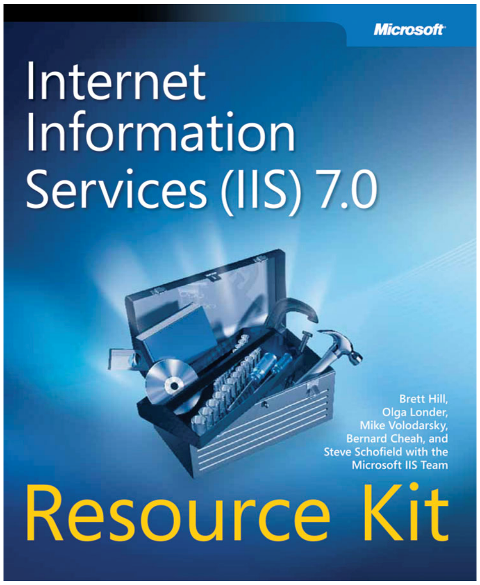


Internet Information Services (IIS) 7.0 Resource Kit

Mike Volodarsky, Olga Londer, Brett Hill, Bernard Cheah, and Steve Schofield with the Microsoft IIS Team

PREVIEW CONTENT This excerpt contains uncorrected manuscript from an upcoming Microsoft Press title, for early preview, and is subject to change prior to release. This excerpt is from *Internet Information Services (IIS) 7.0 Resource Kit* from Microsoft Press (ISBN 978-0-7357356-2441-2, copyright 2008 Mike Volodarsky, Olga Londer, Brett Hill, Bernard Cheah, and Steve Schofield, and Microsoft Corporation, all rights reserved), and is provided without any express, statutory, or implied warranties

To learn more about this book, visit Microsoft Learning at
http://www.microsoft.com/MSPress/books/9550.aspx

Microsoft Press

978-0-7356-2441-2

Table of Contents

Part 4 – Troubleshooting/Performance

Chapter 15 – Logging

Part 4

Troubleshooting and Performance

Chapter 15

Logging

Though not technology's most fascinating topic, Web server log files are extraordinarily important. They are a core resource used, for example, as the basis for billing, reliability, performance, compliance, and forensics. This chapter discusses IIS logging and related features in Microsoft IIS 7.0.

What's New?

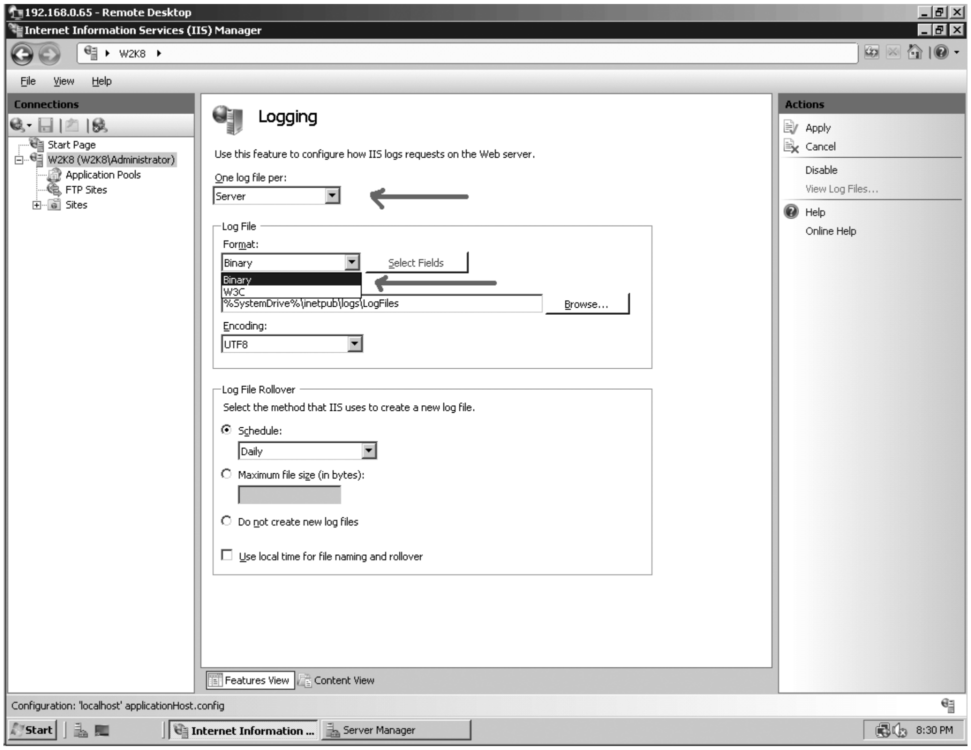
In IIS 7.0, as in IIS 6.0, log files are handled by the HTTP.sys kernel mode device driver. No user code runs in this service because HTTP.sys runs in kernel mode. In general, not a lot has changed related to logging, but a few differences as well as new opportunities are notable. You'll find that many of the enhancements to logging introduced as late as Windows Server 2003 Service Pack 1 (SP1) are included in IIS 7.0. For example, you can use World Wide Web Consortium (W3C) centralized logging and binary logging; you can use standard log formats such as W3C extended, National Center for Supercomputing Applications (NCSA), and IIS; and you can set the custom logging option.

One of the coolest features in IIS 7.0 is its modular architecture and the Integrated Pipeline. (The Integrated Pipeline is covered in depth in Chapter XX.) Logging greatly benefits from the flexibility provided by the modularity in IIS 7.0 and the Integrated Pipeline because you can write your own logging module and inject it into the pipeline. Your custom module can capture just the information needed for your application. We'll demonstrate how to implement your own HTTP SQL logging module later in this chapter.

IIS 7.0 incorporates several changes pertinent to logging. You use the IIS Manager to configure logging-related settings; the new configuration system is based on XML; there are a number of new logging configuration options and a new set of status codes; and IIS 7.0 provides logging for a new service that enables remote administration of an IIS 7.0 server.

The IIS Manager

IIS 7.0 introduces a completely new user interface, the IIS Manager. The IIS Manager makes it easier to browse and make changes to IIS settings, including log file settings. For example, if you wanted to implement the centralized logging in IIS 6.0, you had to use ADSUtil.vbs. Now, the Centralized Logging option is exposed in the IIS Manager, as shown in Figure 15-1.



Logging

Use this feature to configure how IIS logs requests on the Web server.

One log file per: Server

Log File

Format: Binary

Binary
W3C

%SystemDrive%\inetpub\logs\LogFiles

Encoding: UTF8

Log File Rollover

Select the method that IIS uses to create a new log file.

● Schedule: Daily

○ Maximum file size (in bytes):

○ Do not create new log files

☐ Use local time for file naming and rollover

Configuration: 'localhost' applicationHost.config

Figure 15-1 The Centralized Logging option in the IIS Manager

Chapter 7, "Using The IIS Manager," provides an in-depth look at the IIS Manager.

The XML-Based Logging Schema

IIS 7.0 uses a new configuration system that is XML-based and is very similar to ASP.NET. Each configuration section is defined in XML schema files located in *systemRoot*\system32\inetsrv\config\schema. Details on the configuration sections are covered in Chapter 4, "Configuration." Because information is defined in XML files, it is easy to determine what attributes, elements, and enums are used. The schema for IIS 7.0 contains a list of all the configurable options, so looking in the schema fie is a quick way to identify all the configurable settings for any feature, including logging.

The following listing is from the system.applicationHost/log section that is located in *systemRoot*\system32\inetsrv\config\schema\IIS_Schema.xml. (Some long lines have been split to fit on the printed page.) As you can see, the XML clearly defines the names and data types associated with each item.

```
<sectionSchema name="system.applicationHost/log">
    <attribute name="logInUTF8" type="bool" defaultValue="true" />
    <attribute name="centralLogFileMode" type="enum" defaultValue="Site" >
      <enum name="Site" value="0"/>
      <enum name="CentralBinary" value="1"/>
      <enum name="CentralW3C" value="2"/>
    </attribute>
    <element name="centralBinaryLogFile">
      <attribute name="enabled" type="bool" defaultValue="false" />
      <attribute name="directory" type="string" expanded="true"
          defaultValue="%SystemDrive%\inetpub\logs\LogFiles" />
      <attribute name="period" type="enum" defaultValue="Daily">
        <enum name="Hourly" value="4"/>
        <enum name="Daily" value="1"/>
        <enum name="Weekly" value="2"/>
        <enum name="Monthly" value="3"/>
        <enum name="MaxSize" value="0"/>
      </attribute>
      <attribute name="truncateSize" type="int64" defaultValue="20971520"
          validationType="integerRange"
          validationParameter="1048576,4294967295" />
      <attribute name="localTimeRollover" type="bool"
          defaultValue="false"/>
    </element>
    <element name="centralW3CLogFile">
      <attribute name="enabled" type="bool" defaultValue="true" />
      <attribute name="directory" type="string" expanded="true"
          defaultValue="%SystemDrive%\inetpub\logs\LogFiles"
          validationType="nonEmptyString" />
      <attribute name="period" type="enum" defaultValue="Daily">
        <enum name="Hourly" value="4"/>
        <enum name="Daily" value="1"/>
        <enum name="Weekly" value="2"/>
        <enum name="Monthly" value="3"/>
        <enum name="MaxSize" value="0"/>
      </attribute>
      <attribute name="truncateSize" type="int64" defaultValue="20971520"
          validationType="integerRange"
          validationParameter="1048576,4294967295" />
      <attribute name="localTimeRollover" type="bool"
          defaultValue="false"/>
      <attribute name="logExtFileFlags" type="flags"
          defaultValue="Date, Time, ClientIP, UserName, SiteName, ServerIP,
                        Method, UriStem, UriQuery, HttpStatus, Win32Status,
                        ServerPort, UserAgent, HttpSubStatus">
        <flag name="Date" value="1"/>
        <flag name="Time" value="2"/>
        <flag name="ClientIP" value="4"/>
        <flag name="UserName" value="8"/>
        <flag name="SiteName" value="16"/>
        <flag name="ComputerName" value="32"/>
        <flag name="ServerIP" value="64"/>
        <flag name="Method" value="128"/>
        <flag name="UriStem" value="256"/>
        <flag name="UriQuery" value="512"/>
        <flag name="HttpStatus" value="1024"/>
        <flag name="Win32Status" value="2048"/>
        <flag name="BytesSent" value="4096"/>
        <flag name="BytesRecv" value="8192"/>
        <flag name="TimeTaken" value="16384"/>
        <flag name="ServerPort" value="32768"/>
        <flag name="UserAgent" value="65536"/>
        <flag name="Cookie" value="131072"/>
        <flag name="Referer" value="262144"/>
        <flag name="ProtocolVersion" value="524288"/>
        <flag name="Host" value="1048576"/>
        <flag name="HttpSubStatus" value="2097152"/>
      </attribute>
    </element>
  </sectionSchema>
```

Centralized Logging Configuration Options

Here is the logging section defined in the applicationHost.config file that controls Centralized Logging options. You can change this so that your files are stored on another drive or volume. You can enable options you want and disable whatever options you do not need.

```
<log>
  <centralBinaryLogFile enabled="true"
      directory="%SystemDrive%\inetpub\logs\LogFiles" />
  <centralW3CLogFile enabled="true"
      directory="%SystemDrive%\inetpub\logs\LogFiles" />
</log>
```

SiteDefaults Configuration Options

The SiteDefaults section in the applicationHost.config file, shown in the following listing, controls the logging settings that are used when creating new sites. You can configure two options: the format of the log file and the location in which Failed Request tracing files are stored. Failed Request tracing is covered in depth in Chapter XXX.

```
<siteDefaults>
    <logFile logFormat="W3C"
        directory="%SystemDrive%\inetpub\logs\LogFiles" />
    <traceFailedRequestsLogging
        directory="%SystemDrive%\inetpub\logs\FailedReqLogFiles" />
</siteDefaults>
```

Disable HTTP Logging Configuration Options

In some cases, an IIS administrator does not require log files. If you would like to turn off httpLogging at the server level, you can disable logging in the IIS Manager. You can also disable logging at the site level. You might wonder why these options are available. It's so that you can disable logging on your test or development machines to reduce the disk space that unnecessary files use.

You should evaluate your options before disabling HTTP logging. Check with your business or legal department to ascertain what your company's logging requirements and policies are. The default value for this setting, as shown here, is false.

```
<httpLogging dontLog="false" />
```

Default Log File Location

One of the most significant changes in IIS 7.0 is that the folder where IIS stores WWW logs has been changed to *SystemDrive*\Inetpub\Logs\LogFiles (typically C:\Inetpub\Logs\LogFiles). This means that by default in IIS 7.0, all log files are stored under a single folder. Note, however, that log files for the legacy built-in File Transfer Protocol (FTP) and Simple Mail Transfer Protocol (SMTP) services are still located in *windir*\System32\Logfiles. You can manage these files by using the IIS Manager 6.0, an MMC console that is installed when you install the legacy FTP service or the SMTP service.

[NOTE]The new FTP Publishing Service for IIS 7.0 stores its log files in *windir*\Inetpub\Logs by default. You need to download and install this add-on because it does not ship with IIS 7.0. Both an x86 version and an AMD version (for x64-based machines) are available at: *www.iis.net/downloads/default.aspx?tabid=34&g=6&i=1454*

Default UTF-8 Encoding

By default, IIS 7.0 stores log files by using UTF-8 encoding. This changes the default file naming convention so that the files start with u_ (for example, u_exYYMMDD.log). Here is the portion of the IIS_Schema.xml file that sets the UTF-8 encoding option. The default setting is true.

```
  <sectionSchema name="system.applicationHost/log">
    <attribute name="logInUTF8" type="bool" defaultValue="true" />
...
  </sectionSchema>
```

UTF-8 encoding allows for single-byte and multi-byte characters in one string. This encoding enables you to read text-based logs (for example, logs that use W3C Extended, IIS, and NCSA Common formats) in a language other than English. Additionally, if your Web server serves URLs in a language or dialect other than the one supported by the server's default code page.

IIS does not support the UTF-8 format for the built-in FTP Publishing Service log files. UTF-8 encoding is available in IIS 6.0, but it is not enabled by default. If you do not want to have your logs use UTF-8 encoding, you can use ANSI as the format.

New Status Codes

There are new status codes introduced for HTTP and FTP. The additional error codes provide more details about events and better descriptions of how to fix errors, with suggestions on what to look for or what procedures to run. Appendix XXX provides a complete list of all new status codes.

Management Service

IIS 7.0 introduces Management Service, which enables computer and domain administrators to remotely manage a machine using the IIS Manager. The Management Service also enables nonadministrators to control sites and various applications using the IIS Manager from a workstation.

This service has its own logs that are used to track information related to the Management Service. This service is not installed by default. If you install and enable this service, the logs will be saved in *SystemDrive*\Inetpub\logs\WMSvc.

From a logging perspective, you should make sure this option is enabled. The logs can help you audit and troubleshoot issues when clients are connecting to your server. The Management Service is discussed in depth in Chapter XXX.

Log File Formats That Have Not Changed

IIS 7.0 supports all the common logging formats that are available in prior versions of IIS. There have been no changes in IIS 7.0 to the following log file formats:

- Microsoft IIS
- NCSA
- Open Database Connectivity (ODBC)
- W3Svc extended

[NOTE]For descriptions, further discussion, and examples of these log formats, see: *msdn2.microsoft.com/en-us/library/ms525807.aspx*

Centralized Logging

Centralized logging in IIS 7.0 operates the same way it does in IIS 6.0. However, you can now configure this option in the IIS 7.0 Manager. To access this feature, go to Administrative Tools > Internet Information Services (IIS) Manager. Click on the Computer Name and locate the Logging option listed in the IIS section.

Using the Logging option can reduce administrative costs because only one IIS log file is being maintained. If you use binary logging, the log can be stored in a much smaller file than the equivalent text log file.

W3C Centralized Logging Format

W3C centralized logging was first introduced in Windows Server 2003 SP1. W3C centralized logging is a *server*-level setting. When you enable this feature on a server, all Web sites on that server are configured to write log data to a central log file. Data is stored in the log file using the W3C Extended log file format. You can enable this setting through the IIS 7.0 Manager or by using AppCMD. If you use W3C centralized logging, you can view the log file with a text editor such as Notepad.

[Note]W3C centralized logging uses the W3C Extended log format, which includes the following four fields: *HostHeader, Cookie, UserAgent*, and *Referrer*. These fields are not available in centralized binary logging.

Centralized Binary Logging Format

Centralized binary logging is essentially the same as W3C centralized logging, except that the log file uses a proprietary, binary format. Because the resulting file is binary, it is smaller than an equivalent text file so that you can conserve disk space. It cannot be read with a text editor and requires parsing to produce useful information. However, this is easier than you might think when you use the Log Parser tool, which reads the centralized binary file format natively. The Log Parser tool is briefly discussed later in this chapter.

[Important]The built-in FTP and SMTP services do not support W3C centralized logging.

Remote Logging

IIS 7.0 supports writing log files to a network share. This option enables you to have your log files stored in real time to a remote computer. For example, suppose that you have a Web farm configured for logging to a central location. The back end log server could be a server running DFS (Distributed file system). DFS can provide multiple benefits including a central location to collect your log files and automatic replication of your logs to multiple locations. Having such a primary collection point can make handling your reporting processes much easier.

[Important] When you set up your remote logging environment, make sure the A and PTR DNS records are set up so that authentication and resolution happens correctly. This can help avoid problems such as Kerberos authentication errors when HTTP.sys is trying to write log files.

You can use either the IIS 7.0 Manager or AppCMD to set up Universal Naming Convention (UNC) remote logging.

Setting Up Remote Logging by Using the IIS 7.0 Manager

Here are the steps to enable remote logging by using the IIS 7.0 Manager.

1. Create a directory called IISLogs on the remote server that will store the log files. This machine is typically in the same domain as the Web servers. If the remote server is not in the same domain or is a stand-alone machine, you can create a local account with the same user ID and password as is being used to connect from the Web server.

   [Note]If your remote server will be in a different domain, you can set up a Null Session to support remote logging. The following TechNet article describes the steps necessary to configure this scenario: *www.microsoft.com/technet/prodtechnol/WindowsServer2003/Library/IIS/a4939515-b651-4ee0-b327-867b31fd8c9a.mspx?mfr=true*

2. Share the IISLogs folder you created in the previous step. Change the share permissions to at a minimum enable both the remote *machine accounts* Administrators group and the account writing the log files *full control*. Change the NTFS file system (NTFS) permissions so that the remote *machine accounts* Administrators have *full control* and the account writing the log files has *modify* permissions. This example assumes that you are using the NETWORK SERVICE as your application pool account and that the remote server and Web server are in the same domain.

[Note]When the NETWORK SERVICE account accesses a remote resource, it uses the machine account stored in Microsoft Active Directory directory service as the actual account accessing the log folder.

3. In the IIS 7.0 Manager, navigate to your Web site and type in the UNC path to the server. To do so, go to Administrative Tools > Internet Information (IIS) Manager. Select the computer name in the leftmost column and then double-click the Logging icon in the IIS Section. Type the path to the share in the Directory text box by using the syntax \\ServerName\ShareName, as shown in Figure 15-2.

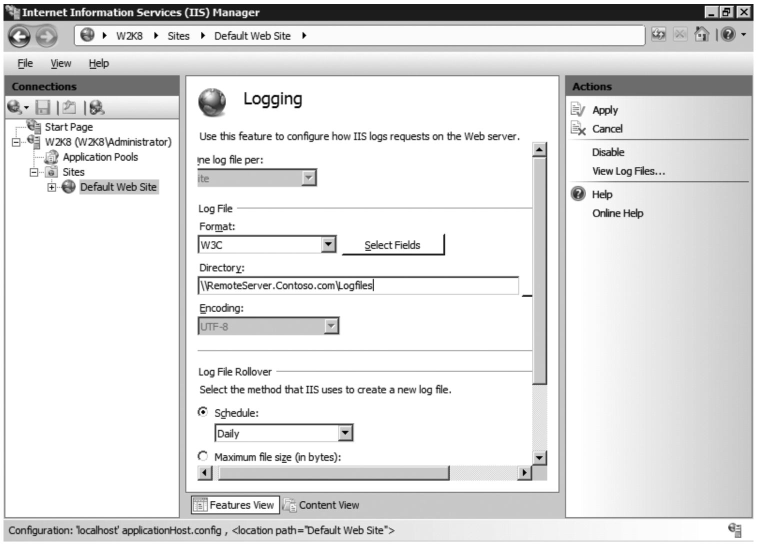
[Note]You can also use the syntax \\\\FQDN\ShareName to specify the logging path, but you might run into issues if you try to use the syntax \\\\IPAddress\ShareName to specify the path. The \\\\IPAddress\\ShareName syntax can cause an authentication issue that prevents the log files from being created. The following is an example of an error generated when trying to use an IP Address when remote logging is enabled.

```
Microsoft-Windows-HttpService , LogFileCreateFailed ,
49,          0,         16,          2,         59,          9,
0x0000000000000800, 0x00000004, 0x000005AC,                    0,
             ,                     ,
{00000000-0000-0000-0000-000000000000},                              ,
128277049412643098,         220,         0, 0xC0000022,
"ResponseLogging ", "Site ", "W3C ",
"\dosdevices\UNC\192.168.0.125\UncLogFiles\W3SVC1\u_ex070630.log",        0
```

4. Click Apply.

5. Browse a Web page in your site.

6. Open a command prompt by using elevated credentials and type **netsh http flush logbuffer**. If this is the first time entries have been logged, HTTP.sys will create the folder and a log file. Open the log file in Notepad to confirm your example entries have been logged.



Use this feature to configure how IIS logs requests on the Web server.

ıne log file per: ite

Format: W3C

Directory: \\\\RemoteServer.Contoso.com\\Logfiles

Encoding: UTF-8

Select the method that IIS uses to create a new log file.

● Schedule: Daily

Configuration: 'localhost' applicationHost.config , <location path="Default Web Site">

Figure 15-2 Configuring the default Web site to enable remote logging

Setting Up Remote Logging by Using AppCMD

You can also use AppCMD to update the logfile directory for a specific Web site. The syntax for configuring UNC remote logging using AppCMD is shown here. (The line has been split to fit it on the printed page.)

```
//AppCMD to set the log directory path for Default Web Site
appcmd set sites "Default Web Site"
    -logFile.directory:\\RemoteServerCMD.Contoso.com\LogFiles
```

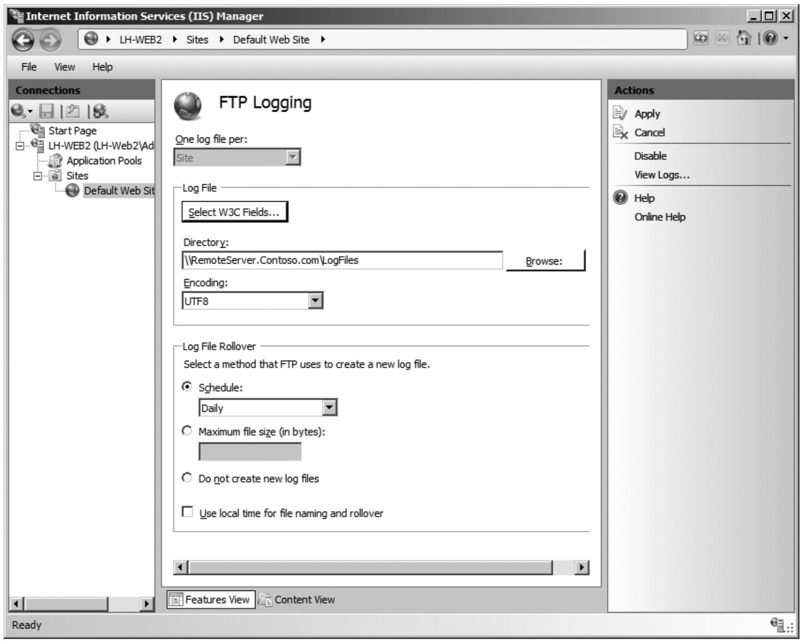
> [Note]To automate configuring remote logging, you could put this example into a script to which you can pass variables.

Executing this command results in the following output.

```
SITE object "Default Web Site" changed
```

Remote Logging Using the FTP 7.0 Publishing Service

The FTP 7.0 Publishing Service is an out-of-band add-on that is meant to replace the built-in FTP service. The FTP 7.0 Publishing Service supports logs stored on a remote computer, which can enhance your ability to track down security breaches. Imagine a particular machine is compromised, but you have your logs stored on a remote system. When the infiltrator tries to cover her tracks by deleting the local log files, those log files will be unavailable because they are stored on a remote share. If your remote share uses DFS, the log files can even be replicated to multiple locations. Remote logging with replication can help in your forensic efforts. To configure the FTP logs to be stored on a remote server, you just have to configure your remote server that houses your logs files the same as you would configure a Web server. Figure 15-3 shows the FTP 7.0 Publishing Service configured to log remotely.



One log file per: Site

Directory: \\RemoteServer.Contoso.com\LogFiles

Encoding: UTF8

Select a method that FTP uses to create a new log file.

● Schedule: Daily

○ Maximum file size (in bytes):

○ Do not create new log files

☐ Use local time for file naming and rollover

Figure 15-3 FTP 7.0 Publishing Service configured to store log files on a remote computer

Custom Logging

The modular architecture of IIS 7.0 enables you to implement your own logging modules or extend or replace existing logging options. Your module can be implemented directly into the request pipeline. Your logging module can be either a native module or a module written using managed code. You can use any .NET language such as C# or Microsoft Visual Basic.NET.

Configuring IIS Logging

IIS 7.0 provides multiple ways to configure and administer your Web server, and that includes configuring your log settings. This section covers how to use the built-in graphical user interface (GUI) as well as command line tools to configure log settings. You'll learn how to use the IIS Manager, AppCMD, and Windows Powershell.

IIS Manager (7.0)

The IIS 7.0 Manager is a completely rewritten tool that administrators can use to manage their Web servers. The intuitive interface enables you to quickly review and adjust all settings, including those that apply to log files. To access the Logging section of the IIS Manager, follow this procedure.

1. Go to Administrative Tools, Internet Information Services (IIS) and select the server name. Figure 15-4 shows the icon for the global Logging section when it is selected.

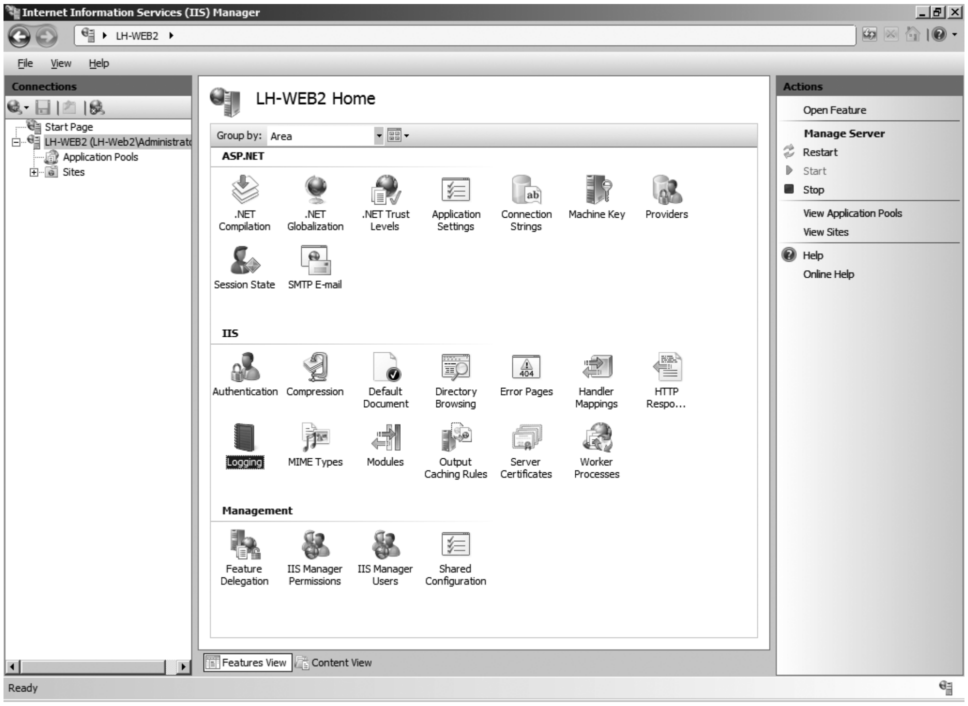
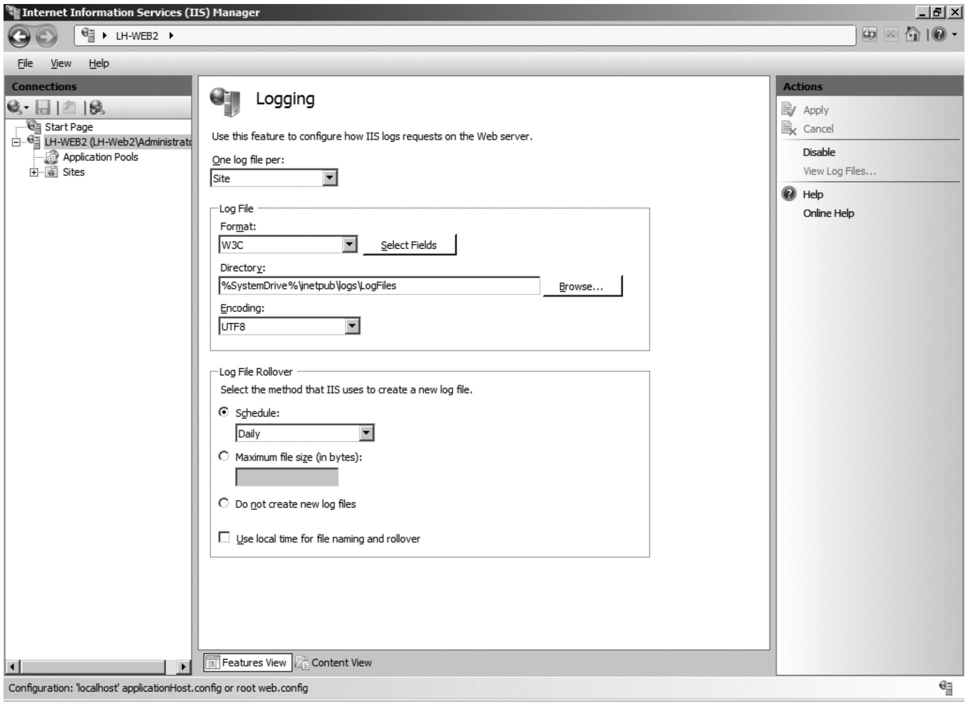


Figure 15-4 The icon for the global Logging section selected in the IIS Manager

2. Double-click the Logging icon to view the interface through which you can administer logging settings for the server.

   The default settings are shown in Figure 15-5. Because is the server node selected in the tree in the left pane, these settings are inherited by all Web sites configured on the server.



Logging

Use this feature to configure how IIS logs requests on the Web server.

One log file per: Site

Log File

Format: W3C

Directory: %SystemDrive%\inetpub\logs\LogFiles

Encoding: UTF8

Log File Rollover

Select the method that IIS uses to create a new log file.

● Schedule: Daily

○ Maximum file size (in bytes):

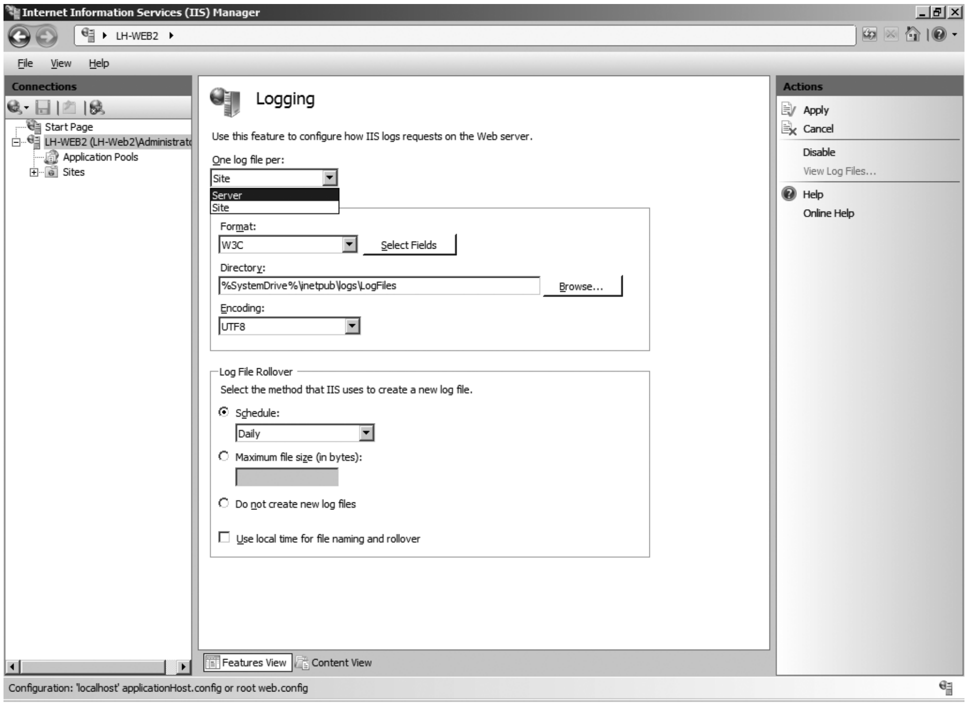
○ Do not create new log files

☐ Use local time for file naming and rollover

Configuration: 'localhost' applicationHost.config or root web.config

Figure 15-5 Default global settings

3. To make changes, select the appropriate drop-down box and select the option you want. For example, to change the server from site-level logging (creating one log file per site) to server-level logging (creating one log file per server), select Server in the One Log File Per drop-down list, as shown in Figure 15-6.



Use this feature to configure how IIS logs requests on the Web server.

One log file per: Site

Format: W3C

Directory: %SystemDrive%\inetpub\logs\LogFiles

Encoding: UTF8

Log File Rollover

Select the method that IIS uses to create a new log file.

● Schedule: Daily

○ Maximum file size (in bytes):

○ Do not create new log files

☐ Use local time for file naming and rollover

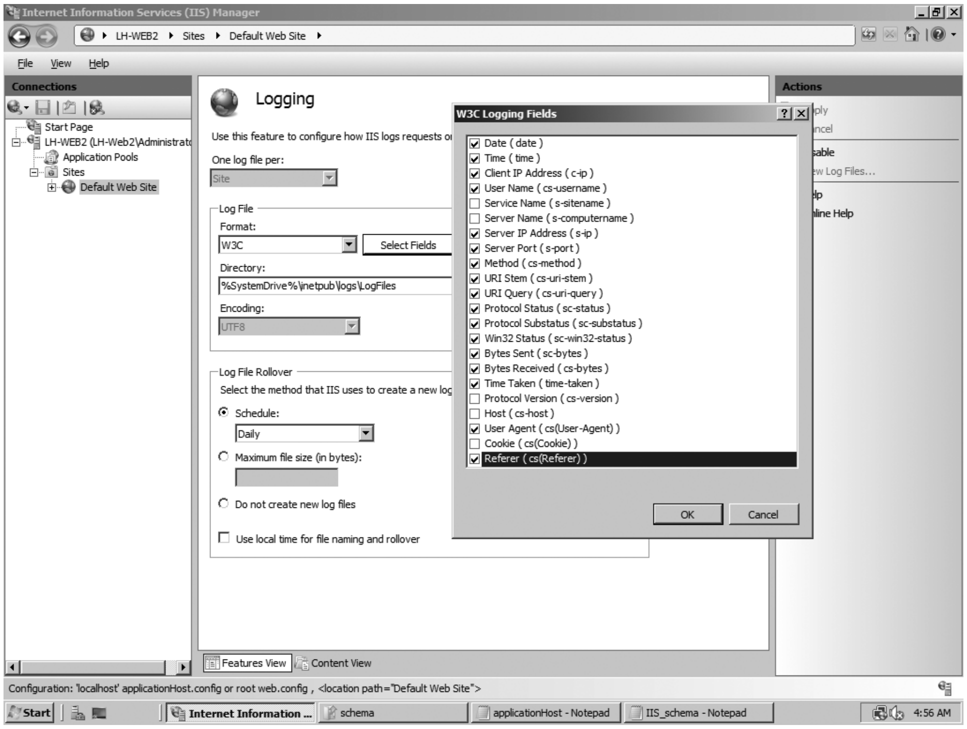
Configuration: 'localhost' applicationHost.config or root web.config

Figure 15-6 Go to the IIS Manager to change logging from site-level logging to server-level logging

In IIS 6.0, you need to write a script to change the *CentralW3CLoggingEnabled* metabase property. This is one example of how the IIS Manager is more powerful and easier to use. (For more information about this metabase attribute, visit the following link at TechNet: *www.microsoft.com/technet/prodtechnol/WindowsServer2003/Library/IIS/6d593c76-94a2-4360-b93d-8ec2bc384f5a.mspx?mfr=true*)

> [Note]When you configure IIS 7.0 to use server-level logging, the Binary format is selected by default. To have your server-level log use W3C extended logging, simply select W3C in the Format drop-down list.

IIS 7.0 also enables you to make changes on individual Web sites. For example, you can click the Select Fields button to adjust which options are logged for a specific Web site, as shown in Figure 15-7. In this figure, the Bytes Sent (sc-bytes), Bytes Received (cs-bytes), Time Taken (time-taken) and Referrer (cs(Referer)) options have been selected. You can also adjust the log Directory setting, the Log File Rollover setting, and the Use Local Time For File Naming And Rollover setting.



Logging

Use this feature to configure how IIS logs requests o

One log file per: Site

Log File

Format: W3C Select Fields

Directory: %SystemDrive%\inetpub\logs\LogFiles

Encoding: UTF8

Log File Rollover

Select the method that IIS uses to create a new log

● Schedule: Daily

○ Maximum file size (in bytes):

○ Do not create new log files

☐ Use local time for file naming and rollover

W3C Logging Fields

☑ Date (date)
☑ Time (time)
☑ Client IP Address (c-ip)
☑ User Name (cs-username)
☐ Service Name (s-sitename)
☐ Server Name (s-computername)
☑ Server IP Address (s-ip)
☑ Server Port (s-port)
☑ Method (cs-method)
☑ URI Stem (cs-uri-stem)
☑ URI Query (cs-uri-query)
☑ Protocol Status (sc-status)
☑ Protocol Substatus (sc-substatus)
☑ Win32 Status (sc-win32-status)
☑ Bytes Sent (sc-bytes)
☑ Bytes Received (cs-bytes)
☑ Time Taken (time-taken)
☐ Protocol Version (cs-version)
☐ Host (cs-host)
☑ User Agent (cs(User-Agent))
☐ Cookie (cs(Cookie))
☑ Referer (cs(Referer))

OK Cancel

Configuration: 'localhost' applicationHost.config or root web.config , <location path="Default Web Site">

Figure 15-7 Clicking the Select Fields button lets you choose which options are logged for a given Web site.

AppCMD

The IIS Manager is a great tool for managing individual settings using a GUI. AppCMD is a tool that is intended to perform all administrative functions from a command line. AppCMD replaces a variety of scripts and tools used in previous IIS versions.

All the logging settings you might need to adjust are located in two sections of applicationHost.config: the system.applicationHost/log section and the system.applicationHost/sites section**.**

The previous example uses IIS Manager to configure server-level logging. To use AppCMD to perform this same operation, follow this procedure:

1. Open a command prompt and navigate to the *systemRoot*\System32\inetsrv folder where AppCMD is deployed.

> [Note] If you add this path to your global *PATH* environment variable, you can execute AppCMD from any folder location. Be careful when adjusting global settings.

2. Execute the following command from the command prompt to list the current settings.

   ```
   appcmd list config -section:log
   ```

   This will display the applicationHost.config section where the centralLogFileMode settings are stored.

---

3. Next execute the following command to configure server-level logging.

   ```
   appcmd set config -section:log -centralLogFileMode:CentralW3C
   ```

4. After you have executed the command in step 3, execute the following command to list the current settings and verify the settings have been changed.

   ```
   appcmd list config -section:log
   ```

   The result, showing that the centralLogFileMode has changed to CentralW3C, should look like the following. (Some lines have been split to fit on the printed page.)

   ```
   C:\Windows\System32\inetsrv>appcmd list config -section:log
   <system.applicationHost>
     <log centralLogFileMode="CentralW3C">
       <centralBinaryLogFile enabled="true"
           directory="%SystemDrive%\inetpub\logs\LogFiles" />
       <centralW3CLogFile enabled="true"
           directory="%SystemDrive%\inetpub\logs\LogFiles" />
     </log>
   </system.applicationHost>

   C:\Windows\System32\inetsrv>
   ```

Notice *log centralLogFileMode*="*CentralW3C*"> setting, shown in bold. Before executing the *appcmd set config* command, there was no value listed because the Site option is the default setting as defined in the schema.

The section titled "Advanced AppCMD Details" later in this chapter covers how to find out which options can be set.

As another example, assuming you have already set the global *Server* attribute, if you want to adjust the global localTimeRollover setting use this command.

```
appcmd set config -section:log -centralW3CLogFile.localTimeRollover:True
```

The result should look like this.

```
Applied configuration changes to section "system.applicationHost/log" for
"MACHINE/WEBROOT/APPHOST" at configuration commit path "MACHINE/WEBROOT/APPHOST"
```

Or, for example, you might want to change the siteDefaults log format to NCSA so that all new sites will inherit this setting unless otherwise configured on a specific site. You can adjust the global Format option to NCSA with this command.

```
appcmd set config -section:sites -siteDefaults.logFile.logFormat:NCSA
```

Here's the result.

```
Applied configuration changes to section "system.applicationHost/sites" for
"MACHINE/WEBROOT/APPHOST" at configuration commit path "MACHINE/WEBROOT/APPHOST"
```

AppCMD enables you to quickly use the command line to make changes to your IIS log settings. You can create a set of scripts that use AppCMD to replace the repetitive changes typically required when using the IIS Manager GUI. Such scripts can help streamline and automate your server configuration and deployment.

AppCMD Required for Windows Vista

By default, Windows Vista does not provide a GUI to manage your log files. You need to use AppCMD to make adjustments to your log file settings. Microsoft has provided an out-of-band add-on for IIS 7.0 on Windows Vista. Here is the link to obtain the Vista logging UI add-on: *www.iis.net/downloads/default.aspx?tabid=34&g=6&i=1328*

Advanced AppCMD Details

AppCMD enables you to perform many advanced operations. Here are some tips for using AppCMD to configure advanced properties.

When you configure the *centralLogFileMode* attribute, the only way to view which properties (also known as enums) are available is to open the IIS_Schema.xml file. This is OK to do once in a while, but it's more efficient to use AppCMD to list the available properties. For example, the following command lists all the properties that can be set in the *system.applicationHost/log* section.

```
//List all properties available the system.applicationHost/log section
appcmd set config –section:log -?
```

The output looks like this.

```
ERROR ( message:-logInUTF8
-centralLogFileMode
-centralBinaryLogFile.enabled
-centralBinaryLogFile.directory
-centralBinaryLogFile.period
-centralBinaryLogFile.truncateSize
-centralBinaryLogFile.localTimeRollover
-centralW3CLogFile.enabled
-centralW3CLogFile.directory
-centralW3CLogFile.period
-centralW3CLogFile.truncateSize
-centralW3CLogFile.localTimeRollover
-centralW3CLogFile.logExtFileFlags
 )
```

To adjust a property value, use the following syntax. (You can adjust multiple attributes by putting a space between each property value.)

```
appcmd set config –section:log –property1Name:Value –property2Name:Value
```

If you are not sure which values are available to set on a particular property, you can use the following command to find out the values. This example shows how to get all values that can be set for the *centralLogFileMode* property**.**

```
//Find out which values can be set.
Appcmd set config –section:log –centralLogFileMode -?
```

The resulting error message lists the valid values, in this case Site, CentralBinary, and CentralW3C.

```
ERROR ( message:Unknown attribute "centralLogFileMode".. Reason: Enum must be one of
Site, CentralBinary, CentralW3C. )
```

You can change the site's log settings. To list all the properties that are available and their syntax, type this command:

```
//List all properties available on the Sites section
appcmd set config -section:sites -?
```

The output shows all properties related to the Sites section. The options starting with -*siteDefaults.logFile*, shown in the next lines of code in bold, enable you to adjust the defaults inherited by new sites. (Some lines have been split to fit on the printed page.)

```
C:\Windows\System32\inetsrv>appcmd set config -section:sites -?
ERROR ( message:-siteDefaults.name
-siteDefaults.id
-siteDefaults.serverAutoStart
-siteDefaults.bindings.
    [protocol='string',bindingInformation='string'].protocol
-siteDefaults.bindings.
    [protocol='string',bindingInformation='string'].bindingInformation
-siteDefaults.limits.maxBandwidth
-siteDefaults.limits.maxConnections
-siteDefaults.limits.connectionTimeout
-siteDefaults.logFile.logExtFileFlags
-siteDefaults.logFile.customLogPluginClsid
-siteDefaults.logFile.logFormat
-siteDefaults.logFile.directory
-siteDefaults.logFile.period
-siteDefaults.logFile.truncateSize
-siteDefaults.logFile.localTimeRollover
-siteDefaults.logFile.enabled
-siteDefaults.traceFailedRequestsLogging.enabled
-siteDefaults.traceFailedRequestsLogging.directory
-siteDefaults.traceFailedRequestsLogging.maxLogFiles
-siteDefaults.traceFailedRequestsLogging.maxLogFileSizeKB
-siteDefaults.traceFailedRequestsLogging.customActionsEnabled
-applicationDefaults.path
-applicationDefaults.applicationPool
-applicationDefaults.enabledProtocols
-virtualDirectoryDefaults.path
-virtualDirectoryDefaults.physicalPath
-virtualDirectoryDefaults.userName
-virtualDirectoryDefaults.password
-virtualDirectoryDefaults.logonMethod
-virtualDirectoryDefaults.allowSubDirConfig
-[name='string',id='unknown'].name
-[name='string',id='unknown'].id
-[name='string',id='unknown'].serverAutoStart
-[name='string',id='unknown'].bindings.
    [protocol='string',bindingInformation='string'].protocol
-[name='string',id='unknown'].bindings.
    [protocol='string',bindingInformation='string'].bindingInformation
-[name='string',id='unknown'].limits.maxBandwidth
-[name='string',id='unknown'].limits.maxConnections
-[name='string',id='unknown'].limits.connectionTimeout
-[name='string',id='unknown'].logFile.logExtFileFlags
-[name='string',id='unknown'].logFile.customLogPluginClsid
-[name='string',id='unknown'].logFile.logFormat
-[name='string',id='unknown'].logFile.directory
-[name='string',id='unknown'].logFile.period
-[name='string',id='unknown'].logFile.truncateSize
-[name='string',id='unknown'].logFile.localTimeRollover
-[name='string',id='unknown'].logFile.enabled
-[name='string',id='unknown'].traceFailedRequestsLogging.enabled
-[name='string',id='unknown'].traceFailedRequestsLogging.directory
-[name='string',id='unknown'].traceFailedRequestsLogging.maxLogFiles
-[name='string',id='unknown'].traceFailedRequestsLogging.maxLogFileSizeKB
-[name='string',id='unknown'].
    traceFailedRequestsLogging.customActionsEnabled
-[name='string',id='unknown'].applicationDefaults.path
-[name='string',id='unknown'].applicationDefaults.applicationPool
-[name='string',id='unknown'].applicationDefaults.enabledProtocols
-[name='string',id='unknown'].virtualDirectoryDefaults.path
-[name='string',id='unknown'].virtualDirectoryDefaults.physicalPath
-[name='string',id='unknown'].virtualDirectoryDefaults.userName
-[name='string',id='unknown'].virtualDirectoryDefaults.password
-[name='string',id='unknown'].virtualDirectoryDefaults.logonMethod
-[name='string',id='unknown'].virtualDirectoryDefaults.allowSubDirConfig
-[name='string',id='unknown'].[path='string'].path
-[name='string',id='unknown'].[path='string'].applicationPool
-[name='string',id='unknown'].[path='string'].enabledProtocols
-[name='string',id='unknown'].[path='string'].virtualDirectoryDefaults.path
-[name='string',id='unknown'].[path='string'].
    virtualDirectoryDefaults.physicalPath
-[name='string',id='unknown'].[path='string'].
    virtualDirectoryDefaults.userName
-[name='string',id='unknown'].[path='string'].
    virtualDirectoryDefaults.password
-[name='string',id='unknown'].[path='string'].
    virtualDirectoryDefaults.logonMethod
-[name='string',id='unknown'].[path='string'].
    virtualDirectoryDefaults.allowSubDirConfig
-[name='string',id='unknown'].[path='string'].[path='string'].path
-[name='string',id='unknown'].[path='string'].[path='string'].physicalPath
-[name='string',id='unknown'].[path='string'].[path='string'].userName
-[name='string',id='unknown'].[path='string'].[path='string'].password
-[name='string',id='unknown'].[path='string'].[path='string'].logonMethod
-[name='string',id='unknown'].[path='string'].[path='string'].
    allowSubDirConfig
 )
```

You can also adjust settings for specific Web sites by using the properties starting with -*[name='string',id='unknown'].logFile***.** You need to just replace the '*unknown*' value with the Web site name. Here is an example of how to do a specific site. Notice the example for the Default Web Site contains double quotation marks. This is necessary to handle spaces in the Web site name. Remember to change the name and id when using the example.

```
//Example how to set the logFile.directory property with a
//Site with spaces in the name.

C:\Windows\System32\inetsrv>appcmd set config -section:sites /[name='"Default Web
Site"',id='1'].logFile.directory:c:\wwwlogs

//Example how to setup logFile.directory property with no spaces
//in the Site name.

C:\Windows\System32\inetsrv>appcmd set config -section:sites
/[name='Contoso.om',id='2'].logFile.directory:c:\wwwlogs
```

You can also use Windows Powershell 1.0 to administer your IIS 7.0 server. This section shows a few examples of setting the Logfile directory value. In the following sample script, you first load Microsoft.Web.Administration.dll into your Windows Powershell session. Next, you assign an instance of the *ServerManager* object to the *$sm* variable, which allows you to query and set Logfile values. (In the following listing, some lines have been split so that they fit on the printed page.)

```
//Load the dll into the Powershell session
[System.Reflection.Assembly]::LoadFrom
    ( "C:\windows\system32\inetsrv\Microsoft.Web.Administration.dll" )

//Load an instance of the Server Manager object into the $sm variable
$sm = new-object Microsoft.Web.Administration.ServerManager

//List Default Web Site LogFile Directory value.
$sm.Sites["Default Web Site"].LogFile.Directory

//List SiteDefaults LogFile Directory value.
$sm.SiteDefaults.LogFile.Directory

//Set Default Website LogFile Directory
$sm.Sites["Default Web Site"].LogFile.Directory =
    "\\RemoteServer.Contoso.com\Logfiles"
$sm.CommitChanges()

//Set SiteDefaults LogFile Directory
$sm.SiteDefaults.Logfile.Directory = "\\RemoteServer.Contoso.com\Logfiles"
$sm.CommitChanges()
```

Using Windows Powershell to administer IIS 7.0 is covered in Chapter 8, "Name." For information about building a cmdlet to administer many common functions in IIS 7.0, see the following Web site: *www.iis.net/articles/view.aspx/IIS7/Use-IIS7-Administration-Tools/Scripting-IIS7/Writing-PowerShell-Command-lets-for-IIS7*

Immediately flushing log entries to disk is introduced in Windows Server 2008. The HTTP.sys service holds requests until they are periodically flushed to disk. When you are troubleshooting an immediate issue, you can use the following *netsh* command, which can be especially useful for troubleshooting HTTP.sys-related errors.

```
//Flush log entries to disk immediately
Netsh http flush logbuffer
```

HTTP.sys Logging

In IIS 6.0, the HTTP.sys process was introduced and took over logging duties that used to be handled by Inetinfo.exe. HTTP.sys introduced another log called httperr.log. The HTTPERR logs for Windows Server 2008 are located in the same location as for Windows Server 2003. The path is *SystemRoot*\System32\LogFiles\httperr. This log records all errors that are not handed off to a valid worker process, typically responses to clients, connection time-outs, and orphaned requests. This additional information can help you troubleshoot HTTP-based errors, which are logged before the request reaches IIS.

Windows Vista and Windows Server 2008 introduce enhancements to the HTTP.sys logging process. You use ETW (Event Tracing for Windows) to obtain the enhanced information. Here are steps to start, capture, and display information from an ETW tracing session:

1. Open a command prompt (click Start, select Run, and then type **cmd.exe**).
2. Start the ETW trace session for HTTP.sys by using the following command.

```
logman.exe start httptrace -p Microsoft-Windows-HttpService 0xFFFF -o
    httptrace.etl -ets
```

3. Reproduce or perform the steps or tests that need to be traced.
4. To stop the ETW trace session for HTTP.sys, use the following command.

   ```
   logman stop httptrace –ets
   ```

5. To convert the ETL file to a CSV file, use this command.

   ```
   tracerpt httptrace.etl -of csv -o httptrace.csv /y
   ```

   The CSV files can then be viewed in a text editor or spreadsheet application. This complete procedure is covered in the following white paper: *http://download.microsoft.com/download/3/b/a/3ba6d659-6e39-4cd7-b3a2-9c96482f5353/HTTP.sys%20Manageability%20in%20Windows%20Vista%20and%20Longhorn%20Server.doc*

The following site discusses the new networking features in Windows Vista and Windows Server 2008: *http://technet.microsoft.com/en-us/library/bb726965.aspx*

Application Logging

Besides the standard IIS type logs, other items can be logged. Many of these options can be set wither with the IIS Manager or by using AppCMD.

Process Recycling Logging

In IIS 7.0, events are logged with more granularity when an application pool recycles. You can control eight configuration settings if each option listed in Table 15-1 generates an event log message. Table 15-1 lists these settings.

Table 15-1 Recycling Options Under Generate Recycle Event Log Entry

Option	Description	Default Setting
Application Pool Configuration Changed	Event is logged when the application pool recycles due to a change in its configuration	No
Isapi Report Unhealthy	Event is logged because an ISAPI extension has reported itself as unhealthy	No
Manual Recycle	Event is logged when the application pool has been manually recycled	No
Private Memory Limit Exceeded	Event is logged when the application pool recycles after exceeding its private memory limit	Yes
Regular Time Interval	Event is logged when the application pool recycles on its scheduled interval	Yes
Request Limit Exceeded	Event is logged when the application pool recycles after exceeding its request limit	No
Specific Time	Event is logged when the application pool recycles at a scheduled time	No
Virtual Memory Limit Exceeded	Event is logged when the application pool recycles after exceeding its virtual memory limits	Yes

ASP

Classic ASP is alive and well in IIS 7.0, and you can configure options for logging ASP errors under the ASP section in the IIS Manager. Use the following options to track down issues when migrating your Classic ASP applications to IIS 7.0.

- **Enable Log Error Requests** Controls whether the Web server writes ASP errors to the application event log
- **Log Errors To The NT Log** Specifies that ASP errors are recorded in the Windows event log

These options are available in IIS 6, but you have to use ADSUtil.vbs to enable in the Metabase. Now, in IIS 7.0, you can use the IIS Manager to enable these options.

ASP.NET

All ASP.NET 2.0 unhandled exceptions are written to the Application Event log. Along with application pool recycle events or other errors in the event logs, this can be very helpful in troubleshooting application errors. You can turned off ASP.NET logging by following the instructions in this KB article: *support.microsoft.com/kb/911816*

> [Important]Microsoft recommends you not do this because it might result in leaked resources and abandon locks.

IIS Events

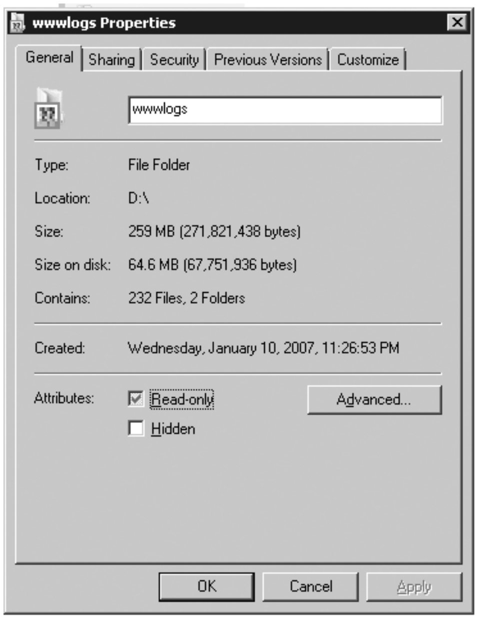
Other processes related to IIS also log to the Windows Event log. This includes the HTTP, IISAdmin, FTP Publishing Service, W3SVC services. For a complete list of events, see:

- *www.microsoft.com/technet/prodtechnol/WindowsServer2003/Library/IIS/af0a2e97-0c46-4e0a-9f2f-bdbe4a220c49.mspx?mfr=true*
- www.microsoft.com/technet/prodtechnol/WindowsServer2003/Library/IIS/d91983bf-6c67-46f8-9db9-a8ab502e55d3.mspx?mfr=true

Folder Compression Option

Log files are necessary to keep track of Web site statistics and trends, and Web developers and business people use them to ensure their Web sites continue to grow. One of the biggest challenges administrators face is how to retain and manage log files. By default, IIS rolls over log files once a day. Your log files can become quite large even if you use the default log file rollover setting.

Windows Server 2008 allows for folder compression to help save space. You can enable this option by using Windows Explorer. Figure 15-8 shows a folder with compression enabled. In this example, the size of the folder is 259 megabytes (MB), but the actual space the folder uses on the disk is only 64.6 MB. If your uncompressed log files take up several gigabytes (GB), you could save yourself a lot of disk space by using folder compression.



wwwlogs

Type: File Folder
Location: D:\
Size: 259 MB (271,821,438 bytes)
Size on disk: 64.6 MB (67,751,936 bytes)
Contains: 232 Files, 2 Folders
Created: Wednesday, January 10, 2007, 11:26:53 PM
Attributes: ☑ Read-only ☐ Hidden

Figure 15-8 Folder compression enabled on the WWWLogs folder

Because HTTP.sys buffers information written to the IIS log files, there will not be a performance hit if your log files are in a folder for which compression is enabled. Some people use third-party log compression products or free tools such as Gzip along with scripts to compress their log files. Unless you have a tool that searches inside zip files, this is an acceptable method only if you rarely need to unzip and search your archived log files. If you have compression enabled, however, you can leave your files in their original, easily searchable state. At most, you'll need to implement some type of archival and deletion script by using your favorite script or third-party program.

Using the built-in compression feature provided by Windows Server 2008 can save you disk space and simplify how you retain your log files. For more information on managing log files, Chapter 6, "Name," of the Microsoft Log Parser book discusses conversion, archival, and repudiation strategies. The ISBN is 1932266526.

Logging Analysis Using Log Parser

A chapter on logging would not be complete without mentioning Log Parser. This is one of the most useful tools for searching your logs. Teaching you Log Parser is beyond the scope of this book, we'll give you some examples you can use in your environment. You can download Log Parser at: *www.iis.net/default.aspx?tabid=2&subtabid=29#LogParser*

Members of the Microsoft.com team are big fans of Log Parser. Look at this Web site for an article that discusses how they use Log Parser: *blogs.technet.com/mscom/archive/2005/10/19/412745.aspx*

Here are three examples of using Log Parser to extract common information from your IIS logs:

1. List the top 10 .ASPX WebRequests.

```
LogParser -i:iisw3c "SELECT TOP 10 cs-uri-stem,
    COUNT(*) AS HitCount INTO Results.csv FROM LOGFILENAME.LOG
    GROUP BY cs-uri-stem ORDER BY HitCount DESC" -o:csv
```

2. Show the 20 requests that take the longest to execute.

```
//Change the date to fit your needs
SELECT
TOP 20
CS-URI-STEM,
TIME-TAKE,time-taken
FROM LOGFILENAME.LOG
WHERE DATE > '2007-03-26'
ORDER BY TIME-TAKEN DESC
```

3. Select information between two dates and pipe results to a text file named Output.txt.

```
SELECT
DATE,
TIME,
CS-URI-STEM,
SC-STATUS,
COUNT(*) AS MaxTime
INTO Output.txt
FROM LOGFILENAME.LOG
WHERE TO_TIME(time)
BETWEEN
TIMESTAMP('01/01 13:50:00', 'MM/dd hh:mm:ss') AND
    TIMESTAMP('01/01 18:30:00', 'MM/dd hh:mm:ss') AND SC-STATUS = 500
GROUP BY
CS-URI-STEM,
DATE,
TIME,
SC-STATUS
ORDER BY MaxTime DESC
```

If you are responsible for maintaining an IIS environment, take a look at Log Parser. You'll want to make it one of your main tools when troubleshooting all kinds of issues. (For more information about Log Parser, visit the community forums at: *forums.iis.net/default.aspx?GroupID=51)*

Summary

IIS 7.0 takes the best features first introduced in Windows Server 2003 and builds on them. The modular architecture and Integrated Pipeline open up a lot of opportunities to enhance your application logging options. The IIS Manager exposes and simplifies how you manage your log settings. You can set your default logging settings to be on a per-site or per-server basis.

IIS 7.0 also introduces many tools for automating your log file configuration. You can use AppCMD or Windows Powershell along with Microsoft.Web.Administration to configure or search for information. The new UTF-8 encoding helps standardize your logs.

IIS 7.0 exposes more data in logging. You can use the new tools provided by IIS 7.0 and by Windows Server 2008 to browse the additional information as you track down and eliminate problems in your environment.
