats

The file format for access logs can be configured in several different ways. You can configure standard logging, Open Database Connectivity (ODBC) logging, and extended logging. With standard logging, you choose a log file format and rely on the format to record the user access information you need. With ODBC logging, you record user access directly to an ODBC-compliant database, such as Microsoft SQL Server. With extended logging, you can customize the logging process and record exactly the information you need to track user access.

IIS can be configured to use per-server or per-site logging. With per-server logging, IIS tracks requests for all websites configured on a server in a single log file. With per-site logging, IIS tracks requests for each website in separate log files. You'll find that per-server logging is more efficient than per-site logging and can reduce the overhead associated with logging.

# Choosing Your Logging Configuration

Per-server logging is the ideal when an IIS server has a large number of sites, such as with Internet service providers (ISPs), and for busy commercial sites, such as those for large organizations. For small and medium installations, you'll find that per-site logging is easier to work with because you'll have separate log files for each site and can use just about any tracking software to review access statistics.

With per-server logging, you can use one of two logging formats:

- **Centralized Binary Logging** Use centralized binary logging when you want all websites running on a server to write log data to a single log file. With centralized binary logging, the log files contain fixed-length and index records that are written in a raw binary format called the Internet Binary Log (IBL) format. Professional software applications and other tools, such as LogParser, can read this format. Because IIS writes the logs in binary format, this logging technique is the most efficient and is recommended for busy commercial sites and ISPs.
- **Centralized World Wide Web Consortium (W3C) Extended Log File Format** Use the centralized extended format when you want all websites running on a server to write log data to a single log file and must customize the tracked information and obtain detailed information. With this format, log entries can become large, and this greatly increases the amount of storage space required. Because recording lengthy entries also can affect the performance of a busy server, this format is not as efficient as centralized binary logging. However, a single centralized extended log is still more efficient than having multiple decentralized extended logs.

With per-site logging, the available formats are:

- **National Center for Supercomputing Applications (NCSA) Common Log File Format** Use the common log format when your reporting and tracking needs are basic. With this format, log entries are small, so not as much storage space is required for logging.
- **Microsoft Internet Information Services (IIS) Log File Format** Use the IIS format when you need a bit more information from the logs but don't need to tailor the entries to get detailed information. With this format, log entries are compact, so not as much storage space is required for logging.
- **World Wide Web Consortium (W3C) Extended Log File Format** Use the extended format when you must customize the tracked information and obtain detailed information. With this format, log entries can become large, and this greatly increases the amount of storage space required. Recording lengthy entries

can also affect the performance of a busy server.

- **Custom (ODBC Logging)** Use the ODBC format when you want to write access information directly to an ODBC-compliant database. With ODBC logging, you'll need tracking software capable of reading from a database. Entries are compact, however, and data can be read much more quickly than from a standard log file. Keep in mind that ODBC logging is more processor-intensive when you write logs directly to a local database instance.

**TIP** With NCSA, IIS, and W3C logging, you have two choices for text encoding. You can use standard ANSI encoding or you can use UTF-8 encoding. ANSI encoding is best used with sites and file names that use standard English characters. UTF-8 encoding is best used with sites and file names that use standard English characters in addition to non-English characters. By default, IIS uses UTF-8 encoding. Regardless of whether you use per-server or per-site logging, you configure text encoding at the server level, and all text-based log files created on the server use this encoding.

Because an understanding of what is written to log files is important to understanding logging itself, the sections that follow examine the main file formats. I'll start with the most basic format and then work toward the most advanced format. After this discussion, you'll be able to determine what each format has to offer and hopefully better determine when to use each format.

# Working with the NCSA Common Log File Format

The NCSA common log file format is the most basic log format. The common log format is a fixed ASCII or UTF-8 format in which each log entry represents a unique file request. You'll use the common log format when your tracking and reporting needs are basic. More specifically, the common log format is a good choice when you need to track only certain items, such as:

- Hits (the number of unique file requests)
- Page views (the number of unique page requests)
- Visits (the number of user sessions in a specified period)
- Other basic access information

With this format, log entries are small, so not as much storage space is required for logging. Each entry in the common log format has only seven fields. These fields are:

- Host
- Identification
- User Authentication
- Time Stamp
- HTTP Request Type
- Status Code
- Transfer Volume

As you'll see, the common log format is easy to understand, which makes it a good stepping-stone to more advanced log file formats. The following listing shows entries in a sample access log that are formatted using the NCSA common log file format. As you can see from the sample, log fields are separated by spaces:

```
192.168.11.15 - ENGSVR01\wrstane [15/Dec/2017:18:44:57 -0800]
"GET / HTTP/1.1" 200 1970 192.168.11.15 - ENGSVR01\wrstane [15/Dec/2017:18:45:06 -0800]
"GET /home.gif HTTP/1.1" 200 5032 192.168.11.15 - ENGSVR01\wrstane [15/Dec/2017:18:45:28 -0800]
"GET /main.htm HTTP/1.1" 200 5432 192.168.11.15 - ENGSVR01\wrstane [15/Dec/2017:18:45:31 -0800]
"GET /details.gif HTTP/1.1" 200 1211 192.168.11.15 - ENGSVR01\wrstane [15/Dec/2017:18:45:31 -0800]
"GET /menu.gif HTTP/1.1" 200 6075 192.168.11.15 - ENGSVR01\wrstane [15/Dec/2017:18:45:31 -0800]
"GET /sidebar.gif HTTP/1.1" 200 9023 192.168.11.15 - ENGSVR01\wrstane [15/Dec/2017:18:45:31 -0800]
"GET /sun.gif HTTP/1.1" 200 4706 192.168.11.15 - ENGSVR01\wrstane [15/Dec/2017:18:45:38 -0800]
"GET /moon.gif HTTP/1.1" 200 1984 192.168.11.15 - ENGSVR01\wrstane [15/Dec/2017:18:45:41 -0800]
"GET /stars.gif HTTP/1.1" 200 2098
```

Most other log file formats are based on the NCSA file format, so it is useful to examine how these fields are used.

# Host Field

Host is the first field in the common log format. This field identifies the host computer requesting a file from your Web server. The value in this field is either the IP address of the remote host, such as 192.168.11.15, or the fully qualified domain name of the remote host, such as net48.imaginedlands.com. The following example shows an HTTP query initiated by a host that was successfully resolved to a domain name:

```
net48.imaginedlands.com - ENGSVR01\wrstanek [15/Dec/2017:18:44:57 -0800] "GET / HTTP/1.1" 200 1970
```

IP addresses are the numeric equivalent of fully qualified domain names. You can often use a reverse DNS lookup to determine the actual domain name from the IP address. When you have a domain name or resolve an IP address to an actual name, you can examine the name to learn more about the user accessing your server. Divisions within the domain name are separated by periods. The final division identifies the domain class, which can tell you where the user lives and works.

Domain classes are geographically and demographically organized. Geographically organized domain classes end in a two- or three-letter designator for the state or country in which the user lives. For example, the .ca domain class is for companies in Canada. Demographically organized domain classes tell you the type of company providing network access to the user.

Table 13-1 summarizes the basic domain classes.

**TABLE 13-1** Basic Domain Classes

.com	Commercial; users from commercial organizations
.edu	Education; users from colleges and universities
.gov	U.S. government; users from U.S. government agencies (except military)
.mil	U.S. military; users who work at military installations
.net	Network; users who work at network service providers and other network-related organizations
.org	Nonprofit organizations; users who work for nonprofit organizations



# Identification Field

The Identification field is the second field in the common log format. This field is meant to identify users by their user name but in practice is rarely used. Because of this, you will generally see a hyphen (-) in this field, as in the following:

```
net48.imaginedlands.com - ENGSVR01\wrstanek [15/Dec/2017:18:44:57 -0800] "GET / HTTP/1.1" 200 1970
```

If you do see a value in this field, keep in mind that the user name is not validated. This means that it could be fictitious and shouldn't be trusted.

# User Authentication Field

The User Authentication field is the third field in the common log format. If you have a password-protected area on your website, users must authenticate themselves with a user name and password that is registered for this area. After users validate themselves with their user name and password, their user name is entered in the User Authentication field. In unprotected areas of a site, you will usually see a hyphen (-) in this field. In protected areas of a site, you will see the account name of the authenticated user. The account name can be preceded by the name of the domain in which the user is authenticated, as shown in this example:

net48.imaginedlands.com - **ENGSVR01\wrstane k** [15/Dec/2017:18:44:57 -0800] "GET / HTTP/1.1" 200 1970

# Time Stamp Field

The Time Stamp field is the fourth field in the common log format. This field tells you exactly when someone accessed a file on the server. The format for the Time Stamp field is as follows:

DD/MMM/YYYY:HH:MM:SS OFFSET

such as:

15/Dec/2017:18:44:57 -0800

The only designator that probably doesn't make sense is the offset. The offset indicates the difference in the server's time from Greenwich Mean Time (GMT) standard time. In the following example, the offset is -8 hours, meaning that the server time is eight hours behind GMT:

net48.imaginedlands.com - ENGSVR01\wrstanek [15/Dec/2017:18:44:57 -0800] "GET / HTTP/1.1" 200 1970

# HTTP Request Field

The HTTP Request field is the fifth field in the common log format. Use this field to determine the method that the remote client used to request the resource, the resource that the remote client requested, and the HTTP version that the client used to retrieve the resource. In the following example, the HTTP Request field information is bold:

```
192.168.11.15 - ENGSVR01\wrstanek [15/Dec/2015:18:45:06 -0800]  
"GET /home.gif HTTP/1.1" 200 5032
```

Here, the transfer method is GET, the resource is /Home.gif, and the transfer method is HTTP 1.1. One thing you should note is that resources are specified using relative Uniform Resource Locators (URLs). The server interprets relative URLs. For example, if you request the file <http://www.imaginedlands.com/home/main.htm>, the server will use the relative URL /Home/Main.htm to log where the file is found. When you see an entry that ends in a slash, keep in mind that this refers to the default document for a directory, which is typically called Index.htm or Default.asp.

# Status Code Field

The Status Code field is the sixth field in the common log format. Status codes indicate whether files were transferred correctly, were loaded from cache, were not found, and so on. Generally, status codes are three-digit numbers. As shown in Table 13-2, the first digit indicates the class or category of the status code.

**TABLE 13-2** Status Code Classes

1XX	Continue/protocol change
2XX	Success
3XX	Redirection
4XX	Client error/failure
5XX	Server error

Because you'll rarely see a status code beginning with 1, you need to remember only the other four categories. A status code that begins with 2 indicates that the associated file transferred successfully. A status code that begins with 3 indicates that the server performed a redirect. A status code that begins with 4 indicates some type of client error or failure. Last, a status code that begins with 5 tells you that a server error occurred.

## Transfer Volume Field

The last field in the common log format is the Transfer Volume field. This field indicates the number of bytes transferred to the client because of the request. In the following example, 4096 bytes (or 4 megabytes) were transferred to the client:

```
net48.imaginedlands.com - ENGSVR01\wrstanek [15/Dec/2017:18:45:06 -0800]  
"GET / HTTP/1.1" 200 4096
```

You'll see a transfer volume only when the status code class indicates success. If another status code class is used in field six, the Transfer Volume field will contain a hyphen (-) or a 0 to indicate that no data was transferred.

## Working with the Microsoft IIS Log File Format

Like the common log format, the Microsoft IIS log file format is a fixed ASCII format. This means that the fields in the log are of a fixed type and cannot be changed. It also means that the log is formatted as standard ASCII text and can be read with any standard text editor or compliant application.

The following listing shows entries from a sample log using the IIS log file format. The IIS log entries include common log fields such as the client IP address, authenticated user name, request date and time, HTTP status code, and number of bytes received. IIS log entries also include detailed items such as the Web service name, the server IP address, and the elapsed time. Note that commas separate log fields, and entries are much longer than those in the common log file format.

```
192.14.16.2, -, 04/15/2017, 15:42:25, W3SVC1, ENGSVR01, 192.15.14.81, 0, 594, 3847, 401, 5, GET, /start.asp, -,
192.14.16.2, ENGSVR01\wrstaneek, 04/15/2017, 15:42:25, W3SVC1, ENGSVR01, 192.15.14.81, 10, 412, 3406, 404, 0,
GET, /localstart.asp, |-|0|404_Object_Not_Found,
192.14.16.2, -, 04/15/2017, 15:42:29, W3SVC1, ENGSVR01, 192.15.14.81, 0, 622, 3847, 401, 5, GET, /default.asp, -,
192.14.16.2, ENGSVR01\wrstaneek, 04/15/2017, 15:42:29, W3SVC1, ENGSVR01, 192.15.14.81, 10, 426, 0, 200, 0,
GET, /default.asp, -,
192.14.16.2, ENGSVR01\wrstaneek, 04/15/2017, 15:42:29, W3SVC1, ENGSVR01, 192.15.14.81, 10, 368, 0, 200, 0,
GET, /contents.asp, -,
192.14.16.2, -, 04/15/2017, 15:42:29, W3SVC1, ENGSVR01, 192.15.14.81, 0, 732, 3847, 401, 5, GET, /navbar.asp, -,
192.14.16.2, -, 04/15/2017, 15:42:29, W3SVC1, ENGSVR01, 192.15.14.81, 0, 742, 3847, 401, 5, GET, /core.htm, -,
192.14.16.2, ENGSVR01\wrstaneek, 04/15/2017, 15:42:29, W3SVC1, ENGSVR01, 192.15.14.81, 20, 481, 0, 200, 0,
GET, /navbar.asp, -,
192.14.16.2, ENGSVR01\wrstaneek, 04/15/2017, 15:42:29, W3SVC1, ENGSVR01, 192.15.14.81, 91, 486, 6520, 200, 0,
GET, /core.htm, -,
```

The fields supported by IIS are summarized in Table 13-3. Note that the listed field order is the general order used by IIS to record fields.

**TABLE 13-3** Fields for the IIS Log File Format

Client IP	IP address of the client, such as 192.14.16.2.
User Name	Authenticated name of the user, such as ENGSVR01\wrstaneek.
Date	Date when the transaction was completed, such as 04/15/2015.
Time	Time when the transaction was completed, such as 15:42:29.
Service	Name of the Web service logging the transaction, such as W3SVC1.
Computer Name	Name of the computer that made the request, such as ENGSVR01.

Server IP	IP address of the Web server, such as 192.15.14.81.
Elapsed Time	Time taken (in milliseconds) for the transaction to be completed, such as 40.
Bytes Received	Number of bytes received by the server in the client request, such as 486.
Bytes Sent	Number of bytes sent to the client, such as 6520.
Status Code	HTTP status code, such as 200.
Windows Status Code	Error status code from Microsoft Windows, such as 0.
Method Used	HTTP request method, such as GET.
File URI	The requested file, such as /start.asp.
Referrer	The referrer—that is, the location where the user came, such as <a href="http://www.imaginedlands.com/">http://www.imaginedlands.com/</a> .



# Working with the W3C Extended Log File Format

The W3C extended log file format is much different from either of the previously discussed log file formats. With this format, you can customize the tracked information and obtain detailed information. When you customize an extended log file, you select the fields you want the server to log, and the server handles the logging for you. Keep in mind that each additional field you track adds to the size of entries recorded in the access logs, and this can greatly increase the amount of storage space required.

The following listing shows sample entries from an extended log. Note that, as with the common log format, extended log fields are separated with spaces.

```
#Software: Microsoft Internet Information Services 10.0
#Version: 1.0
#Date: 2017-12-15 06:28:11
#Fields: date time c-ip cs-username s-ip s-port cs-method
#cs-uri-stem cs-uri-query sc-status cs(User-Agent)
2017-04-05 06:27:58 192.14.16.2 ENGSRV01\wrstane 192.14.15.81 80 GET /cust.htm - 304
Mozilla/4.0+(compatible;+MSIE+7.01;+Windows+NT+6.0;+SLCC1;+.NET+CLR+
2.0.50727) 2017-12-15 06:28:00 192.14.16.2 ENGSRV01\wrstane 192.14.15.81 80 GET /data.htm - 304
Mozilla/4.0+(compatible;+MSIE+7.01;+Windows+NT+6.0;+SLCC1;+.NET+CLR+
2.0.50727) 2017-12-15 06:28:02 192.14.16.2 ENGSRV01\wrstane 192.14.15.81 80 GET /store.htm - 200
Mozilla/4.0+(compatible;+MSIE+7.01;+Windows+NT+6.0;+SLCC1;+.NET+CLR+
2.0.50727) 2017-12-15 06:28:02 192.14.16.2 ENGSRV01\wrstane 192.14.15.81 80 GET /prodadd.htm - 200
Mozilla/4.0+(compatible;+MSIE+7.01;+Windows+NT+6.0;+SLCC1;+.NET+CLR+
2.0.50727) 2017-12-15 06:28:05 192.14.16.2 ENGSRV01\wrstane 192.14.15.81 80 GET /datastop.htm - 200
Mozilla/4.0+(compatible;+MSIE+7.01;+Windows+NT+6.0;+SLCC1;+.NET+CLR+2.0.50727)
```

The first time you look at log entries that use the extended format, you might be a bit confused. The reason for this is that the extended logs are written with server directives in addition to file requests. The good news is that server directives are always preceded by the hash symbol (#), easily allowing you to distinguish them from actual file requests. The key directives you'll see are the directives that identify the server software and the fields being recorded. These directives are summarized in Table 13-4.

**TABLE 13-4** Directives Used with the Extended Log File Format

Date	Identifies the date and time the entries were made in the log
End-Date	Identifies the date and time the log was finished and then archived
Fields	Specifies the fields and the field order used in the log file
Remark	Specifies comments
Software	Identifies the server software that created the log entries
Start-Date	Identifies the date and time the log was started
Version	Identifies the version of the extended log file format used

Most extended log fields have a prefix. The prefix tells you how a particular field is used or how the field was obtained. For example, the cs prefix tells you that the field was obtained from a request sent by the client to the server. Field prefixes are summarized in Table 13-5.

**TABLE 13-5** Prefixes Used with the Extended Log Fields

c	Identifies a client-related field
s	Identifies a server-related field
r	Identifies a remote server field
cs	Identifies information obtained from a request sent by the client to the server
sc	Identifies information obtained from a request sent by the IIS server to the client
sr	Identifies information obtained from a request sent by the Web server to a remote server (used by proxies)
rs	Identifies information obtained from a request sent by a remote server to the IIS server (used by proxies)
x	Application-specific prefix

All fields recorded in an extended log have a field identifier. This identifier details the type of information a particular field records. To create a named field, the IIS server can combine a field prefix with a field identifier, or it can simply use a field identifier. The most commonly used field names are summarized in Table 13-6. As you examine the table, keep in mind that most of these fields relate directly to the fields we've already discussed for the common and extended log file formats. Again, the key difference is that the extended format can give you information that is much more detailed.

**TIP** If you are testing HTTP 2.0 and want to determine whether clients are using it, be sure to add the Protocol Version field to your W3C logs.

**TABLE 13-6** Field Identifiers Used with the Extended File Format

Bytes Received (cs-bytes)	Number of bytes received by the server.
Bytes Sent (sc-bytes)	Number of bytes sent by the server.
Client IP Address (c-ip)	IP address of the client that accessed the server.
Cookie [cs(Cookie)]	Content of the cookie sent or received (if any).
Date (Date)	Date on which the activity occurred.
Method Used (cs-method)	HTTP request method.
Protocol Status (sc-status)	HTTP status code, such as 404.
Protocol Substatus (sc-substatus)	HTTP substatus code, such as 2.

Protocol Version (cs-protocol)	Protocol version used by the client.
Referrer [cs(Referer)]	Previous site visited by the user. This site provided a link to the current site.
Server IP (s-ip)	IP address of the IIS server.
Server Name (s-computername)	Name of the IIS server.
Server Port (s-port)	Port number through which the client is connected.
Service Name and Instance Number (s-sitename )	Internet site and instance number that was running on the server.
Time (Time)	Time the activity occurred.
Time Taken (time-taken)	Time taken (in milliseconds) for the transaction to be completed.
URI Query (cs-uri-query)	Query parameters passed in request (if any).
URI Stem (cs-uri-stem)	Requested resource.
User Agent [cs(User-Agent)]	Browser type and version used on the client.
User Name (c-username)	Name of an authenticated user (if available).
Win32 Status (sc-win32-status)	Error status code from Windows.

**REAL WORLD** When a server is using centralized extended logging, be sure to track the Service Name because this field ensures that the site name and identity is written with each log entry. To ensure proper tracking of errors, you should track both the protocol status and substatus. Protocol Status logs the HTTP status code of the request, such as 404. Protocol Substatus logs the HTTP substatus code of the request, such as 2. When used together, the fields provide the complete status of the request, such as 404.2.

Unlike IIS 6, IIS 10 cannot log process accounting information related to HTTP requests. The reason for this is that process accounting applies only to resources used by out-of-process applications. Process accounting does not cover resources used by pooled or in-process applications.

## Working with ODBC Logging

You can use the ODBC logging format when you want to write access information directly to an ODBC-compliant database, such as Microsoft Office Access or Microsoft SQL Server. The key advantage of ODBC logging is that access entries are written directly to a database in a format that can be quickly read and interpreted by compliant software. The major disadvantages of ODBC logging are two-fold. First, it requires basic database administration skills to configure and maintain. Second, direct ODBC-logging can use a great deal of system resources, so it could be extremely inefficient.

When using ODBC logging, you must configure a Data Source Name (DSN) that allows IIS to connect to your ODBC database. You must also create a database that can be used for logging. This database must have a table with the appropriate fields for the logging data.

Typically, you'll use the same database for logging information from multiple sites with each site writing to a separate table in the database. For example, if you wanted to log Corporate Web, Support Web, and Sales Web access information in your database, and these services were running on separate sites, you would create three tables in your database, such as the following:

- [CorpLog](#)
- [SupportLog](#)
- [SalesLog](#)

These tables would have the columns and data types for field values summarized in Table 13-7. The columns must be configured exactly as shown in the table. Don't worry; IIS includes an SQL script that you can use to create the necessary table structures. This script is located in the %SystemRoot%\System32\Inetsrv directory and is named Logtemp.sql.

<b>NOTE</b> If you use the Logtemp.sql script, be sure to edit the table name set in the CREATE TABLE statement. The default table name is inetlog.
---

**TABLE 13-7** Table Fields for ODBC Logging

ClientHost	IP address of the client that accessed the server. Set as varchar(255).
Username	Name of an authenticated user (if available). Set as varchar(255).
LogTime	Date and time when the activity occurred. Set as datetime.
Service	Internet site and instance number that was running on the server. Set as varchar(255).
Machine	Name of the computer that made the request. Set as varchar(255).

ServerIP	IP address of the IIS server. Set as varchar(50).
ProcessingTime	Time taken (in milliseconds) for the transaction to be completed. Set as int.
BytesRecvd	Number of bytes received by the server. Set as int.
BytesSent	Number of bytes sent by the server. Set as int.
ServiceStatus	HTTP status code. Set as int.
Win32Status	Error status code from Windows. Set as int.
Operation	HTTP request method. Set as varchar(255).
Target	Requested resource. Set as varchar(255).
Parameters	Query parameters passed in request (if any). Set as varchar(255).

## Working with Centralized Binary Logging

You can use centralized binary logging when you want all websites running on a server to write log data to a single log file. With centralized binary logging, the log files are written in a raw binary format called the Internet Binary Log (IBL) format. This format can be read by many professional software applications or by using other tools, such as LogParser.

On a large IIS installation where the server is running hundreds or thousands of sites, centralized binary logging can dramatically reduce the overhead associated with logging activities. Two types of records are written to the binary log files:

- **Index** Act as record headers, similar to the W3C extended log file format, where software, version, date, and field information is provided.
- **Fixed-length** Provide the detailed information about requests. Each value in each field in the entry is stored with a fixed length.

For more information on centralized binary logging, see the “Configuring Centralized Binary Logging” section in Chapter 14, “Configuring Logging.”

# Understanding Logging

In IIS, the following role services make it possible for you to use logging:

- **HTTP Logging** Makes available the standard logging features
- **Custom Logging** Makes available custom logging features (including features required for ODBC logging)
- **ODBC Logging** Makes available ODBC logging features
- **Logging Tools** Makes available additional resources and tools for working with logs

Once the appropriate role services are installed, you can enable and configure IIS logging so that new log entries are generated whenever users access the server. This causes a steady increase in log file size and eventually, in the number of log files. On a busy server, log files can quickly grow to several gigabytes, so therefore, you might need to balance the need to gather information against the need to limit log files to a manageable size.

**NOTE** Keep in mind that log files are stored as ASCII or UTF-8 text files, and if you must, you can split or combine log files as you would with any text file. If your server runs out of disk space when IIS is attempting to add a log entry to a file, IIS logging shuts down and logs a logging error event in the Application log. When disk space is available again, IIS resumes logging file access and writes a start-logging event in the Application log.

When you configure logging, you specify how log files are created and saved. Logs can be created according to a time schedule, such as hourly, daily, weekly, and monthly. Logs can also be set to a fixed file size, such as 100 MB, or they can be allowed to grow to an unlimited file size. The name of a log file indicates its log file format in addition to the time frame or sequence of the log. The various naming formats are summarized in Table 13-8. If a log file uses UTF-8 encoding rather than ASCII, the log file will have a U\_ prefix, such as u\_ex171231.log for a tracking log in W3C extended log format that has UTF-8 encoding.

**TABLE 13-8** Conventions for Log File Names by Log Format

IIS Log Format	
By file size	Inetsvnn.log
Unlimited	Inetsvnn.log
Hourly	Inyymmddhh.log
Daily	Inyymmdd.log

Weekly	Inyymmww.log
Monthly	Inyymm.log
<b>NCSA Common Log Format</b>	
By file size	Ncsann.log
Unlimited	Ncsann.log
Hourly	Ncyymmddhh.log
Daily	Ncyymmdd.log
Weekly	Ncyymmww.log
Monthly	Ncyymm.log
<b>W3C Extended Log Format</b>	
By file size	Extendnn.log
Unlimited	Extendnn.log
Hourly	Exyymmddhh.log
Daily	Exyymmdd.log
Weekly	Exyymmww.log
Monthly	Exyymm.log
<b>Centralized Binary Log Format</b>	
Hourly	Rayymmddhh.ibl
Daily	Rayymmdd.log
Weekly	Rayymmww.ibl
Monthly	Rayymm.ibl

By default, log files are written to the %SystemDrive%\Inetpub\Logs\LogFiles directory. You can configure logging to a different directory, such as D:\LogFiles. Regardless of whether you use the default directory location or assign a new directory location for logs, you'll find separate subdirectories for each service that is enabled for logging under the primary directory.

Subdirectories for sites are named W3SVCN where N is the index number of the service or a random tracking value. The only exception is when you use centralized binary logging or centralized extended logging. Here, website logs are stored in the



%SystemDrive%\Inetpub\Logs\LogFiles\W3SVC directory.

The default server created is number 1. If you create additional sites, an incremental numeric identifier is used. Following this, you could have site directories named W3SVC, W3SVC1, W3SVC2, and so on. To correlate the identifier value to specific websites, in IIS Manager, select the Sites node, and then look at the Name and ID columns to determine which identifier belongs to which site.

## Chapter 14. Configuring Logging

Now that you know how log files are used and created, let's look at how you can enable and configure logging. The sections that follow examine each of the available logging formats. Keep the following in mind:

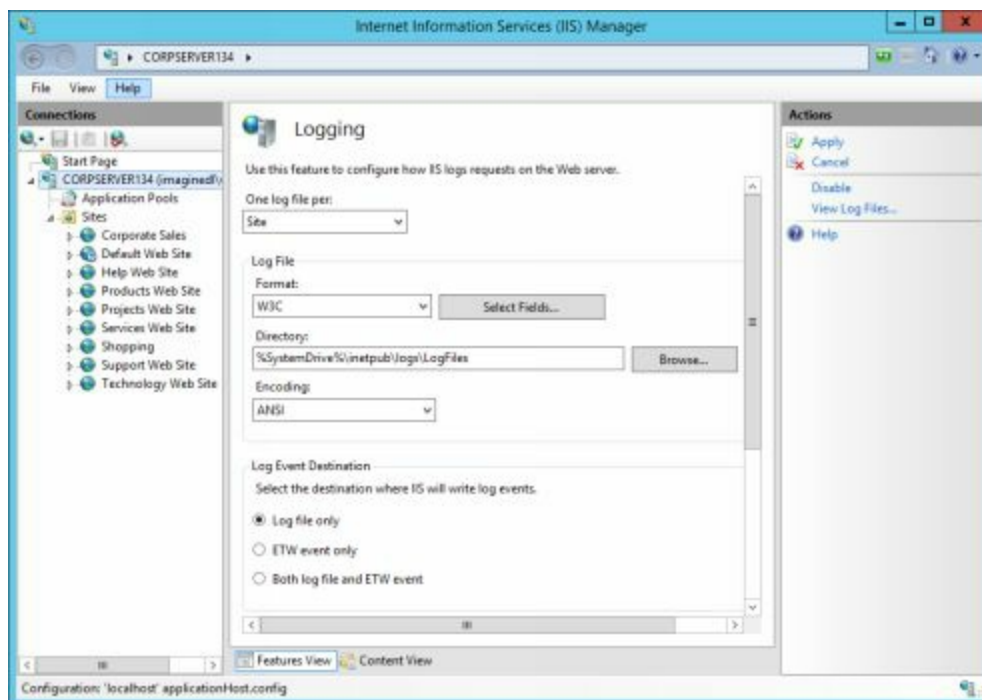
- When you change logging formats for a server, the format is used the next time you start the World Wide Web Publishing service. If you want the new logging format to be used immediately, you should restart the server process. To do this in IIS Manager, in the left pane, select the server node, and then in the Actions pane, click Restart.
- When you change logging formats for a site, the format is used the next time you start the World Wide Web Publishing service or the selected site. If you want the new logging format to be used immediately, you should restart the site. To do this in IIS Manager, in the left pane, select the site node, and then in the Actions pane, click Restart.

# Logging Essentials

An IIS server can use either per-server or per-site logging. As discussed previously, with per-server logging, IIS tracks requests for all websites configured on a server in a single log file. With per-site logging, IIS tracks requests for each website in separate log files.

You can enable logging and configure how IIS logs requests by completing the following steps:

1. In IIS Manager, select the node for the server you want to manage. If the server you want to use isn't listed, connect to it.
2. In the main pane, when you group by area, the Logging feature is listed under IIS. Double-click Logging to open this feature. You should now see the current top-level logging configuration.
3. If logging is currently disabled at the server level, all logging options are dimmed and cannot be selected. To enable logging, in the Actions pane, click Enable.
4. In the One Log File Per drop-down list, select the desired logging technique. If you want the server to use per-server logging, select Server. If you want the server to use per-site logging, select Site.
5. In the Encoding drop-down list, select the desired text encoding for logs formatted with the NSA, IIS, or W3C logging format. Choose either ANSI or UTF8.
6. Click Apply to save your settings.



With NCSA, IIS, and W3C logging, you can use ANSI or UTF-8 text encoding. ANSI supports standard English characters; UTF-8 supports standard English characters and non-English characters. Each IIS server has one text-encoding format, and that format is configured at the server level. All text-based log files created on the server use this encoding.

# Configuring the NCSA Common Logs

The NCSA common log file format is used with per-site logging only. You enable logging and configure the common log file format by completing the following steps:

1. In IIS Manager, navigate to the site you want to manage. In the main pane, when you group by area, the Logging feature is listed under IIS. Double-click Logging to open this feature.
2. If all logging options are dimmed and the server is configured for per site logging, you can click Enable in the Actions pane to enable logging for this site.
3. On the Format list, select NCSA as the log format. By default, log files are located in a subdirectory under %SystemDrive%\System32\inetpub\Logs\Logfiles. If you want to change the default logging directory, type the directory path in the Directory field, or click Browse to look for a directory that you want to use.
4. To configure logging during a specific time period, select Schedule, and then choose one of the following options:
  - **Hourly** IIS creates a new log each hour.
  - **Daily** IIS creates a new log daily at midnight.
  - **Weekly** IIS creates a new log file each Saturday at midnight.
  - **Monthly** IIS creates a new log file at midnight on the last day of the month.
5. To configure logging using an unlimited file size, select Do Not Create New Log Files. With this option, IIS doesn't end the log file automatically. You must manage the log file.
6. To set a maximum log file size in bytes, select Maximum File Size (In Bytes), and then type the desired maximum file size, such as **1024000**. When the log file reaches this size, a new log file is created.
7. Click Apply to save your settings. The service directory and log file are created automatically, if necessary. If IIS doesn't have Read/Write permission on the logging directory, an error is generated.

# Configuring Microsoft IIS Logs

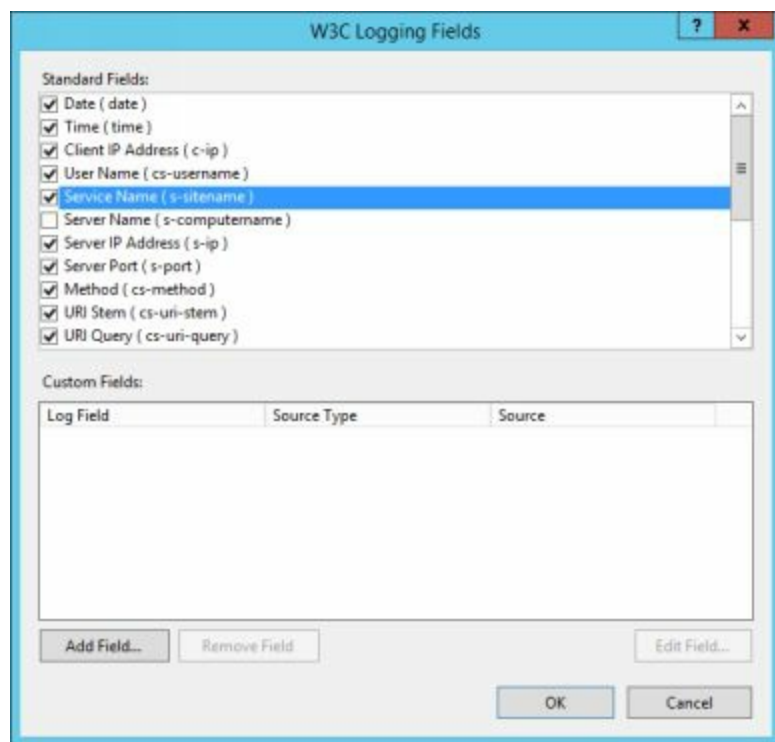
The Microsoft IIS log file format is used with per-site logging only. You enable logging and configure the IIS log file format by completing the following steps:

1. In IIS Manager, navigate to the site you want to manage. In the main pane, when you group by area, the Logging feature is listed under IIS. Double-click Logging to open this feature.
2. If all logging options are dimmed and the server is configured for per site logging, you can click Enable in the Actions pane to enable logging for this site.
3. On the Format list, select IIS as the log format. By default, log files are located in a subdirectory under %SystemDrive%\System32\inetpub\Logs\Logfiles. If you want to change the default logging directory, type the directory path in the Directory field, or click Browse to look for a directory that you want to use.
4. To configure logging using a specific time period, select Schedule, and then choose one of the following options:
  - **Hourly** IIS creates a new log each hour.
  - **Daily** IIS creates a new log daily at midnight.
  - **Weekly** IIS creates a new log file each Saturday at midnight.
  - **Monthly** IIS creates a new log file at midnight on the last day of the month.
5. To configure logging using an unlimited file size, select Do Not Create New Log Files. With this option, IIS doesn't end the log file automatically. You must manage the log file.
6. To set a maximum log file size in bytes, select Maximum File Size (In Bytes), and then type the desired maximum file size, such as **1024000**. When the log file reaches this size, a new log file is created.
7. Click Apply to save your settings. The service directory and log file are created automatically, if necessary. If IIS doesn't have Read/Write permission on the logging directory, an error is generated.

# Configuring W3C Extended Logs

The W3C extended log file format can be used with per site and per server logging. You enable logging and configure the W3C extended log file format by completing the following steps:

1. In IIS Manager, navigate to the node you want to manage. To configure centralized extended logging for a server, in the left pane, select the server node. To configure extended logging for a site, in the left pane, select the site node.
2. In the main pane, when you group by area, the Logging feature is listed under IIS. Double-click Logging to open this feature. If you selected a site node previously and the Actions pane displays a warning that the server is configured for per server logging, you must manage logging through the server node. If logging is otherwise disabled, you must click Enable in the Actions pane to turn on the logging feature.
3. Using the Format list, select W3C as the log format, and then click Select Fields.
4. In the W3C Logging Fields dialog box, select the extended properties that you want to log, and then click OK. The fields you'll want to track in most cases are: Date/Time, Client IP Address, Server IP Address, Service Name, Method, URI Stem, URI Query, Protocol Status, Protocol Substatus, Bytes Sent, Bytes Received, User Agent, Cookie, and Referer.



**NOTE** The more fields you track, the larger the log entries. Moreover, the larger log entries are, the longer it takes IIS to write them.

5. By default, log files are located in a subdirectory under %SystemDrive%\System32\Inetpub\Logs\Logfiles. If you want to change the default logging directory, type the directory path in the Directory field, or click Browse to look for a directory that you want to use.
6. To configure logging using a specific time period, select Schedule, and then choose one of the following options:
  - **Hourly** IIS creates a new log each hour.
  - **Daily** IIS creates a new log daily at midnight.
  - **Weekly** IIS creates a new log file each Saturday at midnight.
  - **Monthly** IIS creates a new log file at midnight on the last day of the month.
7. To configure logging using an unlimited file size, select Do Not Create New Log Files. With this option, IIS doesn't end the log file automatically. You must manage the log file.
8. To set a maximum log file size in bytes, select Maximum File Size (In Bytes), and then type the desired maximum file size, such as **1024000**. When the log file reaches this size, a new log file is created.
9. Click Apply to save your settings. The service directory and log file are created automatically, if necessary. If IIS doesn't have Read/Write permission on the logging directory, an error is generated.



# Configuring ODBC Logging

You can configure ODBC Logging as a type of custom logging with per site logging. Use the ODBC format when you want to write access information directly to an ODBC-compliant database. With ODBC logging, you'll need tracking software capable of reading from a database. Entries are compact, however, and data can be read much more quickly than from a standard log file.

To use ODBC logging, you must perform the following tasks:

1. Create a database using ODBC-compliant database software. As long as IIS can connect to the database using an ODBC connection, the database doesn't have to reside on the IIS server. Microsoft Office Access can be used for small to medium-sized sites with moderate traffic. For large or busy sites, use a more robust solution, such as SQL Server.
2. Within the database, create a table for logging access entries. This table must have the field names and data types listed in Table 13-7. You can use the Logtemp.sql script to create this table.
3. Next, create a Data Source Name (DSN) that IIS can use to connect to the database. You'll probably want to use a system DSN to establish the database connection. With SQL Server, you must specify the technique that should be used to verify the authenticity of the login identification (ID). If you use Microsoft Windows NT authentication, the account you specify when configuring IIS must have permission to write to the database. If you use SQL Server authentication, you can specify an SQL Server login ID and password to use.
4. Complete the process by enabling logging for the site and setting the active log format to ODBC logging. When you configure logging, you must specify the DSN name, the table name, and the logon information.

As discussed in the "HttpLoggingModule" section of the appendix, "Comprehensive IIS Module and Schema Reference," the configuration schema includes default values for ODBC logging. The default values are InternetDb for the database name, InternetLog for the table name, and InternetAdmin for the user name. When configuring DSNs, the database name is the same as the data source name. You can override these settings by assigning specific values at the appropriate configuration level.

The sections that follow describe how you can use SQL Server and IIS to configure ODBC logging. These sections assume a fair amount of knowledge of SQL Server and database administration.

# Creating a Logging Database and Table

You can use SQL Server as your logging server. To do this, you must create a database and configure a logging table. To create a database, complete the following steps:

1. In SQL Server Management Studio, use the Registered Servers view to select the Database Engine server type and the server you want to use.
2. Right-click the Databases folder, and then on the shortcut menu, select New Database. This opens the New Database dialog box.
3. On the General page, in the Database Name box, type **LoggingDB** as the database name.
4. Click OK. SQL Server creates the database.

Next, install the ODBC Logging role service for IIS if this role is not already installed. Then locate the Logtemp.sql script. This script is located in the %SystemRoot%\System32\Inetsrv directory on the IIS server. Edit the script so that it sets the table name you want to use for the site's log entries. For example, if you wanted to name the table HTTPLog, you would update the script as shown in the following listing:

```
use LoggingDB
create table HTTPLog (
ClientHost varchar(255),
username varchar(255),
LogTime datetime,
service varchar(255),
machine varchar(255),
serverip varchar(50),
processingtime int,
bytesrecvd int,
bytessent int,
servicestatus int,
win32status int,
operation varchar(255),
target varchar(255),
parameters varchar(255)
)
```

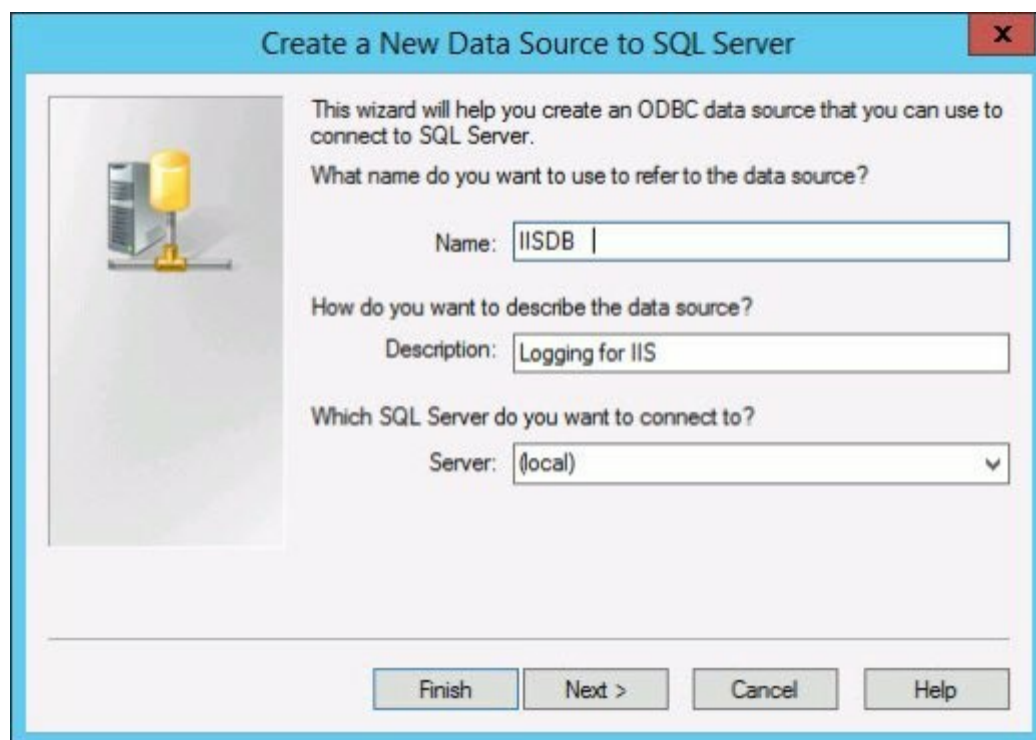
After you update the script, open a Query window in SQL Server Management Studio by selecting New Query on the toolbar. In the Query view, you can access scripts by clicking the Open File button on the toolbar and then typing the location of the script. Alternately, you can copy and paste the script into the newly opened Query view. Run the script by clicking Execute. When the script completes, a new table should be created in the LoggingDB database. If necessary, ensure that you connect to the server running SQL Server using an account with database administrator privileges.

# Creating a DSN for SQL Server

Once you create the logging database and the input table, you can configure IIS to connect to the database. IIS connects to the database using a DSN. You must create the DSN on the IIS server.

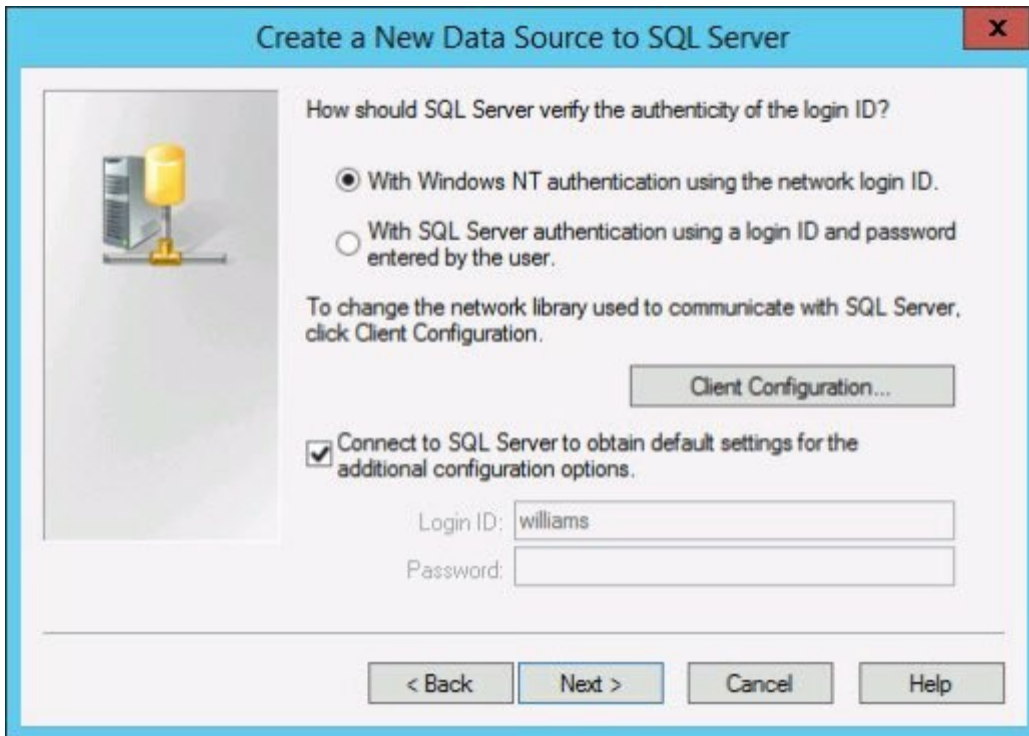
To create a DSN, complete the following steps:

1. On the Tools menu in Server Manager, start ODBC Data Sources 64-bit.
2. On the System DSN tab, click Add. The Create New Data Source dialog box appears.
3. On the Driver list, select SQL Server, and then click Finish. You should now see the Create A New Data Source To SQL Server dialog box.



4. In the Name field, type the name of the DSN, such as **IISDB**.
5. In the Server field, type the name of the SQL Server to which you want to connect, or select (Local) if SQL Server is running on the same hardware as IIS.
6. Next, specify the technique that should be used to verify the authenticity of the login ID. If you use Windows NT authentication, the account you specify when configuring IIS must have permission to write to the logging database. If you use SQL Server authentication, you can specify an SQL Server login ID and password to use.
7. Click Next and then click Finish to complete the process. If Windows is unable to establish a connection to the database, you might need to recheck the information

you've entered for correctness. You might also need to confirm that the account you are using has the appropriate permissions in the database.



**Create a New Data Source to SQL Server**

How should SQL Server verify the authenticity of the login ID?

☒ With Windows NT authentication using the network login ID.

☐ With SQL Server authentication using a login ID and password entered by the user.

To change the network library used to communicate with SQL Server, click Client Configuration.

Client Configuration...

☒ Connect to SQL Server to obtain default settings for the additional configuration options.

Login ID: williams

Password:

< Back   Next >   Cancel   Help

# Configuring ODBC Logging in IIS

ODBC is a type of custom logging that can be configured only when an IIS server is using per-site logging. To complete the configuration process, you must enable and configure ODBC logging in IIS by following these steps:

1. As necessary, use Server Manager to install and enable the Custom Logging and ODBC Logging role services for the IIS server.
2. In IIS Manager, navigate to the server or site you want to manage. In the main pane, when you group by area, the Logging feature is listed under IIS. Double-click Logging to open this feature.
3. If all logging options are dimmed and the server is configured for per site logging, you can click Enable in the Actions pane to enable logging for this site.
4. On the Format list, select Custom as the log format. No additional logging options can be selected in IIS Manager.
5. Edit the configuration file for the site that should use ODBC logging. Use the attributes of the `odbcLogging` element to configure ODBC logging.

# Configuring Centralized Binary Logging

Before you implement centralized binary logging, there are many things you should consider, including how using this format will affect the server and what tools you will use to read the raw binary logs. After planning, you should set up a test installation and determine if it is feasible to switch to centralized binary logging and obtain the information your organization needs from the raw binary log files. Only when you are certain that this format will work for you should you enable binary logging.

When you are ready to implement centralized binary logging, complete the following steps to enable logging and configure the W3C extended log file format:

1. In IIS Manager, select the server you want to manage. In the main pane, double-click Logging to open this feature.
2. If logging is currently disabled at the server level, all logging options are dimmed and cannot be selected. You can enable logging by clicking Enable in the Actions pane.
3. To use per server logging, in the One Log File Per drop-down list, select Server.
4. By default, log files are located in a subdirectory under %SystemDrive%\System32\Inetpub\Logs\Logfiles. If you want to change the default logging directory, type the directory path in the Directory field, or click Browse to look for a directory that you want to use.
5. To configure logging using a specific time period, select Schedule, and then choose one of the following options:
  - **Hourly** IIS creates a new log each hour.
  - **Daily** IIS creates a new log daily at midnight.
  - **Weekly** IIS creates a new log file each Saturday at midnight.
  - **Monthly** IIS creates a new log file at midnight on the last day of the month.
6. To configure logging using an unlimited file size, select Do Not Create New Log Files. With this option, IIS doesn't end the log file automatically. You must manage the log file.
7. To set a maximum log file size in bytes, select Maximum File Size (In Bytes), and then type the desired maximum file size, such as **1024000**. When the log file reaches this size, a new log file is created.
8. Click Apply to save your settings. The service directory and log file are created automatically, if necessary. If IIS doesn't have Read/Write permission on the logging directory, an error is generated.

# Disabling Logging

If you don't plan to generate reports from access logs, you might not want to log user access to the sites on a server. In this case, you can disable logging for the server. You can disable logging for the server and all sites by completing the following steps:

1. In IIS Manager, select the node for the server you want to manage. If the server you want to use isn't listed, connect to it.
2. In the main pane, when you group by area, the Logging feature is listed under IIS. Double-click Logging to open this feature.
3. If logging is currently enabled at the server level, logging options are available and can be selected. You can disable logging by clicking Disable in the Actions pane.

You can enable or disable logging for individual sites by completing the following steps:

1. In IIS Manager, select the node for the site you want to manage. In the main pane, when you group by area, the Logging feature is listed under IIS. Double-click Logging to open this feature.
2. If logging is currently enabled for the site and you want to disable it, in the Actions pane, click Disable. If logging is currently disabled for the site and you want to enable it, in the Actions pane, click Enable.

<p><b>NOTE</b> If you've configured per-server logging, you cannot manage or enable logging at the site level. You can, however, disable logging for individual sites.</p>
--

This eBook was posted by AlenMiler!

Many Interesting eBooks You can also Download from my Blog: [Click Here!](#)

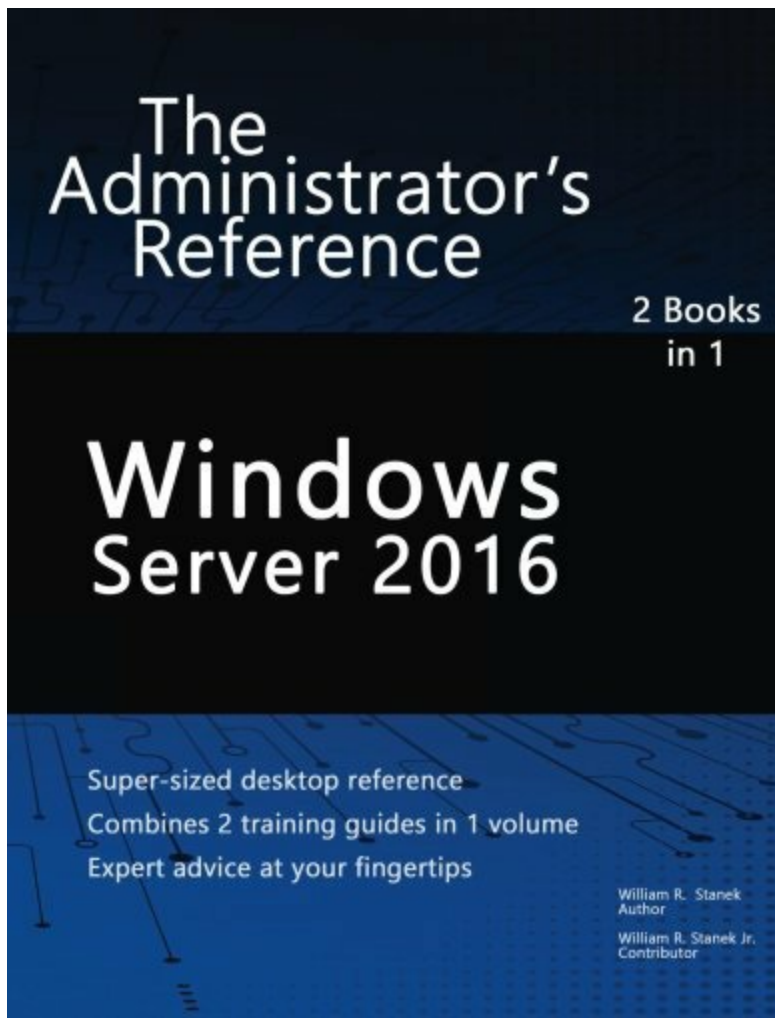
Mirror: [Click Here!](#)





Find William on Twitter at <http://www.twitter.com/WilliamStanek> and on Facebook at <http://www.facebook.com/William.Stanek.Author>.

Connect with Will by visiting him on LinkedIn @ <http://linkedin.com/in/will-stanek/>.

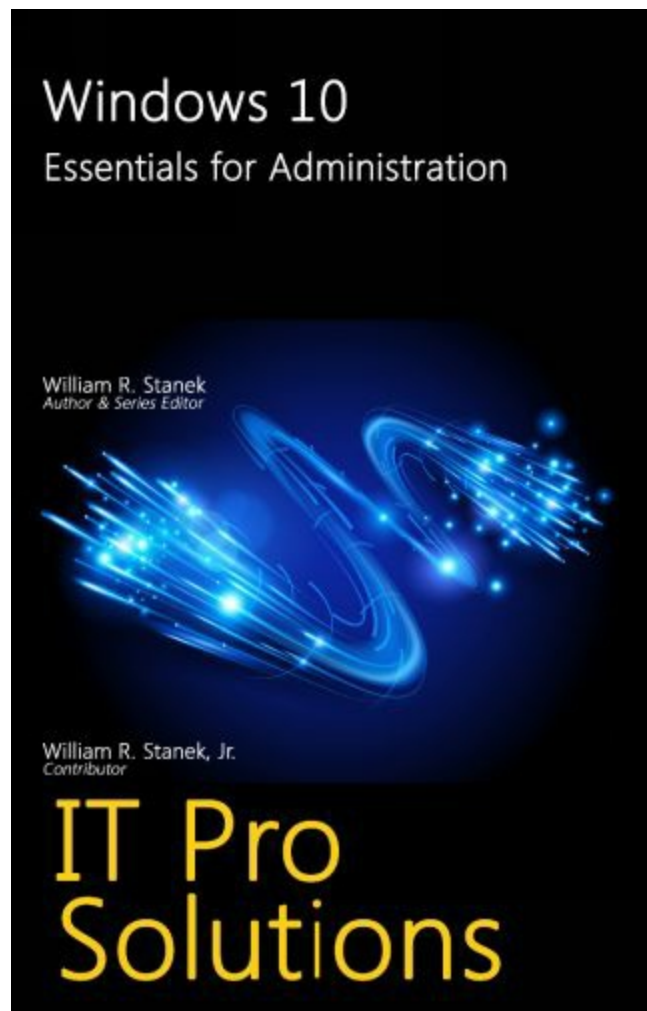


Thank you for purchasing this book. If you found this book to be useful, helpful or informative, raise your voice and support this work by sharing online.

Unsure how to share? Here are some tips:

- [Blog about the book](#)
- [Write a review at your favorite online store](#)
- [Post about the book on Facebook or elsewhere](#)
- [Tweet about the book](#)

Stay in touch!



This eBook was posted by AlenMiler!

Many Interesting eBooks You can also Download from my Blog:

[Click Here!](#)

Mirror:

[Click Here!](#)