**Windows Server 2008 R2: Best Practices for Server Core Deployments**

*Working with Server Core, as opposed to a full Windows Server deployment, conveys numerous benefits like simplicity and small attack surface.*

**Brien M. Posey**

Both Windows Server 2008 and Windows Server 2008 R2 give you considerable flexibility when it comes to deployment concerns. You can deploy either as a full installation or in a Server Core configuration.

A Server Core deployment is a lightweight Windows Server deployment with a minimal footprint. Server Core deployments lack many of the components are taken for granted in full Windows Server installations, so there are numerous myths surrounding Server Core. I’ll debunk some of these myths while also providing some best practices for working with Server Core deployments. All of the following examples are based on Windows Server 2008 R2.

**Server Core Configuration**

One of the big myths surrounding Server Core is that it’s difficult to configure because you have to perform the entire configuration from the Command Prompt. Thankfully, there are a couple of tools that make configuring Server Core much easier.

Windows Server 2008 R2 comes with a built-in utility you can invoke by entering the SCONFIG command at the command prompt. This menu-driven utility lets you perform basic configuration tasks such as specifying an IP address, changing the computer name or joining a domain.

Another tool that greatly simplifies initial configuration is Server Core Configurator 2.0 (see **Figure 1**). This tool goes much further in simplifying the initial configuration than SCONFIG does. It includes the same functionality as SCONFIG, but also lets you add or remove server roles and manage various Control Panel settings. It even lets you enter a product key and activate the server. Download Core Configurator from [coreconfig.codeplex.com/releases/view/36678](http://coreconfig.codeplex.com/releases/view/36678).

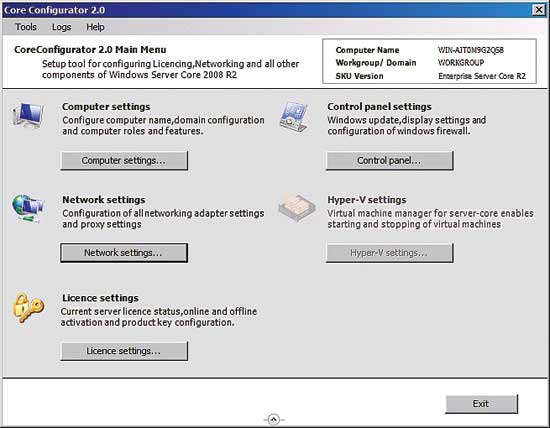


Figure 1 **The Server Core Configurator provides a GUI interface for configuring Server Core.**

**Limited Use**

Because Server Core offers a limited set of services, it’s not suitable for all of the same tasks for which you would use a full Windows Server 2008 R2 deployment. Instead, Server Core can only accommodate a limited number of roles, none of which are installed by default. The roles that you can install include:

* Active Directory Certificate Services
* Active Directory Domain Services
* Active Directory Lightweight Directory Services
* BranchCache Hosted Cache
* DHCP Server
* DNS Server
* File Services
* Media Services
* Print Services
* Web Services (IIS)

**Patch Management Is Essential**

Some IT pros have thought that because Server Core has such a small footprint, they can treat it as an appliance, rather than a real server. This is not true—patch management is an essential administrative task, regardless of whether a system is running Server Core or a full Windows Server deployment.

However, because Server Core does have such a small footprint, many of the Windows Server patches don’t apply to Server Core environments. Therefore, running Server Core can help reduce the patch management burden, but it never goes away entirely.

**Best Practices Analyzer**

Your best option for ensuring your Server Core deployment adheres to Microsoft’s recommended best practices is to use the Best Practices Analyzer. This can be a bit tricky, though, without a GUI. You have two main options for running a best practices scan. One is to run the scan directly from the Server Core desktop. The other option is to run the scan from another machine that does have a GUI.

One thing to keep in mind is that because Server Core consists of such a minimal set of services, there really isn’t any such thing as a set of best practices for configuring the core OS. That being the case, the Best Practices Analyzer is server role-specific. If your server is running only the core OS without any additional roles, you won’t be able to perform a baseline scan.

The scanning process is based on the use of BPA models. Those models are role-specific. For example, there’s a Hyper-V model and an Active Directory model.

**Command Line-Based Scan**

If you want to perform a best practices scan from the command line, you’ll need to install Windows PowerShell. The easiest way to do that is to run the SCONFIG command and then click “Configure Remote Management,” followed by “Enable Windows PowerShell” (see **Figure 2**).

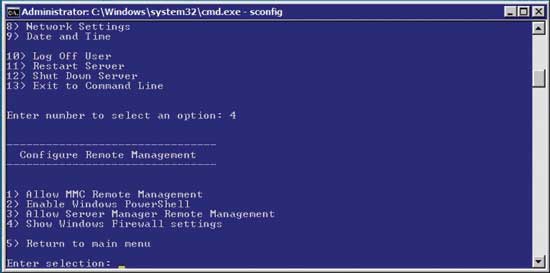


Figure 2 **You can use the SCONFIG utility to enable Windows PowerShell.**

Once you’ve installed Windows PowerShell, you’ll need to install the Server Manager cmdlets for Windows PowerShell, as well as the Best Practices Analyzer cmdlets. You can do this with the Deployment Image Servicing and Management (DISM) tool.

Normally, you’d use the DISM tool to manage deployment images, but by using the /Online switch, you can direct DISM to act against the current OS. You can install the necessary Windows PowerShell cmdlets by entering the following commands at the Command Prompt (see **Figure 3**):

1. DISM /Online /Enable-Feature /FeatureName:ServerManager-PSH-Cmdlets
2. DISM /Online /Enable-Feature /FeatureName:BestPractices-PSH-Cmdlets

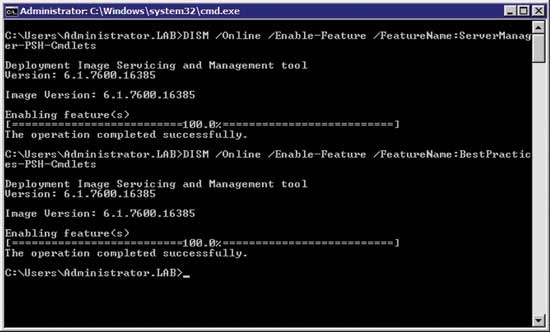


Figure 3 **You can use the DISM command to install the necessary Windows PowerShell cmdlets.**

Now that you have the necessary components in place, you can run a best practices scan. Because you have to run the scan through Windows PowerShell, enter the PowerShell.exe command. When you do, the Command Prompt window will transform into a Windows PowerShell window.

The next thing you’ll have to do is import the cmdlets you enabled earlier. Do this with two simple Windows PowerShell commands:

1. Import-Module ServerManager
2. Import-Module BestPractices

Now that all of the pieces are in place, you should verify that the necessary Best Practices Analyzer models are in place. Do this by entering the following command:

1. Get-BPAModel

The results should look something like what you see in **Figure 4**. If there were no results returned, then the Best Practices Analyzer won’t do anything if you attempt to perform a scan.

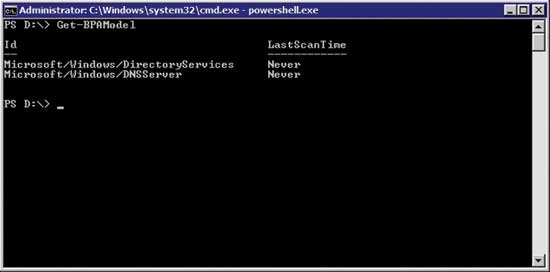


Figure 4 **You can use the Get-BPAModel command to see which models are installed.**

The easiest way to run a best practices scan is to use this command:

1. Get-BPAModel | Invoke-BPAModel

This command, which is shown in **Figure 5**, runs a best practices scan against all of the models installed on the server.

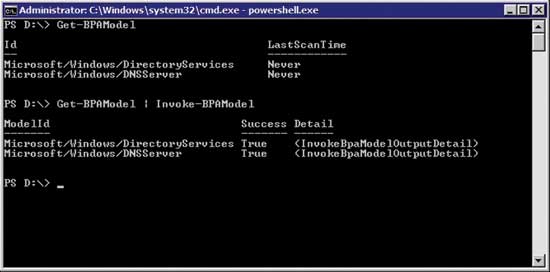


Figure 5 **The best practices scan won’t display any results if the models aren’t in place.**

Because the results weren’t displayed automatically, enter the following command to see them:

1. Get-BPAModel | Get-BPAResult | Out-File "C:\BPA.txt"

This command will create a text file with the scan results. You can copy the file to another system or you can use the following command to see the results:

1. Type C:\BPA.TXT

You can see what the scan results look like in **Figure 6**.



Figure 6 **This is what the scan results look like.**

**Performing a GUI Best Practices Scan**

You could also use a GUI to perform a best practices scan of your Server Core deployment. To do so, you’ll need to run the scan from a server running Windows Server 2008 R2 or you can use a computer running Windows 7 with the Remote Server Administration Tools. If you decide to use Windows 7, then you can download the Remote Server Administration Tools from [microsoft.com/download/en/details.aspx?displaylang=en&id=7887](http://www.microsoft.com/download/en/details.aspx?displaylang=en&id=7887).

Like the command line-based best practices scan, scanning with the GUI requires that you have Windows PowerShell installed on the core server. You’ll also need to enable Remote MMC Management and Server Manager Remote Management. You can prepare the server by entering the SCONFIG command and following these steps:

Enter 4 (Configure Remote Management)  
Enter 2 (Enable Windows PowerShell)  
Restart the server  
Log in and enter SCONFIG  
Enter 4 (Configure Remote Management)  
Enter 1 (Allow MMC Remote Management)  
Enter 3 (Allow Server Manager Remote Management)  
Enter 5 (Return to Main Menu)  
Enter 13 (Exit to Command Line)

To perform the scan, open Server Manager and right click on the listing for your server within the console tree. Then select the “Connect to Another Computer” from the shortcut menu. When prompted, enter the name of the computer you want to scan.

Once you’ve connected to the remote Server Core machine, expand the Roles container and then select the role for which you want to run a best practices scan. As you scroll through the results, you should see a section labeled Best Practices Analyzer. You can scan the role by clicking the Scan This Role link. The scan results are displayed within the Best Practices Analyzer section, as shown in **Figure 7**.

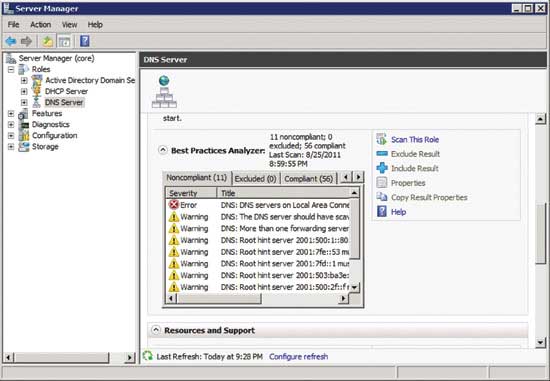


Figure 7 **You can scan a core server by using another server’s GUI.**

**Windows PowerShell**

Many of the processes described here rely on Windows PowerShell being installed on the Server Core computer. Unless you plan on running an application that requires Windows PowerShell, you should probably move it when you’re done with the initial configuration and best practices testing.

This may sound strange because there are a lot of posts on the Internet claiming that if you want to manage Server Core from the command line, you must do so through Windows PowerShell. However, Server Core is designed to be managed from a Command Prompt, not from Windows PowerShell. In fact, the first version of Server Core wouldn’t even allow Windows PowerShell to be installed. Technically, there’s nothing wrong with having Windows PowerShell installed on your server. However, most people use Server Core because of its small footprint and small attack surface because having Windows PowerShell installed increases the server’s attack surface.

You can remove Windows PowerShell by opening a Command Prompt window and entering the following command (see **Figure 8**):

1. Start /W ocsetup MicrosoftWindowsPowerShell /Uninstall

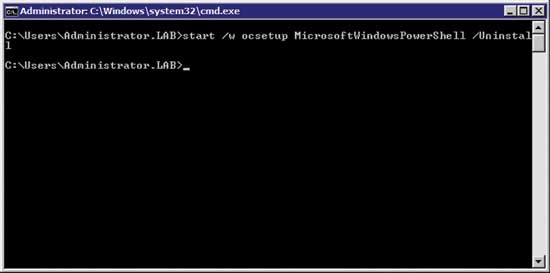


Figure 8 **You can use the Start /W OCSETUP Microsoft Windows PowerShell /Uninstall command to remove Windows PowerShell.**

Some of you may wonder why I used the Start /W OCSETUP command instead of ServerManagerCMD. There are two reasons why I didn’t use ServerManagerCMD:

* First, ServerManagerCMD is the command-line interface to Server Manager. Server Manager doesn’t exist in Server Core, so using ServerManagerCMD isn’t even an option.
* The other reason for not using ServerManagerCMD is because although the command exists in full installations of Windows Server 2008 R2, the command has been deprecated. Microsoft wants you to begin using Windows PowerShell cmdlets to add and remove features instead of relying on ServerManagerCMD. Naturally, you can’t use a Windows PowerShell command to remove Windows PowerShell.

As you can see, the techniques used for managing Server Core differ from those used for managing a typical Windows Server 2008 deployment. Even so, Server Core is ideal for use in high-security environments or when you need to conserve server resources.

Making Sense of Server Core

Greg Shields

http://i.msdn.microsoft.com/Global/Images/clear.gifContents

[Windows Server 2008, MS-DOS Edition](http://technet.microsoft.com/en-us/magazine/2009.02.geekofalltrades.aspx?pr=blog#id0080005)  
[The Cheat Sheet](http://technet.microsoft.com/en-us/magazine/2009.02.geekofalltrades.aspx?pr=blog#id0080012)  
[Core Management](http://technet.microsoft.com/en-us/magazine/2009.02.geekofalltrades.aspx?pr=blog#id0080071)

In the world of IT publications, the needs of the jack-of-all-trades IT professional are perhaps the least represented. With all the books and articles that explain today's technology in deep and gory detail, too often the needs of the smaller-environment IT pro don't get met. Does this describe you? Are you ultimately responsible for every facet of your computing environment? Is it your job to manage everything from network routing to server administration to changing toner cartridges in the printers?

If so, this column is dedicated to you. My name is Greg Shields and I've spent the last 15 years working in IT environments both large and small, and have devoted the last few to writing about what I've seen and learned. I'm excited about this monthly opportunity to share the best real-world tips and tricks that you—the master of none—can implement today to improve your small IT environment.

You'll note that I'm specifically not saying "your small business" here. While those of us in small businesses today typify the jack-of-all-trades IT professional, there are others whose job it is to manage small parts of a much bigger organization. No matter, if you're in charge of it all, you've likely got a "just-get-it-done" outlook on work. In this column, I'll attempt to tell you exactly how to do that, and hopefully get you home on time every night as an added bonus.

To start you out right, this first column will talk about a new form of Windows Server you might not have paid attention to yet. Server Core in Windows Server 2008 is a compelling new addition that makes sense for the small computing environment. In this column, I'll tell you why that might be the case and where it might fit best. I'll also take you through the steps you'll need to set up your first instance of Server Core on Windows Server 2008.

**Windows Server 2008, MS-DOS Edition**

The MSDN Web site refers to Server Core as "a minimal server installation option for computers running on the Windows Server 2008 operating system. Server Core provides a low-maintenance server environment with limited functionality." Lots of words, but what they really say is that Server Core arrives as another installation option for Windows Server 2008, one that strips away vast amounts of functionality from the operating system. There is no Explorer shell. There is no Internet Explorer. There is no support for managed code.

Server Core is one result of some componentization activities that were completed by Microsoft in the development of Windows Server 2008. From a high-level perspective, this componentization looked at the codebase that was Windows Server and broke it down into logical groups that connect to each other. The detailed knowledge thus gained of the components that make up Windows Server, along with the interfaces between those components, better enabled Microsoft to make decisions about which components could be safely set side.

By stripping Windows down to its bare essentials, even eliminating nearly all its graphical user interfaces, the end result is a smaller OS with a reduced attack surface and fewer requirements for hardware resources. Those reduced hardware requirements mean Server Core can be installed on less-powerful servers. With Server Core, you can continue using older hardware that you might otherwise dispose of.

As noted earlier, this slimmer and trimmer OS eliminates virtually every graphical tool used to administer a server at its console. For all practical purposes, Server Core can be (jokingly) considered Windows Server 2008, MS-DOS Edition. So if you're to use it in your environment, you'll need to dust off your old command-prompt skills and relearn the text commands you may not have used in years.

For some, that's the bad news. But once you've gotten through that initial re-learning process, you'll find you've developed some command-line techniques you can immediately use in managing the other "real" Windows servers in your environment. This is the case because, with few exceptions, every command you use to manage your Server Core instance can be used on the full Windows Server to accomplish the same task. More command-line management results in more scripts and batch files, which lets you solve problems and administer your environment both faster and with more reliability.

If Server Core's command-line requirements still scare you, you will be happy to know that most of the command-line configuration happens when you first build your server. Once the server is built, networked, and connected to your domain, you'll use your regular administrative consoles like those you use for your other servers.

**The Cheat Sheet**

Toward that end, let me help you through your first Server Core installation. This process includes all the commands that you'll need to run from the command prompt after the initial installation in order to get your server ready for operation.

Server Core arrives not as a separate edition of Windows Server 2008, but instead as an installation option. This means that no matter what edition of Windows Server you purchase—Standard, Enterprise, or Datacenter—you have the option of installing that instance as either a full version or a Server Core version. You'll see these options when you drop the installation DVD into a candidate server's drive and boot the machine.

After answering some initial questions about the installation, you'll see a screen similar to **Figure 1**, where you can select the installation option for the edition you've chosen. In this case, we'll install the Standard Edition of Windows Server 2008 with the Server Core Installation option. Finish answering WinPE's preinstallation questions to kick off the installation.

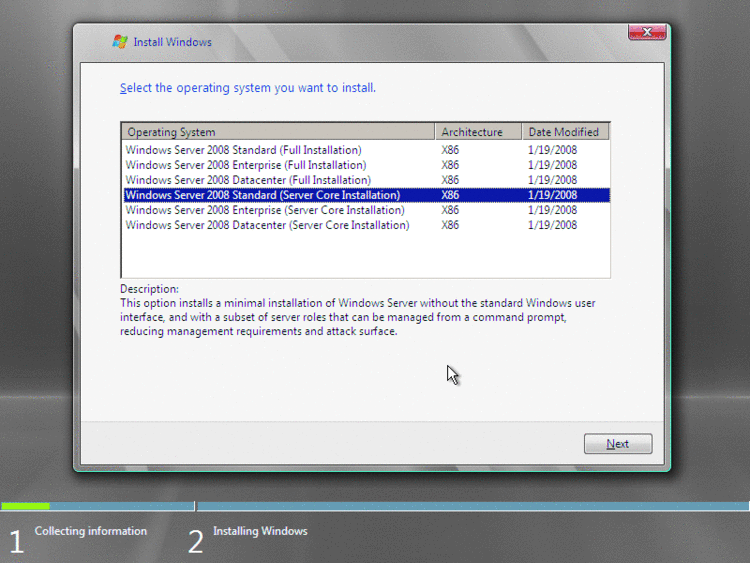


Figure 1 **Server Core is an installation option**

Once the installation routine completes, log in as administrator with a blank password. You'll immediately be asked to change that password. When that's done, you'll be logged in and presented with Server Core's undeniably minimalist user interface. **Figure 2** shows the sparse style of the console of your Server Core instance.

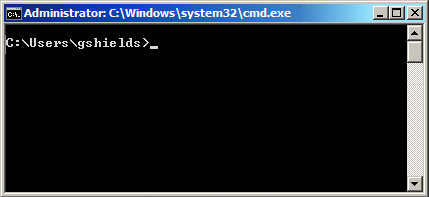


Figure 2 **Server Core’s interface shows little more than a command prompt**

A freshly installed Server Core instance arrives with none of the needed settings preconfigured. To get that instance on the network, in your domain, and ready for work involves a set of configurations that you'll accomplish through the command prompt. Though this may sound daunting, feel free to use what follows as a cheat sheet to get you through the initial configuration:

**Change the Computer's Name** Your computer may have some random name. Change that name with the following command:

[Copy Code](javascript:CopyCode('ctl00_MTContentSelector1_mainContentContainer_ctl04_code');" \o "Copy Code)

netdom renamecomputer %computername%

/newname:{newComputerName}

After that, you need to reboot the computer, which you can do with the following command:

[Copy Code](javascript:CopyCode('ctl00_MTContentSelector1_mainContentContainer_ctl05_code');" \o "Copy Code)

shutdown /r /f /t 0

**Set Static IP Address, Subnet Mask, and Default Gateway** If you use DHCP for your servers, ignore this step. To set a static IP address and configure the default gateway for your server, use this command:

[Copy Code](javascript:CopyCode('ctl00_MTContentSelector1_mainContentContainer_ctl06_code');" \o "Copy Code)

netsh interface ipv4 set address {interfaceName} static

{ipAddress} {subnetMask} {defaultGateway}

For most servers the value for {interfaceName} will be set to "Local Area Connection." You can verify that name first with this command:

[Copy Code](javascript:CopyCode('ctl00_MTContentSelector1_mainContentContainer_ctl07_code');" \o "Copy Code)

netsh interface show interface

**Set a DNS Server** Finish up your network configuration by setting your server's DNS server address. Do it with this command:

[Copy Code](javascript:CopyCode('ctl00_MTContentSelector1_mainContentContainer_ctl08_code');" \o "Copy Code)

netsh interface ipv4 add dnsserver name={interfaceName}

address={dnsServerIpAddress} index=1

**Join the Domain** Once the computer's name and networking are set, join it to the domain with this command:

[Copy Code](javascript:CopyCode('ctl00_MTContentSelector1_mainContentContainer_ctl09_code');" \o "Copy Code)

netdom join {computerName} /domain:{domainName}

/userD:{domainUsername} /passwordD:{domainPassword}

Note that you'll need to reboot the computer again after you've completed this process.

**Configure a Proxy Server** If your organization makes use of a proxy server to connect to the Internet, set its configuration with this command:

[Copy Code](javascript:CopyCode('ctl00_MTContentSelector1_mainContentContainer_ctl10_code');" \o "Copy Code)

netsh winhttp set proxy {proxyServerName}:{proxyPortNumber}

Although Server Core doesn't have the ability to run a Web browser, environments that use proxies will need to configure this setting if they will be using Automatic Updates to download updates from the Internet.

**Add and Activate a License Key** Server Core doesn't include Server Manager, so the command line is the mechanism for licensing your server. Enter its license key with this command:

[Copy Code](javascript:CopyCode('ctl00_MTContentSelector1_mainContentContainer_ctl11_code');" \o "Copy Code)

slmgr –ipk {licenseKey}

You can separate the character strings in that key with dashes. Activate that license with this command:

[Copy Code](javascript:CopyCode('ctl00_MTContentSelector1_mainContentContainer_ctl12_code');" \o "Copy Code)

slmgr –ato

Be patient during the long pause you are likely to experience after entering these commands, before you see a pop-up window with their result.

**Configure Automatic Updates** Automatic Updates can be configured via the command line or through Group Policy. Note that it is usually easier to use Group Policy to configure your Server Core instance along with the rest of your environment. However, if your Group Policy configuration for Automatic Updates is set to notify before downloading or installing, it will not work with Server Core.

With no explorer shell, the balloon notifications used by Automatic Updates do not function with Server Core. However, you can use the screg­edit.wsf script found in C:\Windows\System32 to set Automatic Updates to download updates and install them at 3:00 A.M. every day, like so:

[Copy Code](javascript:CopyCode('ctl00_MTContentSelector1_mainContentContainer_ctl13_code');" \o "Copy Code)

Cscript c:\windows\system32\scregedit.wsf /au 4

**Manage the Windows Firewall with Advanced Security** As with Automatic Updates, the Windows Firewall with Advanced Security is best managed with Group Policy. However, if you wish to turn the firewall on or off on a standalone instance, you can do so with this command:

[Copy Code](javascript:CopyCode('ctl00_MTContentSelector1_mainContentContainer_ctl14_code');" \o "Copy Code)

netsh advfirewall set {profileName} state {on|off}

Use the command

[Copy Code](javascript:CopyCode('ctl00_MTContentSelector1_mainContentContainer_ctl15_code');" \o "Copy Code)

netsh advfirewall set /?

to determine the values for {profileName}.

**Enable Remote Desktop** Finally, if you ever want to manage your Server Core instance's console via Terminal Services, you would use this command:

[Copy Code](javascript:CopyCode('ctl00_MTContentSelector1_mainContentContainer_ctl16_code');" \o "Copy Code)

cscript c:\windows\system32\scregedit.wsf /ar 0

That's it. Now you're ready to install whatever components you need onto your Server Core instance. Remember that Server Core is limited to only a subset of the possible roles that can be installed to a full Windows Server 2008 instance:

* Active Directory Domain Services
* Active Directory Lightweight Domain Services
* DHCP Server
* DNS Server
* File Services
* Media Services
* Print Services
* Hyper-V
* Web Services (IIS) (Limited)

The limited roles available mean that your Server Core instance will likely fulfill only a few needs in your environment. Server Core instances make great lightweight infrastructure servers like domain controllers, DNS servers, or file servers, for example. Because you need only limited direct, on-console interaction with these types of roles, you can generally get by with lesser and/or older hardware.

Server Core's reduced hardware also needs help when hosting Hyper-V. With Hyper-V, your goal is to squeeze as much performance out of your hardware as possible for the running of virtual machines. Since a primary partition operating system instance is required for the processing of Hyper-V's virtualization layer, Server Core's more limited hardware needs means more hardware resources will be available for the operation of your virtual machines.

To take advantage of this, there are two commands you'll need to know to install components into Server Core. The first, oclist, lists the available and installed components on the server. The second, ocsetup {componentName} installs a desired component. For example, to install the Hyper-V role, use this command:

[Copy Code](javascript:CopyCode('ctl00_MTContentSelector1_mainContentContainer_ctl17_code');" \o "Copy Code)

ocsetup Microsoft-Hyper-V

Be aware that unlike virtually every Windows command, ocsetup is case-sensitive. Thus, to install a component, you'll need to use oclist to get the exact name with proper case to later install it with ocsetup.

Another command, Wusa, is handy for the installation of patches. To manually install MSU or MSP patches, use this command:

[Copy Code](javascript:CopyCode('ctl00_MTContentSelector1_mainContentContainer_ctl18_code');" \o "Copy Code)

wusa {patchFileName}

**Core Management**

Once Server Core is installed, the remainder of your administration is likely to be done through the management consoles already installed to your desktop. For example, once the Hyper-V role is installed, all other work with Hyper-V is then done through the Hyper-V Manager. This makes working with Server Core instances easy once they're built.

On occasion, you'll still need to interact with the server itself to work with files or install a new component or patch. The most obvious and easiest way to accomplish this is by connecting to the server using Terminal Services. But there is another set of command-line tools, the Microsoft PsTools, that can help.

The PsTools have long been used by smart administrators to accomplish command line-based actions remotely on systems all around the network. The tools are of particular use with Server Core since its installation is all about the command line.

If you haven't yet downloaded these tools, you can get them from the PsTools site at technet.microsoft.com/sysinternals/bb896649.aspx. The download includes an HTML help file with usage information for all of the tools.

Then copy them after unpacking to a location in the path of your management desktop. Helpful hint: I copy them to c:\windows where they'll always be available when I open a command prompt.

There are many PsTools, but two in particular are useful for working with Server Core. The first, PsExec, provides a way for you to run a command on another computer remotely. This is useful for initiating individual commands on remote Server Core instances, but even more compelling when you use it to launch a remote command prompt. As shown in **Figure 3**, you can do this from your management desktop with this command:

**psexec \\{computerName} cmd**

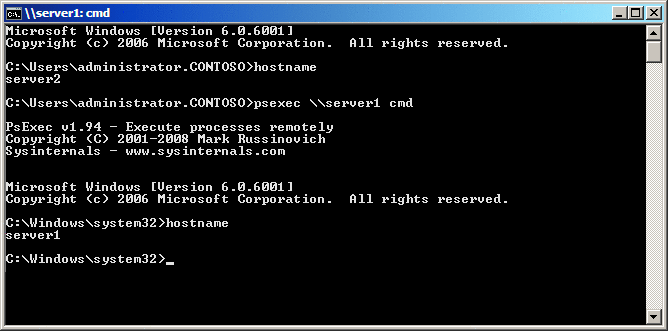


Figure 3 **PsExec brings Server Core’s command prompt to your local desktop**

If you run this command against your Server Core instance, you'll see that your command prompt happens to be the very same command prompt on the remote server.

Sometimes you merely want to discover how processes are behaving on your server. Maybe a process is using more than its fair share of the processor or memory. PsList can help. To launch a regularly updating text-based view of processes and view their resource use (as shown in **Figure 4**), use this command:

**pslist \\{computerName} /s**

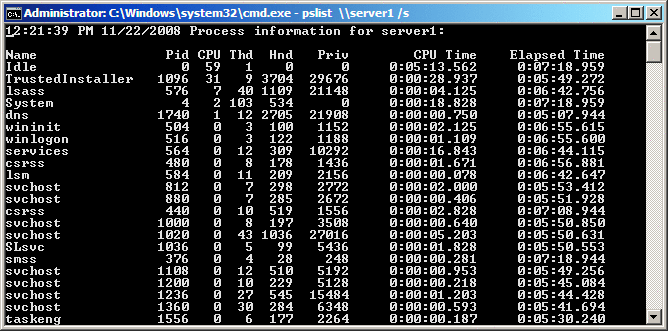


Figure 4 **PsList lets you display a list of running processes**

The resulting list of processes repeatedly updates and is ordered by processor use. If you happen to find some processes that are misbehaving, you can kill any offending processes with this command:

[Copy Code](javascript:CopyCode('ctl00_MTContentSelector1_mainContentContainer_ctl23_code');" \o "Copy Code)

**pskill \\{computerName} {processNameOrPid}**

Finally, you will also find the two scripting languages VBScript and Windows PowerShell very useful for managing your Server Core instance. While Server Core comes equipped with the necessary VBScript runtime code to launch commands directly from its console, it does not presently include the ability to use Windows PowerShell. So, while you can run Windows PowerShell commands on your management desktop against a Server Core instance, you can't directly launch them within your Server Core instance.

So, there you have it. I've provided you with enough information to get started using Server Core in your small environment today.

Got a problem in your small environment or something you'd like a little help or insight into? Send your questions and suggestions to me at [gshields@concentratedtech.com](mailto:gshields@concentratedtech.com).

**Administer Windows Server 2008 Server Core from the Command Prompt**

To become an expert in administering Server Core from the command prompt, you need the following:

* Familiarity with using the command-line tools included in Windows Server 2008
* Knowledge of which tools can be used to administer a particular role or feature, as well as to perform a particular type of task

Here’s a quick guide to some of the more useful command-line tools you can use for performing specific types of administrative tasks in Server Core. The list is not meant to be complete and the task areas in it are not ordered in any particular way. Also, while most of the tools listed in this table are command-line tools, a few of them are scripts or GUI tools.

Tips RSS Feed

Subscribe to the [TechNet Magazine Tips RSS feed](http://technet.microsoft.com/magazine/rss/default.aspx?z=z&Tips=1).

Task: View system information

* Msinfo32
* Set
* Systeminfo

Task: View user information

* Whoami

Task: Manage users and groups

* Net accounts
* Net group
* Net localgroup
* Net user

Task: View or change computer name

* Hostname
* Netdom renamecomputer

Task: Join or leave a domain

* Netdom join

Task: Log off or shut down

* Logoff
* Shutdown

Task: Configure networking

* Ipconfig
* Netsh interface
* Netsh routing
* Route

Task: Configure Windows Firewall

* Netsh advfirewall

Task: Configure Internet Protocol security (IPsec)

* Netsh ipsec
* Scregedit.wsf

Task: Activate Windows

* Slmgr.vbs

Task: Manage services

* Net continue
* Net pause
* Net start
* Net stop
* Sc
* Tasklist

Task: Manage processes

* Taskkill
* Tasklist
* Taskmgr

Task: Manage tasks

* At
* Schtasks

Task: Collect and analyze performance data

* Logman
* Relog
* Typeperf

Task: View events and manage event logs

* Wevtutil

Task: Manage disks and storage

* Compact
* Defrag
* Diskpart
* Diskraid
* Mountvol

Task: Manage Volume Shadow Copy Service (VSS)

* Vssadmin
* Cacl

Task: Manage file systems and file permissions

* Convert
* Fsutil
* Icacls
* Takeown

Task: Manage files

* Openfiles
* Sigverif

Task: Manage shares and share permissions

* Net share

Task: Manage the registry

* Reg
* Regedit

Task: Install and manage drivers

* Driverquery
* Pnputil
* Sc

Task: Install and manage updates

* Pkgmgr
* Scregedit.wsf
* Systeminfo
* Wuauclt
* Wusa

Task: Install roles and features

* Oclist
* Ocsetup

Task: Install applications

* Msiexec

Task: Manage Group Policy

* Gpresult
* Gpupdate
* Secedit

Task: Manage certificates

* Certreq
* Certutil

Task: Manage Terminal Services (Remote Desktop for Administration)

* Change
* Logoff
* Msg
* Mstsc
* Qappsrv
* Qprocess
* Query
* Qwinsta
* Reset session
* Rwinsta
* Shadow
* Tscon
* Tsdiscon
* Tskill

From the Microsoft Press book [*Windows Server 2008 Server Core Administrator’s Pocket Consultant*](http://www.microsoft.com/learning/en/us/books/12977.aspx) by Mitch Tulloch.

**Commands and Tools for Managing Windows Server 2008 Server Core**

Full server and server core installations are different when it comes to local console administration. With a full server installation, you have a UI that includes a full desktop environment for local console management of the server. With a core server installation, you have a minimal UI that includes a limited desktop environment for local console management of the server. This minimal interface includes:

* Windows Logon screen for logging on and logging off
* Notepad for editing files
* Regedit for managing the registry
* Task Manager for managing tasks and starting new tasks
* Command Prompt for administration via the command line

After you log on to a core-server installation, you have a limited desktop environment with an Administrator command prompt. You can use this command prompt for administration of the server. If you accidentally close the command prompt, you can start a new command prompt by following these steps:  
**1.** Press Ctrl+Shift+Esc to display Task Manager.  
**2.** On the Applications tab, click New Task.  
**3.** In the Create New Task dialog box, type cmd in the Open field and then click OK.  
  
You can start Notepad and Regedit directly from a command prompt by entering **notepad.exe** or **regedit.exe** as appropriate. To open Control Panel, type **intl.cpl**.

Tips RSS Feed

Subscribe to the [TechNet Magazine Tips RSS feed](http://technet.microsoft.com/magazine/rss/default.aspx?z=z&Tips=1).

At the command prompt, you’ll find that you have all the standard commands and command-line utilities available for managing the server. However, keep in mind that commands, utilities, and programs will only run if all of their dependencies are available in the core-server installation.  
  
While core-server installations support a limited set of roles and role services, you can install most features. The key exceptions are those that depend on the .NET Framework. Because the Microsoft .NET Framework is not supported in the original implementation, you cannot add features such as Windows PowerShell. And you can use Terminal Services to manage a core-server installation remotely.  
  
Here is an overview of key commands and utilities you’ll use for managing server core installations while logged on locally:   
**Control desk.cpl**- View or set display settings.  
**Control intl.cpl**- View or set regional and language options, including formats and the keyboard layout.  
**Control sysdm.cpl**- View or set system properties.  
**Control timedate.cpl**- View or set the date, time, and time zone.  
**Cscript slmgr.vbs –ato**- Activate the operating system.  
**DiskRaid.exe**- Configure software RAID.  
**ipconfig /all**- List information about the computer’s IP address configuration.  
**NetDom RenameComputer**- Set the server’s name and domain membership.  
**OCList.exe**- List roles, role services, and features.  
**OCSetup.exe**- Add or remove roles, role services, and features.  
**PNPUtil.exe**- Install or update hardware device drivers.  
**Sc query type=driver**- List installed device drivers.  
**Scregedit.wsf**- Configure the operating system. Use the /cli parameter to list available configuration areas.  
**ServerWerOptin.exe**- Configure Windows Error Reporting.  
**SystemInfo**- List the system configuration details.  
**WEVUtil.exe**- View and search event logs.  
**Wmic datafile where name=“FullFilePath” get version**- List a file’s version.  
**Wmic nicconfig index=9 call enabledhcp**- Set the computer to use dynamic IP addressing rather than static IP addressing.  
**Wmic nicconfig index=9 call enablestatic(“IPAddress”), (“SubnetMask”)**- Set a computer’s static IP address and network mask.  
**Wmic nicconfig index=9 call setgateways(“GatewayIPAddress”)**- Set or change the default gateway.  
**Wmic product get name /value “**- List installed MSI applications by name.  
**Wmic product where name=“Name” call uninstall**- Uninstall an MSI application.  
**Wmic qfe list**- List installed updates and hotfixes.  
**Wusa.exe PatchName.msu /quiet**- Apply an update or hotfix to the operating system.