PART I

Introduction and Deployment

- ▶ **CHAPTER 1:** Background on IIS and New Features in IIS 8.0
- ▶ **CHAPTER 2:** IIS 8.0 Architecture
- ▶ **CHAPTER 3:** Planning Your Deployment
- ▶ **CHAPTER 4:** Installing IIS 8.0

1

Background on IIS and New Features in IIS 8.0

WHAT'S IN THIS CHAPTER?

- A background of IIS
- Windows Server 2012 features
- New features in IIS 8.0

Microsoft's Internet Information Services (IIS) has been around for more than 15 years, from its first incarnation in Windows NT 3.51 to the current release of IIS 8.0 on the Windows Server 2012 and Windows 8 platforms. It has evolved from providing basic service as an HTTP server, as well as additional Internet services such as Gopher and WAIS, to a fully configurable application services platform integrated with the operating system.

IIS 8.0 is not as dramatic a change as IIS 7.0 was, but IIS 8.0 benefits from the improvements in the Windows Server 2012 operating system. These benefits make IIS 8.0 far more scalable, more appropriate for cloud and virtual systems, and more integral to Microsoft's application and programming environment.

This chapter provides an overview of the changes in IIS 8.0 as well as a sampling of some of the new technologies. If you are familiar with IIS 7.0, you will want to skim through this chapter for changes before digging into future chapters for specifics. If you are new to IIS, this chapter will provide an introduction to the features in IIS 8.0 and provide you with a basis for understanding future chapters. And if you're the kind of reader who just wants to skip to the part that applies to your immediate needs, this chapter can help you figure out in what area those needs lie.

IIS VERSIONS 1.0 TO 4.0

IIS was released with Service Pack 3 for Windows NT 3.51, as a set of services providing HTTP, Gopher, and WAIS functionality. Although the functions were there, most users chose alternatives from third-party vendors, such as O'Reilly's website or Netscape's server. Although these services had been available for years with the various flavors of UNIX operating systems, native Internet services for Windows were mostly an afterthought, with little integration with the Windows operating system.

With the advent of Windows NT 4.0, IIS also matured in version 2.0. The most notable improvement in IIS version 2.0 was closer integration with the Windows NT operating system, taking advantage of Windows security accounts and providing integrated administration through a management console similar to many other Windows services. IIS 2.0 introduced support for HTTP Host headers, which allowed multiple sites to run on a single IP address, and aligned Microsoft's IIS development with National Computer Security Association (NCSA) standards, providing for NCSA common log formats and NCSA-style map files. IIS 2.0 also introduced a web browser interface for management and content indexing through Microsoft's Index Server.

IIS version 3.0 was introduced with Windows NT Service Pack 3 and introduced the world to ASP (Active Server Pages) and Microsoft's concept of an *application server*. A precursor to the ASP.NET environment, ASP (now referred to as *classic ASP*) is a server-side scripting environment for the creation of dynamic web pages. Using VBScript, JScript, or any other active scripting engine, programmers finally had a viable competitor to Common Gateway Interface (CGI) and scripting technologies available on non-Microsoft platforms, such as Perl.

IIS 4.0, available in the NT Option Pack, introduced ASP 2.0, an object-based version of ASP that included six built-in objects to provide standardized functionality in ASP pages. IIS 4.0 was the last version of IIS that coumld be downloaded and installed outside of the operating system.

IIS 5.0 AND 5.1

With the release of Windows 2000, IIS became integrated with the operating system. Version numbers reflected the operating system, and there were no upgrades to IIS available without upgrading the operating system. IIS 5.0 shipped with Windows 2000 Server versions and Windows 2000 Professional, and IIS version 5.1 shipped with Windows XP Professional, but not Windows XP Home Edition. For all essential functions, IIS 5.0 and IIS 5.1 are identical, differing only slightly as needed by the changes to the operating system.

With Windows 2000 and IIS 5.0, IIS became a service of the operating system, meant to be the base for other applications, especially for ASP applications. The IIS 5.0 architecture served static content, Internet Server Application Programming Interface (ISAPI) functions, or ASP scripts, with ASP script processing handed off to a script engine based on the file extension. Using file extensions to determine the program that handles the file has always been a common part of Windows functionality, and in the case of ASP processing, the speed of serving pages was increased by the automatic handoff of ASP scripts directly to the ASP engine, bypassing the static content handler. This architecture has endured in IIS to the current version.

IIS 6.0

IIS 6.0 shipped with Windows Server 2003 editions and Windows XP Professional 64-Bit Edition, which was built on the Windows Server 2003 Service Pack 1 code base. IIS 6.0 was identical among operating system versions, but there were restrictions or expansions depending on the version of Server 2003 under which IIS was running. For example, Server 2003 Web Edition would only run IIS and a few ancillary services; it could not be used to run Microsoft SQL Server. On the other end of the spectrum, only the Enterprise and Data Center versions of Server 2003 included clustering technology.

Operating system changes also expanded the capabilities of IIS as an application server. Native XML Web Services appeared in Server 2003. Process-independent session states made web farms easier to configure and manage, allowing session states to be stored outside of the application for redundancy and failover. Web farms also became easier with Server 2003's improved Network load-balancing features, such as the NLB Manager, which provided a single management point for NLB functions.

Secure by Default

Windows Server 2003 and IIS 6.0 shipped in a secure state, with IIS no longer installed by default. Even when IIS was installed, the default installation would serve only static HTML pages; all dynamic content was locked down. Managed through web service extensions, applications such as ASP and ASP.NET had to be specifically enabled, minimizing default security holes with unknown services open to the world.

IIS 6.0 also ran user code under a low-privilege account, Network Service, which had few privileges on the server outside of the IIS processes and the website hierarchy. Designed to reduce the damage exposure from rogue code, access to virtual directories and other resources had to be specifically enabled by the administrator for the Network Service account.

IIS 6.0 also allowed delegation for the authentication process; thus, administrators and programmers could further restrict account access. Passport authentication was also included with IIS 6.0, although in real-world use, it never found widespread favor among administrators. Kerberos authentication, on the other hand, allowed secure communication within an Active Directory domain and solved many remote resource permission issues.

IIS 6.0 also would serve only specific file requests, by default not allowing execution of command-line code or even the transfer of executable files. Unless the administrator assigned a specific MIME (Multipurpose Internet Mail Extensions) type to be served, IIS would return a 404 error to the request, reporting the file not found. Earlier versions of IIS included a wildcard mapping and would serve any file type.

Request Processing

IIS 6.0 changed the way IIS processed requests, eliminating what had been a major performance hurdle in scaling prior IIS versions to serve multiple sites. IIS 6.0 used the Http.sys listener to receive requests and then handed them off to worker processes to be addressed. These worker processes were isolated to application pools, and the administrator could assign application pools to specific sites and applications. This meant that many more requests could be handled simultaneously, and it also provided for an isolated architecture in cases of error. If a worker process failed, the effects would not be seen outside of the application pool, providing stability across the server's sites. In addition, worker processes could be assigned a processor affinity, allowing multiprocessor systems to split the workload.

Additional Features

As did its predecessors, IIS 6.0 included additional features and functionality. Some internal features, such as HTTP compression and kernel mode caching, increased performance of the web server and applications served from it. Other features affected configuration, such as the move to an XML metabase, or stability, such as being able to configure individual application pools and isolate potential application failures. Still others added or expanded utility and ancillary functions, such as the improved FTP services or the addition of POP services to the existing SMTP service.

Application Pools

IIS 6.0 changed the way applications behaved in memory, isolating applications into memory pools. Administrators could configure separate memory pools for separate applications, thus preventing a faulty application from crashing other applications outside of its memory pool. This is particularly important in any shared web server environment, especially with ASP.NET applications.

FTP Service

The FTP service grew up in IIS 6.0, providing for greater security and separation of accounts through a new isolation mode using either Active Directory or local Windows accounts. Using Windows accounts or Active Directory accounts, users could be restricted to their own available FTP locations without resorting to naming the home directories the same as the FTP accounts. In addition, users were prevented from traversing above their home directories and seeing what other accounts may exist on the server. Even without NT File System (NTFS) permissions to the content, security in FTP before IIS 6.0 was still compromised because a user could discover other valid user accounts on the system.

SMTP and POP Services

The SMTP service in Windows Server 2003 didn't change much from previous versions, allowing for greater flexibility and security but not altering the core SMTP functions. Most administrators would not use the SMTP service in IIS for anything other than outbound mail, instead relying on third-party servers or Microsoft's Exchange Server for receiving and distributing mail. But the addition of a POP3 service in Server 2003 allowed a rudimentary mail server configuration, useful for testing or small mail domains. Although SMTP can be used to transfer mail, most mail clients such as Microsoft Outlook rely on the POP3 or IMAP protocols to retrieve mail, which was unavailable without additional products until Windows Server 2003 and IIS 6.0.

IIS 7.0 AND 7.5

IIS 7.0 was a complete rewrite of the base code from IIS 6.0 and earlier. Available on Windows Vista and Windows Server 2008, IIS 7.0 adapted to several operating systems, including the new Windows Core Edition and the Windows Web Server edition. IIS 7.5, introduced with Windows 7, consisted of IIS 7.0 plus all the inline updates that had been made to IIS 7.0 since its introduction. Users could essentially update IIS 7.0 to the functionality of IIS 7.5 by installing the appropriate updates and modules.

IIS 7.0 was a ground-up rewrite of IIS 6.0, designed as an integrated web application platform. Integration with the ASP.NET framework combined with fully exposed application programming interfaces (APIs) for complete extensibility of the platform and management interfaces made IIS 7.0 a programmer's dream. Security that included delegation of configuration and a complete diagnostic suite with request tracing and advanced logging satisfied several of the administrator's desires.

Although the most substantial change in IIS 7.0 may have been the integration of ASP.NET into the request pipeline, the extensibility of IIS 7.0, configuration delegation and the use of XML configuration files, request tracing and diagnostics, and the new administration tools were all welcome changes from previous versions of IIS.

Unlike previous versions of IIS, the modular design of IIS 7.0 allowed for easy implementation of custom modules and additional functionality. This increased functionality came from in-house development, third-party sources, or even Microsoft. Because these modules and additional programs could be plugged into IIS at any time, without changing core operating system functions, the Microsoft IIS development team shipped additional supported and unsupported modules outside of Microsoft's standard Service Pack process. IIS 7.5 included most of these inline updates and modules, such as FTP 7.5, that did not originally exist for IIS 7.0. Microsoft's website at `www.iis.net` is the source for these additional downloads, for the IIS 7.0 and 7.5 versions, as well as for future add-on modules and updates for IIS 8.0.

ASP.NET Integration

One of the most radical changes in IIS 7.0 was its close integration with ASP.NET and the ASP.NET processes. There was a unified event pipeline in IIS 7.0 that merged the previously separate IIS and ASP.NET pipelines from IIS 6.0 and earlier. ASP.NET HTTP modules that previously only listened for events within the ASP.NET pipeline could be used for any request in IIS 7.0. For backward compatibility, IIS 7.0 maintained a Classic pipeline mode, which emulated the separate IIS and ASP.NET pipeline model from IIS 6.0.

IIS 7.0 also changed IIS configuration to match the process used for configuring ASP.NET applications. This greatly improved and simplified the implementation of IIS into the ASP.NET programming environment and allowed for better configurability and easier deployment of both sites and applications. It also made deployment across multiple systems in web farms more straightforward and allowed for extensibility of the configurations. IIS 7.0 introduced the concept of shared configuration, wherein multiple web servers can point to the same physical file for configuration, making deploying configuration changes to web farms nearly instantaneous.

IIS 7.0 introduced the `applicationHost.config` file for storing settings and added configuration options for individual websites or web applications to the `web.config` files, alongside ASP.NET settings, in a new `system.webServer` section.

Extensibility

IIS 7.0 greatly increased the extensibility of IIS as a web application platform. Because of the changes to the request-processing pipeline, the core server itself was now extensible, using both native and managed code. Instead of having to work with ISAPI filters to modify the request process, developers could now inject their own components directly into the processing pipeline. These components could represent the developers' own code, third-party utilities and components, and existing Microsoft core components. This meant that if you didn't like Microsoft's Windows authentication process, you could not only choose to use forms authentication on all files, but also choose to bypass all built-in authentication and roll your own. In addition, if you didn't need to process classic ASP files, you could simply not load that component. Unlike in previous versions, in which components were loaded into memory in a single DLL, IIS 7.0 reduced the memory footprint by not loading unnecessary modules or code.

Security

Componentization also increased the already strong security that existed in IIS 6.0. A perennial complaint against Microsoft had always been that IIS installed by default and that all services were active by default. IIS 6.0 and Server 2003 reversed that course—almost nothing was installed by default, and even when you did install it, the majority of components were disabled by default. To enable ASP.NET, you had to choose to allow ASP.NET as a web service extension. Classic ASP had to be enabled separately, as did third-party CGI application processors such as Perl or PHP.

With the exception of third-party software, however, IIS 6.0 still loaded all the services into memory—it just loaded them as disabled. For example, if you didn't want to use Windows authentication, as would be the case if you were using your own authentication scheme, you could choose not to *enable* it, but the code still resided in memory. Similarly, default IIS 6.0 installations were locked down to processing static HTML files, a good choice from a security standpoint. But what if you were never going to use static HTML files in your application or site? In IIS 7.0, you had the option of never loading the code in the first place.

Minimal Installation

IIS 7.0 continued the tradition of its predecessor with minimal installation the default. IIS was not installed with the default operating system installation, and a basic install only selected those options needed for serving static HTML files. The installation graphical user interface (GUI) for IIS 6.0 allowed a choice of eight different options, including installing FTP, whereas IIS 7.0's setup allowed for more than 40 options. This granularity of setup reduced the memory footprint of IIS 7.0, but more importantly, it reduced the security footprint as well.

Management Delegation

Management of IIS in previous versions meant either granting local administrator privileges to the user or working through Windows Management Instrumentation (WMI) and Active Directory Services Interfaces (ADSI) options to manage the site configurations directly. The only other option was for developers to work through the IIS administrators to change configurations—an option that could often be frustrating for both administrators and programmers. IIS 7.0 changed this through delegation of administration permissions at the server, site, and application levels.

Unified Authentication and Authorization

In IIS 7.0, the authentication and authorization process merged the traditional IIS authentication options with ASP.NET options. This allowed administrators and developers to use ASP.NET authentication across all files, folders, and applications in a site.

In IIS 6.0 and previous versions, controlling access to an Adobe Acrobat (PDF) file was difficult through ASP.NET authentication schemes. You would need to enable Windows authentication or basic authentication on the website, folder, or file and create a Windows account to have access to the file. Then you would need to require the user to provide valid credentials for that Windows account, even if he or she already had logged into your ASP.NET application, to be able to access that PDF file. The alternative was to use impersonation in ASP.NET to access the file using the ASP .NET process account—all to prevent someone from opening the PDF file by pasting the direct URL into their browser. Options involving streaming the content from a protected location were just as cumbersome, and redirecting files to be processed by the ASP.NET DLL was even more problematic.

In IIS 7.0, using ASP.NET authentication no longer required the file to be processed as an ASPX extension; thus, file extensions of all types could be secured with Forms authentication or any other ASP.NET method. This reduced the requirement for Windows Client Access Licenses (CALs) to provide access control, which was prohibitive in an Internet environment.

Remote Management

Although IIS could be remotely managed in previous versions using the IIS Manager over RPC, this wasn't firewall-friendly. An HTML-based management option also existed; however, this didn't allow management of all IIS features. In both cases, users were required to be in the local Administrators group on the machine.

IIS 7.0 introduced a new remote Management Service that permitted the IIS Manager tool to administer remote IIS 7.0 installations over HTTPS. By using the new delegation features in IIS 7.0, remote users could be given access to the entire server, a single website, or even just a single web application. Additionally, features that have not been delegated will not be visible to the end user when connecting remotely.

The Remote Management service also introduced the concept of IIS Users. These user accounts do not exist outside of IIS. An administrator can choose to permit either Windows users or IIS Users access to administer IIS remotely. IIS Users do not consume Windows CALs, nor do they have any permissions outside of IIS itself; thus, they are a cheaper and more secure option for permitting external IIS administration.

IIS Manager

IIS 7.0 introduced a new, unified IIS Manager that combined all management functions for both IIS and ASP.NET in one location. Developers could now manage individual sites and applications without needing local administrator access to the server. The IIS Manager is also extensible through the addition of modules.

AppCmd.exe Command-Line Utility

IIS 7.0 introduced a new command-line utility, `AppCmd.exe`, which replaced the functionality provided by the various VBScript command-line utilities included with previous versions. `AppCmd.exe` also expanded command-line control to all IIS configuration functions. For example, to create a virtual directory using `AppCmd.exe`, you would enter at a command prompt:

```
C:\Windows\System32\inetsrv\appcmd add vdir
/app.name: "Default Web Site/" / /path: /VirtualDiretory1
/physicalPath: C:\InetPub\VirtualDirectory1
```

PowerShell Integration

IIS 7.0 saw the integration of PowerShell commands into IIS management and deployment scenarios with the IIS PowerShell Snap-In. PowerShell has become the scripting tool of choice for Windows administrators, and integration with IIS through cmdlets and specific functions has made enterprise management of IIS servers simpler.

In PowerShell, creating a virtual directory would look something like the following:

```
PS IIS:\> New-Item 'IIS:\Sites\Default Web Site\VirtualDirectory1'
-type VirtualDirectory -physicalPath C:\InetPub\VirtualDirectory1
```

Diagnostics

IIS 7.0 made diagnostic tracing and server state management simple for both administrators and developers. The new Request Tracing module allowed for tracing any request through the pipeline to the point of exit or failure, and provides a logging function for those traces.

Using the Request Tracing module, you could configure logging and tracing of any type of content or result code. Like most IIS settings, request tracing can be configured at the server, site, or application level.

WINDOWS SERVER 2012 FEATURES

Because IIS is integrated into the Windows operating system, many of the changes to IIS 8.0 have to do with changes to the Windows operating system itself. Windows Server 2012 has many new features that affect and enhance IIS 8.0.

Server Versions

Windows Server 2008 came in multiple versions, including Standard, Enterprise, and Datacenter, primarily differentiated by the amount of memory and number of processors accessible. Each was targeted, and licensed, for specific deployment types, and changing the version required a full reinstall.

Windows Server 2008 also had several special editions available. Windows Server 2008 Web Edition was designed to run a web server but could not run applications such as Microsoft Exchange or be used as an Active Directory server. The HPC Edition was designed for high-performance computing, using computing clusters and expandable into a Microsoft Azure data center for cost-effective high-performance tasks. Windows Server 2008 was also available for Itanium processors and also in a Foundation version for the low-cost and low-performance server needs of small companies.

Windows Server 2012 will not support 32-bit or Itanium processors. There is no longer a Web Edition or Foundation version, and the features of the HPC version have been incorporated into the standard operating system. In short, you can buy Windows Server 2012 in only one edition and install it for any system configuration you need, physical or virtual.

Windows Server 2008 and Windows Server 2008 R2 may be directly upgraded to Windows Server 2012, providing that the system meets hardware requirements, but Windows Server 2008 will not be upgradable to future versions of Windows.

The New User Interface

One of the most obvious changes for Windows Server 2012 is the availability of the new graphical user interface. Microsoft designed this interface to unify all forms of Windows, from servers to desktops to tablets to phones, and everything else imaginable. Although seemingly targeted toward the consumers and end user, the new graphical interface (shown in Figure 1-1) also expands the abilities of the server administrator with a new Server Manager interface as well. Live Tiles displays a real-time view of the server and provides a dashboard with live statistics for the administrator.

Windows Server 2012 does not default to the new interface, termed *Server with a GUI*. There are, in fact, three separate interfaces available for Windows Server 2012: the standard Server with a GUI interface used on the desktop; a command-line interface similar to the Server Core installation available in Windows Server 2008; and a new hybrid version, *Minimal Server Interface*, that allows you to run the graphical Server Manager and Microsoft Management Console (MMC) without adding the burden of the browser and interface graphics. Administrators can switch between these versions without having to reinstall Windows, unlike in Windows Server 2008. A simple PowerShell cmdlet allows the change, switching to the Server with a GUI interface from the command-line interface:

```
PS> Install-WindowsFeature Server-Gui-Mgmt-Infra,Server-Gui-Shell –Restart
```

Reversing this and reverting to the Server Core interface is simple:

```
PS> Uninstall-WindowsFeature Server-Gui-Mgmt-Infra,Server-Gui-Shell –Restart
```

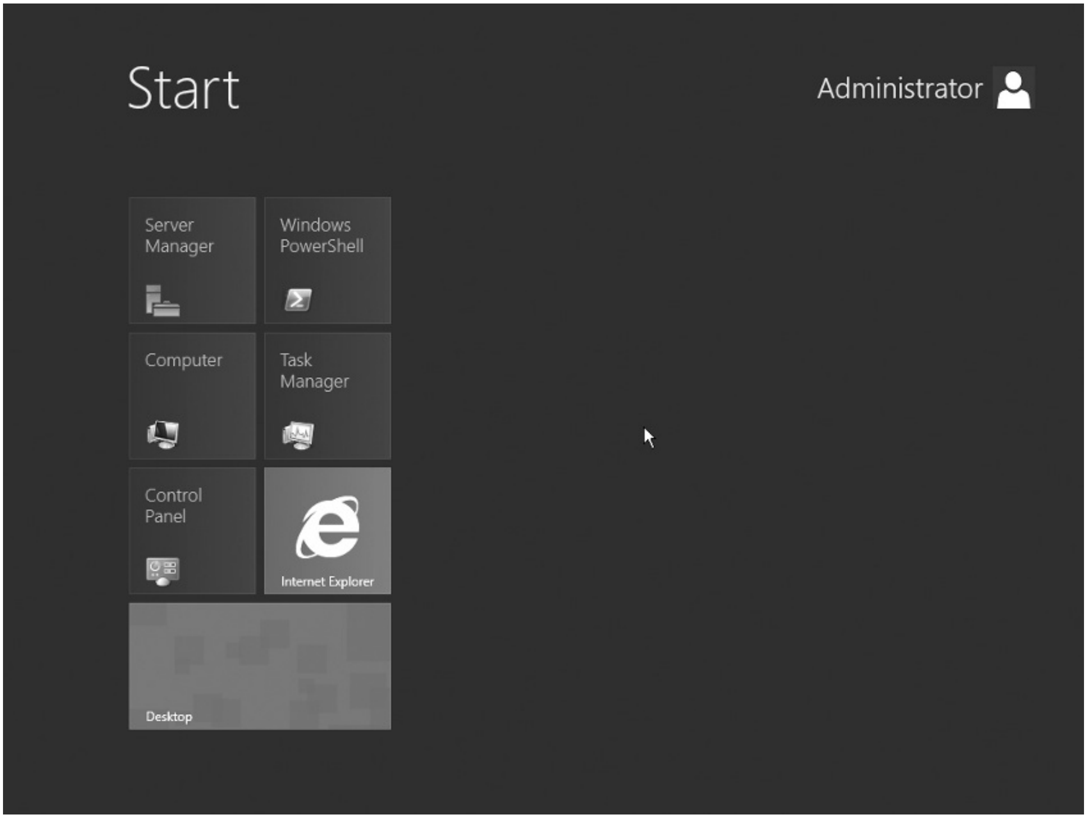
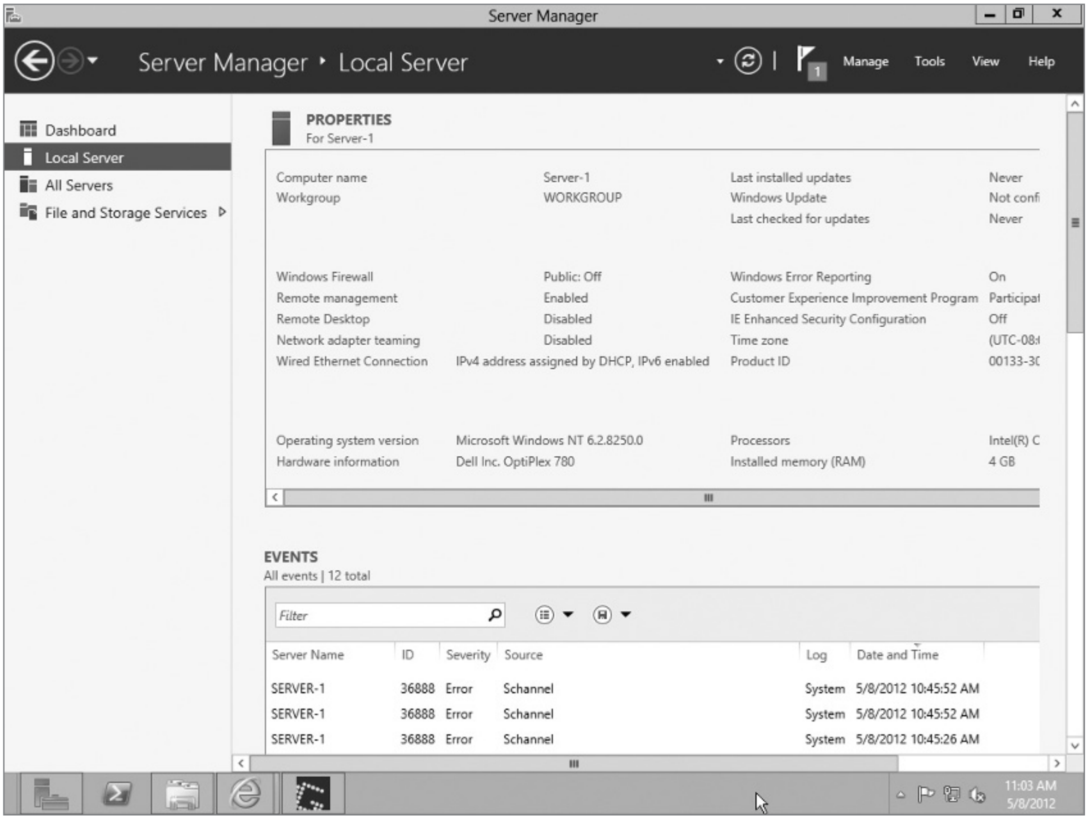


FIGURE 1-1

Most administrators will rarely see the Server with a GUI interface and instead will primarily use the new Minimal Server Interface and the Server Manager (as shown in Figure 1-2) to manage all servers in the enterprise, virtual or physical, whether or not they are based in the cloud.

The new Server Manager allows multiple servers to be administered, even from a Windows 8 workstation. New in Windows Server 2012 is the ability to manage multiple servers with credentials differing from the user's default credentials. These servers can be virtual or physical and may be located in the cloud. Server Manager in Windows Server 2012 will even aggregate server information by server role and other groupings.

> **NOTE** *When you are adding or installing a feature, the requisite source files need to be available. If they are not available as part of the Windows installation, they will be downloaded from the Windows Update website; optionally, the administrator can specify a local Windows Imaging (WIM) file as an installation source. For more information, see* `http://technet.microsoft.com/en-us/library/hh831786.aspx`.



PROPERTIES
For Server-1

Computer name	Server-1	Last installed updates	Never
Workgroup	WORKGROUP	Windows Update	Not confi
		Last checked for updates	Never
Windows Firewall	Public: Off	Windows Error Reporting	On
Remote management	Enabled	Customer Experience Improvement Program	Participat
Remote Desktop	Disabled	IE Enhanced Security Configuration	Off
Network adapter teaming	Disabled	Time zone	(UTC-08:
Wired Ethernet Connection	IPv4 address assigned by DHCP, IPv6 enabled	Product ID	00133-3C
Operating system version	Microsoft Windows NT 6.2.8250.0	Processors	Intel(R) C
Hardware information	Dell Inc. OptiPlex 780	Installed memory (RAM)	4 GB

EVENTS
All events | 12 total

Filter

Server Name	ID	Severity	Source	Log	Date and Time
SERVER-1	36888	Error	Schannel	System	5/8/2012 10:45:52 AM
SERVER-1	36888	Error	Schannel	System	5/8/2012 10:45:52 AM
SERVER-1	36888	Error	Schannel	System	5/8/2012 10:45:26 AM

FIGURE 1-2

Virtualization and Private Cloud

Windows Server 2008 supported virtualization and Microsoft's Hyper-V technology, but in Windows Server 2012 virtualization and cloud deployment are the driving force in many of the operating system's architecture changes. Active Directory's changes to accommodate rapid cloud deployment and the virtualization of Active Directory servers make for a seamless management interface. Virtual images and physical servers are treated identically in Windows Server 2012 and can be managed through the same Server Manager interface.

Windows Server 2012 supports both public and private clouds, as well as hybrid clouds, but the private cloud is where the operating system really shines. Management of virtual environments and resources, especially in conjunction with Microsoft System Center 2012, is fully integrated into all levels of the operating system. Hyper-V v.3, Microsoft's latest version of its hypervisor technology, fully integrates PowerShell for local and remote management of all virtual systems. This eases the burden of virtual management by allowing fully scripted and automated solutions.

Windows Server 2012 with Hyper-V also expands the ability to access resources, without the limits on physical versus virtual process imposed in Windows Server 2008. This means that the only limit to virtual machines is the limits of the hardware. Storage management has also become easier in Windows Server 2012 with scalable virtual disks and a new virtual disk format, VHDX. With VHDX, Hyper-V can use virtual fiber channel connections to SMB storage devices. VHDX also allows virtual disks to be merged in real time without taking the system down.

Hyper-V Clustering and Replication

Hyper-V replication is simple in Windows Server 2012, requiring only that a snapshot be sent to the remote site and then enabling replication. Asynchronous replication can support active–passive failover scenarios as well as active–active options between sites. Failover is automatic, with fully integrated updating of IP addressing to the backup virtual machine, allowing for near real-time disaster recovery. Migration of virtual machines can also be done in real time now, without the normal associated downtime and with no shared data between migrating virtual machines.

Clustering in Hyper-V is also greatly enhanced from Windows Server 2008. Windows Server 2008 R2 allowed clusters of up to 16 nodes. A Hyper-V failover cluster was limited to 1,000 virtual machines across all the nodes in the cluster. Any single node in the cluster was limited to running a maximum of 384 virtual machines. Windows Server 2012 now supports up to 64 nodes in a cluster and up to 4,000 virtual machines across the cluster. A single node in the cluster can run a maximum of 1,024 virtual machines. A cost savings for many organizations is that clustering is now included in Windows Server 2012 Standard Edition at no extra charge.

Hyper-V Virtual Networking

Networking in Windows Server 2012 has also been drastically modified to allow for complete virtual networking. Isolated virtual networks can now be created with the same physical infrastructure, a process that could barely be imitated on Windows Server 2008. Windows Server 2012 introduces functions such as DHCP Guard, which prevents a virtual server from exposing services to other virtual networks. This allows for isolating multitenant networks and controlling bandwidth use on the virtual networks, valuable to both hosters and those organizations where a single server farm handles multiple subsidiaries.

Unified Remote Access

Remote access for Windows users has gone from being a convenience to being a necessity, both for mobile clients as well as administrators. Previous Windows Server versions had three separate technologies, virtual private networks (VPNs) and DirectAccess, as well as cross-premises connectivity. In Windows Server 2012, DirectAccess becomes the connection technology, whether the client is using a Windows device or VPN to connect. With wizards to walk the user or administrator through the process, DirectAccess allows for remote client access to systems behind Network Address Translation (NAT) firewalls as well as DMZ use, and simplifies end-user connectivity.

TLS/SSL

The Schannel security support provider (Schannel SSP) provides Transport Layer Security (TLS), Secure Sockets Layer (SSL), and Datagram Transport Layer Security (DTLS) authentication protocols for Windows Server 2012 and adds support for Server Name Indicator (SNI) extensions. Both

SNI and DTLS directly affect the implementation and configuration of IIS 8.0 under Windows Server 2012. SSL and TLS are covered in Chapter 15, “SSL and TSL.”

SNI

In Windows Server 2008 and IIS 7.0, a common issue was running multiple virtual web servers on a single server with multiple certificates handled by the one server. During an SSL request, the client requests a certificate from the server to secure the communication and then uses that certificate to encrypt communication to the server. In versions before Windows Server 2012, the client did not inform the server of the target domain during this negotiation, so the server could only issue a single certificate for the request.

Using SNI, the client informs the server of the target domain when it requests the certificate, allowing the server to send a certificate targeted for the specific website so that the secure session is established to the correct virtual web server. In its simplest form, SNI allows an IIS 8.0 server to host multiple SSL sites and certificates on a single IP address, allowing SSL with host headers with individual SSL certificates for each site. In Windows Server 2008 R2, using SSL on sites with host headers allowed for only a single, wild-card certificate across all sites.

DTLS

DTLS comes into play with streaming media, which often uses datagrams for applications such as videoconferencing. DTLS allows secure communications using the Windows Security Support Provider Interface (SSPI). Note that applications must be designed for this functionality, but this now allows secure sessions for gaming applications, video streaming, and other datagram uses.

IIS 8.0 FEATURES

IIS 8.0 has a number of new features and improvements, some of which have been released for IIS 7.5 as out-of-band updates on `www.iis.net`. Many of these new features are due to the updates within the new operating system, though, and cannot be ported back to older versions of IIS. The Application Warm-Up module, for example, was released for IIS 7.5 and is built into IIS 8.0, but a central store for SSL certificates requires Windows Server 2012 and is available only on IIS 8.0.

SSL Changes

Changes to SSL within Windows Server 2012 naturally affect IIS 8.0 as well. In IIS 8.0, certificates are no longer restricted to a site but are managed through a central certificate store, making management of multiple sites in large web farms far less time-consuming. In addition, SSL certificates no longer need to be bound to an IP address, and deployment or configuration of SSL can now be done through simple PowerShell cmdlets. SSL and TLS are covered in Chapter 15.

CPU Throttling

The CPU throttling process in IIS 8.0 has been improved, allowing sites to use more CPU when needed but throttling CPU cycles back to preset limits when there is contention between sites for the CPU. IIS 7.0 on Windows Server 2008 would simply kill a process that required too much CPU, effectively making CPU throttling a dangerous practice on heavily used servers. In IIS 8.0, administrators can set a limit for CPU use that the system will allow a process to exceed if the CPU is available. Otherwise, the process is restricted to the limit and not summarily killed.

Application Warm-Up

The Application Warm-Up module was released for IIS 7.5 and Windows Server 2008 R2, and has been fully implemented in IIS 8.0 on Windows Server 2012. The application is identical; the only real difference is that it ships as part of the server operating system now and there's no need to install it as an add-on module. If you are upgrading from Server 2008, you will need to remove the IIS 7.5 version and upgrade to the shipping version to avoid conflicts.

Application Warm-Up fixes issues with complex applications that take significant time to load caches or generate content for the first HTTP request by allowing administrators to preload or preconfigure those tasks. In addition, the Application Warm-Up module enables administrators to configure a splash page that is displayed to end users while the application is starting. In previous situations, programmers needed to write the routines to handle this; otherwise, users would essentially see a dead browser.

WebSocket

WebSocket is a W3C-standardized API that allows full-duplex bidirectional communications over a single IP address and port. WebSocket requires both client and server support, which is now built into Windows Server 2012 and IIS 8.0, as well as Internet Explorer 10 and above. The web application must also be written to support the WebSocket API.

The big advantage to this API support is for developers, who will now find it much easier to use HTML and connect to data sources asynchronously in cloud deployments. Connections using the WebSocket API are bidirectional and full-duplex, using a single TCP connection but sending streams of messages instead of streams of bytes, thereby greatly increasing the access speed of data connections over standard TCP connections. The WebSocket API uses the standard HTTP port 80, allowing for communication through most firewalls.

Additional Features

As in previous versions, IIS 8.0 includes FTP and SMTP services. SMTP remains unchanged from Windows Server 2008 and IIS 7.5, and FTP is nearly identical to the FTP server available for download from Microsoft's website at `www.iis.net`. Both FTP and SMTP are covered in detail in Chapter 10, "Configuring Other Services."

FTP

Windows Server 2008 shipped with exactly the same FTP code and functions found in Windows Server 2003 and IIS 6.0, whereas Windows Server 2008 R2 shipped with the updated FTP 7.5, which was released as an inline update through Microsoft's website at `www.iis.net`. FTP 7.5

included secure FTP using SSL certificates, which had been one of the primary reasons for using third-party FTP servers. In addition, FTP 7.5 integrated with the IIS management functions, including extensibility of the authentication process. This means that FTP can use ASP.NET authentication, including membership and roles features, and does not require Windows CALs.

Windows Server 2012 ships with essentially the same FTP server as in FTP 7.5, with some additional functionality. FTP is covered in detail in Chapter 10.

SMTP

SMTP is again available on Windows Server 2012, as it was on Windows Server 2008, without the need to purchase Microsoft Exchange Server. Unchanged from the Windows Server 2008 implementation, SMTP code is actually developed and owned by the Windows Exchange Server development team. The SMTP service in Windows Server 2012 is not meant to be a full-featured implementation, but rather a simplified service that provides minimum functionality without the need for additional services. Most professional users of IIS will want to install another mail server product, such as Microsoft's Exchange Server.

That doesn't mean that SMTP in Windows Server 2012 is a lightweight product. It is still functional for sending mail from applications on IIS 8.0, and it is a fully compliant implementation of SMTP that functions well in an Internet environment. While not having the configurability of Microsoft's Exchange Server, it will still function with multiple virtual servers and serve multiple SMTP domains while providing for security through relay permissions and IP restrictions as well as Windows login account access.

Chapter 10 goes into more detail on SMTP installation and configuration.