



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

# Microsoft Ignite

Nov 19–22, 2024

Register now >



## Enable-PSRemoting

Reference

Feedback

Module: [Microsoft.PowerShell.Core](#)

### In this article

[Syntax](#)

[Description](#)

[Examples](#)

[Parameters](#)

[Show 4 more](#)

Configures the computer to receive remote commands.

## Syntax

```
Enable-PSRemoting  
    [-Force]  
    [-SkipNetworkProfileCheck]  
    [-WhatIf]  
    [-Confirm]  
    [<CommonParameters>]
```

## Description

This cmdlet is only available on the Windows platform.

The `Enable-PSRemoting` cmdlet configures the computer to receive PowerShell remote commands that are sent by using the WS-Management technology. WS-Management based PowerShell remoting is currently supported only on Windows platform.

PowerShell remoting is enabled by default on Windows Server platforms. You can use `Enable-PSRemoting` to enable PowerShell remoting on other supported versions of Windows and to re-enable remoting if it becomes disabled.

You have to run this command only one time on each computer that will receive commands. You do not have to run it on computers that only send commands. Because the configuration starts listeners to accept remote connections, it is prudent to run it only where it is needed.

Enabling PowerShell remoting on client versions of Windows when the computer is on a public network is normally disallowed, but you can skip this restriction by using the **SkipNetworkProfileCheck** parameter.

For more information, see the description of the **SkipNetworkProfileCheck** parameter.

Multiple PowerShell installations can exist side-by-side on a single computer. Running `Enable-PSRemoting` will configure a remoting endpoint for the specific installation version that you are running the cmdlet in. So if you run `Enable-PSRemoting` while running PowerShell 6.2, a remoting endpoint will be configured that runs PowerShell 6.2. If you run `Enable-PSRemoting` while running PowerShell 7-preview, a remoting endpoint will be configured that runs PowerShell 7-preview.

`Enable-PSRemoting` creates two remoting endpoint configurations as needed. If the endpoint configurations already exist, then they are simply ensured to be enabled. The created configurations are identical but have different names. One will have a simple name corresponding to the PowerShell version that hosts the session. The other configuration name contains more detailed information about the PowerShell version which hosts the session. For example, when running `Enable-PSRemoting` in PowerShell 6.2, you will get two configured endpoints named **PowerShell.6**, **PowerShell.6.2.2**. This allows you to create a connection to the latest PowerShell 6 host version by using the simple name **PowerShell.6**. Or you can connect to a specific PowerShell host version by using the longer name **PowerShell.6.2.2**.

To use the newly enabled remoting endpoints, you must specify them by name with the **ConfigurationName** parameter when creating a remote connection using the `Invoke-Command`, `New-PSSession`, `Enter-PSSession` cmdlets. For more information, see Example 4.

The `Enable-PSRemoting` cmdlet performs the following operations:

- Runs the [Set-WSManQuickConfig](#) cmdlet, which performs the following tasks:
  - Starts the WinRM service.
  - Sets the startup type on the WinRM service to Automatic.
  - Creates a listener to accept requests on any IP address.

- Enables a firewall exception for WS-Management communications.
- Creates the simple and long name session endpoint configurations if needed.
- Enables all session configurations.
- Changes the security descriptor of all session configurations to allow remote access.
- Restarts the WinRM service to make the preceding changes effective.

To run this cmdlet on the Windows platform, start PowerShell by using the Run as administrator option. This cmdlet is not available on Linux or MacOS versions of PowerShell.

#### ⊗ Caution

This cmdlet does not affect remote endpoint configurations created by Windows PowerShell. It only affects endpoints created with PowerShell version 6 and greater. To enable and disable PowerShell remoting endpoints that are hosted by Windows PowerShell, run the `Enable-PSRemoting` cmdlet from within a Windows PowerShell session.

## Examples

### Example 1: Configure a computer to receive remote commands

This command configures the computer to receive remote commands.

```
Enable-PSRemoting
```

```
WARNING: PowerShell remoting has been enabled only for Power
```

```
does not affect Windows PowerShell remoting configurations.  
PowerShell to affect all PowerShell remoting configurations.
```

## Example 2: Configure a computer to receive remote commands without a confirmation prompt

This command configures the computer to receive remote commands. The **Force** parameter suppresses the user prompts.

```
Enable-PSRemoting -Force
```

```
WARNING: PowerShell remoting has been enabled only for Power  
does not affect Windows PowerShell remoting configurations.  
PowerShell to affect all PowerShell remoting configurations.
```

## Example 3: Allow remote access on clients

This example shows how to allow remote access from public networks on client versions of the Windows operating system. The name of the firewall rule can be different for different versions of Windows. Use `Get-NetFirewallRule` to see a list of rules. Before enabling the firewall rule, view the security settings in the rule to verify that the configuration is appropriate for your environment.

```
Get-NetFirewallRule -Name 'WINRM*' | Select-Object -Property
```

```
Name
```

```
----
```

```
WINRM-HTTP-In-TCP-NoScope
```

```
WINRM-HTTP-In-TCP
```

```
WINRM-HTTP-Compat-In-TCP-NoScope
```

```
WINRM-HTTP-Compat-In-TCP
```

```
Enable-PSRemoting -SkipNetworkProfileCheck -Force  
Set-NetFirewallRule -Name 'WINRM-HTTP-In-TCP' -RemoteAddress
```

By default, `Enable-PSRemoting` creates network rules that allow remote access from private and domain networks. The command uses the **SkipNetworkProfileCheck** parameter to allow remote access from public networks in the same local subnet. The command specifies the **Force** parameter to suppress confirmation messages.

The **SkipNetworkProfileCheck** parameter does not affect server versions of the Windows operating system, which allow remote access from public networks in the same local subnet by default.

The `Set-NetFirewallRule` cmdlet in the **NetSecurity** module adds a firewall rule that allows remote access from public networks from any remote location. This includes locations in different subnets.

## Example 4: Create a remote session to the newly enabled endpoint configuration

This example shows how to enable PowerShell remoting on a computer, find the configured endpoint names, and create a remote session to one of the endpoints.

The first command enables PowerShell remoting on the computer.

The second command lists the endpoint configurations.

The third command creates a remote PowerShell session to the same machine, specifying the **PowerShell.7** endpoint by name. The remote session will be hosted with the latest PowerShell 7 version (7.2.7).

The last command accesses the `$PSVersionTable` variable in the remote session to display the PowerShell version that is hosting the session.

```
Enable-PSRemoting -Force

Get-PSSessionConfiguration

$session = New-PSSession -ComputerName localhost -Configurat


Invoke-Command -Session $session -ScriptBlock { $PSVersionTa

WARNING: PowerShell remoting has been enabled only for Power
does not affect Windows PowerShell remoting configurations.
PowerShell to affect all PowerShell remoting configurations.

Name          : PowerShell.7
PSVersion      : 7.2
StartupScript  :
RunAsUser      :
Permission     : NT AUTHORITY\INTERACTIVE AccessAllowed,
                  BUILTIN\Administrators AccessAllowed,
                  BUILTIN\Remote Management Users AccessAllowe

Name          : PowerShell.7.2.7
PSVersion      : 7.2
StartupScript  :
RunAsUser      :
Permission     : NT AUTHORITY\INTERACTIVE AccessAllowed,
                  BUILTIN\Administrators AccessAllowed,
                  BUILTIN\Remote Management Users AccessAllowe

Name          Value
----
PSCompatibleVersions {1.0, 2.0, 3.0, 4.0...}
PSEdition          Core
PSRemotingProtocolVersion 2.3
Platform           Win32NT
SerializationVersion 1.1.0.1
GitCommitId        6.2.2
WSManStackVersion   3.0
PSVersion           6.2.2
OS                  Microsoft Windows 10.0.18363
```

 **Note**

The name of the firewall rule can be different depending on the version of Windows. Use the `Get-NetFirewallRule` cmdlet to list the names of the rules on your system.

# Parameters

## -Confirm

Prompts you for confirmation before running the cmdlet.

 Expand table

Type:	SwitchParameter
Aliases:	cf
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

## -Force

Forces the command to run without asking for user confirmation.

 Expand table

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False



Accept pipeline input:	False
Accept wildcard characters:	False

**-SkipNetworkProfileCheck**

Indicates that this cmdlet enables remoting on client versions of the Windows operating system when the computer is on a public network. This parameter enables a firewall rule for public networks that allows remote access only from computers in the same local subnet.

This parameter does not affect server versions of the Windows operating system, which, by default, have a local subnet firewall rule for public networks. If the local subnet firewall rule is disabled on a server version, `Enable-PSRemoting` re-enables it, regardless of the value of this parameter.

To remove the local subnet restriction and enable remote access from all locations on public networks, use the `Set-NetFirewallRule` cmdlet in the **NetSecurity** module.

This parameter was introduced in PowerShell 3.0.

 Expand table

Type:	<a href="#">SwitchParameter</a>
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

**-WhatIf**

Shows what would happen if the cmdlet runs. The cmdlet is not run.

 Expand table

Type:	SwitchParameter
Aliases:	wi
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

## Inputs

None

You can't pipe objects to this cmdlet.

## Outputs

String

This cmdlet returns strings that describe its results.

## Notes

This cmdlet is only available on Windows platforms.

On server versions of the Windows operating system, `Enable-PSRemoting` creates firewall rules for private and domain networks that allow remote access, and creates a firewall rule for public networks that allows remote access only from computers in the same local subnet.

On client versions of the Windows operating system, `Enable-PSRemoting` creates firewall rules for private and domain networks that allow unrestricted remote access. To create a firewall rule for public networks that allows remote access from the same local subnet, use the `SkipNetworkProfileCheck` parameter.

On client or server versions of the Windows operating system, to create a firewall rule for public networks that removes the local subnet restriction and allows remote access, use the `Set-NetFirewallRule` cmdlet in the NetSecurity module to run the following command: `Set-NetFirewallRule -Name "WINRM-HTTP-In-TCP-PUBLIC" -RemoteAddress Any`

`Enable-PSRemoting` enables all session configurations by setting the value of the **Enabled** property of all session configurations to `$True`.

`Enable-PSRemoting` removes the **Deny\_All** and **Network\_Deny\_All** settings. This provides remote access to session configurations that were reserved for local use.

## Related Links

- [Disable-PSSessionConfiguration](#)
- [Enable-PSSessionConfiguration](#)
- [Get-PSSessionConfiguration](#)
- [Register-PSSessionConfiguration](#)
- [Set-PSSessionConfiguration](#)
- [Disable-PSRemoting](#)
- [WSMan Provider](#)
- [about\\_Remote](#)
- [about\\_Session\\_Configurations](#)


### Collaborate with us on GitHub


The source for this content can be found on GitHub, where you can also create and review issues and pull requests. For more information, see [our contributor guide](#).




### PowerShell feedback



PowerShell is an open source project. Select a link to provide feedback:

 [Open a documentation issue](#)

 [Provide product feedback](#)

 English (United States)

 Your Privacy Choices


 Theme 

[Manage cookies](#)


[Previous Versions](#)

[Blog](#) 

[Contribute](#)

[Privacy](#) 

[Terms of Use](#)

[Trademarks](#) 

© Microsoft 2024