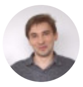
Windows PowerShell Scripting Tutorial for Beginners



Ian Skur (/author/ian_skur/)

Published: February 21, 2018

Automate it. Now, where's that script…

Warning: PowerShell is addictive.

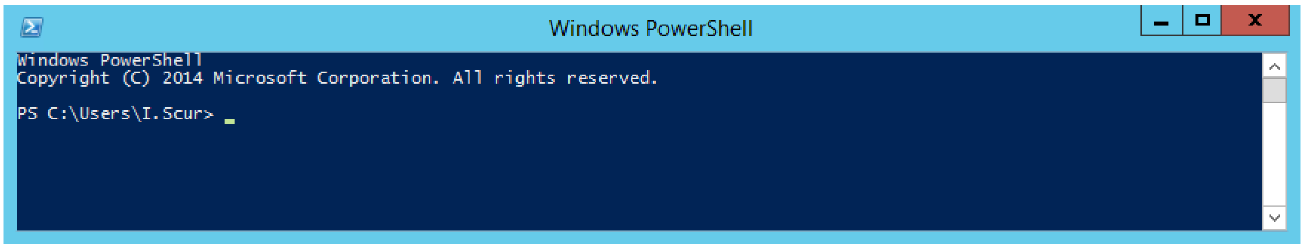
Meet PowerShell Tutorial

Windows PowerShell is an object-oriented automation engine and scripting language with an interactive command-line shell designed to help IT professionals configure systems and automate administrative tasks. You can find it in every modern Windows OS starting with Windows 2008R2. Learning Windows PowerShell is like learning to use a universal multi-tool. In this post, I cover PowerShell scripting basics so you can more easily perform virtually any administration task related to your Windows IT environment.

So let's start learning PowerShell.

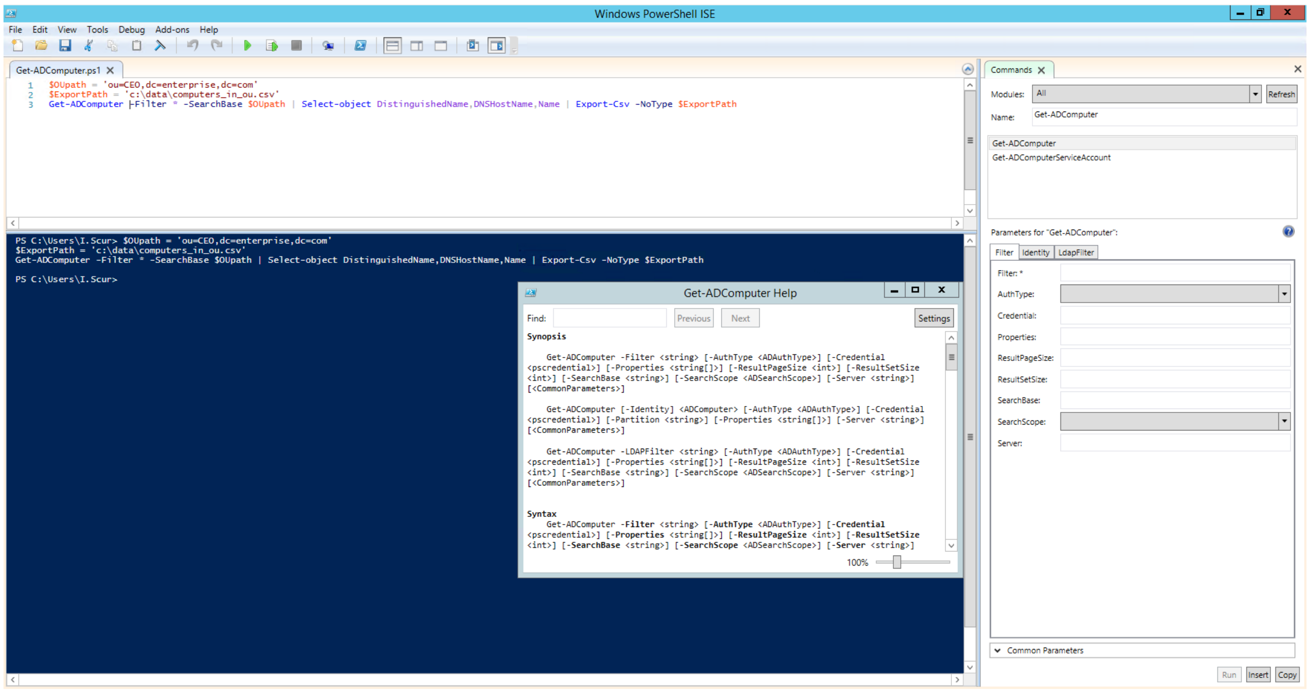
PowerShell offers both a command-line option and an integrated scripting environment (ISE):

- To launch the PowerShell command line, type *powershell.exe* in the Windows Start menu. You'll see a screen like the following:



```
Windows PowerShell
Copyright (C) 2014 Microsoft Corporation. All rights reserved.

PS C:\Users\I.Scur> _
```

- To launch the PowerShell ISE, type *powershell_ise.exe* in the Start menu. Using the PowerShell ISE is the preferred way to work with the scripting language, because it provides syntax highlighting, auto-filling of commands and other automation features that simplify script development and testing.



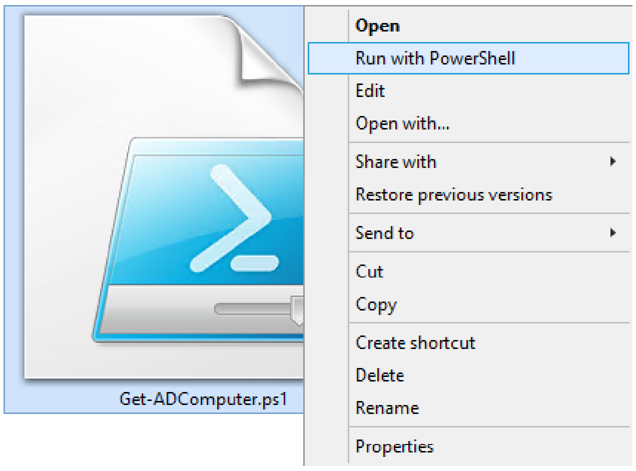
```
$OUpath = 'ou=CEO,dc=enterprise,dc=com'
$ExportPath = 'c:\data\computers_in_ou.csv'
Get-ADComputer -Filter * -SearchBase $OUpath | Select-object DistinguishedName,DNSHostName,Name | Export-Csv -NoType $ExportPath
```

```
PS C:\Users\I.Scur> $OUpath = 'ou=CEO,dc=enterprise,dc=com'
$ExportPath = 'c:\data\computers_in_ou.csv'
Get-ADComputer -Filter * -SearchBase $OUpath | Select-object DistinguishedName,DNSHostName,Name | Export-Csv -NoType $ExportPath

PS C:\Users\I.Scur>
```

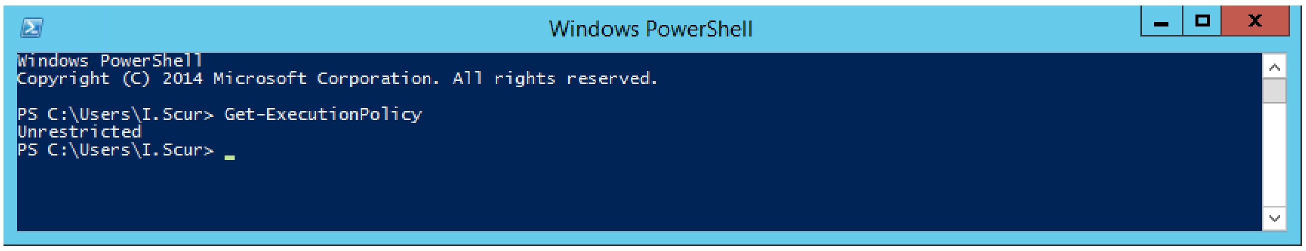
```
Synopsis

    Get-ADComputer -Filter <string> [-AuthType <ADAuthType>] [-Credential
<pscredential>] [-Properties <string[]>] [-ResultPageSize <int>] [-ResultSetSize
<int>] [-SearchBase <string>] [-SearchScope <ADSearchScope>] [-Server <string>]
[<CommonParameters>]

    Get-ADComputer [-Identity] <ADComputer> [-AuthType <ADAuthType>] [-Credential
<pscredential>] [-Partition <string>] [-Properties <string[]>] [-Server <string>]
[<CommonParameters>]

    Get-ADComputer -LDAPFilter <string> [-AuthType <ADAuthType>] [-Credential
<pscredential>] [-Properties <string[]>] [-ResultPageSize <int>] [-ResultSetSize
<int>] [-SearchBase <string>] [-SearchScope <ADSearchScope>] [-Server <string>]
[<CommonParameters>]


Syntax
    Get-ADComputer -Filter <string> [-AuthType <ADAuthType>] [-Credential
<pscredential>] [-Properties <string[]>] [-ResultPageSize <int>] [-ResultSetSize
<int>] [-SearchBase <string>] [-SearchScope <ADSearchScope>] [-Server <string>]
```

Preparing to Run PowerShell Scripts

PowerShell scripts are stored in.ps1 files. By design, you cannot run a script by simply double-clicking a file; this helps avoid accidental harm to your systems. Instead, to execute a script, right-click it and click “Run with PowerShell”:



In addition, there is policy that restricts script execution. You can check this policy by running the Get-ExecutionPolicy command in PowerShell:



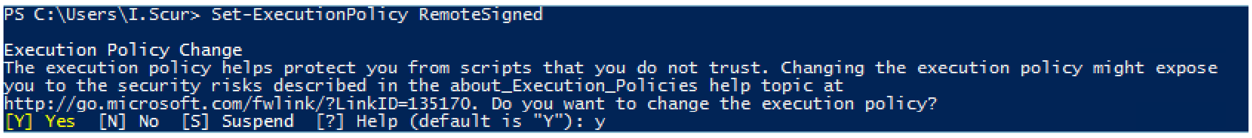
```
Windows PowerShell
Copyright (C) 2014 Microsoft Corporation. All rights reserved.

PS C:\Users\I.Scur> Get-ExecutionPolicy
Unrestricted
PS C:\Users\I.Scur> _
```

You will get one of the following values:

- **Restricted**— No scripts are allowed. This is the default setting, so you will see it the first time you run the command.
- **AllSigned**— You can run scripts signed by a trusted developer. With this setting in place, before executing, a script will ask you to confirm that you want to run it.
- **RemoteSigned**— You can run your own scripts or scripts signed by a trusted developer.

- **Unrestricted**— You can run any script you want.

To start working with PowerShell, you'll need to change the policy setting from Restricted to RemoteSigned using the Set-ExecutionPolicy RemoteSigned command:



```
PS C:\Users\I.Scur> Set-ExecutionPolicy RemoteSigned

Execution Policy Change
The execution policy helps protect you from scripts that you do not trust. Changing the execution policy might expose
you to the security risks described in the about_Execution_Policies help topic at
http://go.microsoft.com/fwlink/?LinkID=135170. Do you want to change the execution policy?
[Y] Yes  [N] No  [S] Suspend  [?] Help (default is "Y"): y
```

PowerShell Cmdlets

About Cmdlets

A cmdlet is a PowerShell command with a predefined function, similar to an operator in a programming language. Here are some key things to know about cmdlets:

- There are system, user and custom cmdlets.
- Cmdlets output results as an object or as an array of objects.
- Cmdlets can get data for analysis or transfer data to another cmdlet using pipes (I'll discuss pipes more in a moment).
- Cmdlets are case-insensitive. For example, it doesn't matter whether you type "Get-ADUser", "get-aduser" or "gEt-AdUsEr".
- If you want to use several cmdlets in one string, you must separate them with a semicolon (;).

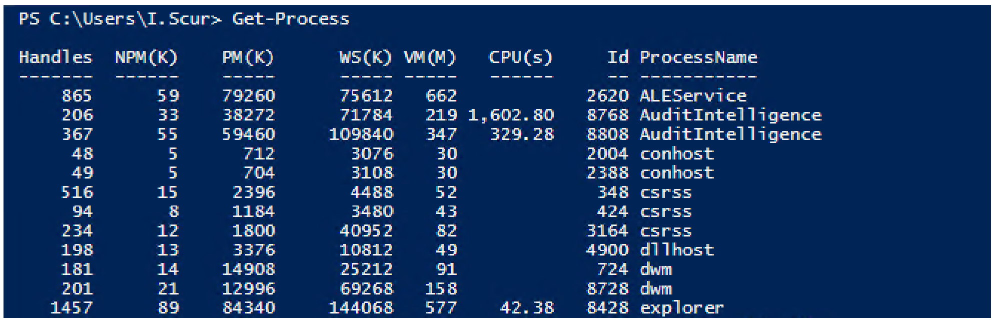
Cmdlet Format

A cmdlet always consists of a verb (or a word that functions as a verb) and a noun, separated with a hyphen (the "verb-noun" rule). For example, some of the verbs include:

- **Get** — To get something
- **Set** — To define something
- **Start** — To run something
- **Stop** — To stop something that is running
- **Out** — To output something
- **New** — To create something ("new" is not a verb, of course, but it functions as one)

For practice, try executing the following cmdlets:

- Get-Process — Shows the processes currently running on your computer:

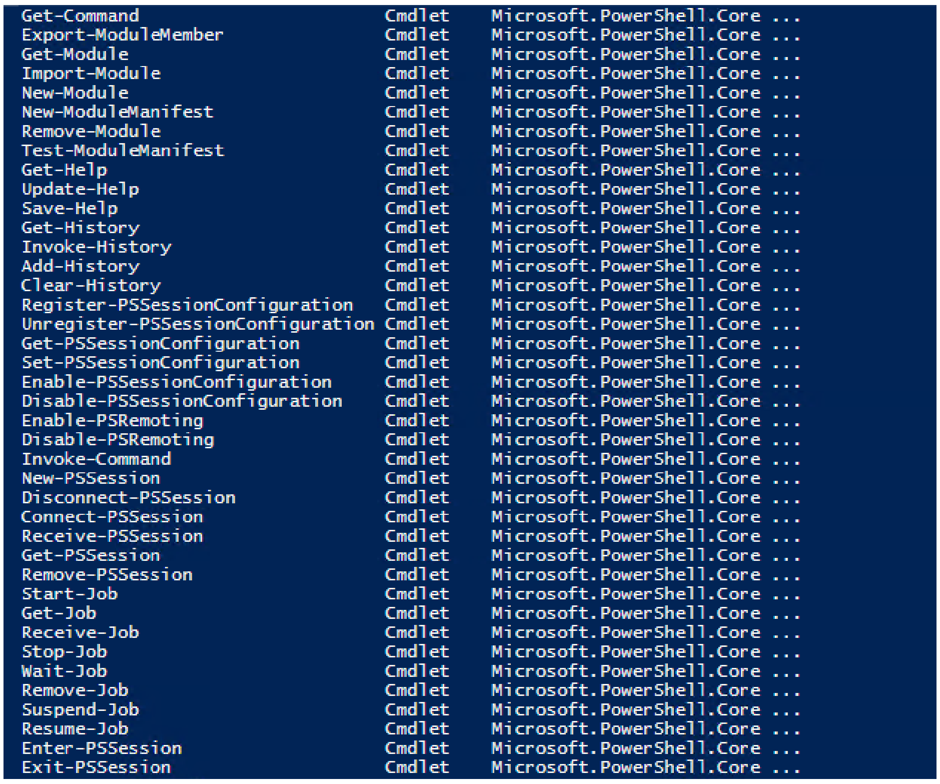


```
PS C:\Users\I.Scur> Get-Process

Handles  NPM(K)    PM(K)      WS(K) VM(M)   CPU(s)     Id ProcessName
-------  ------    -----      ----- -----   ------     -- -----------
    865      59    79260      75612   662            2620 ALEService
    206      33    38272      71784   219 1,602.80   8768 AuditIntelligence
    367      55    59460     109840   347   329.28   8808 AuditIntelligence
     48       5      712       3076    30            2004 conhost
     49       5      704       3108    30            2388 conhost
    516      15     2396       4488    52             348 csrss
     94       8     1184       3480    43             424 csrss
    234      12     1800      40952    82            3164 csrss
    198      13     3376      10812    49            4900 dllhost
    181      14    14908      25212    91             724 dwm
    201      21    12996      69268   158            8728 dwm
   1457      89    84340     144068   577    42.38   8428 explorer
```

- Get-Service — Shows the list of services with their status
- Get-Content — Shows the content of the file you specify (for example, Get-Content C:\Windows\System32\drivers\etc\hosts)

Available Cmdlets

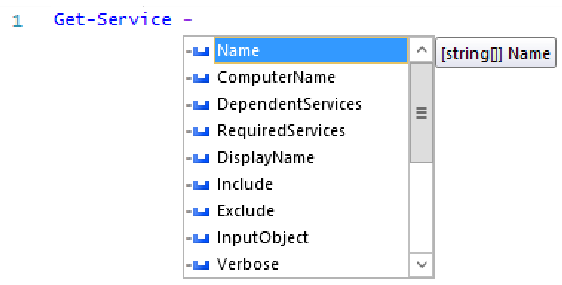
Good news — you don't need to memorize all cmdlets. You can list all cmdlets by executing the Get-Help -Category cmdlet, which will return the following:


```
Get-Command                        Cmdlet    Microsoft.PowerShell.Core ...
Export-ModuleMember                Cmdlet    Microsoft.PowerShell.Core ...
Get-Module                         Cmdlet    Microsoft.PowerShell.Core ...
Import-Module                      Cmdlet    Microsoft.PowerShell.Core ...
New-Module                         Cmdlet    Microsoft.PowerShell.Core ...
New-ModuleManifest                 Cmdlet    Microsoft.PowerShell.Core ...
Remove-Module                      Cmdlet    Microsoft.PowerShell.Core ...
Test-ModuleManifest                Cmdlet    Microsoft.PowerShell.Core ...
Get-Help                           Cmdlet    Microsoft.PowerShell.Core ...
Update-Help                        Cmdlet    Microsoft.PowerShell.Core ...
Save-Help                          Cmdlet    Microsoft.PowerShell.Core ...
Get-History                        Cmdlet    Microsoft.PowerShell.Core ...
Invoke-History                     Cmdlet    Microsoft.PowerShell.Core ...
Add-History                        Cmdlet    Microsoft.PowerShell.Core ...
Clear-History                      Cmdlet    Microsoft.PowerShell.Core ...
Register-PSSessionConfiguration    Cmdlet    Microsoft.PowerShell.Core ...
Unregister-PSSessionConfiguration Cmdlet    Microsoft.PowerShell.Core ...
Get-PSSessionConfiguration         Cmdlet    Microsoft.PowerShell.Core ...
Set-PSSessionConfiguration         Cmdlet    Microsoft.PowerShell.Core ...
Enable-PSSessionConfiguration      Cmdlet    Microsoft.PowerShell.Core ...
Disable-PSSessionConfiguration     Cmdlet    Microsoft.PowerShell.Core ...
Enable-PSRemoting                  Cmdlet    Microsoft.PowerShell.Core ...
Disable-PSRemoting                 Cmdlet    Microsoft.PowerShell.Core ...
Invoke-Command                     Cmdlet    Microsoft.PowerShell.Core ...
New-PSSession                      Cmdlet    Microsoft.PowerShell.Core ...
Disconnect-PSSession               Cmdlet    Microsoft.PowerShell.Core ...
Connect-PSSession                  Cmdlet    Microsoft.PowerShell.Core ...
Receive-PSSession                  Cmdlet    Microsoft.PowerShell.Core ...
Get-PSSession                      Cmdlet    Microsoft.PowerShell.Core ...
Remove-PSSession                   Cmdlet    Microsoft.PowerShell.Core ...
Start-Job                          Cmdlet    Microsoft.PowerShell.Core ...
Get-Job                            Cmdlet    Microsoft.PowerShell.Core ...
Receive-Job                        Cmdlet    Microsoft.PowerShell.Core ...
Stop-Job                           Cmdlet    Microsoft.PowerShell.Core ...
Wait-Job                           Cmdlet    Microsoft.PowerShell.Core ...
Remove-Job                         Cmdlet    Microsoft.PowerShell.Core ...
Suspend-Job                        Cmdlet    Microsoft.PowerShell.Core ...
Resume-Job                         Cmdlet    Microsoft.PowerShell.Core ...
Enter-PSSession                    Cmdlet    Microsoft.PowerShell.Core ...
Exit-PSSession                     Cmdlet    Microsoft.PowerShell.Core ...
```

You can also create your own custom cmdlets.

Parameters

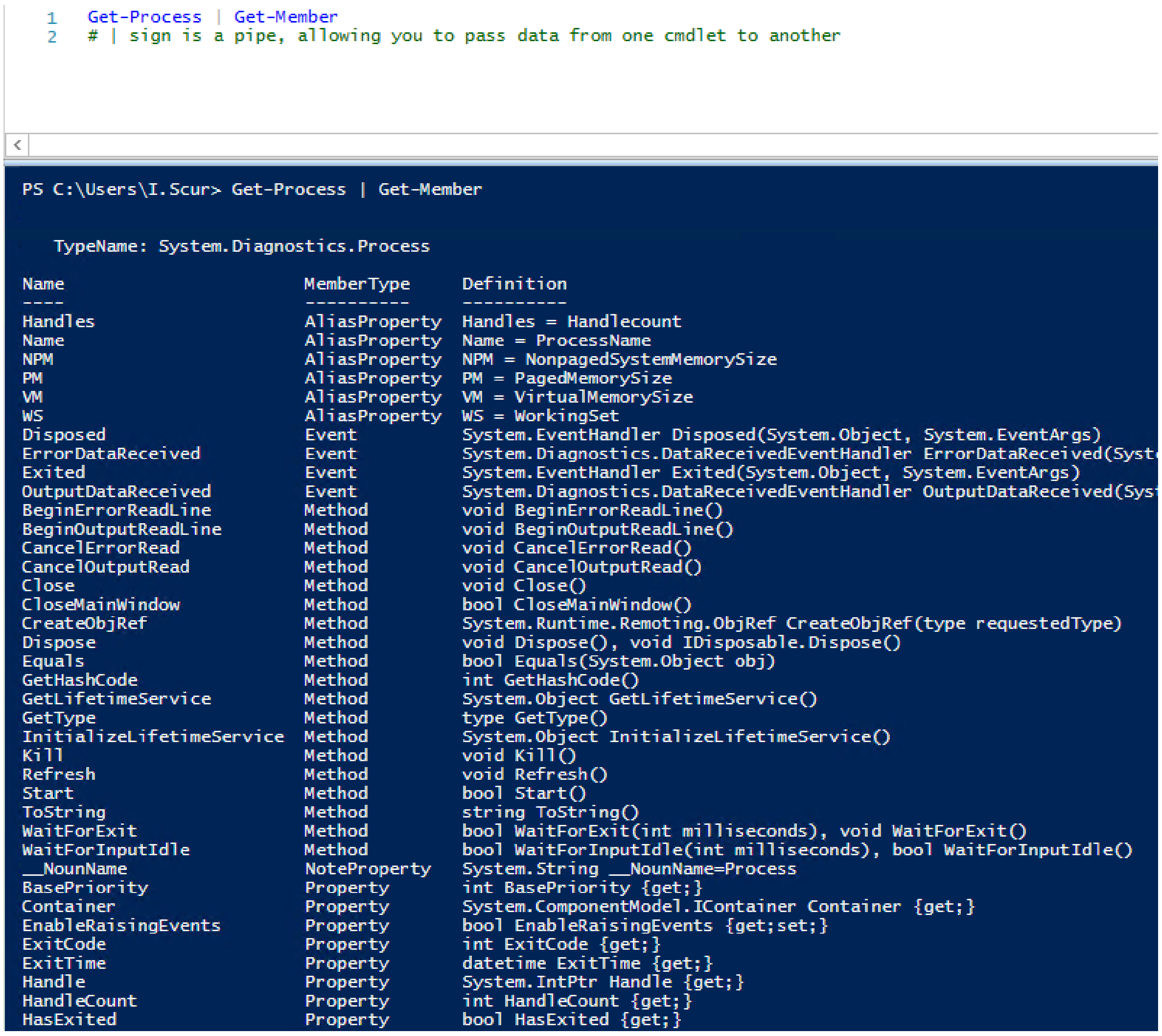
Each cmdlet has several parameters that customize what it does. The PowerShell ISE will automatically suggest all valid parameters and their types after you type a cmdlet and a hyphen (-):


```
Get-Service -
```

For example, the following cmdlet shows all services whose names start with “W”:
Get-Service -Name W*

If you forget a cmdlet’s parameters, just use a script like the following, which will display the parameters for the Get-Process cmdlet:

Get-Process | Get-Member

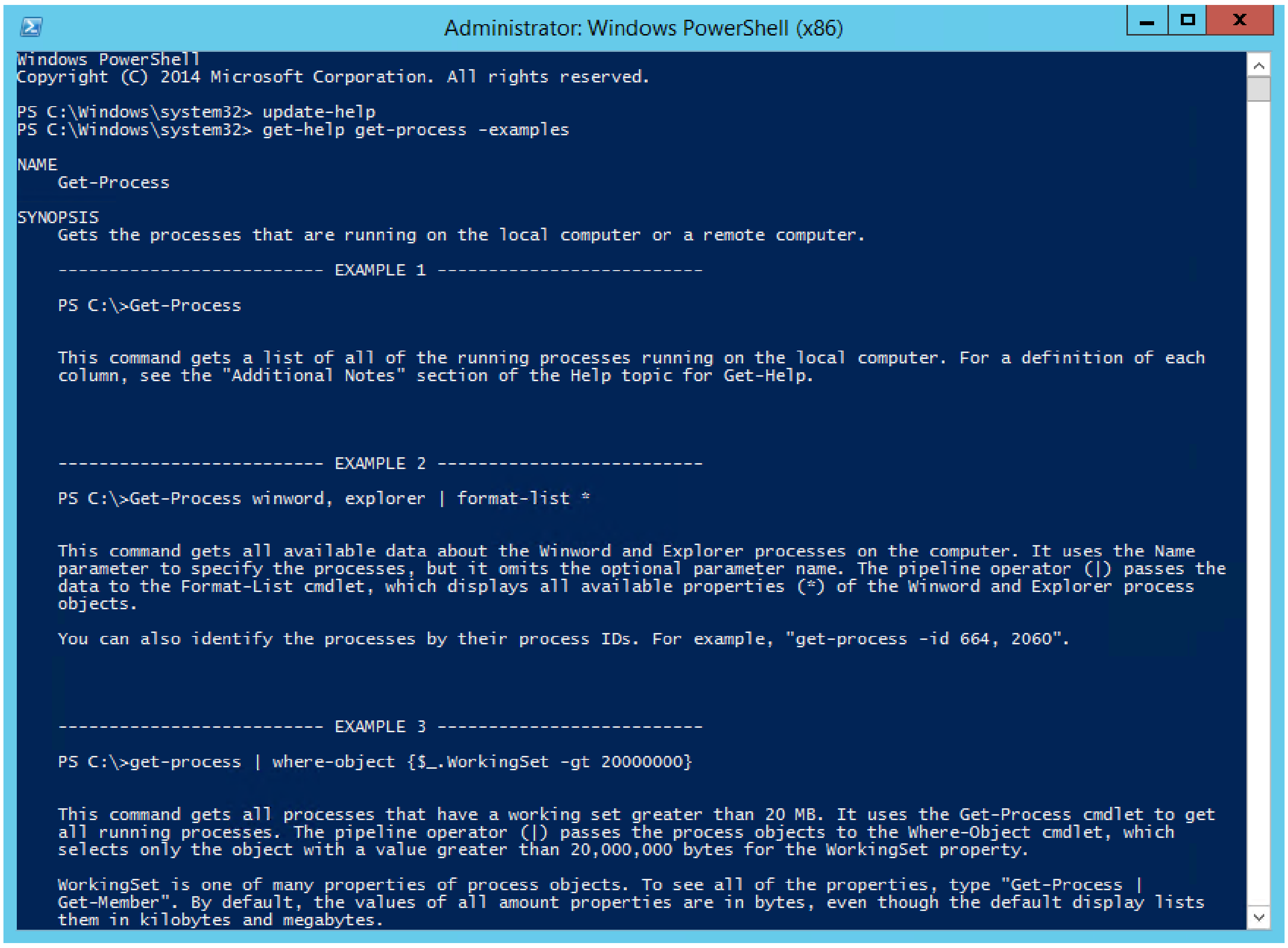


```
Get-Process | Get-Member
# | sign is a pipe, allowing you to pass data from one cmdlet to another
```

```
PS C:\Users\I.Scur> Get-Process | Get-Member


   TypeName: System.Diagnostics.Process

Name                       MemberType     Definition
----                       ----------     ----------
Handles                    AliasProperty  Handles = Handlecount
Name                       AliasProperty  Name = ProcessName
NPM                        AliasProperty  NPM = NonpagedSystemMemorySize
PM                         AliasProperty  PM = PagedMemorySize
VM                         AliasProperty  VM = VirtualMemorySize
WS                         AliasProperty  WS = WorkingSet
Disposed                   Event          System.EventHandler Disposed(System.Object, System.EventArgs)
ErrorDataReceived          Event          System.Diagnostics.DataReceivedEventHandler ErrorDataReceived(Syst
Exited                     Event          System.EventHandler Exited(System.Object, System.EventArgs)
OutputDataReceived         Event          System.Diagnostics.DataReceivedEventHandler OutputDataReceived(Sys
BeginErrorReadLine         Method         void BeginErrorReadLine()
BeginOutputReadLine        Method         void BeginOutputReadLine()
CancelErrorRead            Method         void CancelErrorRead()
CancelOutputRead           Method         void CancelOutputRead()
Close                      Method         void Close()
CloseMainWindow            Method         bool CloseMainWindow()
CreateObjRef               Method         System.Runtime.Remoting.ObjRef CreateObjRef(type requestedType)
Dispose                    Method         void Dispose(), void IDisposable.Dispose()
Equals                     Method         bool Equals(System.Object obj)
GetHashCode                Method         int GetHashCode()
GetLifetimeService         Method         System.Object GetLifetimeService()
GetType                    Method         type GetType()
InitializeLifetimeService  Method         System.Object InitializeLifetimeService()
Kill                       Method         void Kill()
Refresh                    Method         void Refresh()
Start                      Method         bool Start()
ToString                   Method         string ToString()
WaitForExit                Method         bool WaitForExit(int milliseconds), void WaitForExit()
WaitForInputIdle           Method         bool WaitForInputIdle(int milliseconds), bool WaitForInputIdle()
__NounName                 NoteProperty   System.String __NounName=Process
BasePriority               Property       int BasePriority {get;}
Container                  Property       System.ComponentModel.IContainer Container {get;}
EnableRaisingEvents        Property       bool EnableRaisingEvents {get;set;}
ExitCode                   Property       int ExitCode {get;}
ExitTime                   Property       datetime ExitTime {get;}
Handle                     Property       System.IntPtr Handle {get;}
HandleCount                Property       int HandleCount {get;}
HasExited                  Property       bool HasExited {get;}
```

If you still don't find the cmdlet you need, you can make sure the help is current and then get examples for a cmdlet (such as Get-Process) using a script like this:

```
Update-Help #to update the help data
Get-Help Get-Process -Examples
```



```
Windows PowerShell
Copyright (C) 2014 Microsoft Corporation. All rights reserved.

PS C:\Windows\system32> update-help
PS C:\Windows\system32> get-help get-process -examples

NAME
    Get-Process

SYNOPSIS
    Gets the processes that are running on the local computer or a remote computer.

    -------------------------- EXAMPLE 1 --------------------------

    PS C:\>Get-Process


    This command gets a list of all of the running processes running on the local computer. For a definition of each
    column, see the "Additional Notes" section of the Help topic for Get-Help.




    -------------------------- EXAMPLE 2 --------------------------

    PS C:\>Get-Process winword, explorer | format-list *


    This command gets all available data about the Winword and Explorer processes on the computer. It uses the Name
    parameter to specify the processes, but it omits the optional parameter name. The pipeline operator (|) passes the
    data to the Format-List cmdlet, which displays all available properties (*) of the Winword and Explorer process
    objects.

    You can also identify the processes by their process IDs. For example, "get-process -id 664, 2060".




    -------------------------- EXAMPLE 3 --------------------------

    PS C:\>get-process | where-object {$_.WorkingSet -gt 20000000}


    This command gets all processes that have a working set greater than 20 MB. It uses the Get-Process cmdlet to get
    all running processes. The pipeline operator (|) passes the process objects to the Where-Object cmdlet, which
    selects only the object with a value greater than 20,000,000 bytes for the WorkingSet property.

    WorkingSet is one of many properties of process objects. To see all of the properties, type "Get-Process |
    Get-Member". By default, the values of all amount properties are in bytes, even though the default display lists
    them in kilobytes and megabytes.
```

Aliases

You can also use aliases, which are shortened cmdlet names. For instance, instead of “Get-Help” you can use just “Help”. Try running the following two commands and see whether you get the same result:

- Start-Process notepad
- start notepad

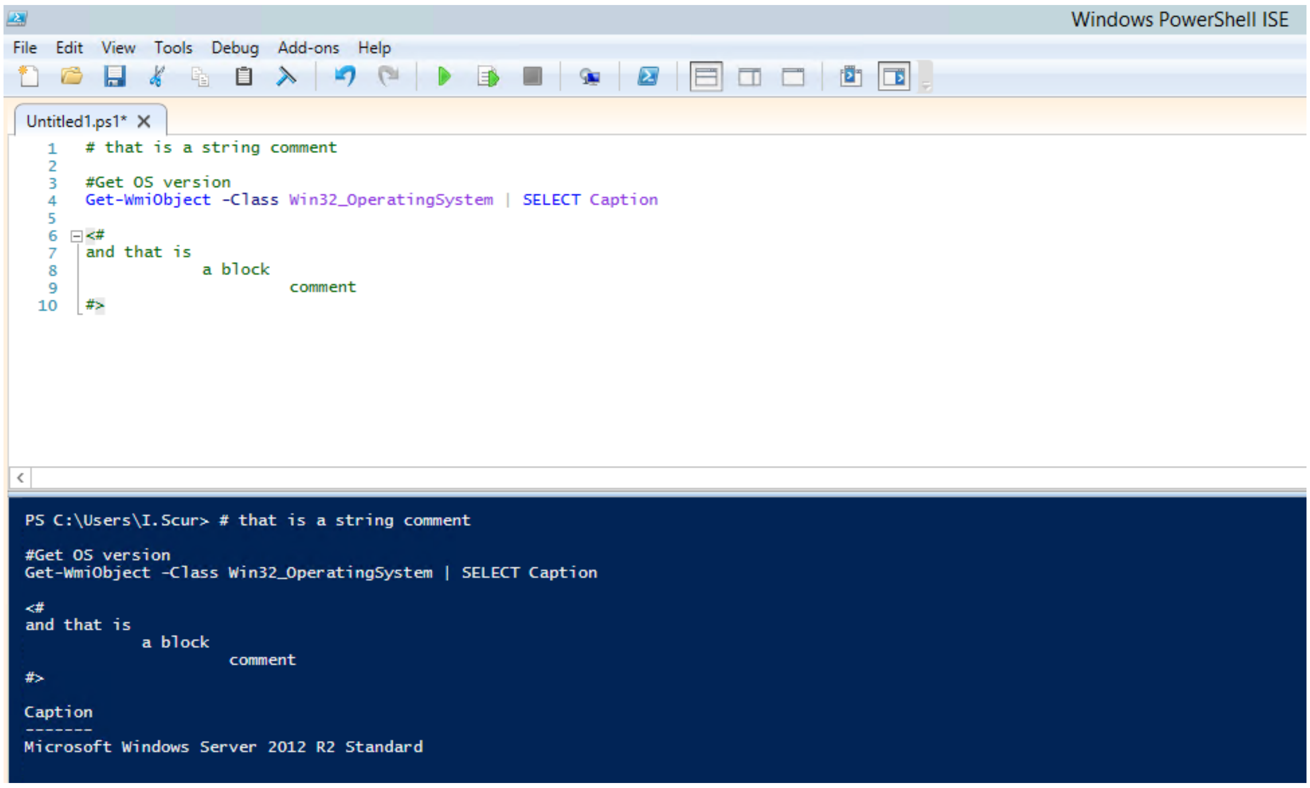
Similarly, to stop this process, you can use either of the following commands:

- Stop-Process -Name notepad
- spps -Name notepad

To see all aliases, execute the Get-Alias cmdlet.

Comments

Leaving comments in a script will help you — and your colleagues — better understand what the script does. A string comment is a single line that starts with a number sign (#), and block comments start and end with number signs and angle brackets and spread across multiple lines.



```
# that is a string comment

#Get OS version
Get-WmiObject -Class Win32_OperatingSystem | SELECT Caption

<#
and that is
            a block
                     comment
#>
```

```
PS C:\Users\I.Scur> # that is a string comment

#Get OS version
Get-WmiObject -Class Win32_OperatingSystem | SELECT Caption

<#
and that is
            a block
                     comment
#>

Caption
-------
Microsoft Windows Server 2012 R2 Standard
```

Pipes

A pipe passes data from one cmdlet to another. I used a pipe earlier to get all properties of an object.

For example, if you execute the following script, you'll get all services sorted by their status:

```
Get-Service | Sort-Object -property Status
```

You can also use a pipe to output text to a file using a script like the following:

```
"Hello, World!" | Out-File C:\ps\test.txt
```

You can use multiple pipes. For instance, the following script lists all services, with the first pipe excluding stopped services and the second pipe limiting the list to display names only:

```
Get-Service | WHERE {$_.status -eq "Running"} | SELECT displayname
# “$_.” defines current element in the pipe
```

Summary

Let's quickly summarize the key points of this Windows PowerShell tutorial. Now you know how to run PowerShell, how to change execution policy, what a cmdlet is, how to pass data using pipe and how to get object properties. Keep in mind that if you forget something, you can always use the Get-Help cmdlet.

I hope you found this PowerShell for beginners useful!

In the next parts of this free PowerShell tutorial, I'll cover variables, arrays and cycles, which are used in PowerShell scripting to simplify the administration of Windows Servers.

In the meantime, you might want to check out the following PowerShell scripts used by IT professionals to address specific cases:

1. How to Export Members of a Particular AD Group (https://www.netwrix.com/how_to_export_members_from_active_directory.html)
2. How to Get Local Group Membership Reports (https://www.netwrix.com/how_to_get_local_group_membership_report.html)
3. How to Export Folder Permissions to Excel or CSV File (https://www.netwrix.com/how_to_export_folder_permissions.html)
4. How to Export Group Policy Settings in Minutes (https://www.netwrix.com/how_to_report_on_gpo_security_settings.html)
5. How to Detect Every Active Directory User's Last Logon Date (https://www.netwrix.com/how_to_determine_last_logon_date.html)
6. How to Get Server Inventory across Your Network (https://www.netwrix.com/how_to_collect_server_inventory.html)
7. How to Get AD User Group Membership Reports (https://www.netwrix.com/how_to_get_ad_user_group_membership.html)
8. How to Export Active Directory Objects to CSV (https://www.netwrix.com/how_to_export_active_directory_objects_to_csv.html)
9. How to Detect Who Installed What Software on Your Windows Server (https://www.netwrix.com/how_to_detect_software_installations.html)
10. How to List All User Accounts on a Windows System (https://www.netwrix.com/how_to_list_all_user_accounts_on_a_windows_system.html)
11. How to Find Inactive Computers in Active Directory (https://www.netwrix.com/how_to_find_inactive_computers_active_directory_powershell.html?itm_source=blog&itm_medium=context&itm_campaign=powershell&itm_content=none&cID=70170000000kgEZ)
12. How Find Disabled or Inactive Users in AD (https://www.netwrix.com/how_to_get_disabled_users.html?itm_source=blog&itm_medium=context&itm_campaign=powershell&itm_content=none&cID=70170000000kgEZ)
13. Advanced Event Log Filtering Using PowerShell (https://blog.netwrix.com/2015/04/29/advanced-event-log-filtering-using-powershell/)
14. How to Get a List of Expired User Accounts in AD (https://www.netwrix.com/how_to_find_expired_accounts.html?itm_source=blog&itm_medium=context&itm_campaign=powershell&itm_content=none&cID=70170000000kgEZ)
15. How to Get a List of AD Users Whose Passwords Never Expire (https://blog.netwrix.com/2017/07/20/how-to-get-a-list-of-ad-users-whose-passwords-never-expire-using-powershell/)
16. How to Collect AD Site Information Using PowerShell (https://blog.netwrix.com/2017/07/20/how-to-collect-ad-site-information-using-powershell/)
17. How to Find Locked Out User Accounts in Active Directory (https://www.netwrix.com/how_to_find_locked_accounts_powershell.html?itm_source=blog&itm_medium=context&itm_campaign=powershell&itm_content=none&cID=70170000000kgEZ)
18. How to Create AD Users in Bulk and Email Their Credentials (https://blog.netwrix.com/2017/03/30/create-ad-users-in-bulk-and-email-their-credentials-using-powershell/)
19. How to Disable Inactive User Accounts (https://blog.netwrix.com/2015/03/17/how-to-disable-inactive-user-accounts-using-powershell/)
20. Ransomware Protection Using FSRM and PowerShell (https://blog.netwrix.com/2016/04/11/ransomware-protection-using-fsrm-and-powershell/)

And if you are just getting started with PowerShell, I recommend the following resources:

- PowerShell documentation (https://docs.microsoft.com/en-us/powershell/)
- "Hey, Scripting Guy!" blog (https://blogs.technet.microsoft.com/heyscriptingguy/)

If you want to get all the chapters at once, we've got you covered – the PowerShell series has been combined into one PDF document available for free download:

(https://www.netwrix.com/powershell_tutorial_pdf.html?itm_source=blog&itm_medium=banner&itm_campaign=powershell&itm_content=lower-banner&cID=70170000000kgEZ)

Cmdlet (https://blog.netwrix.com/tag/cmdlet/) PowerShell (https://blog.netwrix.com/tag/powershell/)

6 Comments **Netwrix Corporation Blog** 1 **Login**

Recommend 2 **Sort by Newest**

Join the discussion…

LOG IN WITH **OR SIGN UP WITH DISQUS** ?

Name

Alan Prage • 2 days ago
This was very informative and easy to follow. Thank you so much.
Reply • Share ›

Kan Moharana • 2 months ago
Hi Netwrix team
Thanks lot for the infos. Very useful for my job.
Reply • Share ›

Russell Abrogena • 3 months ago
This is very useful & I really like it...Thanks
Reply • Share ›

NetwrixMod Mod Russell Abrogena • 2 months ago
You're welcome, Russell!
Reply • Share ›

Pavitra Golchha • 5 months ago
this is so easier to understand! Thanks!
Reply • Share ›

NetwrixMod Mod Pavitra Golchha • 2 months ago
Thank you, Pavitra!
Reply • Share ›

Subscribe **Add Disqus to your siteAdd DisqusAdd** **Disqus' Privacy PolicyPrivacy PolicyPrivacy**

Previous post

The Ten Best Free Active Directo... (https://blog.netwrix.com/2018/02/15/the-ten-best-free-active-directory-management-tools/)

Next post

SysAdmin Magazine: Keep a Close ... (https://blog.netwrix.com/2018/02/22/sysadmin-magazine-keep-a-close-eye-on-permissions/)

POPULAR

Windows PowerShell Scripting Tutorial for Beginners
February 21, 2018 / Ian Skur

Top 5 Free Tools for NTFS Permissions Reporting
August 18, 2017 / Oleg Lalaev

Tutorial: Learn the Basics of Active Directory
April 20, 2017 / Ryan Brooks

FEATURED TAGS

Active Directory (/tag/active-directory/) Change audit (/tag/change-auditing/) Data breach (/tag/data-breach/)

Cyber security (/tag/cyber-security/) IT management (/tag/it-management/)

Active Directory audit (/tag/active-directory-audit/) Password management (/tag/password-management/)

PowerShell (/tag/powershell/) IT audit (/tag/it-audit/) IT security (/tag/security/) ... (/tags/)

Sysadmin Magazine

May the PowerShell Be with You

Read Online

Get top-notch cybersecurity insights and advice right in your inbox every week

Enter your business email

Sign Up

WRITE FOR US

Share useful insights with IT pros

Drop us a message (mailto:blog@netwrix.com)

STAY CONNECTED

Netwrix Auditor

Platform Overview (https://www.netwrix.com/auditor.html?itm_source=blog&itm_medium=footer&itm_campaign=none&itm_content=none&cID=70170000000kgEZ)

Feature Tour (https://www.netwrix.com/it_change_tracking_solution_features.html?itm_source=blog&itm_medium=footer&itm_campaign=none&itm_content=none&cID=70170000000kgEZ)

Request a Price Quote (https://www.netwrix.com/how_to_buy.html?itm_source=blog&itm_medium=footer&itm_campaign=none&itm_content=none&cID=70170000000kgEZ)

Solutions (https://www.netwrix.com/solutions.html?itm_source=blog&itm_medium=footer&itm_campaign=none&itm_content=none&cID=70170000000kgEZ)

Virtual Appliance (https://www.netwrix.com/virtual_appliances.html?itm_source=blog&itm_medium=footer&itm_campaign=none&itm_content=none&cID=70170000000kgEZ)

Cloud Vision (https://www.netwrix.com/cloud_opportunities.html?itm_source=blog&itm_medium=footer&itm_campaign=none&itm_content=none&cID=70170000000kgEZ)

Netwrix Freeware

Free Netwrix Auditor for Active Directory (https://www.netwrix.com/netwrix_change_notifier_for_active_directory.html?itm_source=blog&itm_medium=footer&itm_campaign=none&itm_content=none&cID=70170000000kgEZ)

Account Lockout Examiner (https://www.netwrix.com/account_lockout_examiner.html?itm_source=blog&itm_medium=footer&itm_campaign=none&itm_content=none&cID=70170000000kgEZ)

Top 7 Free Tools (https://www.netwrix.com/top_7_freeware_tools.html?itm_source=blog&itm_medium=footer&itm_campaign=none&itm_content=none&cID=70170000000kgEZ)

Audited Systems

Active Directory (https://www.netwrix.com/active_directory_auditing.html?itm_source=blog&itm_medium=footer&itm_campaign=none&itm_content=none&cID=70170000000kgEZ)

Azure AD (https://www.netwrix.com/azure_ad_auditing.html?itm_source=blog&itm_medium=footer&itm_campaign=none&itm_content=none&cID=70170000000kgEZ)

Office 365 (https://www.netwrix.com/office_365_auditing.html?itm_source=blog&itm_medium=footer&itm_campaign=none&itm_content=none&cID=70170000000kgEZ)

Compliance

PCI compliance (https://www.netwrix.com/PCI_Compliance.html?itm_source=blog&itm_medium=footer&itm_campaign=none&itm_content=none&c

HIPAA compliance (https://www.netwrix.com/HIPAA_Compliance.html?itm_source=blog&itm_medium=footer&itm_campaign=none&itm_content=none&c

SOX compliance (https://www.netwrix.com/SOX_Compliance.html?itm_source=blog&itm_medium=footer&itm_campaign=none&itm_content=none&c

Windows File Servers (https://www.netwrix.
itm_source=blog&itm_medium=footer&itm_

FISMA compliance (https://www.netwrix.com/FISMA_Compliance.html?
itm_source=blog&itm_medium=footer&itm_campaign=none&itm_content=none&c

EMC (https://www.netwrix.com/emc_storag
itm_source=blog&itm_medium=footer&itm_

ISO 27001 compliance (https://www.netwrix.com/ISO_IEC_Compliance.html?
itm_source=blog&itm_medium=footer&itm_campaign=none&itm_content=none&c

NetApp (https://www.netwrix.com/netapp_
itm_source=blog&itm_medium=footer&itm_

GLBA compliance (https://www.netwrix.com/GLBA_Compliance.html?
itm_source=blog&itm_medium=footer&itm_campaign=none&itm_content=none&c

Windows Server (https://www.netwrix.com/
itm_source=blog&itm_medium=footer&itm_

FERPA compliance (https://www.netwrix.com/FERPA_Compliance.html?
itm_source=blog&itm_medium=footer&itm_campaign=none&itm_content=none&c

Exchange (https://www.netwrix.com/exchange
itm_source=blog&itm_medium=footer&itm_

NERC compliance (https://www.netwrix.com/NERC_Compliance.html?
itm_source=blog&itm_medium=footer&itm_campaign=none&itm_content=none&c

SQL Server (https://www.netwrix.com/sql_server_auditing.html?
itm_source=blog&itm_medium=footer&itm_

GDPR compliance (https://www.netwrix.com/GDPR_Compliance.html?
itm_source=blog&itm_medium=footer&itm_campaign=none&itm_content=none&c

Oracle Database (https://www.netwrix.com/oracle_database_monitoring_and_auditing.html?
itm_source=blog&itm_medium=footer&itm_

CJIS compliance (https://www.netwrix.com/CJIS_Compliance.html?
itm_source=blog&itm_medium=footer&itm_campaign=none&itm_content=none&c

VMware (https://www.netwrix.com/vmware_auditing_reporting.html?
itm_source=blog&itm_medium=footer&itm_campaign=none&itm_content=none&cID=70170000000kgEZ)

SharePoint (https://www.netwrix.com/sharepoint_auditing.html?
itm_source=blog&itm_medium=footer&itm_campaign=none&itm_content=none&cID=70170000000kgEZ)

Support

Support Programs (https://www.netwrix.com/support.html?
itm_source=blog&itm_medium=footer&itm_

Knowledge Base (https://www.netwrix.com/knowledge_base.html?
itm_source=blog&itm_medium=footer&itm_

Submit Ticket (https://www.netwrix.com/open_a_ticket.html?
itm_source=blog&itm_medium=footer&itm_

Customer Portal (https://www.netwrix.com/tickets.html?
itm_source=blog&itm_medium=footer&itm_

Renew Maintenance (https://www.netwrix.com/support.html?
itm_source=blog&itm_medium=footer&itm_campaign=none&itm_content=none&cID=70170000000kgEZ#renewform)

Company

About us (https://www.netwrix.com/company.html?
itm_source=blog&itm_medium=footer&itm_campaign=none&itm_content=none&c

Contact Us (https://www.netwrix.com/contact.html?
itm_source=blog&itm_medium=footer&itm_campaign=none&itm_content=none&c

Our Customers (https://www.netwrix.com/customer_case_studies.html?
itm_source=blog&itm_medium=footer&itm_campaign=none&itm_content=none&c

News Room (https://www.netwrix.com/newsroom.html?
itm_source=blog&itm_medium=footer&itm_campaign=none&itm_content=none&c

Stay Connected

(https://www.linkedin.com/company/455932) (https://twitter.com/netwrix) (https://www.facebook.com/Netwrix)
(https://www.youtube.com/Netwrix)
(https://community.spiceworks.com/pages/NetWrix)
(https://www.instagram.com/netwrix/)