Microsoft

Administering Windows Server Hybrid Core Infrastructure

Exam Ref AZ-800

Orin Thomas

FREE SAMPLE CHAPTER |

Microsoft

Exam Ref AZ-800 Administering Windows Server Hybrid Core Infrastructure

Orin Thomas

Exam Ref AZ-800 Administering Windows Server Hybrid Core Infrastructure

Published with the authorization of Microsoft Corporation by:
Pearson Education, Inc.

ISBN-13: 978-0-13-772926-5
ISBN-10: 0-13-772926-X

Library of Congress Control Number: 2022938820

ScoutAutomatedPrintCode

EDITOR-IN-CHIEF
Brett Bartow

EXECUTIVE EDITOR
Loretta Yates

SPONSORING EDITOR
Charvi Arora

DEVELOPMENT EDITOR
Songlin Qiu

TECHNICAL EDITOR
Andrew Warren

MANAGING EDITOR
Sandra Schroeder

SENIOR PROJECT EDITOR
Tracey Croom

COPY EDITOR
Elizabeth Welch

INDEXER
Tim Wright

PROOFREADER
Barbara Mack

EDITORIAL ASSISTANT
Cindy Teeters

COVER DESIGNER
Twist Creative, Seattle

COMPOSITOR
codeMantra

Pearson's Commitment to Diversity, Equity, and Inclusion

Pearson is dedicated to creating bias-free content that reflects the diversity of all learners. We embrace the many dimensions of diversity, including but not limited to race, ethnicity, gender, socioeconomic status, ability, age, sexual orientation, and religious or political beliefs.

Education is a powerful force for equity and change in our world. It has the potential to deliver opportunities that improve lives and enable economic mobility. As we work with authors to create content for every product and service, we acknowledge our responsibility to demonstrate inclusivity and incorporate diverse scholarship so that everyone can achieve their potential through learning. As the world's leading learning company, we have a duty to help drive change and live up to our purpose to help more people create a better life for themselves and to create a better world.

Our ambition is to purposefully contribute to a world where

- Everyone has an equitable and lifelong opportunity to succeed through learning
- Our educational products and services are inclusive and represent the rich diversity of learners
- Our educational content accurately reflects the histories and experiences of the learners we serve
- Our educational content prompts deeper discussions with learners and motivates them to expand their own learning (and worldview)

While we work hard to present unbiased content, we want to hear from you about any concerns or needs with this Pearson product so that we can investigate and address them.

Please contact us with concerns about any potential bias at https://www.pearson.com/report-bias.html.

This page intentionally left blank

Contents at a glance

This page intentionally left blank

Contents

Introduction

The AZ-800 exam deals with advanced topics that require candidates to have an excellent working knowledge of Windows Server and Azure Hybrid functionality. Some of the exam comprises topics that even experienced Windows Server Hybrid administrators may rarely encounter unless they are consultants who manage hybrid cloud workloads on a regular basis. To be successful in taking this exam, not only do candidates need to understand how to deploy and manage AD DS, hybrid identity, Windows Servers, virtual machines, containers, hybrid networks, and storage services, but they also need to know how to perform these tasks with on-premises and Azure IaaS instances of Windows Server.

Candidates for this exam are information technology (IT) professionals who want to validate their advanced Windows Server Hybrid administration skills and knowledge. To pass, candidates require a thorough theoretical understanding as well as meaningful practical experience implementing the technologies involved.

This edition of this book covers Windows Server and the AZ-800 exam objectives as of mid-2022. As Windows Server hybrid technologies evolve, so do the AZ-800 exam objectives, so you should check carefully if any changes have occurred since this edition of the book was authored and study accordingly.

This book covers every major topic area found on the exam, but it does not cover every exam question. Only the Microsoft exam team has access to the exam questions, and Microsoft regularly adds new questions to the exam, making it impossible to cover specific questions. You should consider this book a supplement to your relevant real-world experience and other study materials. If you encounter a topic in this book that you do not feel completely comfortable with, use the "Need more review?" links you'll find in the text to find more information and take the time to research and study the topic. Great information is available on Microsoft Docs, Microsoft Learn, and in blogs and forums.

Organization of this book

This book is organized by the "Skills measured" list published for the exam. The "Skills measured" list is available for each exam on Microsoft Learn: *https://microsoft.com/learn*. Each chapter in this book corresponds to a major topic area in the list, and the technical tasks in each topic area determine a chapter's organization. If an exam covers six major topic areas, for example, the book will contain six chapters.

Microsoft certifications

Microsoft certifications distinguish you by proving your command of a broad set of skills and experience with current Microsoft products and technologies. The exams and corresponding certifications are developed to validate your mastery of critical competencies as you design and develop, or implement and support, solutions with Microsoft products and technologies both on-premises and in the cloud. Certification brings a variety of benefits to the individual and to employers and organizations.

NEED MORE REVIEW **ALL MICROSOFT CERTIFICATIONS**

For information about Microsoft certifications, including a full list of available certifications, go to *http://www.microsoft.com/learn.com/learn*.

Check back often to see what is new!

Quick access to online references

Throughout this book are addresses to webpages that the author has recommended you visit for more information. Some of these addresses (also known as URLs) can be painstaking to type into a web browser, so we've compiled all of them into a single list that readers of the print edition can refer to while they read.

Download the list at

MicrosoftPressStore.com/ExamRefAZ800/downloads

The URLs are organized by chapter and heading. Every time you come across a URL in the book, find the hyperlink in the list to go directly to the webpage.

Errata, updates, & book support

We've made every effort to ensure the accuracy of this book and its companion content. You can access updates to this book—in the form of a list of submitted errata and their related corrections—at:

MicrosoftPressStore.com/ExamRefAZ800/errata

If you discover an error that is not already listed, please submit it to us at the same page.

For additional book support and information, please visit *http://www.MicrosoftPressStore.com/Support*.

Please note that product support for Microsoft software and hardware is not offered through the previous addresses. For help with Microsoft software or hardware, go to *http://support.microsoft.com*.

Stay in touch

Let's keep the conversation going! We're on Twitter: *http://twitter.com/MicrosoftPress*.

This page intentionally left blank

About the author

ORIN THOMAS is a Principal Cloud Advocate at Microsoft and has written more than 3 dozen books for Microsoft Press on such topics as Windows Server, Windows Client, Azure, Office 365, System Center, Exchange Server, Security, and SQL Server. He has authored Azure Architecture courses at Pluralsight and has authored multiple Microsoft Official Curriculum and EdX courses on a variety of IT Pro topics. You can follow him on Twitter at *http://twitter.com/orinthomas*.

This page intentionally left blank

CHAPTER 2

Manage Windows Servers and workloads in a hybrid environment

A critical element in any complex hybrid cloud deployment is the set of tools used to manage, maintain, and monitor workloads. Windows Server hybrid administrators have several options when it comes to choosing which tools they will use to manage the Windows Server instances that they are responsible for. Some tools allow you to manage up to the cloud from an on-premises administrative workstation; other tools allow you to manage down from the cloud from the Azure portal or Azure CLI.

Skills covered in this chapter:

- Skill 2.1: Manage Windows Servers in a hybrid environment
- Skill 2.2: Manage Windows Servers and workloads by using Azure Services

Skill 2.1: Manage Windows Servers in a hybrid environment

This objective deals with the technologies and techniques that you can use to manage Windows Server instances in on-premises and cloud environments. You'll learn about choosing and configuring administration tools as well as constrained delegation and Just Enough Administration.

This skill covers how to:

- Choose administration tools
- Deploy a WAC gateway server
- Configure a target machine for WAC
- Manage Azure hybrid services with WAC
- Configure PowerShell remoting
- Configure CredSSP or Kerberos Delegation for second hop remoting
- Configure Just Enough Administration for PowerShell remoting

Choose administration tools

You can use a variety of tools to manage Windows Server 2019. Some, such as PowerShell, the Microsoft Management Console, and Server Manager, are built into the operating systems. You'll need to download others, such as Windows Admin Center, for free from the Microsoft website.

The company's general systems administration philosophy is that while you can do almost everything with a graphical console such as Windows Admin Center, Active Directory Administrative Center, or the Server Manager console, any task that you do repeatedly should be automated using PowerShell. Microsoft best practice is that almost all administration tasks should be performed remotely rather than by signing in directly to the server and performing them locally.

Remote not local

Windows Server is designed to be administered remotely rather than locally. This "remote first" philosophy shouldn't come as a surprise to experienced administrators. The vast majority of Windows Server instances are running as virtual machines, either in datacenters or in the cloud, and we are long past the days where your primary method of switching between different servers that you were working on was by selecting different options on a KVM switch.

You need to be familiar with how to use your tools remotely. You should avoid signing in to each server individually using Remote Desktop and firing up the console that is relevant to the role or feature that you want to manage. You should also avoid using Remote Desktop to connect to a server just to run a PowerShell script because this is a task more appropriately performed using PowerShell remoting.

Privileged Access Workstations

Servers are only as secure as the computers that you use to manage them. An increasing number of security incidents have occurred because a privileged user's computer was infected with malware and that computer was then used to perform server administration tasks. Privileged Access Workstations (PAWs) are specially configured computers that you use to perform remote administration tasks. The idea of a PAW is that you have a computer with a locked-down configuration that you only use to perform server administration tasks. You don't use this computer to read your email or browse the internet; you just use it to perform server administration tasks.

Consider configuring a PAW in the following way:

- Configure Windows Defender Application Control to allow only specifically authorized and digitally signed software to run on the computer.
- Configure Credential Guard to protect credentials stored on the computer.
- Use BitLocker to encrypt the computer's storage and protect the boot environment.
- The computer should not be used to browse the internet or to check email. Server administrators should have completely separate computers to perform their other

daily job tasks. Block internet browsing on the PAW both locally and on the perimeter network firewall.

- Block the PAW from accessing the internet. Software updates should be obtained from a dedicated secure update server on the local network. External tools should be obtained from another computer and transferred to the PAW.
- Server administrators should not sign in to the PAW using an account that has administrative privileges on the PAW.
- Only specific user accounts used by server administrators should be able to sign on to the PAW. Consider additional restrictions such as sign-in hours. Block privileged accounts from signing in to computers that are not PAWs or servers to be managed, such as the IT staff's everyday work computers.
- Configure servers to only accept administrator connections from PAWs. This can be done through Windows Defender Firewall with Advanced Security.
- Use configuration-management tools to monitor the configuration of the PAW. Some organizations rebuild PAWs entirely every 24 hours to ensure that configurations are not altered. Use these tools to restrict local group membership and ensure that the PAW has all appropriate recent software updates applied.
- Ensure that audit logs from PAWs are forwarded to a separate secure location.
- Disable the use of unauthorized storage devices. For example, you can configure policies so that only USB storage devices that have a specific BitLocker organizational ID can be used with the computer.
- Block unsolicited inbound network traffic to the PAW using Windows Defender Firewall.

Jump servers

Jump servers are another security procedure that can be used in conjunction with privileged-access workstations. Jump servers allow servers to accept administrative connections only from specific hosts. For example, you only allow domain controllers to be administered from computers that have a specific IP address and a computer certificate issued by a specific certification authority. You can configure jump servers to only accept connections from PAWs and servers to be administered to only accept connections from jump servers. As mentioned earlier, some organizations that use jump servers have them rebuilt and redeployed every 24 hours to ensure that their configuration does not drift from the approved configuration. Azure provides a service, Azure Bastion, that functions as a managed jump server. You'll learn more about using Azure Bastion to access Windows Server IaaS VMs in Chapter 3.

Remote Desktop

Remote Desktop is the way that many administrators are likely to remotely perform one-off tasks on servers running the GUI version of Windows Server. While best practice is to use PowerShell or Windows Admin Center for remote administration, sometimes it's quicker to just establish a Remote Desktop session. This is because using Remote Desktop allows you to

perform tasks on the remote server in a manner that appears similar to being directly signed in at the console.

By default, Remote Desktop is disabled on newly deployed computers running Windows Server (though this is not the case for new Azure IaaS instances of Windows Server). You enable Remote Desktop either through the Remote tab of the System Properties dialog box or by running the following PowerShell command:

```
Set-ItemProperty -Path "HKLM:\System\CurrentControlSet\Control\Terminal Server" -Name
"fDenyTSConnections" –Value 0
```

You can make Remote Desktop connections to computers running the Server Core installation option if Remote Desktop is enabled.

By default, Remote Desktop Connection connects to Remote Desktop services on port 3389. When you enable Remote Desktop using the GUI, a remote desktop related firewall is automatically enabled. If you enable Remote Desktop using PowerShell, you also need to manually enable a firewall rule to allow connections. You can do this using the following PowerShell command:

```
Enable-NetFirewallRule -DisplayGroup "Remote Desktop"
```

By default, the **Allow connections only from computers running Remote Desktop with Network Level Authentication** option is selected. Network Level Authentication requires that a user be authenticated prior to the Remote Desktop session being established. Network Level Authentication is supported by the Remote Desktop Connection client, which is available on all Windows operating systems, but it might not be supported by third-party Remote Desktop clients.

Only users who are members of the local Administrators group and members of the local Remote Desktop Users group can make connections via Remote Desktop. If you want to grant a user account permission to access the server without the account full administrative privileges, add the account to the local Remote Desktop Users group.

You can map local volumes to a remote host in an active Remote Desktop Connection session by configuring the **Local Resources and Devices** setting on the **Local Resources** tab of the **Remote Desktop Connection** dialog box. While it is less effective over low-bandwidth connections, it can provide a simple way to transfer files from your client computer to a remote server instead of setting up FTP or another file transfer method.

Deploy a WAC gateway server

Windows Admin Center (WAC) is a web-based console that allows you to remotely manage Windows Server through a web browser. You can connect to and use Windows Admin Center using Edge, Chrome, or any standards-compliant browser. You can use WAC to manage computers running Windows Server 2012 and later and Windows 10 or later client computers.

You can install WAC on computers running Windows 10 and later and Windows Server 2016 and later. You can install WAC on a Windows Server instance deployed using the Server Core installation option.

When you deploy WAC on a Windows Server instance, it functions as a *gateway server*. Gateway servers allow any client on the network to connect to the Windows Admin Center instance using their standards-compliant web browser without requiring Windows Admin Center be installed locally. A WAC gateway server can function as an administration connection point for multiple administrative sessions from different administrative users. Some organizations only deploy a single highly available gateway server and have all WAC administration tasks performed using that single WAC gateway instance. You should not deploy Windows Admin Center on a Windows Server instance that hosts the AD DS role.

Installing WAC

Windows Admin Center isn't included in Windows Server. You have to download the installation files from the Microsoft website. There are four Windows Admin Center deployment options:

- **Local client** When you choose this installation option, you install Windows Admin Center on your workstation. You connect to the WAC instance locally, which is similar to installing the Remote Server Administration Tools (RSAT) on a local workstation. When you install WAC locally, a shortcut to the WAC console is placed on your desktop.
- **Gateway server** When you install WAC in the gateway server configuration, you install it on a computer running Windows Server 2016 or later and then make remote connections to the WAC instance hosted on that computer through your preferred browser. Once connected to the WAC instance, you can add servers that you want to manage to the web-based console. When you perform an administrative task, the instructions to perform that task are issued from the gateway server and are run against the target server.
- **Managed server** The managed server deployment is a version of WAC in a gateway server configuration deployed on a cluster node to manage the cluster.
- **Failover cluster** The gateway server is deployed as a highly available service. This requires the configuration of a Cluster Shared Volume to store persistent data used by WAC. A script is available from the Microsoft website that simplifies the process of performing a high availability deployment.

When you install WAC on a Windows Server instance, you get the option of configuring which port will be used. You can choose between using a self-signed SSL (TLS) certificate or an SSL (TLS) certificate that is already installed on the computer. If you're deploying a gateway server, things will be a lot simpler if you deploy a TLS certificate from a trusted CA because it won't be necessary to go through the hassle of responding to dialog boxes about whether to trust the self-signed certificate when connecting to the gateway server from a variety of different administrative systems.

You can install Windows Admin Center on a Server Core instance of Windows Server using `msiexec` and by specifying the management port and SSL certificate option. (It should be the TLS certificate since the SSL protocol has been phased out, but most of the world still uses the

legacy terminology.) The syntax of the command-line installation where a trusted certificate is used is as follows:

```
msiexec /i <WACInstallerName>.msi /qn /L*v log.txt SME_PORT=<port> SME_
THUMBPRINT=<thumbprint> SSL_CERTIFICATE_OPTION=installed
```

`SME_PORT` is the port you want to use, and `SME_THUMBPRINT` is the thumbprint of the installed SSL (TLS) certificate. By default, installing WAC updates the computer's trusted host files. When you deploy WAC, you can configure it to update automatically or manually. When you configure WAC to update automatically, new versions will be installed as they become available through Microsoft Update. If you don't configure this option, you'll need to manually install newer versions of WAC as they become available.

To update an expired certificate on a WAC gateway server, you need to obtain and install the new certificate, obtain the certificate's thumbprint, and then rerun Setup and change the certificate used by WAC by specifying the new thumbprint.

NEED MORE REVIEW? **DEPLOY WAC GATEWAY**

You can learn more about deploying a WAC gateway at *https://docs.microsoft.com/en-us/windows-server/manage/windows-admin-center/deploy/install*.

Windows Admin Center extensions

Windows Admin Center extensions allow for the extension of WAC functionality. Windows Admin Center includes extensions for roles built into Windows Server such as Storage Migration Services and third-party extensions. Microsoft encourages third-party partners to add extensions to Windows Admin Center as an alternative to requiring systems administrators to use product-specific consoles.

By default, Windows Admin Center will display extensions published to the Microsoft official NuGet feed. This feed includes extensions published and updated by Microsoft as well as those published by trusted third-party vendors. Also, you can configure Windows Admin Center to display extensions or installations from any NuGet feed that supports the NuGet V2 APIs or a specially configured file share accessible to the computer hosting Windows Admin Center.

Extensions are available in Windows Admin Center by selecting the **Settings** icon and then selecting **Extensions**. The **Available Extension** pane displays all extensions that are available but not installed from the currently configured feed. You can update currently installed extensions if new versions of those extensions are available through the **Installed Extensions** pane. You can also configure Windows Admin Center to automatically update extensions.

Show script

When you perform a task in Windows Admin Center, you can select the PowerShell icon in the Windows Admin Center title bar to view PowerShell source code relevant to the tasks.

This allows you to copy and save useful PowerShell code for reuse later rather than having to perform all tasks through WAC.

Configure a target machine for WAC

Just like you need to configure a Windows Server instance so that you can connect to it using Remote Desktop, a Microsoft Management Console, or a remote PowerShell session, you will also have to configure a Windows Server instance so that it can be managed from a remote WAC instance.

To allow administration from a WAC instance, Remote Management must be enabled on a Windows Server instance you intend to manage. WAC traffic from the WAC instance to target servers uses PowerShell and WMI over WinRM. WinRM connections over HTTP use port 5985 and WinRM connections over HTTPS uses port 5986. If WinRM over HTTPS is not configured, you can configure a WinRM HTTPS listener using the following command:

```
winrm quickconfig -transport:https
```

In addition to the WinRM ports, WAC uses the SMB file sharing protocol for some file copying tasks. To configure a target machine for remote management by WAC, you will need to ensure any firewalls between the WAC instance and the target computer allow inbound connections on ports 445, 5985, and 4986.

To use Windows Admin Center from the Azure portal to manage Windows Server instances in Azure, it's necessary to deploy Windows Admin Center to each Windows Server Azure IaaS instance.

NEED MORE REVIEW? **CONFIGURE TARGET MACHINE**

You can learn more about configuring a target machine at *https://docs.microsoft.com/windows-server/manage/windows-admin-center/azure/manage-vm*.

Manage Azure hybrid services with WAC

Windows Admin Center can also be used to manage Azure hybrid services, such as Azure Backup, Azure Software Update, Azure Site Recovery, Azure Network Adapter, and Azure Monitor. Before you can integrate Azure hybrid services with WAC, you need to register the Windows Admin Center gateway with your Azure subscription. This process requires that you have access to an Azure AD account with the necessary permissions to configure an Azure AD application that has access to the Azure AD tenancy associated with your Azure subscription.

Configure PowerShell remoting

PowerShell is the primary scripting, automation, and management tool from Microsoft. In almost all cases, you can access greater functionality and settings through PowerShell than you can through WAC or the Azure console.

PowerShell includes a substantial amount of documentation explaining what each cmdlet can do and how you can do it. Once you know the name of the command you want to use to perform a task, you can use the PowerShell built-in help to learn the precise details of how to use that cmdlet to perform that task. You can get help for each cmdlet by typing **help cmdletname**. For example, to get help with the `get-service` cmdlet, type **help get-service** into a PowerShell session.

Modules

Modules are collections of PowerShell cmdlets. In older versions of PowerShell, you needed to manually load a module each time you wanted to use one of its associated cmdlets. In Windows Server 2016 and later, any module that is installed will load automatically when you try to run an associated cmdlet. Viewing cmdlets by module using the `Get-Command-Module <modulename>` cmdlet allows you to view just those cmdlets associated with a specific role or feature.

PowerShell Gallery

The PowerShell Gallery is a collection of modules published by the community that extend the functionality of PowerShell beyond what is available with a default installation of Windows Server. Table 2-1 lists the commands that you can use to get started with the PowerShell Gallery.

TABLE 2-1 PowerShell Gallery basics

Command	Functionality
`Find-Module -Repository PSGallery | out-host -paging`	This will list the available modules in the PowerShell Gallery in a paged format. You'll be prompted to install the `NuGetProvider` to interact with the PowerShell Gallery.
`Find-Module -Repository PSGallery -Name <ModuleName>`	This will list the modules with a specific name. You can use wildcards. For example, to view all modules that start with the name AzureRM, run the command `Find-Module -Repository PSGallery -Name AzureRM*`.
`Install-Module -Repository PSGallery -Name <ModuleName>`	This will install the Modulename module. For example, to install the AzureRM module, run the command `Install-Module -Repository PSGallery -Name AzureRM`.
`Update-Module`	This will update any module that you've installed using `Install-Module`.
`Get-InstalledModule`	Use this command to view all modules installed from the PowerShell Gallery.

PowerShell remoting

PowerShell remoting allows you to establish a remote interactive PowerShell session from a local PowerShell session on an administrative workstation or Cloud Shell. By default, PowerShell remoting is enabled on Windows Server instances but also requires a connection from a

private network and an account that is a member of the local Administrators group. PowerShell uses WMI over WinRM. WinRM connections over HTTP use port 5985, and WinRM connections over HTTPS uses port 5986. If PowerShell remoting has been disabled, you can enable it using the `Enable-PSRemoting` cmdlet. If WinRM over HTTPS is not configured, you can configure a WinRM HTTPS listener using the following command:

```
winrm quickconfig -transport:https
```

You initiate a remote PowerShell session using the `enter-pssession` command. If you do not specify alternate credentials, the credentials of the currently signed-on user will be used. If you want to use alternate credentials, one method to do so securely is by using the `get-credential` command and assigning it to a PowerShell variable, and then using the variable with the `enter-pssession` command. When you use `get-credential`, you will be prompted to enter a set of credentials. For example, to enter a set of credentials and then to use those credentials to establish a remote PowerShell session to a host named dc1.tailwindtraders.com, use the following commands:

```
$creds = get-credential
Enter-pssession -Computername dc1.tailwindtraders.com -credential $creds
```

To enable PowerShell remoting to computers that are not domain-joined, you must configure the trusted hosts list on the client computer from which you want to establish the remote session. You do this on the client computer using the `set-item` cmdlet. For example, to trust the computer at IP address 192.168.3.200, run this command:

```
Set-Item wsman:\localhost\Client\TrustedHosts -Value 192.168.3.200 -Concatenate
```

Once you've run the command to configure the client, you'll be able to establish a PowerShell remote session using the `Enter-PSSession` cmdlet. If you want more information about remoting, run the following command to bring up help text on the subject:

```
Help about_Remote_faq -ShowWindow
```

PowerShell allows you to run one command against many machines, which is known as *one-to-many remoting* or *fan-out administration*. You can use one-to-many remoting to run the same command against any number of computers. Rather than signing in to each computer to check whether a particular service is running, you can use PowerShell remoting to run the same command that checks the status of the service against each computer within the scope of the command.

For example, you could use the following command to read a list of computers from a text file named computers.txt:

```
$Computers = Get-Content c:\Computers.txt
```

You could then use the following command to get the properties of the Windows Update service:

```
Invoke-Command -ScriptBlock { get-service wuauserv } -computername $Computers
```

You can also use the `Invoke-Command` cmdlet to run a script from the local computer against a number of remote computers. For example, to run the script FixStuff.ps1 against the computers in the file computers.txt, run this command:

```
$Computers = Get-Content c:\Computers.txt
Invoke-Command -FilePath c:\FixStuff.ps1
```

> ***NEED MORE REVIEW?*** **POWERSHELL REMOTING**
>
> You can learn more about PowerShell remoting at *https://docs.microsoft.com/powershell/scripting/learn/remoting/powershell-remoting-faq*.

Configure CredSSP or Kerberos Delegation for second hop remoting

Second hop remoting is when you are signed in to one host, make a remote PowerShell connection to a second host, and perform a task that requires resource access to a third host that requires your account credentials. Unless the second host has a way of forwarding your credentials to the third host, the task may not complete because your credentials can't be used for that task. The process of a server acting on behalf of a signed-on user is termed *delegation*.

Kerberos delegation allows a computer to interact with the Kerberos Key Distribution Center to obtain a service ticket derived from the user's permissions that is used to access resources on the network.

For example, say you need to allow users with accounts in the tailwindtraders.com domain to use a WAC to manage a server named app1.adatum.com in the adatum.com domain. The following conditions exist:

- You have deployed a WAC gateway server on host wac.tailwindraders.com.
- There is a two-way forest trust between the adatum.com and the tailwindtraders.com single-domain forests.

You can configure constrained delegation in this scenario by running the following PowerShell command:

```
Set-ADComputer -Identity (Get-ADComputer wac.tailwindtraders.com) -PrincipalsAllowed
ToDelegateToAccount (Get-ADComputer app1.adatum.com)
```

Kerberos constrained delegation allows you to limit which of a computer's services can interact with the KDC to obtain the appropriate ticket on the user's behalf. You can configure constrained delegation on the **Delegation** tab of a computer account's properties in Active Directory Users and Computers. When you do this, you specify the service type, the user or computer account that can leverage delegated credentials, the port, and the service principal name of the service that can perform the action.

NEED MORE REVIEW? **SECOND HOP REMOTING**

You can learn more about second hop remoting at *https://docs.microsoft.com/powershell/scripting/learn/remoting/ps-remoting-second-hop*.

Configure Just Enough Administration for PowerShell remoting

Just Enough Administration (JEA) allows you to implement role-based access control (RBAC) functionality through Windows PowerShell remoting. JEA allows you to specify which PowerShell cmdlets and functions can be used when connected to a specific endpoint. You can go further and specify which parameters within those cmdlets and functions are authorized and even specify which values can be used with those parameters.

For example, you could create a JEA endpoint where a user is able to run the `Restart-Service` command, but only where the Name parameter is set to DHCPServer. This would allow the user to restart the DHCPServer on the computer they connected to, but it would not restart any other service on the computer.

You can also configure a JEA endpoint to allow other command-line commands such as `whoami` to be run, though the drawback of this is that you don't have the same level of control when restricting how that command can be run.

JEA endpoints can leverage virtual accounts. This means that activities performed on the computer through the endpoint use a special temporary virtual account rather than the user's account. This temporary virtual account has local administrator privileges but is constrained to only using the cmdlets, functions, parameters, and values defined by JEA. The benefits of this include:

- The user's credentials are not stored on the remote system. If the remote system is compromised, the user's credentials are not subject to credential theft and cannot be used to traverse the network to gain access to other hosts.
- The user account used to connect to the endpoint does not need to be privileged. The endpoint simply needs to be configured to allow connections from specified user accounts.
- The virtual account is limited to the system on which it is hosted. The virtual account cannot be used to connect to remote systems. Attackers cannot use a compromised virtual account to access other protected servers.
- The virtual account has local administrator privileges but is limited to performing only the activities defined by JEA. You have the option of configuring the virtual account with the membership of a group other than the local administrators group, to further reduce privileges.

Role-capability files

A role-capability file is a special file that allows you to specify what tasks can be performed when connected to a JEA endpoint. Only tasks that are explicitly allowed in the role-capability file can be performed.

You can create a new blank role-capability file by using the `New-PSRoleCapabilityFile` cmdlet. Role-capability files use the .psrc extension. For example, run this command to create a new role-capability file for a role that allows someone to manage a DNS server:

```
New-PSRoleCapabilityFile -Path .\DNSOps.psrc
```

Once the PSRC file is created, you edit the role-capability file and add the cmdlets, functions, and external commands that are available when a user is connected to the endpoint. You can allow entire Windows PowerShell cmdlets or functions or list which parameters and parameter values can be used.

You can edit a role-capability file in PowerShell ISE, Visual Studio Code (though only the first is available on Windows Server), or any capable text editor. Editing the file involves commenting out the appropriate sections and filling them in with the configuration items that you want to set.

Authoring role-capability files is one of those few times when you need to know whether something in PowerShell is a cmdlet or a function. Mostly, people refer to commands in PowerShell as cmdlets, but some are actually functions and others are aliases. You need to know the appropriate type when configuring a role-capability file because if you put a function in as an allowed cmdlet, you won't get the expected result. You can figure out which designation is appropriate by using the `Get-Command` cmdlet.

Table 2-2 describes the different options that you can configure in a role-capability file.

TABLE 2-2 Role-capability files

Capability	Description
ModulesToImport	JEA auto-loads standard modules, so you probably don't need to use this unless you need to import custom modules.
VisibleAliases	Specifies which aliases to make available in the JEA session. Even if an aliased cmdlet is available, the alias won't be available unless it's here.
VisibleCmdlets	Lists which Windows PowerShell cmdlets are available in the session. You can extend this by allowing all parameters and parameter values to be used or you can limit cmdlets to particular parameters and parameter values. For example, use the following syntax, if you wanted to allow the `Restart-Service` cmdlet to only be used to restart the DNS service: `VisibleCmdlets = @{ Name = 'Restart-Service'; Parameters = @{ Name='Name'; ValidateSet = 'DNS'}}`

Capability	Description
VisibleFunctions	This field lists which Windows PowerShell functions are available in the session. You can choose to list functions, allowing all parameters and parameter values to be used, or you can limit functions to particular parameters and parameter values. For example, if you wanted to allow the `Add-DNSServerResourceRecord`, `Get-DNSServer ResourceRecord`, and `Remove-DNSServerResource` functions to be used, you would use the following syntax: `VisibleFunctions = 'Add-DNSServerResourceRecord', 'Get-DNSServerResourceRecord','Remove-DNSServerResourceRecord'`
VisibleExternal Commands	This field allows users who are connected to the session to run external commands. For example, you can use this field to allow access to *c:\windows\system32\whoami.exe* so that users connected to the JEA session can identify their security context by using the following syntax: `VisibleExternalCommands = 'C:\Windows\System32\whoami.exe'`
VisibleProviders	This field lists Windows PowerShell providers that are visible to the session.
ScriptsToProcess	This field allows you to configure Windows PowerShell scripts to run automatically when the session is started.
AliasDefinitions	This field allows you to define Windows PowerShell aliases for the JEA session.
FunctionDefinitions	This field allows you to define Windows PowerShell functions for the JEA session.
VariableDefinitions	This field allows you to define Windows PowerShell variables for the JEA session.
EnvironmentVariables	This field allows you to specify environment variables for the JEA session.
TypesToProcess	This field allows you to configure Windows PowerShell type files to load for the JEA session.
FormatsToProcess	This field allows you to configure Windows PowerShell formats to load for the JEA session.
AssembliesToLoad	This field allows you to specify which assemblies to load for the JEA session.

Session-configuration files

Session-configuration files determine which role capabilities are mapped to specific security groups. For example, if you wanted to allow only members of the CONTOS\DNSOps security group to connect to the JEA endpoint that is defined by the DNSOps role-capability file, you would configure this security group in the session-configuration file.

You use the `New-PSSessionConfigurationFile` cmdlet to create a session-configuration file. These files use the .pssc extension. For example, to create a new session-configuration file for the DNSOps role, run the following command:

```
New-PSSessionConfigurationFile -Path .\DNSOps.pssc -Full
```

Session-configuration files have elements described in Table 2-3.

TABLE 2-3 Session-configuration files

Field	Explanation
SessionType	This field allows you to configure the session's default settings. If you set this to *RestrictedRemoteServer*, you can use the `Get-Command`, `Get-FormatData`, `Select-Object`, `Get-Help`, `Measure-Object`, `Exit-PSSession`, `Clear-Host`, and `Out-Default` cmdlets. The session execution policy is set to *RemoteSigned*. Example: `SessionType = 'RestrictedRemoteServer'`
RoleDefinitions	You use the *RoleDefinitions* entry to assign role capabilities to specific security groups. These groups do not need to have any privileges and can be standard security groups. Example: `RoleDefinitions =@{'CONTOSO\DNSOps' = @{RoleCapabilities='DNSOps'}}`
RunAsVirtualAccount	When enabled, this field allows JEA to use a privileged virtual account created just for the JEA session. This virtual account has local administrator privileges on member servers and is a member of the Domain Admins group on a domain controller. Use this option to ensure that credentials are not cached on the server that hosts the endpoint. Remember that you can configure the virtual account to be a member of groups other than the local administrators group.
TranscriptDirectory	This field allows you to specify the location where JEA activity transcripts are stored.
RunAsVirtual AccountGroups	If you do not want the virtual account to be a member of the local administrators group (or Domain Admins on a domain controller), you can instead use this field to specify the groups in which the virtual account is a member.

JEA endpoints

A JEA endpoint is a Windows PowerShell endpoint that you configure so that only specific authenticated users can connect to it. When those users do connect, they only have access to the Windows PowerShell cmdlets, parameters, and values defined by the appropriate session-configuration file that links security groups and role capabilities. When you use endpoints with virtual accounts, the actual activity that a user performs on the server that hosts the endpoint occurs using the virtual account. This means that no domain-based administrative credentials are stored on the server that hosts the endpoint.

A server can have multiple JEA endpoints, and each JEA endpoint can be used for a different administrative task. For example, you could have a DNSOps endpoint to perform DNS administrative tasks and an IISOps endpoint to perform Internet Information Server–related administrative tasks. Users are not required to have privileged accounts that are members of groups, such as the local administrators group, to connect to an endpoint. Once connected, users have the privileges assigned to the virtual account configured in the session-configuration file.

You create JEA endpoints by using the `Register-PSSessionConfiguration` cmdlet. When using this cmdlet, you specify an endpoint name and a session-configuration file hosted on the local machine.

For example, to create the endpoint DNSOps using the DNSOps.pssc session-configuration file, issue the following command and then restart the WinRM service:

```
Register-PSSessionConfiguration -Name DNSOps -Path .\DNSOps.pssc
```

You can use the `Get-PSSessionConfigurationFile` cmdlet to determine which endpoints are present on a computer. A user wanting to connect to a JEA session endpoint uses the `Enter-PSSession` cmdlet with the ConfigurationName parameter. For example, to connect to the DNSOps JEA endpoint on server MEL-DNS1, you would use this command:

```
Enter-PSSession -ComputerName MEL-DNS1 -ConfigurationName DNSOps
```

Once you've verified that JEA works, you'll need to lock down the default PowerShell endpoint. By default, only members of the local administrators group can connect to this default endpoint, and if you've implemented JEA properly, this group shouldn't need to have very many members anyway.

NEED MORE REVIEW? **JUST ENOUGH ADMINISTRATION**

You can learn more about Just Enough Administration at *https://docs.microsoft.com/powershell/scripting/learn/remoting/jea/overview*.

EXAM TIP

Remember which PowerShell cmdlets are relevant to specific JEA tasks.

Skill 2.2: Manage Windows Servers and workloads by using Azure Services

This objective deals with managing Windows Server instances in hybrid environments using Azure services, including Azure Arc, Microsoft Defender for Cloud, Microsoft Update, and Desired State Configuration.

Index

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O-P

Q-R

S

T

U

V

W

X-Y-Z