**Windows Server Administration Tools**

**Introduction**

Completed100 XP

* 2 minutes

Select the most appropriate Windows Server administration tool for a given situation and learn how to use that tool.

**Scenario**

Contoso, Ltd. is a financial services company in Seattle with major offices located throughout the world. Most of its compute environment runs on-premises on Windows Server. This includes virtualized workloads on Windows Server 2016 hosts.

Contoso IT staff are migrating Contoso on-premises servers to Windows Server 2022. As a Windows Server administrator, you are responsible for managing and maintaining the server infrastructure that will help Contoso achieve its business goals. You decide to investigate the available Windows Server administration tools, including Windows Admin Center, Remote Server Administration Tools, Server Manager, and Windows PowerShell.

After completing this module, you’ll be able to describe the different administration tools that are available to manage a Windows Server deployment, and select the appropriate tool for a given situation.

**Learning objectives**

After completing this module, you'll be able to:

* Describe Windows Admin Center.
* Describe how to use Remote Server Administration Tools (RSAT) to manage servers.
* Describe Server Manager.
* Describe how to use Windows PowerShell to manage servers.
* Explain how to use Windows PowerShell to remotely administer a server.

**Prerequisites**

To get the best learning experience from this module, you should have knowledge and experience of:

* Windows Server.
* Basic security best practices.
* Windows client operating systems such as Windows 10.
* Working with command-line tools.

**Next unit: Explore Windows Admin Center**

**Explore Windows Admin Center**

**200 XP**

**10 minutes**

**As a senior Windows Server administrator at Contoso, you're responsible for planning a new approach to performing server administration. In the past, managing and administrating the IT environment involved using different tools across multiple consoles. Windows Admin Center combines those tools into a single console that can easily be deployed and accessed through a web interface. You decide to investigate Windows Admin Center.**

**Overview**

**Windows Admin Center is a modular web application comprised of the following four modules:**

**Server manager**

**Failover clusters**

**Hyper-converged clusters**

**Windows 10 clients**

**Note**

**Windows Admin Center manages servers that run Windows Server 2008 R2 and newer. If you want to manage servers other than the local server, you must add those other servers to the console.**

**Windows Admin Center has two main components:**

**Gateway. The Gateway manages servers through PowerShell remoting and Windows Management Instrumentation (WMI) over Windows Remote Management (WINRM).**

**Web server. The Web server component observes HTTPS requests and serves the user interface to the web browser on the management station. This is not a full install of Internet Information Services (IIS), but a mini Web server for this specific purpose.**

**Important**

**Because Windows Admin Center is a web-based tool that uses HTTPS, it requires a X.509 certificate to provide SSL encryption. The installation wizard gives you the option to either use a self-signed certificate or provide your own SSL certificate. The self-signed certificate expires 60 days after it is created.**

**Benefits of Windows Admin Center**

**The following table describes the benefits of Windows Admin Center:**

**Expand table**

**Benefit Description**

**Easy to install and use You can download and install it on Windows 10 or Windows Server through a single Windows Installer (MSI) and access it from a supported web browser.**

**Compliments existing solutions It does not replace but compliments existing solutions such as Remote Server Administration Tools, System Center, and Azure Monitor.**

**Manage from the internet It can be securely published to the public internet so you can connect to and manage servers from anywhere.**

**Enhanced security Role-based access control lets you fine-tune which administrators have access to which management features. Gateway authentication provides support for local groups, Active Directory groups, and Microsoft Entra groups.**

**Azure integration You can easily get to the proper tool within Windows Admin Center, then launch it to the Azure portal for full management of Azure services.**

**Extensibility A Software Development Kit (SDK) will allow Microsoft and other partners to develop new tools and solutions for more products.**

**No external dependencies Windows Admin Center doesn't require internet access or Microsoft Azure. There is no requirement for IIS or SQL server and there are no agents to deploy. The only dependency is to the requirement of Windows Management Framework 5.1 on managed servers.**

**Supported platforms and browsers**

**You can install Windows Admin Center on Windows 10 and Windows Server.**

**Important**

**Windows Admin Center is not supported on domain controllers and will return an error if you try to install it.**

**After downloading and installing Windows Admin Center, you must enable TCP port 6516 on the local firewall.**

**Note**

**You are prompted to select a TCP port during setup.**

**The Windows browser versions of Microsoft Edge on Windows 10 and Google Chrome are tested and supported on Windows 10. Other modern web browsers have not been tested and are not officially supported.**

**Important**

**Internet Explorer is not supported and will return an error if you attempt to launch Windows Admin Center.**

**Authenticate to remote computers**

**When you connect to a remote computer, you must authenticate to that computer. If the computer you want to manage is part of the same AD DS (Active Directory Domain Services) forest, Kerberos authentication is used.**

**Where this is not the case, you must configure the target computers as trusted hosts. For example, if you use a workgroup computer installed with Windows Admin Center to administer your domain controllers.**

**Important**

**When you install Windows Admin Center on a workgroup computer, you are prompted to allow Windows Admin Center to manage the local computer's TrustedHosts setting. If you bypass this automated setting, you must configure TrustedHosts manually.**

**You can configure trusted hosts settings by using the following Windows PowerShell command in an elevated Windows PowerShell window. You can specify the remote hosts by IP, FQDN, or NetBIOS name.**

**PowerShell**

**Copy**

**Set-Item WSMan:localhost\Client\TrustedHosts -Value 'SEA-DC1.Contoso.com'**

**Tip**

**You can also use a wildcard setting: Set-Item WSMan:\localhost\Client\TrustedHosts -Value '\*'**

**Demonstration**

**The following video demonstrates how to administer Windows Server by using Windows Admin Center. The main steps in the process are:**

**Install the Windows Admin Center from a downloadable .msi installer file.**

**Verify that appropriate TCP port is configured during installation.**

**Open Microsoft Edge and open Windows Admin Center.**

**Add a domain controller to Windows Admin Center.**

**Review the options available on the Overview and Tools panes.**

**Review the following: Certificates, Performance Monitoring, Processes, Registry, Roles & Features, Scheduled Tasks, and PowerShell.**

**Quick review**

**1. Using Windows Admin Center, an administrator connects to the domain controller, SEA-DC1. The administrator wants to add a new user account to the Contoso.com AD DS domain. Which of the following procedures would not work?**

**In Windows Admin Center, connect to SEA-DC1 and then, in the navigation pane, select Active Directory. Select Create, then select User. Enter the required details and then select Create.**

**In Windows Admin Center, connect to SEA-DC1 and then, in the navigation pane, select Local users & groups. Select Create, then select User. Enter the required details and then select Create.**

**In Windows Admin Center, connect to SEA-DC1 and then, in the navigation pane, select PowerShell. After signing in, use the New-ADUser cmdlet to create a new user.**

**Check your answers**

**Having an issue? We can help!**

[Skip to main content](https://learn.microsoft.com/en-us/training/modules/describe-windows-server-administration-tools/3-use-server-manager#main)

We use optional cookies to improve your experience on our websites, such as through social media connections, and to display personalized advertising based on your online activity. If you reject optional cookies, only cookies necessary to provide you the services will be used. You may change your selection by clicking “Manage Cookies” at the bottom of the page. [Privacy Statement](https://go.microsoft.com/fwlink/?LinkId=521839) [Third-Party Cookies](https://aka.ms/3rdpartycookies)

Accept Reject Manage cookies

* [**Learn**](https://learn.microsoft.com/en-us/)
* [Documentation](https://learn.microsoft.com/en-us/docs/)
* [Training](https://learn.microsoft.com/en-us/training/)
* [Credentials](https://learn.microsoft.com/en-us/credentials/)
* [Q&A](https://learn.microsoft.com/en-us/answers/)
* [Code Samples](https://learn.microsoft.com/en-us/samples/browse/)
* [Assessments](https://learn.microsoft.com/en-us/assessments/)
* [Shows](https://learn.microsoft.com/en-us/shows/)

Top of Form

Bottom of Form

[Sign in](https://learn.microsoft.com/en-us/training/modules/describe-windows-server-administration-tools/3-use-server-manager)

[**Training**](https://learn.microsoft.com/en-us/training/)

* Products
* Career Paths
* [Browse all training](https://learn.microsoft.com/en-us/training/browse/)
* Educator Center
* Student Hub
* [FAQ & Help](https://learn.microsoft.com/en-us/training/support/)

1. [Learn](https://learn.microsoft.com/en-us/)

1. [Training](https://learn.microsoft.com/en-us/training/)

1. [Browse](https://learn.microsoft.com/en-us/training/browse/)

1. [Describe Windows Server administration tools](https://learn.microsoft.com/en-us/training/modules/describe-windows-server-administration-tools/)

Add

[**Previous**](https://learn.microsoft.com/en-us/training/modules/describe-windows-server-administration-tools/2-explore-windows-admin-center/)

* Unit 3 of 8

[**Next**](https://learn.microsoft.com/en-us/training/modules/describe-windows-server-administration-tools/4-list-remote-server-administration-tools/)

**Use Server Manager**

Completed100 XP

* 5 minutes

As an alternative to using Windows Admin Center, Contoso IT administrators want to use a standard, built-in console to manage groups of server computers. You decide to investigate Server Manager. Server Manager is the built-in management console that most server administrators are familiar with.

**Overview**

You can use the current version of Server Manager to manage the local server and remotely manage up to 100 additional servers.

**Note**

This number will depend on the amount of data that you request from managed servers and the hardware and network resources available to the system running Server Manager.

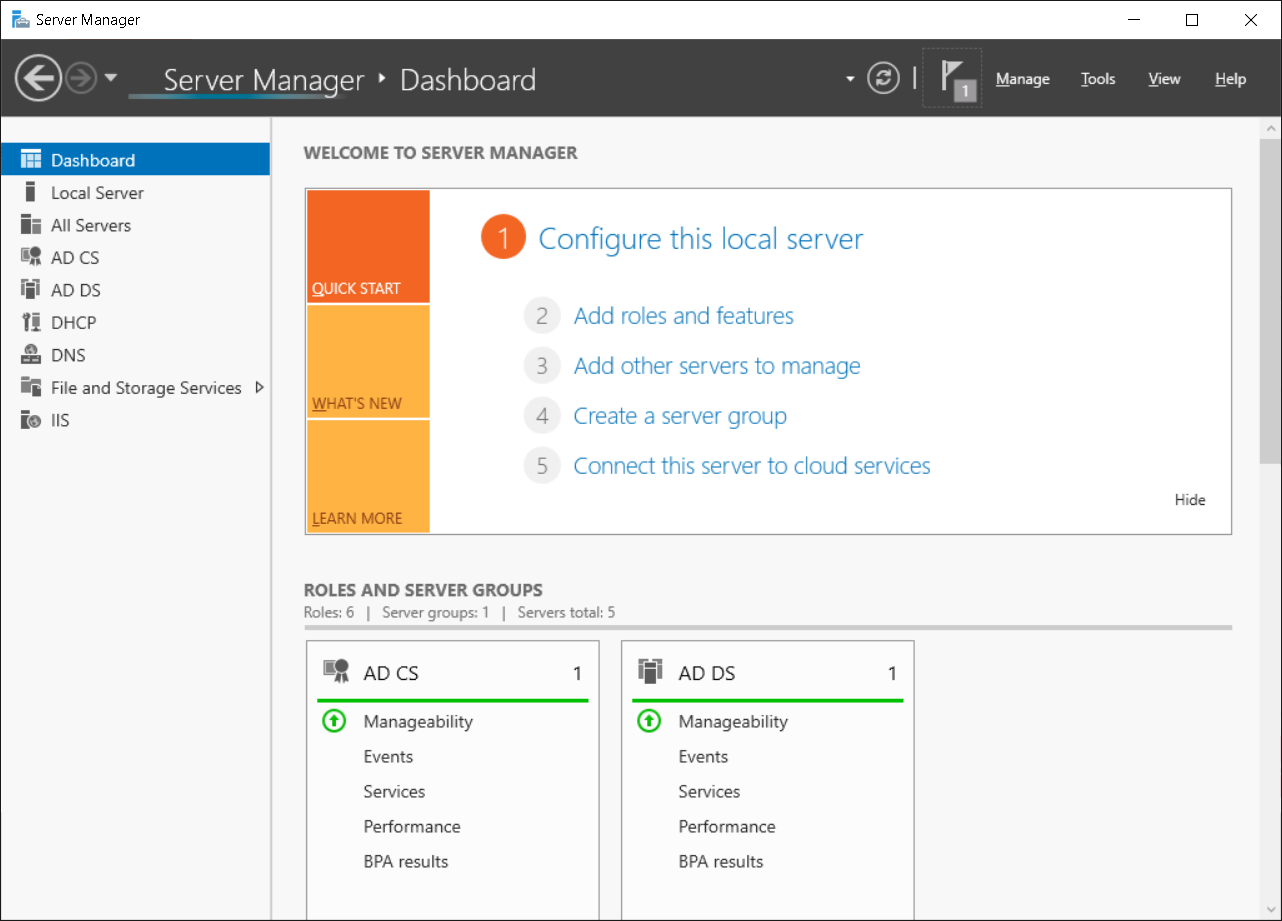
In the Server Manager console, you must manually add remote servers that you want to manage. IT administrators often use Server Manager to remotely manage server core installations.

**Note**

Server Manager is included with RSAT for Windows 10. However, you can only use it to manage remote servers. You can't use Server Manager to manage client operating systems.

Server Manager initially opens to a dashboard which provides quick access to:

* Configuring the local server.
* Adding roles and features.
* Adding other servers to manage.
* Creating a server group.
* Connecting this server to cloud services.

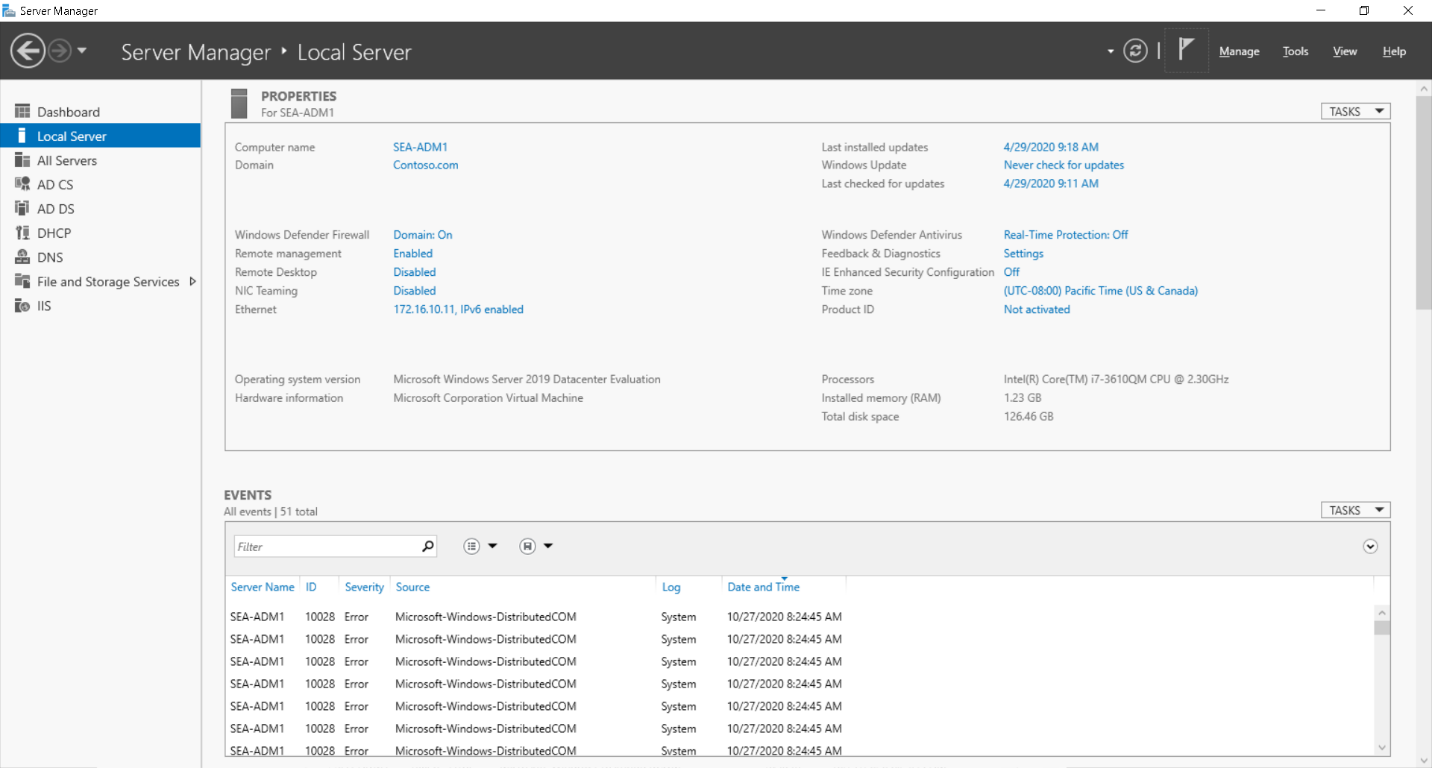


**Note**

The dashboard provides links to web-based articles about new features in Server Manager and links to learn more about Microsoft solutions.

Server Manager has a section for properties of the local server. From the Local Server node, you can perform different types of initial configuration that are similar to those you can configure with the Sconfig tool, including:

* Computer name and domain membership
* Windows Firewall settings
* Remote Desktop
* Network settings
* Windows Update settings
* Time zone
* Windows activation



This section also provides basic information about the hardware, such as:

* Operating system version
* Processor information
* Amount of memory
* Total disk space

There are also sections for:

* Querying specific event logs for various event severity levels over a specific time period.
* Monitoring the status of services and stopping and starting services.
* Best practices analysis to determine if the roles are functioning properly on your servers.
* A display of Performance Monitor that allows you to set alert thresholds on CPU and memory.
* Listing the installed roles and features with the ability to add and remove them.

The navigation pane provides a link to other roles installed on the server, which will provide information about specific roles such as events relating to that role. In some cases, you will observe a sub-menu that allows you to configure aspects about the role, such as File and Storage Services and Remote Desktop Services.

**Next unit: List Remote Server Administration Tools**

[**Continue**](https://learn.microsoft.com/en-us/training/modules/describe-windows-server-administration-tools/4-list-remote-server-administration-tools/)

**Skip to main content**

**We use optional cookies to improve your experience on our websites, such as through social media connections, and to display personalized advertising based on your online activity. If you reject optional cookies, only cookies necessary to provide you the services will be used. You may change your selection by clicking “Manage Cookies” at the bottom of the page. Privacy Statement Third-Party Cookies**

**Accept Reject Manage cookies**

**Learn**

**Documentation**

**Training**

**Credentials**

**Q&A**

**Code Samples**

**Assessments**

**Shows**

**Sign in**

**Training**

**Products**

**Career Paths**

**Browse all training**

**Educator Center**

**Student Hub**

**FAQ & Help**

**Learn Training Browse Describe Windows Server administration tools**

**Add**

**Previous**

**Unit 4 of 8**

**Next**

**List Remote Server Administration Tools**

**Completed**

**100 XP**

**5 minutes**

**Following a recent security audit by a firm specializing in IT security, you are reviewing a report on shortcomings in Contoso's procedures regarding server management. One of the significant comments in the report highlights the security impact of using locally installed, interactive management tools. You decide to investigate what other options are available for remote management. You learn that Remote Server Administration Tools (RSAT) are a group of management tools that enables IT administrators to remotely manage roles and features in Windows Server from a computer that is running Windows 10.**

**Enable RSAT**

**You no longer need to download and install RSAT on Windows 10. With Windows 10 1803 and newer, you enable RSAT from the Settings app. In Settings, search for Manage optional features, select Add a feature, and then select the appropriate RSAT tool(s) from the returned list. Select Install to add the feature.**

**You can install the consoles available within RSAT on computers running Windows 10 or on server computers that are running the Server with Desktop Experience option of a Windows Server installation.**

**Note**

**Until the introduction of Windows Admin Center, RSAT consoles were the primary graphical tools for administering the Windows Server operating system.**

**RSAT for Windows 10 consists of the full complement of available management tools including those described in the following table.**

**Expand table**

**Tool Description**

**Active Directory Certificate Services Tools Includes Certification Authority, Certificate Templates, Enterprise PKI, and Online Responder Management snap-ins.**

**Active Directory Domain Services Tools and Active Directory Lightweight Directory Services Tools Includes Active Directory Administrative Center, Active Directory Domains and Trusts, Active Directory Sites and Services, Active Directory Users and Computers, ADSI Edit, Active Directory module for Windows PowerShell, and tools such as DCPromo.exe, LDP.exe, NetDom.exe, NTDSUtil.exe, RepAdmin.exe, DCDiag.exe, DSACLs.exe, DSAdd.exe, DSDBUtil.exe, DSMgmt.exe, DSMod.exe, DSMove.exe, DSQuery.exe, DSRm.exe, GPFixup.exe, KSetup.exe, KtPass.exe, NlTest.exe, NSLookup.exe, and W32tm.exe.**

**BitLocker Drive Encryption Administration Utilities Includes Manage-bde, Windows PowerShell cmdlets for BitLocker, and BitLocker Recovery Password Viewer for Active Directory.**

**DHCP Server Tools Includes the DHCP Management Console, the DHCP Server cmdlet module for Windows PowerShell, and the Netsh command-line tool**

**DNS Server Tools Includes the DNS Manager snap-in, the DNS module for Windows PowerShell, and the Ddnscmd.exe command-line tool.**

**Failover Clustering Tools Includes Failover Cluster Manager, Failover Clusters (Windows PowerShell cmdlets), MSClus, Cluster.exe, Cluster-Aware Updating management console, and Cluster-Aware Updating cmdlets for Windows PowerShell.**

**File Services Tools Includes the following: Share and Storage Management Tools, Distributed File System Tools, File Server Resource Manager Tools, Services for NFS Administration Tools, iSCSI management cmdlets for Windows PowerShell**

**Group Policy Management Tools Includes Group Policy Management Console, Group Policy Management Editor, and Group Policy Starter GPO Editor.**

**IP Address Management (IPAM) Tools Includes tools for managing remote IPAM server.**

**Network Controller Management Tools Includes PowerShell tools for managing Network Controller on Windows Server.**

**Network Load Balancing Tools Includes the Network Load Balancing Manager, Network Load Balancing Windows PowerShell cmdlets, and the NLB.exe and WLBS.exe command-line tools.**

**Remote Access Management Tools Includes graphical and PowerShell tools for managing the Remote Access role.**

**Remote Desktop Services Tools Includes snap-ins for Remote Desktop Licensing Manager, Remote Desktop Licensing Diagnostics, and Remote Desktop Gateway Manager.**

**Server Manager Includes the Server Manager console.**

**Shielded VM Tools Includes Provisioning Data File Wizard and Template Disk Wizard.**

**Storage Migration Service Management Tools Provides management tools for storage migration tasks.**

**Storage Replica Module for Windows PowerShell Includes the PowerShell module enabling you to remotely manage the Storage Replica feature.**

**System Insights Module for Windows PowerShell Provides System Insights PowerShell module.**

**Volume Activation Tools Manages volume activation through the vmw.exe file.**

**Windows Server Update Services Tools Includes the Windows Server Update Services snap-in, WSUS.msc, and PowerShell cmdlets**

**Next unit: Use Windows PowerShell**

**Continue**

**Skip to main content**

**We use optional cookies to improve your experience on our websites, such as through social media connections, and to display personalized advertising based on your online activity. If you reject optional cookies, only cookies necessary to provide you the services will be used. You may change your selection by clicking “Manage Cookies” at the bottom of the page. Privacy Statement Third-Party Cookies**

**Accept Reject Manage cookies**

**Learn**

**Documentation**

**Training**

**Credentials**

**Q&A**

**Code Samples**

**Assessments**

**Shows**

**Sign in**

**Training**

**Products**

**Career Paths**

**Browse all training**

**Educator Center**

**Student Hub**

**FAQ & Help**

**Learn Training Browse Describe Windows Server administration tools**

**Add**

**Previous**

**Unit 5 of 8**

**Next**

**Use Windows PowerShell**

**Completed**

**100 XP**

**8 minutes**

**While GUI management tools can often be easier to use than command-line tools, Contoso's IT department manager believes you can achieve many administrative tasks more quickly by using a simple script or a single command. For example, the process of updating the same information for several user accounts by using Active Directory Users and Computers can be time consuming. However, using the Active Directory module in Windows PowerShell enables an administrator to perform this repetitive task quickly. You decide to investigate the impact of using Windows PowerShell to administer the server infrastructure at Contoso.**

**Overview**

**Windows PowerShell is a command-line and scripting environment that you can use to manage all aspects of the Windows operating system. Windows PowerShell uses special commands called cmdlets that are composed of verb-noun pairs, such as Restart-Computer.**

**You can also use Windows PowerShell to remotely connect to other computers, and even run Windows PowerShell cmdlets against a list of computers, enabling you to perform actions against multiple computers by using a single command-line instruction.**

**Windows PowerShell commands and cmdlets**

**Commands are building blocks that you piece together by using the Windows PowerShell scripting language. They provide Windows PowerShell’s main functionality. By using commands, you can create custom solutions for complex administrative problems.**

**Cmdlets are the fundamental components of commands. There are thousands of Windows PowerShell cmdlets available in the Windows operating systems and other Microsoft products. As mentioned earlier, cmdlets are comprised a verb-noun pair.**

**Cmdlet verbs**

**The verb portion of the cmdlet name indicates what the cmdlet does. There is a set of approved verbs that cmdlet creators use, which provides consistency in cmdlet names. Common verbs are described in the following table.**

**Expand table**

**Verb Explanation**

**Get Retrieves a resource, such as a file or a user.**

**Set Changes the data associated with a resource, such as a file or user property.**

**New Creates a resource, such as a file or user.**

**Add Adds a resource to a container of multiple resources.**

**Remove Deletes a resource from a container of multiple resources.**

**Cmdlet nouns**

**The noun portion of the cmdlet name indicates what kinds of resources or objects the cmdlet affects. All cmdlets that operate on the same resource should use the same noun. For example, the Service noun is used for cmdlets that work with Windows services, and the Process noun is used for managing processes on a computer.**

**Parameter format**

**Parameters modify the actions that a cmdlet performs. Each cmdlet can have no parameters, one parameter, or many parameters. Parameter names begin with a dash (-). A space separates the value that you want to pass from the parameter name. If the value that you are passing contains spaces, you will need to enclose the text in quotation marks. Some parameters accept multiple values, which are separated by commas and no spaces.**

**Examples**

**You can study the following examples to help determine how you could use Windows PowerShell to perform common administrative tasks. The following command displays a list of running services.**

**PowerShell**

**Copy**

**Get-Service | Where-Object {$\_.Status -eq "Running"}**

**The following command displays a list of services that have a name that begins with “win” and that excludes the service called WinRM.**

**PowerShell**

**Copy**

**Get-Service -Name "win\*" -Exclude "WinRM"**

**This next command outputs a list of all services to a text file formatted for HTML output.**

**PowerShell**

**Copy**

**Get-Service | ConvertTo-Html > File.html**

**A variation of the preceding command outputs only selected data about services, and then exports the output to a CSV file.**

**PowerShell**

**Copy**

**Get-Service | Select-Object Name, Status | Export-CSV c:\service.csv**

**The following command retrieves the specified information (office phone number and user principal name) about Active Directory users.**

**PowerShell**

**Copy**

**Get-ADUser -Filter \* -Properties OfficePhone | FT OfficePhone,UserPrincipalName**

**This final example retrieves a subset of AD DS users (those in the Marketing OU) and modifies their properties by adding a description to each account.**

**PowerShell**

**Copy**

**Get-ADUser -Filter 'Name -like "\*"' -SearchBase "OU=Marketing,DC=Contoso,DC=Com" | Set-ADUser -Description "Member of the Marketing Department"**

**Windows PowerShell ISE**

**The ISE is a fully graphical environment that provides a script editor, debugging capabilities, an interactive console, and several tools that help you discover and learn new Windows PowerShell commands. This module provides a basic familiarity with how the ISE works.**

**Panes**

**The ISE offers two main panes: a Script pane (or script editor) and the Console pane. You can position these one above the other or side-by-side in a two-pane layout. You can also maximize one pane and switch back and forth between the panes. By default, a Command Add-on pane also displays, which enables you to search for or browse available commands, and review and fill in parameters for a command you select. There is also a floating Command window that provides the same functionality.**

**Customizing the view**

**The ISE provides several ways to customize the view. A slider in the lower-right area of the window changes the active font size. The Options dialog box lets you customize font and color selection for many different Windows PowerShell text elements, such as keywords and string values. The ISE supports the creation of visual themes. A theme is a collection of font and color settings that you can apply as a group to customize the appearance of the ISE. There are several built-in themes that package customizations for purposes such as giving presentations. The ISE also gives you the option to create custom themes. Other ISE features include:**

**A built-in, extensible snippets library that you can use to store commonly used commands.**

**The ability to load add-ins created by Microsoft or by third parties that provide additional functionality.**

**Integration with Windows PowerShell’s debugging capabilities.**

**Windows PowerShell remoting**

**The purpose of Windows PowerShell remoting is to connect to remote computers so that you can run commands on them, and then direct the results back to your local computer. This enables you to run Windows PowerShell commands on multiple computers on your network from your client computer, rather than using creating a connection to each computer.**

**A key goal of Windows PowerShell remoting is to enable batch administration, which enables you to run commands on a set of remote computers simultaneously. You can use remoting in one of three ways, described in the following table.**

**Expand table**

**Method Description**

**One-to-One remoting In the One-to-One remoting scenario, (also known as interactive remoting), you connect to a single remote computer and run Windows PowerShell commands on it, exactly as if you had signed in to the computer and opened a Windows PowerShell window.**

**One-to-Many remoting In the One-to-Many remoting scenario, (also known as fan-out remoting), you issue a command that will be executed on one or more remote computers in parallel. You are not working with each remote computer interactively. Instead, your commands are issued and executed in a batch, and the results are returned to your computer for your use.**

**Many-to-One remoting In the Many-to-One remoting scenario, (also known as fan-in remoting), multiple administrators make remote connections to a single computer. Typically, those administrators will have differing permissions on the same remote computer and might be working in a restricted Windows PowerShell session.**

**Caution**

**When you run commands on multiple computers, be aware of the differences between the remote computers, such as differences in operating systems, file system structures, and system registries.**

**PowerShell Direct**

**Many administrators choose to run some of their servers in virtualized environments. To enable a simpler administration of Hyper-V VMs running Windows 10 or Windows Server, you can use a feature called PowerShell Direct.**

**PowerShell Direct enables you to run a Windows PowerShell cmdlet or script inside a VM from the host operating system regardless of network, firewall, and remote management configurations.**

**Next unit: Use Windows PowerShell to remotely administer a server**

**Continue**

**Skip to main content**

**We use optional cookies to improve your experience on our websites, such as through social media connections, and to display personalized advertising based on your online activity. If you reject optional cookies, only cookies necessary to provide you the services will be used. You may change your selection by clicking “Manage Cookies” at the bottom of the page. Privacy Statement Third-Party Cookies**

**Accept Reject Manage cookies**

**Search**

**Sign in**

**Learn Training Browse Describe Windows Server administration tools**

**Use Windows PowerShell to remotely administer a server**

**200 XP**

**7 minutes**

**IT administration staff at Contoso are increasingly focused on performing administrative tasks remotely from the object they're managing. One area of particular interest is the ability to leverage Windows PowerShell to remotely administer and manage Contoso's IT infrastructure. You can use Windows PowerShell remoting to achieve this.**

**Requirements for remoting**

**Remoting requires that you have Windows PowerShell on your local computer, and Windows Remote Management enabled on any remote computers to which you want to connect. You also must enable Windows PowerShell remoting.**

**Note**

**Windows PowerShell remoting is enabled by default in Windows Server, but you must enable it on Windows 10.**

**To enable Windows PowerShell remoting, use one of the following procedures. At an elevated command prompt (or Windows PowerShell (Admin) prompt), run the following command:**

**Copy**

**Winrm quickconfig**

**Alternatively, to enable remoting, you can use the following Windows PowerShell cmdlet:**

**PowerShell**

**Copy**

**Enable-PSremoting -force**

**Important**

**Windows Remote Management communicates via HTTP. By default, Windows Remote Management and Windows PowerShell remoting use TCP port 5985 for incoming unencrypted connections, and TCP port 5986 for incoming encrypted connections.**

**Note**

**Applications that use Windows Remote Management—such as Windows PowerShell—can also apply their own encryption to the data that is passed to the Windows Remote Management service.**

**Any files and other resources that are necessary to run a particular command must be on the remote computer because the remoting commands do not copy any resources. However, you can run local scripts. This is because the script’s contents are sent to the remote computer, rather than the script file itself.**

**To perform remote administration, administrators must have permission to:**

**Connect to the remote computer**

**Run Windows PowerShell**

**Access data stores and the registry on the remote computer**

**Caution**

**Enabling remoting on computers with a network interface card (NIC) assigned to the Public network location profile generates an error. You can use Enable-PSremoting -force to bypass this restriction, and force remoting on a device with a public NIC.**

**Run cmdlets against remote computers**

**Several cmdlets have a ComputerName parameter that enables you to retrieve objects from remote computers. These cmdlets do not use Windows PowerShell remoting to communicate, so you can use the ComputerName parameter in these cmdlets on any computer that runs Windows PowerShell. You do not have to configure the computers for Windows PowerShell remoting, or fulfill system requirements for remoting.**

**The following table provides more information about the ComputerName parameter.**

**Command Description**

**Get-Command –ParameterName ComputerName Finds cmdlets that use the ComputerName parameter.**

**Get-Help <cmdlet-name> -parameter ComputerName Determine whether the ComputerName parameter requires Windows PowerShell remoting.**

**Processing remote commands**

**When you connect to a remote computer and send it a remote command, the command transmits across the network to a Windows PowerShell instance on the remote computer, and then runs on it. The command results are sent back to the local computer and display in the Windows PowerShell session on the local computer.**

**All of the local input to a remote command is collected before any of it is sent to the remote computer. However, the output is returned to the local computer as it is generated. When you connect to a remote computer, the system uses the user name and password credentials on the local computer to authenticate you to the remote computer.**

**Note**

**By default, the Kerberos version 5 (V5) authentication protocol is used to perform the authorization and authentication. Therefore, an AD DS domain is required.**

**Tip**

**In situations where the remote computer is not in a domain or is in an untrusted domain, you can allow a client computer to connect by defining the remote computers as trusted hosts.**

**The following cmdlets support remoting.**

**Invoke-Command**

**Enter-PSSession**

**Exit-PSSession**

**Disconnect-PSSession**

**Receive-PSSession**

**Connect-PSSession**

**Tip**

**You can create remote tabs using Windows PowerShell ISE. To perform this task, use the New Remote PowerShell tab option from the File menu in Windows PowerShell ISE.**

**When you implement remoting, you can connect to the remote machines using a temporary session, or you can create a persistent session.**

**Create a temporary session**

**For a temporary session, you start the session, run the commands, and then end the session. This is an efficient method for running a single command or several unrelated commands, even on a large number of remote computers.**

**Note**

**Variables or functions defined within commands are no longer available after you close the connection.**

**To create a temporary connection, use the Invoke-Command cmdlet with the –ComputerName parameter to specify the remote computers. Then, use the –ScriptBlock parameter to specify the command. For example, the following command runs Get-EventLog on the SEA-DC1 computer:**

**PowerShell**

**Copy**

**Invoke-Command –ComputerName SEA-DC1 –ScriptBlock {Get-EventLog –log system}**

**Create a persistent session**

**To create a persistent connection with another computer, use the New-PSSession cmdlet. For example, the following command creates a session on a remote computer, and saves the session in the $s variable:**

**PowerShell**

**Copy**

**$s = New-PSSession –ComputerName SEA-DC1**

**Use the Enter-PSSession cmdlet to connect to and start an interactive session. For example, after you open a new session on SEA-DC1, the following command starts an interactive session with the computer:**

**PowerShell**

**Copy**

**Enter-PSSession $s**

**Once you enter a session, the Windows PowerShell command prompt on your local computer changes to indicate the connection.**

**A screenshot of the Administrator: Windows PowerShell dialog box. The administrator has connected remotely to SEA-DC1.**

**The interactive session remains open until you close it. This enables you to run as many commands as needed. To end the interactive session, enter the following command:**

**PowerShell**

**Copy**

**Exit-PSSession**

**Run remote commands on multiple computers**

**For temporary sessions, the Invoke-Command cmdlet accepts multiple computer names. For persistent connections, the Session parameter accepts multiple Windows PowerShell sessions. To run a remote command on multiple computers, include all computer names in the ComputerName parameter with the Invoke-Command cmdlet, and separate the names with commas as demonstrated in the following example:**

**PowerShell**

**Copy**

**Invoke-Command -ComputerName SEA-DC1, SEA-SVR1, SEA-SVR2 -ScriptBlock {Get-Culture}**

**For persistent sessions, you also can run a command in multiple Windows PowerShell sessions. The following commands create Windows PowerShell sessions on SEA-DC1, SEA-SVR1, and SEA-SVR2, and then run a Get-Culture command in each Windows PowerShell session:**

**PowerShell**

**Copy**

**$s = New-PSSession -ComputerName SEA-DC1, SEA-SVR1, SEA-SVR2**

**Invoke-Command -Session $s -ScriptBlock {Get-Culture}**

**Tip**

**To include the local computer in the list of computers, enter the name of the local computer, or a period (.), or localhost.**

**How to run a script on remote computers**

**To run a local script on remote computers, use the FilePath parameter with Invoke-Command. The following command runs the Sample.ps1 script on the SEA-DC1 and SEA-SVR1 computers:**

**PowerShell**

**Copy**

**Invoke-Command -ComputerName SEA-DC1, SEA-SVR1 –FilePath C:\Test\Sample.ps1**

**The results of the script are returned to the local computer. By using the FilePath parameter, you do not need to copy any files to the remote computers.**

**Demonstration**

**The following video demonstrates how to manage a remote Windows Server by using Windows PowerShell. The main steps in the process are:**

**Launch an elevated PowerShell prompt.**

**Create a PowerShell remoting session by running the following command:**

**PowerShell**

**Copy**

**Enter-PSSession -ComputerName SEA-DC1**

**Retrieve information about the server, such as name and IP address, using standard Windows PowerShell cmdlets.**

**Check status of IIS service and restart that service using the following command:**

**PowerShell**

**Copy**

**Get-Service -Name IISAdmin | Restart-Service**

**Quick review**

**1. What cmdlet can be run on a remote Windows Server computer to enable PowerShell remoting?**

**Enable-PSRemoting.**

**New-PSSession.**

**Enter-PSSession.**

**Having an issue? We can help!**

**?**

The following cmdlets support remoting.

* Invoke-Command
* Enter-PSSession
* Exit-PSSession
* Disconnect-PSSession
* Receive-PSSession
* Connect-PSSession

**To create a temporary connection, use the Invoke-Command cmdlet with the –ComputerName parameter to specify the remote computers. Then, use the –ScriptBlock parameter to specify the command. For example, the following command runs Get-EventLog on the SEA-DC1 computer:**

**PowerShell**

**Copy**

**Invoke-Command –ComputerName SEA-DC1 –ScriptBlock {Get-EventLog –log system}**