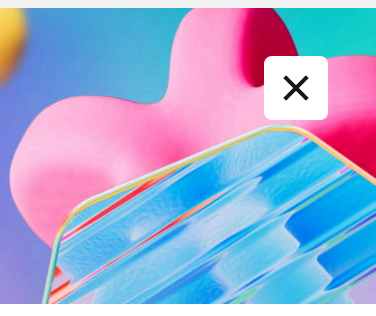

Set-ExecutionPolicy

Reference

Feedback

Module: Microsoft.PowerShell.Security

In this article

Syntax
Description
Examples
Parameters

Show 4 more

Sets the PowerShell execution policies for Windows computers.

Syntax

PowerShell Copy

```
Set-ExecutionPolicy
   [-ExecutionPolicy] <ExecutionPolicy>
   [[-Scope] <ExecutionPolicyScope>]
   [-Force]
   [-WhatIf]
   [-Confirm]
   [<CommonParameters>]
```

Description

The `Set-ExecutionPolicy` cmdlet changes PowerShell execution policies for Windows computers. For more information, see about_Execution_Policies.

Beginning in PowerShell 6.0 for non-Windows computers, the default execution policy is `Unrestricted` and can't be changed. The `Set-ExecutionPolicy` cmdlet is available, but PowerShell displays a console message that it's not supported.

An execution policy is part of the PowerShell security strategy. Execution policies determine whether you can load configuration files, such as your PowerShell profile, or run scripts. And, whether scripts must be digitally signed before they are run.

Download PDF

The `Set-ExecutionPolicy` cmdlet's default scope is `LocalMachine`, which affects everyone who uses the computer. To change the execution policy for `LocalMachine`, start PowerShell with **Run as Administrator**.

To display the execution policies for each scope, use `Get-ExecutionPolicy -List`. To see the effective execution policy for your PowerShell session use `Get-ExecutionPolicy` with no parameters.

Examples

Example 1: Set an execution policy

This example shows how to set the execution policy for the local computer.

PowerShell Copy

```
Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Scope LocalMachine
Get-ExecutionPolicy -List

Scope ExecutionPolicy
        ----- ---------------
MachinePolicy       Undefined
   UserPolicy       Undefined
      Process       Undefined
  CurrentUser    RemoteSigned
 LocalMachine    RemoteSigned
```

The `Set-ExecutionPolicy` cmdlet uses the **ExecutionPolicy** parameter to specify the `RemoteSigned` policy. The **Scope** parameter specifies the default scope value, `LocalMachine`. To view the execution policy settings, use the `Get-ExecutionPolicy` cmdlet with the **List** parameter.

Example 2: Set an execution policy that conflicts with a Group Policy

This command attempts to set the `LocalMachine` scope's execution policy to `Restricted`. `LocalMachine` is more restrictive, but isn't the effective policy because it conflicts with a Group Policy. The `Restricted` policy is written to the registry hive `HKEY_LOCAL_MACHINE`.

PowerShell Copy

```
PS> Set-ExecutionPolicy -ExecutionPolicy Restricted -Scope LocalMachine

Set-ExecutionPolicy : PowerShell updated your local preference successfully, but
overridden by the Group Policy applied to your system. Due to the override, your
its current effective execution policy of "AllSigned". Contact your Group Policy
more information. At line:1 char:20 + Set-ExecutionPolicy <<<< restricted

PS> Get-ChildItem -Path HKLM:\SOFTWARE\Microsoft\PowerShell\1\ShellIds

    Hive: HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\PowerShell\1\ShellIds

Name                   Property
----                   --------
Microsoft.PowerShell   Path            : C:\Windows\System32\WindowsPowerShell\
                       ExecutionPolicy : Restricted
ScriptedDiagnostics    ExecutionPolicy : Unrestricted
```

The `Set-ExecutionPolicy` cmdlet uses the **ExecutionPolicy** parameter to specify the `Restricted` policy. The **Scope** parameter specifies the default scope value, `LocalMachine`. The `Get-ChildItem` cmdlet uses the **Path** parameter with the `HKLM:` drive to specify registry location.

Example 3: Apply the execution policy from a remote computer to a local computer

This command gets the execution policy object from a remote computer and sets the policy on the local computer. `Get-ExecutionPolicy` sends a **Microsoft.PowerShell.ExecutionPolicy** object down the pipeline. `Set-ExecutionPolicy` accepts pipeline input and doesn't require the **ExecutionPolicy** parameter.

PowerShell Copy

```
Invoke-Command -ComputerName Server01 -ScriptBlock { Get-ExecutionPolicy } | Set
```

The `Invoke-Command` cmdlet is executed at the local computer and sends the **ScriptBlock** to the remote computer. The **ComputerName** parameter specifies the remote computer, **Server01**. The **ScriptBlock** parameter runs `Get-ExecutionPolicy` on the remote computer. The `Get-ExecutionPolicy` object is sent down the pipeline to the `Set-ExecutionPolicy`. `Set-ExecutionPolicy` applies the execution policy to the local computer's default scope, `LocalMachine`.

Example 4: Set the scope for an execution policy

This example shows how to set an execution policy for a specified scope, `CurrentUser`. The `CurrentUser` scope only affects the user who sets this scope.

PowerShell Copy

```
Set-ExecutionPolicy -ExecutionPolicy AllSigned -Scope CurrentUser
Get-ExecutionPolicy -List

Scope ExecutionPolicy
        ----- ---------------
MachinePolicy       Undefined
   UserPolicy       Undefined
      Process       Undefined
  CurrentUser       AllSigned
 LocalMachine    RemoteSigned
```

`Set-ExecutionPolicy` uses the **ExecutionPolicy** parameter to specify the `AllSigned` policy. The **Scope** parameter specifies the `CurrentUser`. To view the execution policy settings, use the `Get-ExecutionPolicy` cmdlet with the **List** parameter.

The effective execution policy for the user becomes `AllSigned`.

Example 5: Remove the execution policy for the current user

This example shows how use the `Undefined` execution policy to remove an execution policy for a specified scope.

PowerShell Copy

```
Set-ExecutionPolicy -ExecutionPolicy Undefined -Scope CurrentUser
Get-ExecutionPolicy -List

Scope ExecutionPolicy
        ----- ---------------
MachinePolicy       Undefined
   UserPolicy       Undefined
      Process       Undefined
  CurrentUser       Undefined
 LocalMachine    RemoteSigned
```

`Set-ExecutionPolicy` uses the **ExecutionPolicy** parameter to specify the `Undefined` policy. The **Scope** parameter specifies the `CurrentUser`. To view the execution policy settings, use the `Get-ExecutionPolicy` cmdlet with the **List** parameter.

Example 6: Set the execution policy for the current PowerShell session

The `Process` scope only affects the current PowerShell session. The execution policy is saved in the environment variable `$env:PSExecutionPolicyPreference` and is deleted when the session is closed.

PowerShell Copy

```
Set-ExecutionPolicy -ExecutionPolicy AllSigned -Scope Process

Scope ExecutionPolicy
        ----- ---------------
MachinePolicy       Undefined
   UserPolicy       Undefined
      Process       AllSigned
  CurrentUser    RemoteSigned
 LocalMachine    RemoteSigned
```

The `Set-ExecutionPolicy` uses the **ExecutionPolicy** parameter to specify the `AllSigned` policy. The **Scope** parameter specifies the value `Process`. To view the execution policy settings, use the `Get-ExecutionPolicy` cmdlet with the **List** parameter.

Example 7: Unblock a script to run it without changing the execution policy

This example shows how the `RemoteSigned` execution policy prevents you from running unsigned scripts.

A best practice is to read the script's code and verify it's safe *before* using the `Unblock-File` cmdlet. The `Unblock-File` cmdlet unblocks scripts so they can run, but doesn't change the execution policy.

PowerShell Copy

```
PS> Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Scope LocalMachine

PS> Get-ExecutionPolicy

RemoteSigned

PS> .\Start-ActivityTracker.ps1

.\Start-ActivityTracker.ps1 : File .\Start-ActivityTracker.ps1 cannot be loaded.
The file .\Start-ActivityTracker.ps1 is not digitally signed.
The script will not execute on the system.
For more information, see about_Execution_Policies at https://go.microsoft.com/f
At line:1 char:1
+ .\Start-ActivityTracker.ps1
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~
+ CategoryInfo          : NotSpecified: (:) [], PSSecurityException
+ FullyQualifiedErrorId : UnauthorizedAccess

PS> Unblock-File -Path .\Start-ActivityTracker.ps1

PS> Get-ExecutionPolicy

RemoteSigned

PS> .\Start-ActivityTracker.ps1
```

```
Task 1:
```

The `Set-ExecutionPolicy` uses the **ExecutionPolicy** parameter to specify the `RemoteSigned` policy. The policy is set for the default scope, `LocalMachine`.

The `Get-ExecutionPolicy` cmdlet shows that `RemoteSigned` is the effective execution policy for the current PowerShell session.

The `Start-ActivityTracker.ps1 script is executed from the current directory. The script is blocked by` RemoteSigned` because the script isn't digitally signed.

For this example, the script's code was reviewed and verified as safe to run. The `Unblock-File` cmdlet uses the **Path** parameter to unblock the script.

To verify that `Unblock-File` didn't change the execution policy, `Get-ExecutionPolicy` displays the effective execution policy, `RemoteSigned`.

The script, `Start-ActivityTracker.ps1` is executed from the current directory. The script begins to run because it was unblocked by the `Unblock-File` cmdlet.

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

-ExecutionPolicy

Specifies the execution policy. If there are no Group Policies and each scope's execution policy is set to `Undefined`, then `Restricted` becomes the effective policy for all users.

The acceptable execution policy values are as follows:

- `AllSigned`. Requires that all scripts and configuration files are signed by a trusted publisher, including scripts written on the local computer.
- `Bypass`. Nothing is blocked and there are no warnings or prompts.
- `Default`. Sets the default execution policy. `Restricted` for Windows clients or `RemoteSigned` for Windows servers.
- `RemoteSigned`. Requires that all scripts and configuration files downloaded from the Internet are signed by a trusted publisher. The default execution policy for Windows server computers.
- `Restricted`. Doesn't load configuration files or run scripts. The default execution policy for Windows client computers.
- `Undefined`. No execution policy is set for the scope. Removes an assigned execution policy from a scope that is not set by a Group Policy. If the execution policy in all

scopes is `Undefined`, the effective execution policy is `Restricted`.

- `Unrestricted`. Beginning in PowerShell 6.0, this is the default execution policy for non-Windows computers and can't be changed. Loads all configuration files and runs all scripts. If you run an unsigned script that was downloaded from the internet, you're prompted for permission before it runs.

Expand table

Type:	ExecutionPolicy
Accepted values:	AllSigned, Bypass, Default, RemoteSigned, Restricted, Undefined, Unrestricted
Position:	0
Default value:	None
Required:	True
Accept pipeline input:	True
Accept wildcard characters:	False

-Force

Suppresses all the confirmation prompts. Use caution with this parameter to avoid unexpected results.

Expand table

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

-Scope

Specifies the scope that is affected by an execution policy. The default scope is `LocalMachine`.

The effective execution policy is determined by the order of precedence as follows:

- `MachinePolicy` - Set by a Group Policy for all users of the computer
- `UserPolicy` - Set by a Group Policy for the current user of the computer
- `Process` - Affects only the current PowerShell session
- `LocalMachine` - Default scope that affects all users of the computer
- `CurrentUser` - Affects only the current user

The `Process` scope only affects the current PowerShell session. The execution policy is saved in the environment variable `$env:PSExecutionPolicyPreference`, rather than the registry. When the PowerShell session is closed, the variable and value are deleted.

Execution policies for the `CurrentUser` scope are written to the registry hive `HKEY_LOCAL_USER`.

Execution policies for the `LocalMachine` scope are written to the registry hive `HKEY_LOCAL_MACHINE`.

Expand table

Type:	ExecutionPolicyScope
Accepted values:	CurrentUser, LocalMachine, MachinePolicy, Process, UserPolicy
Position:	1
Default value:	LocalMachine
Required:	False
Accept pipeline input:	True
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

ExecutionPolicy

You can pipe an execution policy object to this cmdlet.

String

You can pipe a string that contains the name of an execution policy to this cmdlet.

Outputs

None

This cmdlet returns no output.

Notes

`Set-ExecutionPolicy` doesn't change the `MachinePolicy` and `UserPolicy` scopes because they are set by Group Policies.

`Set-ExecutionPolicy` doesn't override a Group Policy, even if the user preference is more restrictive than the policy.

If the Group Policy **Turn on Script Execution** is enabled for the computer or user, the user preference is saved, but it's not effective. PowerShell displays a message that explains the conflict.

Related Links

- about_Execution_Policies
- about_Group_Policy_Settings
- about_Providers
- Get-AuthenticodeSignature
- Get-ChildItem
- Get-ExecutionPolicy
- Invoke-Command
- Set-AuthenticodeSignature
- Unblock-File

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