Just Enough Administration in Windows Server 2016

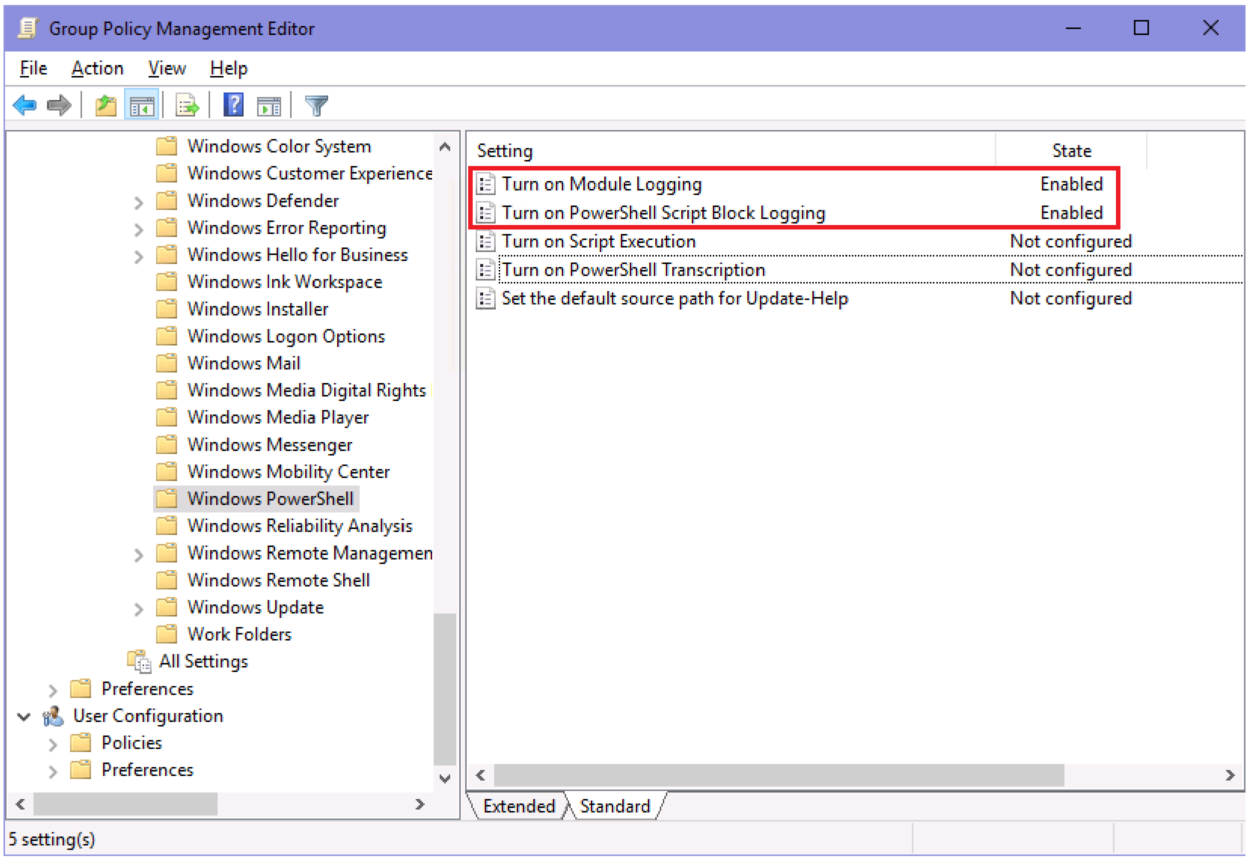
michaelfirsov.wordpress.com/just-enough-administration-in-windows-server-2016

February 14, 2018

You may already know that Windows Server 2016 has a new feature called JEA – Just Enough Administration. In this post I'd like to show in which circumstances it can be used and how it can be configured. As the purpose of JEA is to give administrators/users as few permissions as required for their work to be done, this technology would be pertinent in situations when different company administrators have different working tasks and when there's no other way not to give them excessive user rights or permissions on various servers or workstations. Suppose there's a junior administrator in your company that should have the right to configure the company DNS servers and nothing more, but considering the DNS servers are AD-integrated this administrator must have access to the company DCs because you can't separate GUI-based administration of AD from DNS. This is where JEA comes into play: it allows you to create PoweShell-based administrative endpoints that strictly define which actions may be taken by which users or groups. Now let's see how JEA can be deployed on a single domain member server.

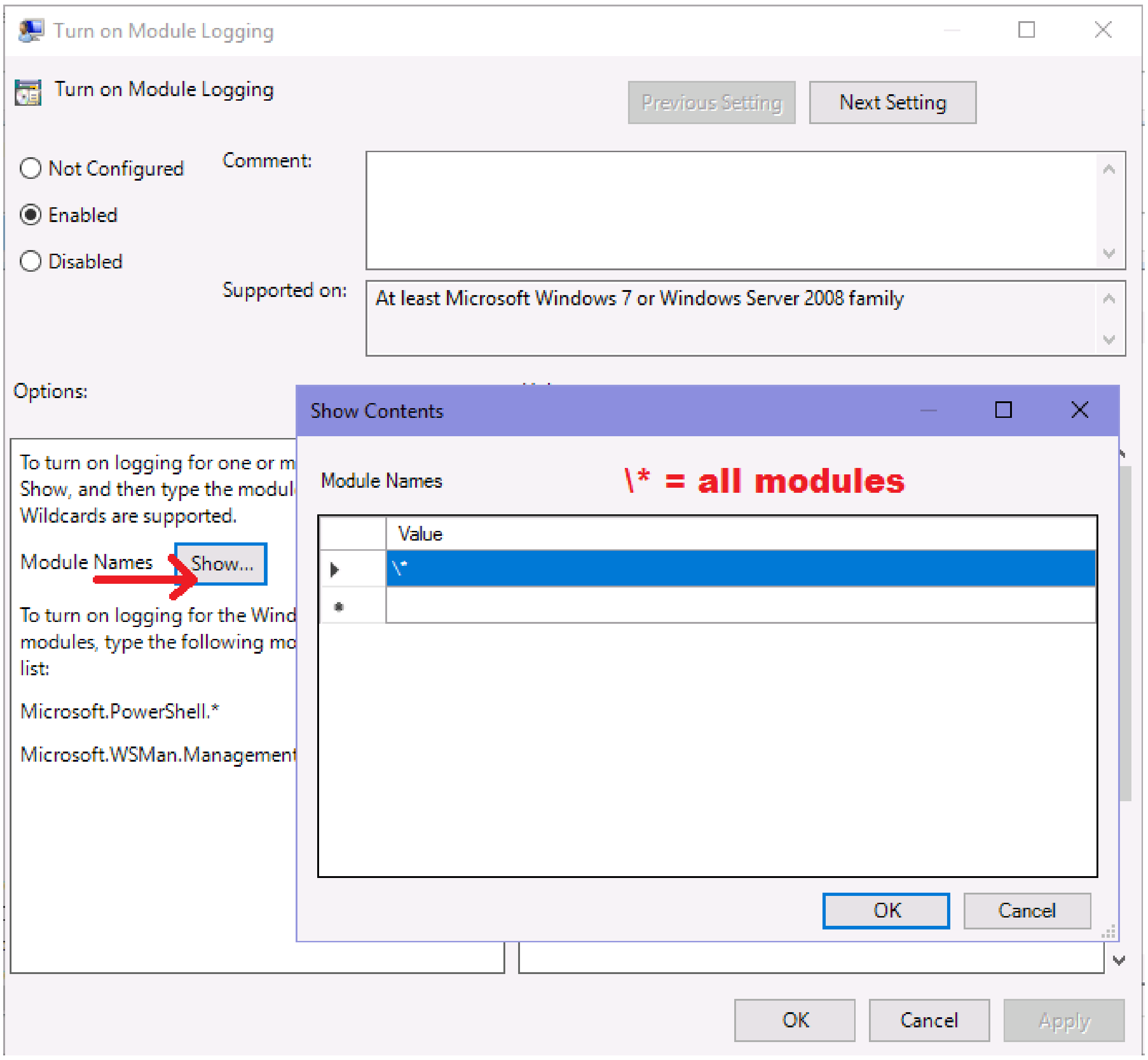
As JEA is a built-in feature of Windows Server 2016 no prerequisites must be installed, but it's highly recommended to enable script block logging in the corresponding GPO wich applies to the server or workstation on which the JEA endpoint to be created:

Computer Configuration\Administrative Templates\Windows Components\Windows PowerShell



Setting	State
Turn on Module Logging	Enabled
Turn on PowerShell Script Block Logging	Enabled
Turn on Script Execution	Not configured
Turn on PowerShell Transcription	Not configured
Set the default source path for Update-Help	Not configured

1) Enable **Turn on Module Logging** and select all modules by clicking on *Show…* and typing *


Turn on Module Logging

Turn on Module Logging

( ) Not Configured
(●) Enabled
( ) Disabled

Comment:

Supported on: At least Microsoft Windows 7 or Windows Server 2008 family

Options:

To turn on logging for one or m
Show, and then type the modul
Wildcards are supported.

Module Names Show...

To turn on logging for the Wind
modules, type the following mo
list:

Microsoft.PowerShell.*

Microsoft.WSMan.Management

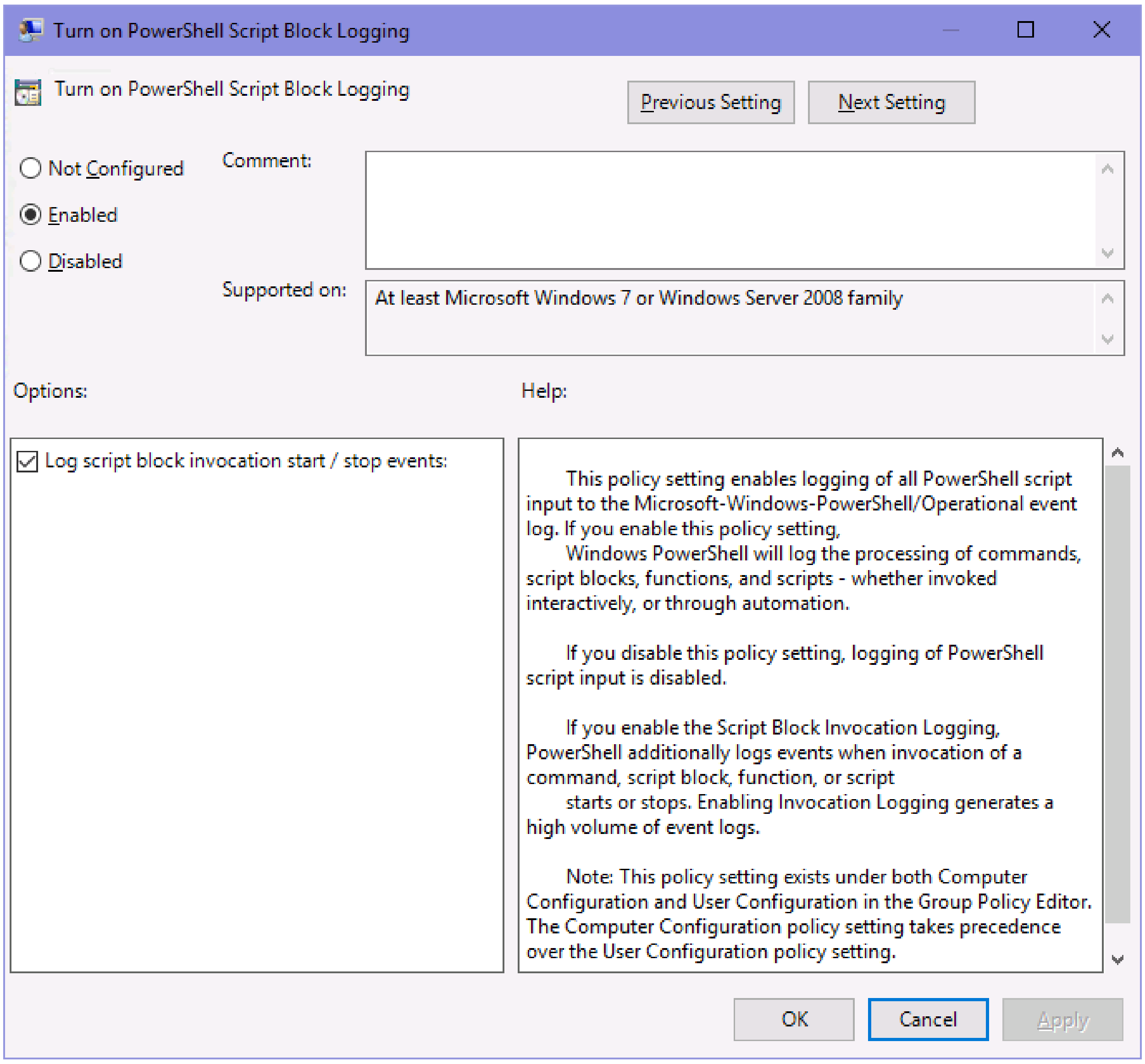
Show Contents

Module Names

* = all modules

	Value
▶	*
*	

2) Enable **Turn on PowerShell Script Block Logging** (you can also select *Log script block invocation start/stop events* checkbox)


Turn on PowerShell Script Block Logging
Turn on PowerShell Script Block Logging
Previous Setting
Next Setting
Comment:
Supported on:
Options:
Help:
OK
Cancel
Apply

Not Configured [✓ Enabled] Disabled

Supported on: At least Microsoft Windows 7 or Windows Server 2008 family

☑ Log script block invocation start / stop events:

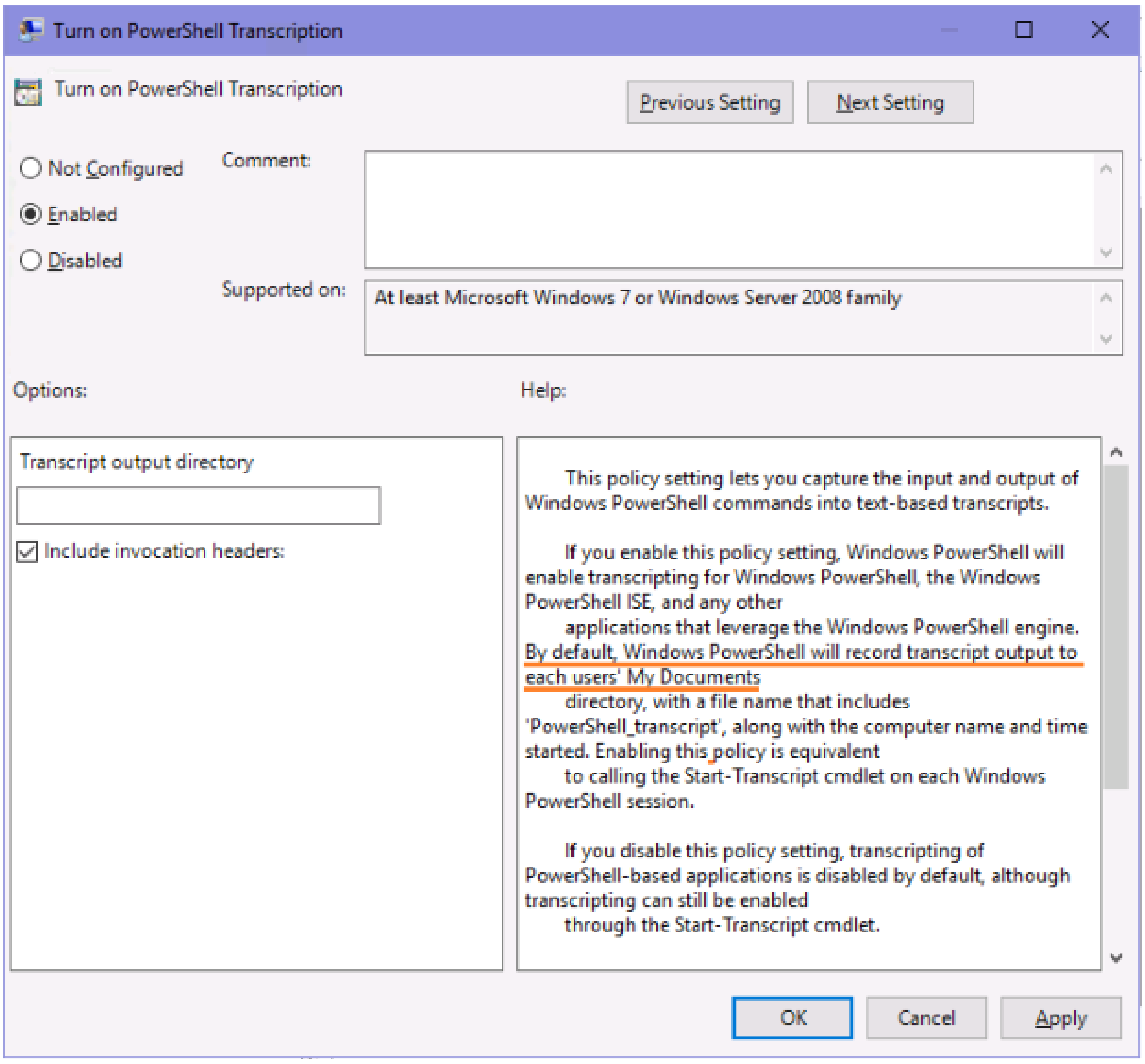
This policy setting enables logging of all PowerShell script input to the Microsoft-Windows-PowerShell/Operational event log. If you enable this policy setting,
Windows PowerShell will log the processing of commands, script blocks, functions, and scripts - whether invoked interactively, or through automation.

If you disable this policy setting, logging of PowerShell script input is disabled.

If you enable the Script Block Invocation Logging, PowerShell additionally logs events when invocation of a command, script block, function, or script
starts or stops. Enabling Invocation Logging generates a high volume of event logs.

Note: This policy setting exists under both Computer Configuration and User Configuration in the Group Policy Editor. The Computer Configuration policy setting takes precedence over the User Configuration policy setting.

There's also one more setting that can be enabled – **Turn on PowerShell Transcription** – this setting when enabled will log all PS-based commands into the specified directory (please note that we can define a separate transcript folder for each endpoint – I'll show it a bit later).


Turn on PowerShell Transcription
Turn on PowerShell Transcription
Previous Setting
Next Setting
OK
Cancel
Apply

Not Configured
[✓ Enabled]
Disabled

Comment:

Supported on: At least Microsoft Windows 7 or Windows Server 2008 family

Options:

Transcript output directory

[✓] Include invocation headers:

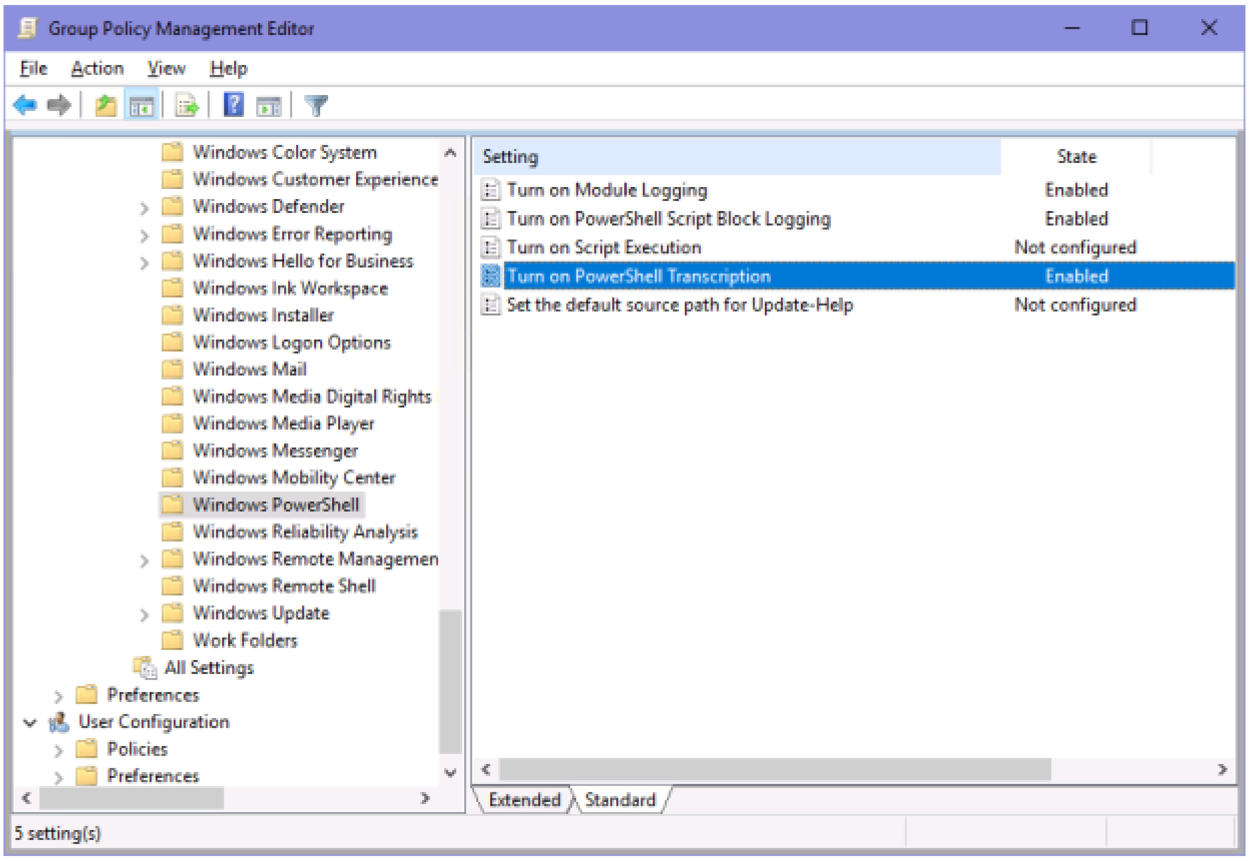
Help:

This policy setting lets you capture the input and output of Windows PowerShell commands into text-based transcripts.

If you enable this policy setting, Windows PowerShell will enable transcripting for Windows PowerShell, the Windows PowerShell ISE, and any other applications that leverage the Windows PowerShell engine. By default, Windows PowerShell will record transcript output to each users' My Documents directory, with a file name that includes 'PowerShell_transcript', along with the computer name and time started. Enabling this policy is equivalent to calling the Start-Transcript cmdlet on each Windows PowerShell session.

If you disable this policy setting, transcripting of PowerShell-based applications is disabled by default, although transcripting can still be enabled through the Start-Transcript cmdlet.

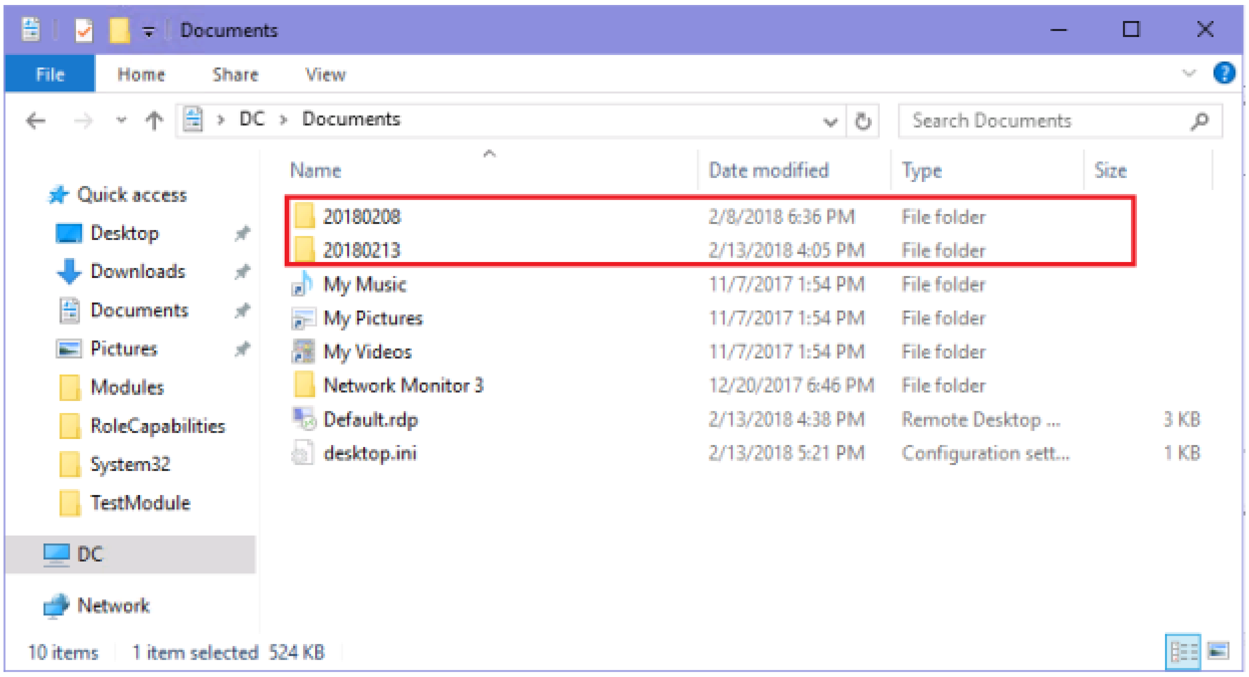
In case this policy setting is enabled the complete list of the PowerShell GPO settings will look as follows:


Group Policy Management Editor

Setting	State
Turn on Module Logging	Enabled
Turn on PowerShell Script Block Logging	Enabled
Turn on Script Execution	Not configured
Turn on PowerShell Transcription	Enabled
Set the default source path for Update-Help	Not configured

5 setting(s)

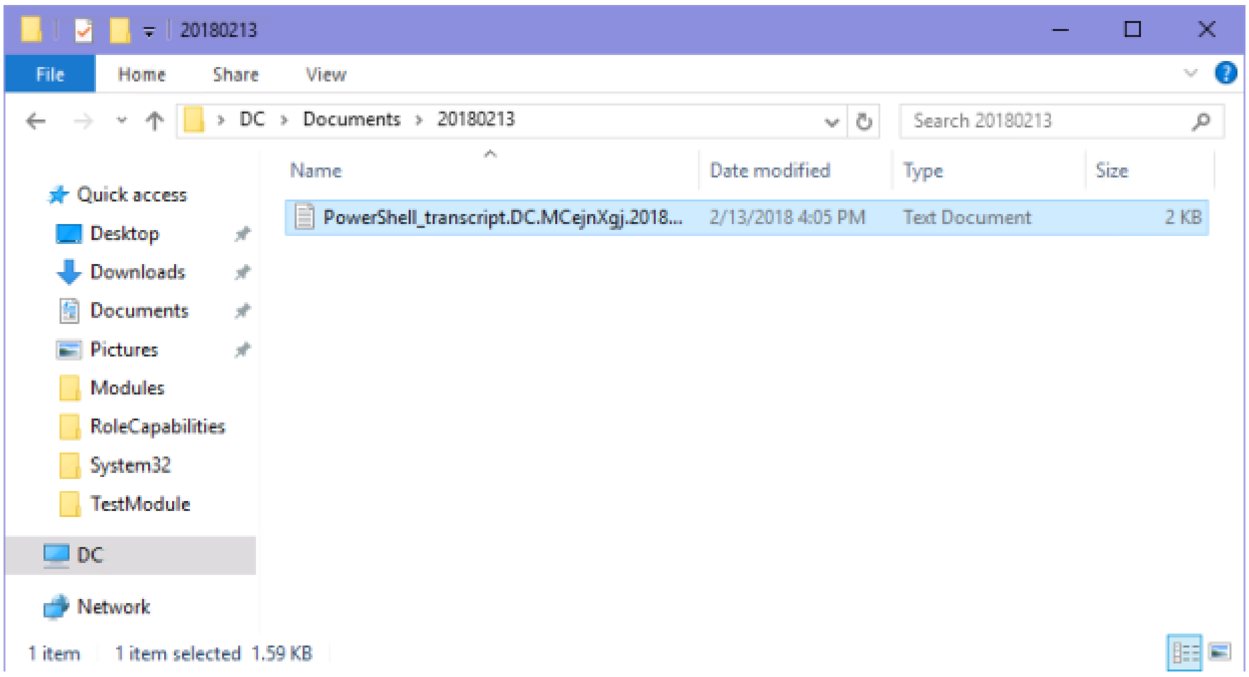
Here's the example of the transcripts being logged into the default user's *Document* folder:

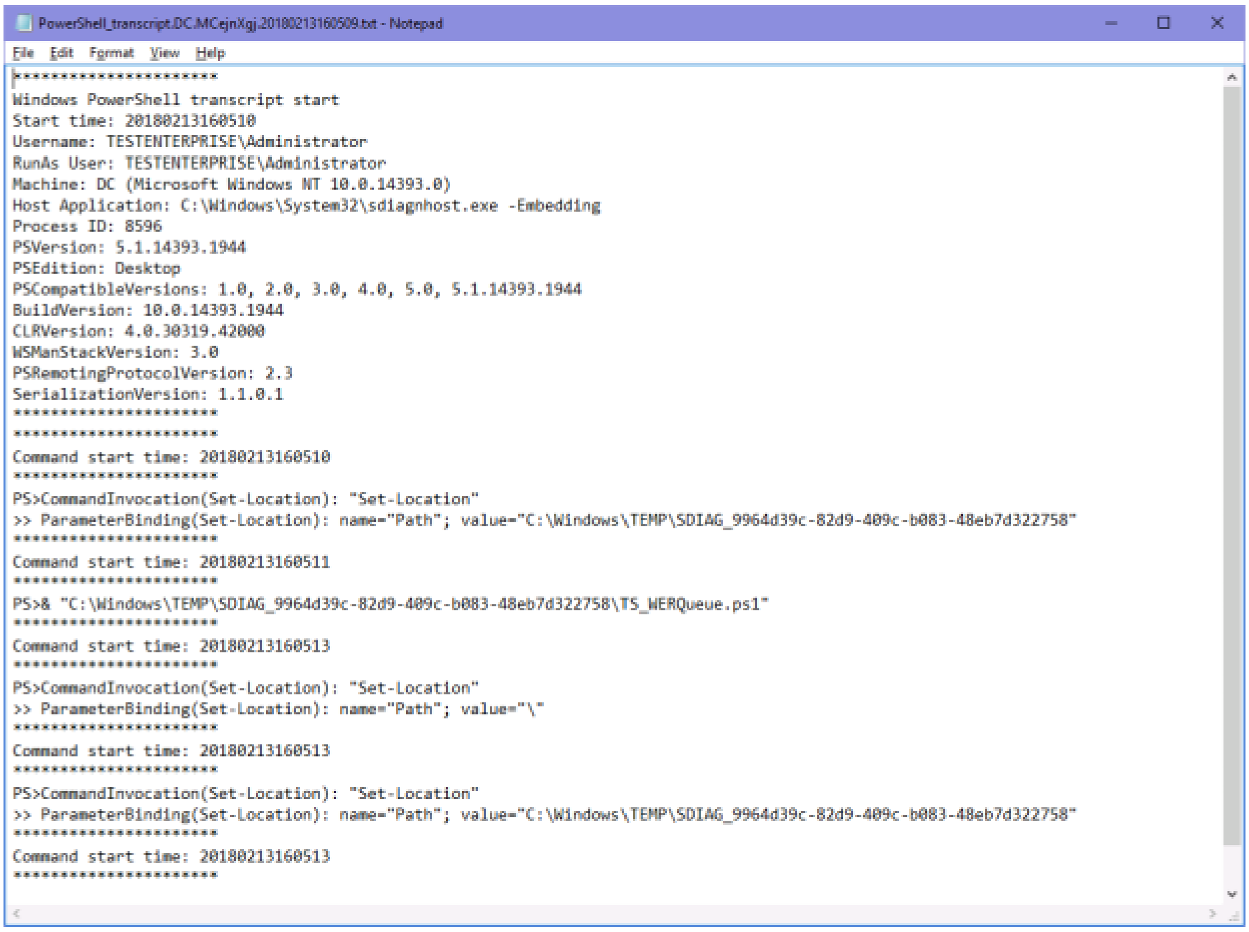

Documents

DC › Documents

Name	Date modified	Type	Size
20180208	2/8/2018 6:36 PM	File folder	
20180213	2/13/2018 4:05 PM	File folder	
My Music	11/7/2017 1:54 PM	File folder	
My Pictures	11/7/2017 1:54 PM	File folder	
My Videos	11/7/2017 1:54 PM	File folder	
Network Monitor 3	12/20/2017 6:46 PM	File folder	
Default.rdp	2/13/2018 4:38 PM	Remote Desktop ...	3 KB
desktop.ini	2/13/2018 5:21 PM	Configuration sett...	1 KB

10 items 1 item selected 524 KB




PowerShell_transcript.DC.MCejnXgj.20180213160509.txt - Notepad
File
Edit
Format
View
Help

```
**********************
Windows PowerShell transcript start
Start time: 20180213160510
Username: TESTENTERPRISE\Administrator
RunAs User: TESTENTERPRISE\Administrator
Machine: DC (Microsoft Windows NT 10.0.14393.0)
Host Application: C:\Windows\System32\sdiagnhost.exe -Embedding
Process ID: 8596
PSVersion: 5.1.14393.1944
PSEdition: Desktop
PSCompatibleVersions: 1.0, 2.0, 3.0, 4.0, 5.0, 5.1.14393.1944
BuildVersion: 10.0.14393.1944
CLRVersion: 4.0.30319.42000
WSManStackVersion: 3.0
PSRemotingProtocolVersion: 2.3
SerializationVersion: 1.1.0.1
**********************
**********************
Command start time: 20180213160510
**********************
PS>CommandInvocation(Set-Location): "Set-Location"
>> ParameterBinding(Set-Location): name="Path"; value="C:\Windows\TEMP\SDIAG_9964d39c-82d9-409c-b083-48eb7d322758"
**********************
Command start time: 20180213160511
**********************
PS>& "C:\Windows\TEMP\SDIAG_9964d39c-82d9-409c-b083-48eb7d322758\TS_WERQueue.ps1"
**********************
Command start time: 20180213160513
**********************
PS>CommandInvocation(Set-Location): "Set-Location"
>> ParameterBinding(Set-Location): name="Path"; value="\"
**********************
Command start time: 20180213160513
**********************
PS>CommandInvocation(Set-Location): "Set-Location"
>> ParameterBinding(Set-Location): name="Path"; value="C:\Windows\TEMP\SDIAG_9964d39c-82d9-409c-b083-48eb7d322758"
**********************
Command start time: 20180213160513
**********************
```

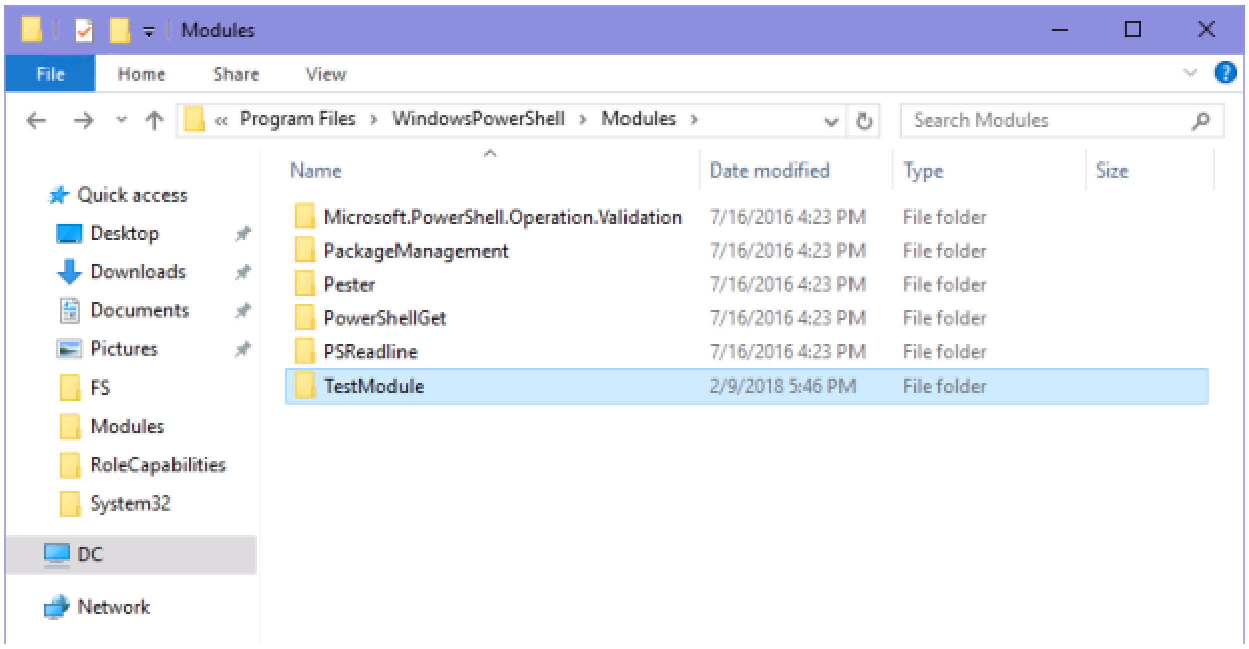
Once the GPO is created we can proceed to the first step in deploying JEA:

Step 1: Creating a new PowerShell module

(All the steps will be performed on my domain controller – DC. This is the server TO WHICH a user named User1 will be given access).

All JEA functionality “contains” in the user-defined PowerShell modules and I prefer creating the new modules beforehand so I can place all needed files in the new module’s folder later on. By default PS looks for its modules in C:\Program Files\WindowsPowerShell\Modules folder so to create a new module I’ll do the following:

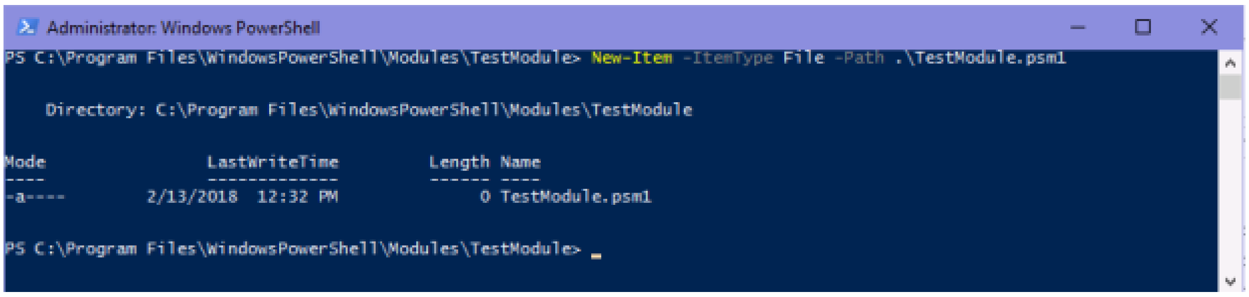
a) Create a folder for the module:



Name	Date modified	Type	Size
Microsoft.PowerShell.Operation.Validation	7/16/2016 4:23 PM	File folder	
PackageManagement	7/16/2016 4:23 PM	File folder	
Pester	7/16/2016 4:23 PM	File folder	
PowerShellGet	7/16/2016 4:23 PM	File folder	
PSReadline	7/16/2016 4:23 PM	File folder	
TestModule	2/9/2018 5:46 PM	File folder	

b) Create a new empty module (at least one file in the module folder must have the same name as the folder – in this case TestModule) – in fact the module is just a **.psm1** file:

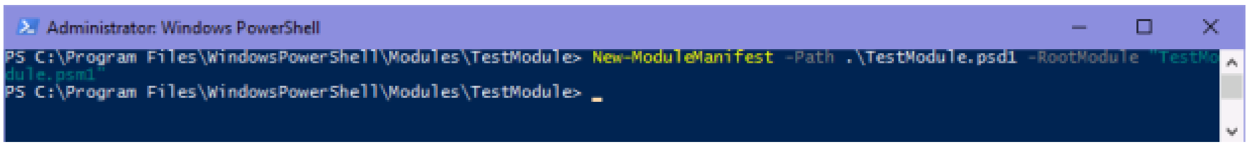
New-Item -ItemType File -Path .\TestModule.psm1



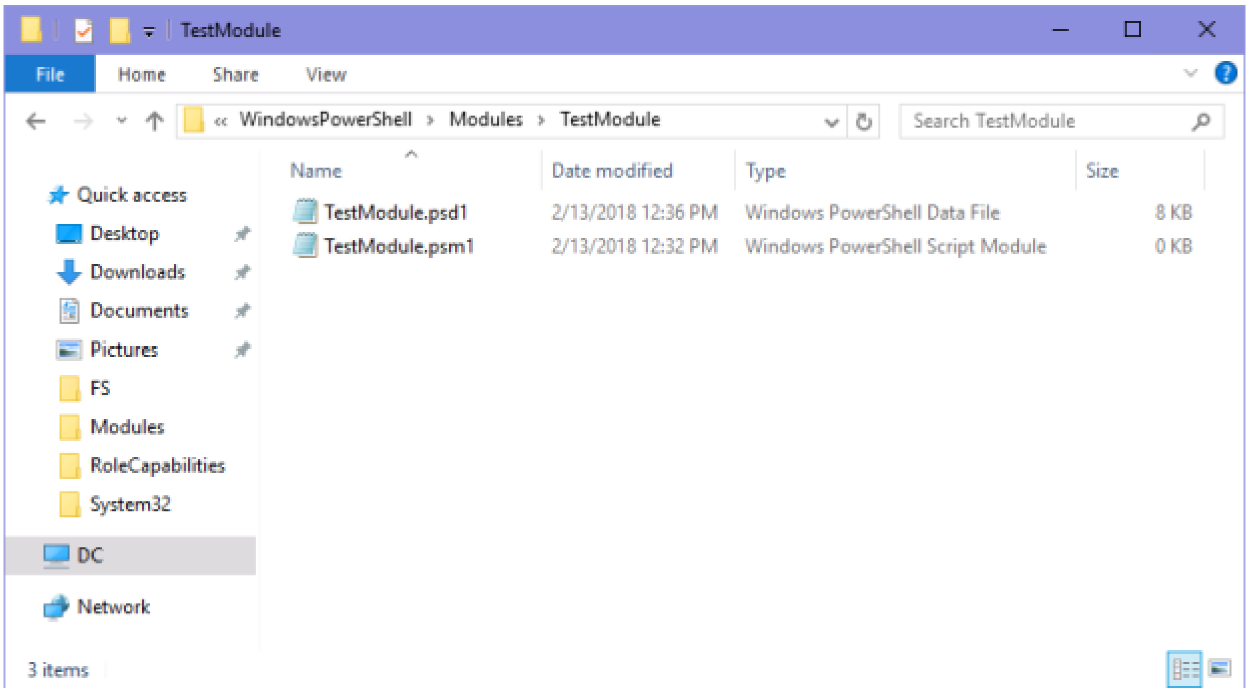
```
PS C:\Program Files\WindowsPowerShell\Modules\TestModule> New-Item -ItemType File -Path .\TestModule.psm1


    Directory: C:\Program Files\WindowsPowerShell\Modules\TestModule


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
-a----        2/13/2018  12:32 PM              0 TestModule.psm1


PS C:\Program Files\WindowsPowerShell\Modules\TestModule> _
```

c) Create a new manifest file (**.psm1** file)

New-ModuleManifest -Path .\TestModule.psd1 -RootModule “TestModule.psm1”



```
PS C:\Program Files\WindowsPowerShell\Modules\TestModule> New-ModuleManifest -Path .\TestModule.psd1 -RootModule "TestMo
dule.psm1"
PS C:\Program Files\WindowsPowerShell\Modules\TestModule> _
```

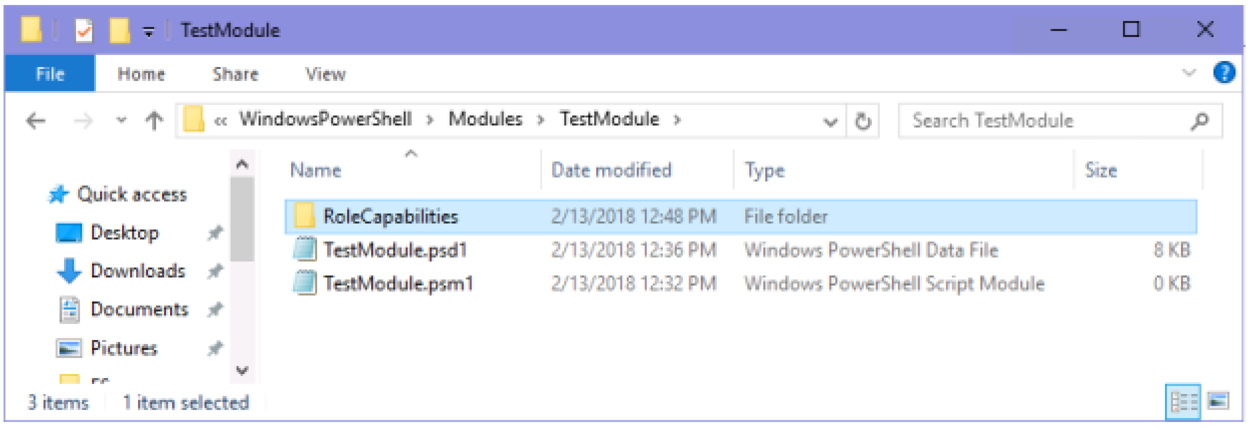


Name	Date modified	Type	Size
TestModule.psd1	2/13/2018 12:36 PM	Windows PowerShell Data File	8 KB
TestModule.psm1	2/13/2018 12:32 PM	Windows PowerShell Script Module	0 KB

Step 2: Creating role capabilities file (.psrc file)

At least one **.psrc** file called role capabilities file must be created which defines what user or administrator can do in a JEA session. This file contains all cmdlets and external programs (it can contain providers and functions as well) which will be permitted to use in the JEA session. For this test I'll create the file which permits a user only to see the list of running process by means of Get-Process cmdlet, the list of services by means of Get-Service cmdlet and to run the single external program – nslookup. It is worth noting that the cmdlets and external programs typicaly run under virtual administrator account which by default is a member of the local *Administrators* or *Domain Admins* groups. The **.psrc** file must be placed in the **RoleCapabilities** subfolder of the module's folder:

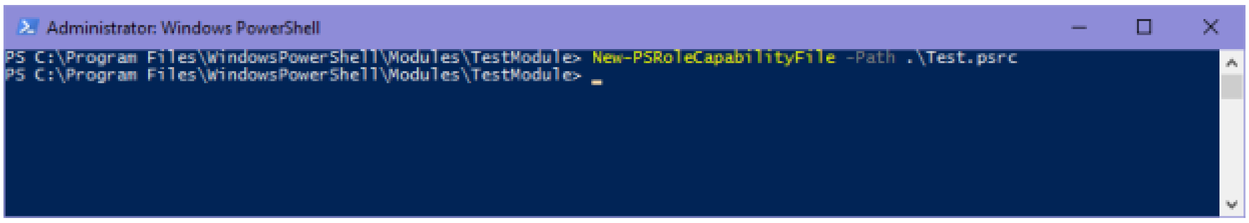
a) I create RoleCapabilities folder in the TestModule folder:



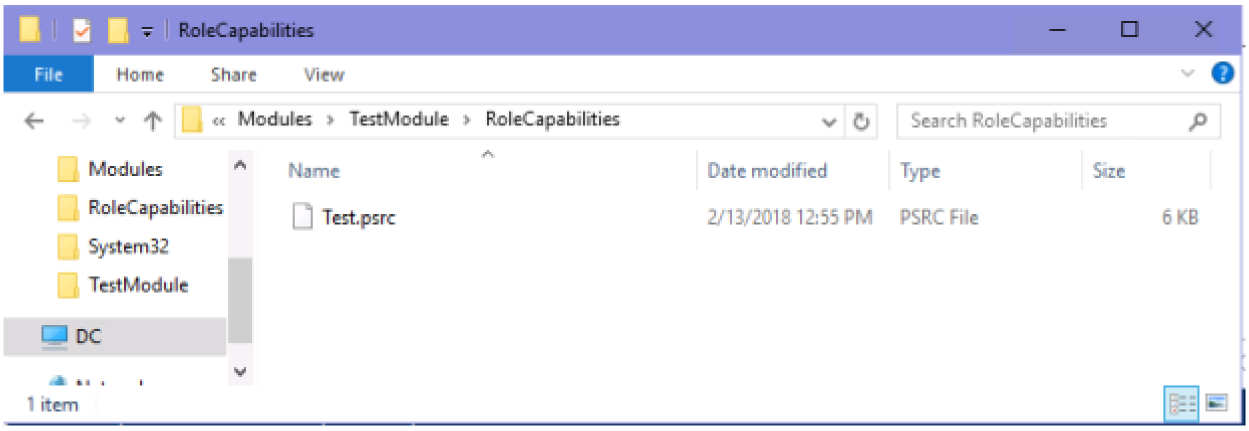
Name	Date modified	Type	Size
RoleCapabilities	2/13/2018 12:48 PM	File folder	
TestModule.psd1	2/13/2018 12:36 PM	Windows PowerShell Data File	8 KB
TestModule.psm1	2/13/2018 12:32 PM	Windows PowerShell Script Module	0 KB

b) and the role capabilities file in it:

New-PSRoleCapabilityFile -Path .\RoleCapabilities\Test.psrc

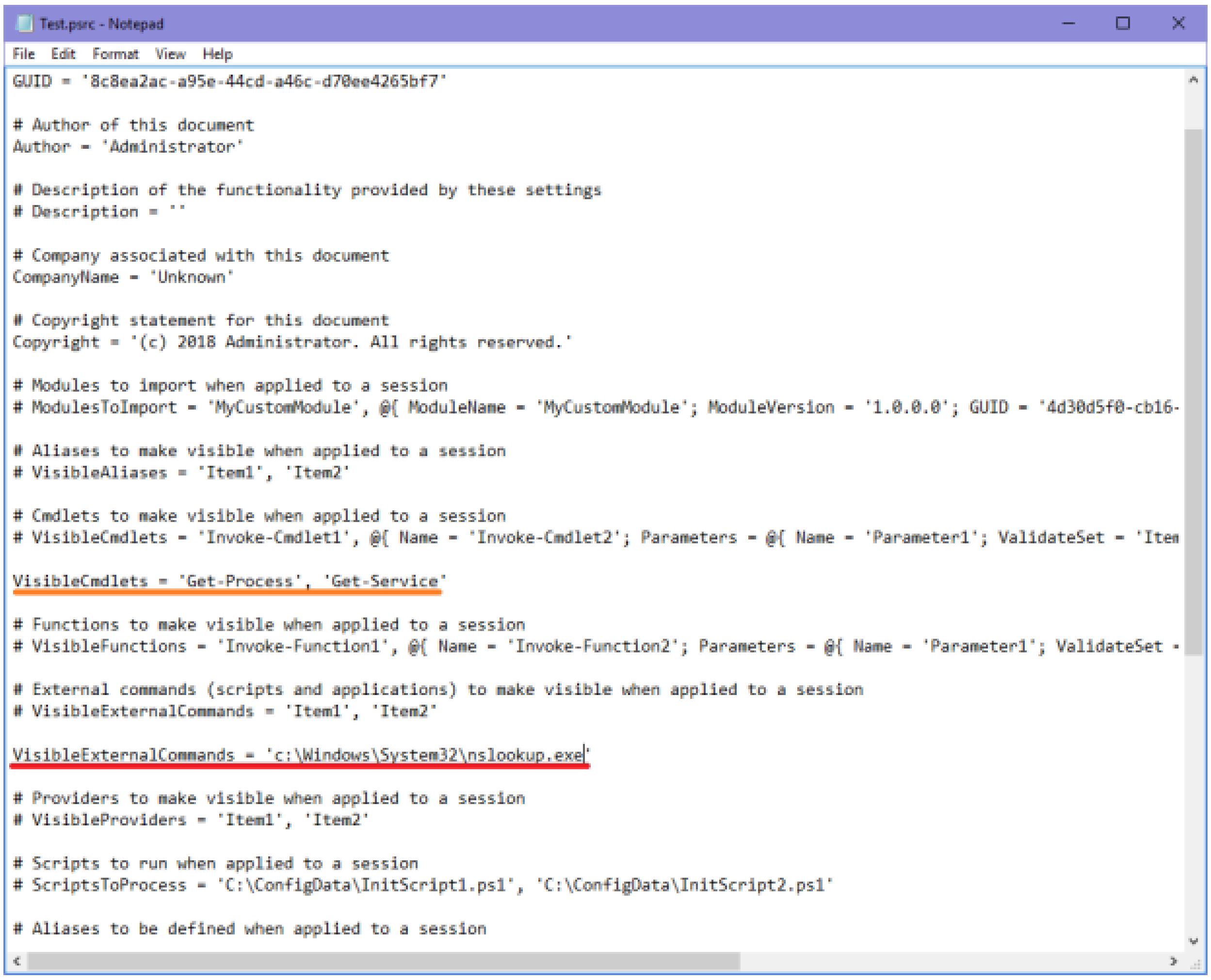


```
PS C:\Program Files\WindowsPowerShell\Modules\TestModule> New-PSRoleCapabilityFile -Path .\Test.psrc
PS C:\Program Files\WindowsPowerShell\Modules\TestModule> _
```



```
Name          Date modified         Type         Size
Test.psrc     2/13/2018 12:55 PM    PSRC File    6 KB
```

Now I can edit the **Test.psrc** with Notepad to permit only a few cmdlets/external commands:



```
GUID = '8c8ea2ac-a95e-44cd-a46c-d70ee4265bf7'

# Author of this document
Author = 'Administrator'

# Description of the functionality provided by these settings
# Description = ''

# Company associated with this document
CompanyName = 'Unknown'

# Copyright statement for this document
Copyright = '(c) 2018 Administrator. All rights reserved.'

# Modules to import when applied to a session
# ModulesToImport = 'MyCustomModule', @{ ModuleName = 'MyCustomModule'; ModuleVersion = '1.0.0.0'; GUID = '4d30d5f0-cb16-

# Aliases to make visible when applied to a session
# VisibleAliases = 'Item1', 'Item2'

# Cmdlets to make visible when applied to a session
# VisibleCmdlets = 'Invoke-Cmdlet1', @{ Name = 'Invoke-Cmdlet2'; Parameters = @{ Name = 'Parameter1'; ValidateSet = 'Item

VisibleCmdlets = 'Get-Process', 'Get-Service'

# Functions to make visible when applied to a session
# VisibleFunctions = 'Invoke-Function1', @{ Name = 'Invoke-Function2'; Parameters = @{ Name = 'Parameter1'; ValidateSet =

# External commands (scripts and applications) to make visible when applied to a session
# VisibleExternalCommands = 'Item1', 'Item2'

VisibleExternalCommands = 'c:\Windows\System32\nslookup.exe'

# Providers to make visible when applied to a session
# VisibleProviders = 'Item1', 'Item2'

# Scripts to run when applied to a session
# ScriptsToProcess = 'C:\ConfigData\InitScript1.ps1', 'C:\ConfigData\InitScript2.ps1'

# Aliases to be defined when applied to a session
```

Step 3: Creating session configuration file (.pssc file)

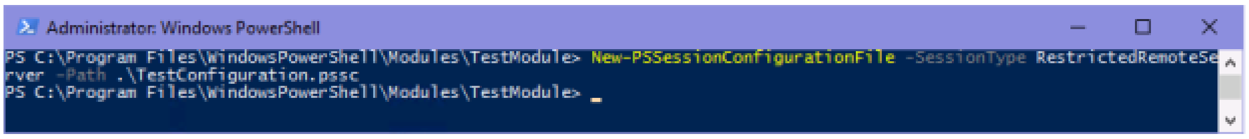
To define who will have access to the new endpoint and under which account the new ps session will run, the session configuration file must be created (I'll create it in the module's folder):

New-PSSessionConfigurationFile -SessionType RestrictedRemoteServer -Path .\TestConfiguration.pssc

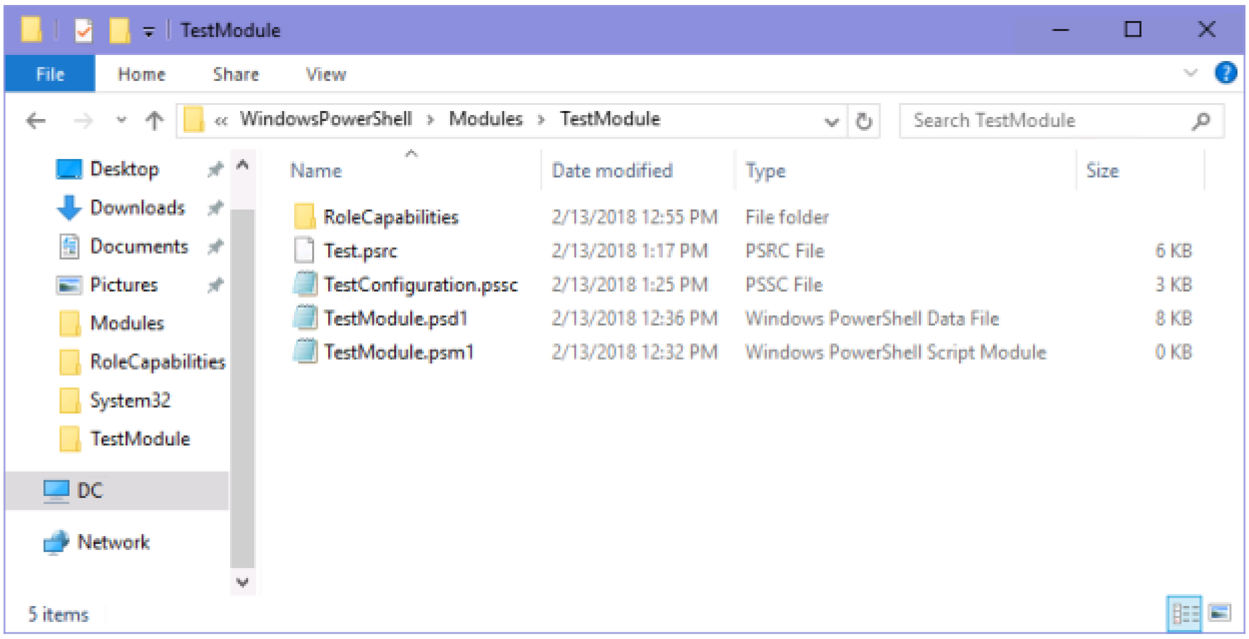
Sessions configured with -*SessionType* RestrictedRemoteServer field will operate in the **Restricted Language** mode which allows by default only the following commands:

- Clear-Host (cls, clear)
- Exit-PSSession (exsn, exit)
- Get-Command (gcm)
- Get-FormatData
- Get-Help
- Measure-Object (measure)
- Out-Default
- Select-Object (select)

More detailed explanation on this mode can be found here – About Language Modes.

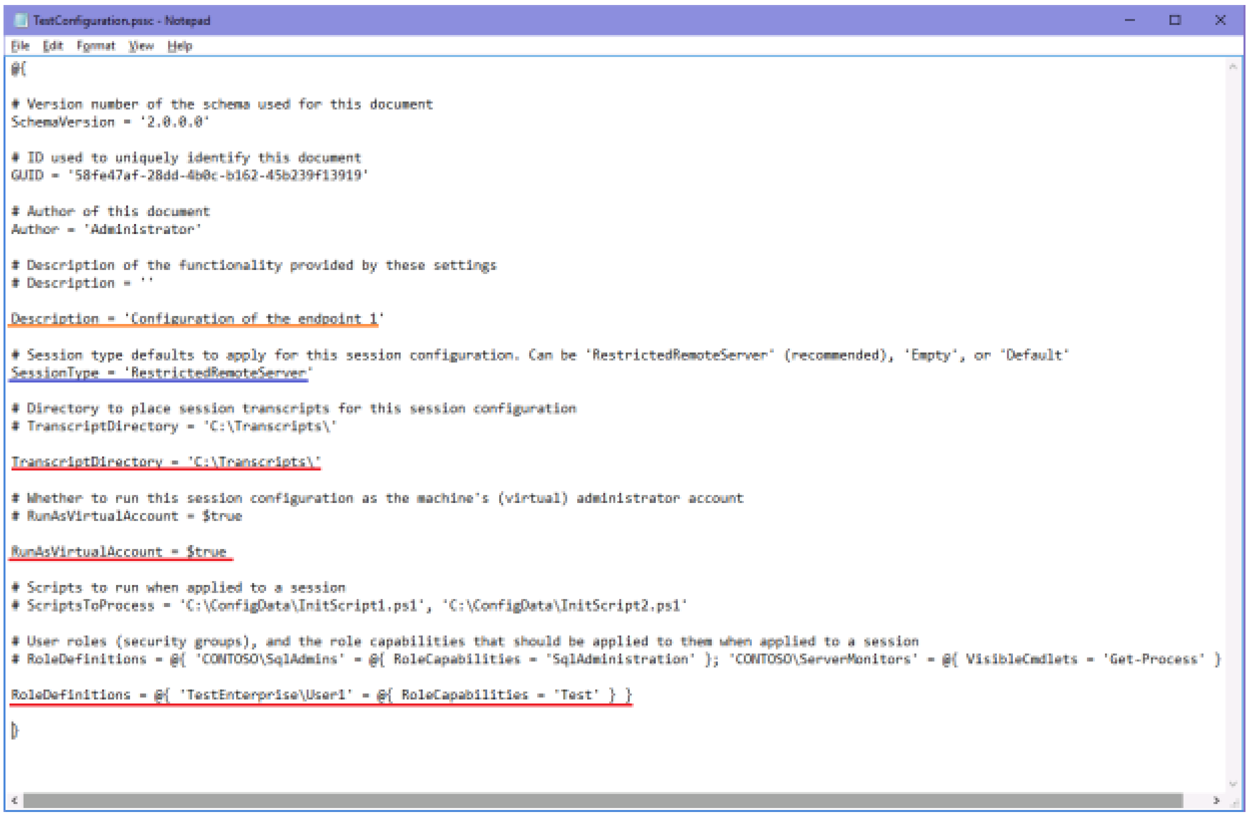


```
PS C:\Program Files\WindowsPowerShell\Modules\TestModule> New-PSSessionConfigurationFile -SessionType RestrictedRemoteSe
rver -Path .\TestConfiguration.pssc
PS C:\Program Files\WindowsPowerShell\Modules\TestModule> _
```



Name	Date modified	Type	Size
RoleCapabilities	2/13/2018 12:55 PM	File folder	
Test.psrc	2/13/2018 1:17 PM	PSRC File	6 KB
TestConfiguration.pssc	2/13/2018 1:25 PM	PSSC File	3 KB
TestModule.psd1	2/13/2018 12:36 PM	Windows PowerShell Data File	8 KB
TestModule.psm1	2/13/2018 12:32 PM	Windows PowerShell Script Module	0 KB

Like **.psd1** and **.psm1** files, the **.pssc** files can be edited in Notepad – I'll configure the following sections:


TestConfiguration.pssc - Notepad

```
@{

# Version number of the schema used for this document
SchemaVersion = '2.0.0.0'

# ID used to uniquely identify this document
GUID = '58fe47af-28dd-4b0c-b162-45b239f13919'

# Author of this document
Author = 'Administrator'

# Description of the functionality provided by these settings
# Description = ''

Description = 'Configuration of the endpoint 1'

# Session type defaults to apply for this session configuration. Can be 'RestrictedRemoteServer' (recommended), 'Empty', or 'Default'
SessionType = 'RestrictedRemoteServer'

# Directory to place session transcripts for this session configuration
# TranscriptDirectory = 'C:\Transcripts\'

TranscriptDirectory = 'C:\Transcripts\'

# Whether to run this session configuration as the machine's (virtual) administrator account
# RunAsVirtualAccount = $true

RunAsVirtualAccount = $true

# Scripts to run when applied to a session
# ScriptsToProcess = 'C:\ConfigData\InitScript1.ps1', 'C:\ConfigData\InitScript2.ps1'

# User roles (security groups), and the role capabilities that should be applied to them when applied to a session
# RoleDefinitions = @{ 'CONTOSO\SqlAdmins' = @{ RoleCapabilities = 'SqlAdministration' }; 'CONTOSO\ServerMonitors' = @{ VisibleCmdlets = 'Get-Process' }

RoleDefinitions = @{ 'TestEnterprise\User1' = @{ RoleCapabilities = 'Test' } }

}
```

TranscriptDirectory – this line enables logging of the commands run in the session (this is the *endpoint-specific* transcript directory as opposed to the default transcript directory defined in the PS GPO above) .

RunAsVirtualAccount – if set to true runs the session under virtual administrative account (member of either local *Administrator*'s or *Domain Admins*' group).

Other options:

1) if you don't want to run the new session under administrative account you can specifiy any group/groups of wich this virtual account should be the member of:

RunAsVirtualAccount = $true
RunAsVirtualAccountGroups = 'Network Configuration Operators'

2) for accessing network resourses from the new session you can use group managed service account:
GroupManagedServiceAccount = 'TestDomain\TestGSA'

RoleDefinitions – defines who (in my test – TestEnterprise\User1) will have access to which endpoint (the name of the endpoint is the RoleCapabilities file name without extension).

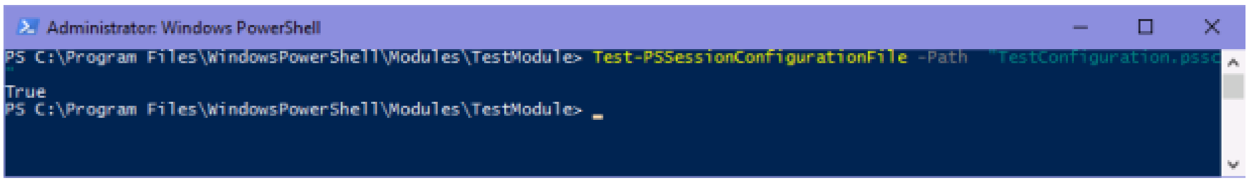
SessionType is already set to RestrictedRemoteServer

Description – the optional field.

Before we proceed to registering JEA endpoint it would be pertinent to test the session configuration file using the Test-PSSessionConfigurationFile cmdlet:

Test-PSSessionConfigurationFile -Path "TestConfiguration.pssc"

Should this test reveal any errors you can edit the file once again in Notepad.



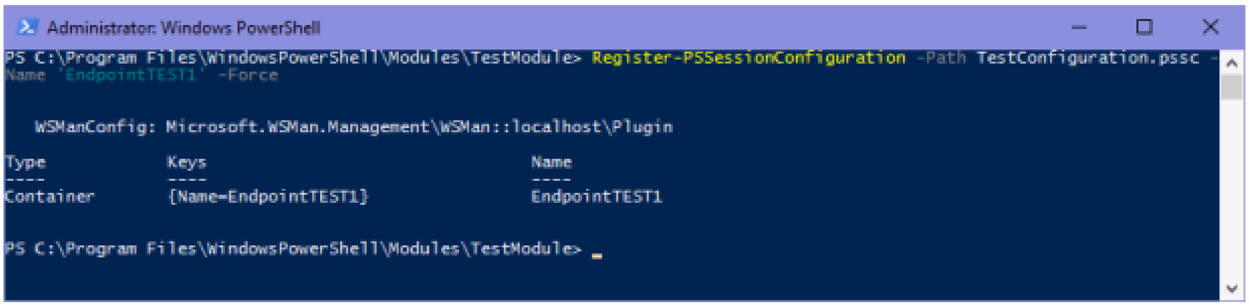
```
PS C:\Program Files\WindowsPowerShell\Modules\TestModule> Test-PSSessionConfigurationFile -Path  "TestConfiguration.pssc
True
PS C:\Program Files\WindowsPowerShell\Modules\TestModule> _
```

True means the file has the correct syntax.

The last step: Registering the endpoint

The following command will register the new endpoint on the system:

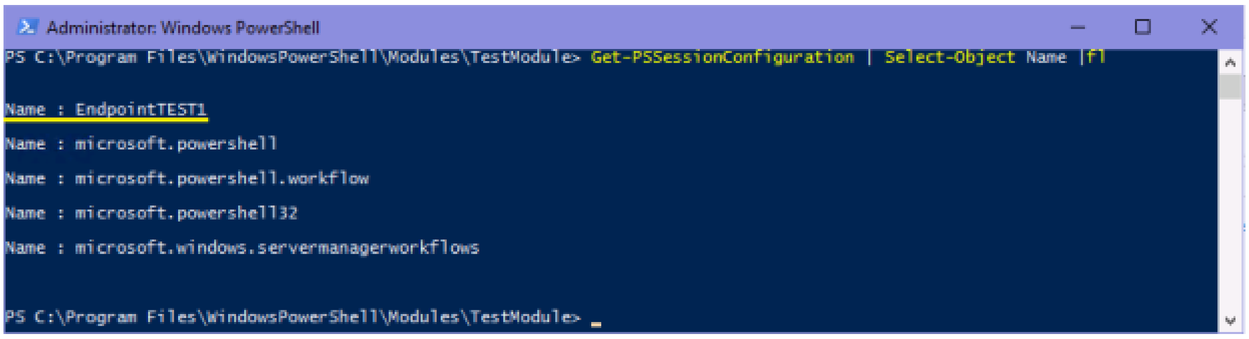
Register-PSSessionConfiguration -Path TestConfiguration.pssc -Name 'EndpointTEST1' -Force



```
PS C:\Program Files\WindowsPowerShell\Modules\TestModule> Register-PSSessionConfiguration -Path TestConfiguration.pssc -
Name 'EndpointTEST1' -Force


   WSManConfig: Microsoft.WSMan.Management\WSMan::localhost\Plugin

Type            Keys                                Name
----            ----                                ----
Container       {Name=EndpointTEST1}                EndpointTEST1


PS C:\Program Files\WindowsPowerShell\Modules\TestModule> _
```

You can see the newly-created endpoint by issuing this command:

Get-PSSessionConfiguration | Select-Object Name



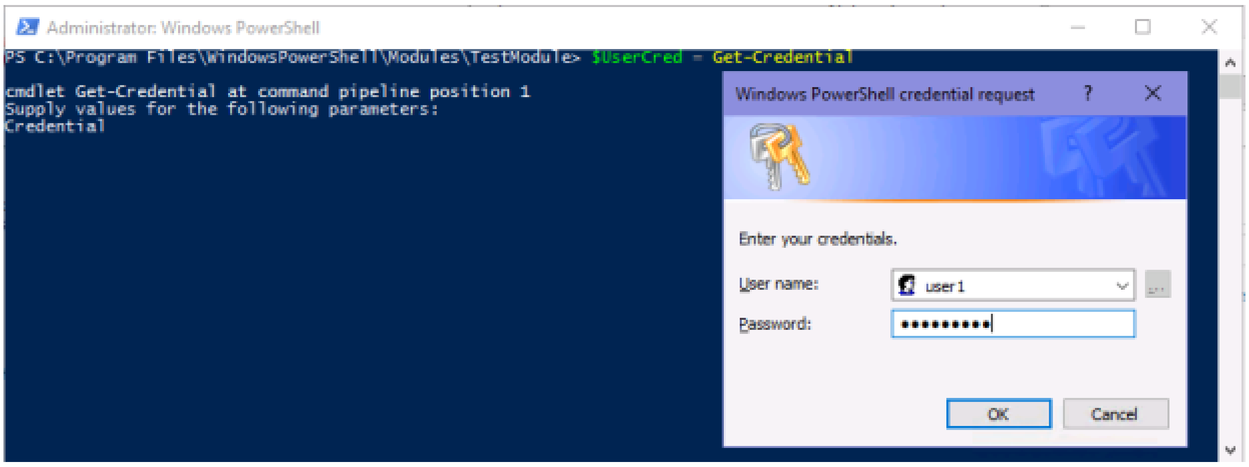
```
PS C:\Program Files\WindowsPowerShell\Modules\TestModule> Get-PSSessionConfiguration | Select-Object Name |fl

Name : EndpointTEST1

Name : microsoft.powershell

Name : microsoft.powershell.workflow

Name : microsoft.powershell32

Name : microsoft.windows.servermanagerworkflows


PS C:\Program Files\WindowsPowerShell\Modules\TestModule> _
```

Now it's time to test the new endpoint: first of all I'd like to make sure the endpoint works as expected by connecting to the local host using User1′ domain credentials.

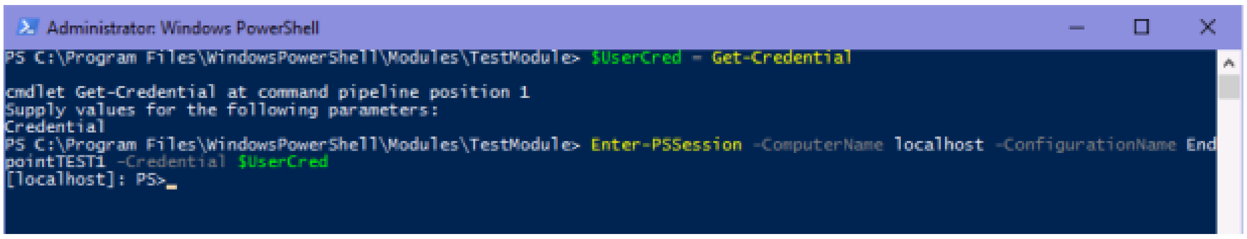
I) Connecting to the local host

$UserCred = Get-Credential

Enter-PSSession -ComputerName localhost -ConfigurationName EndpointTEST1 -Credential $UserCred



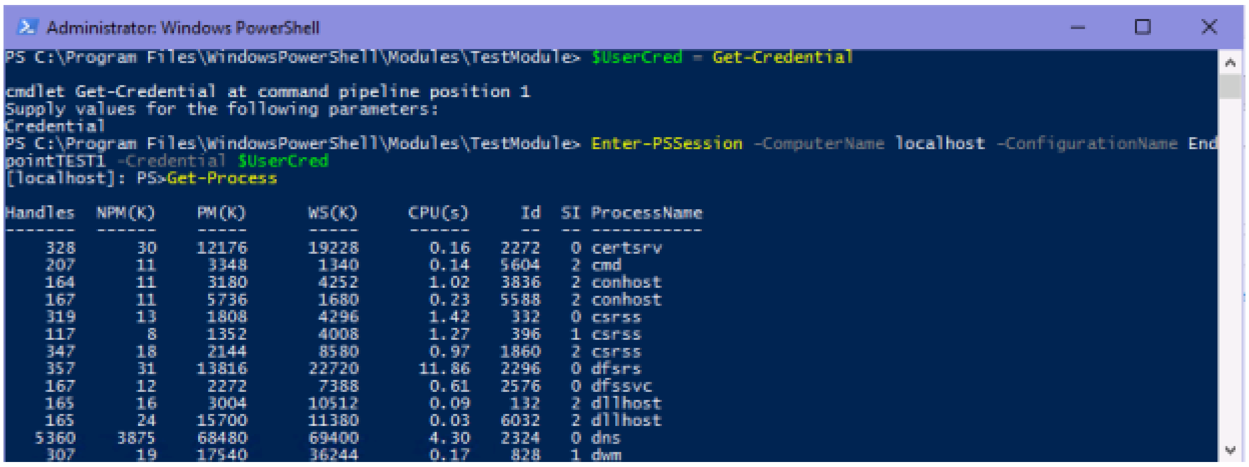
```
Administrator: Windows PowerShell
PS C:\Program Files\WindowsPowerShell\Modules\TestModule> $UserCred = Get-Credential

cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:
Credential
```



```
Administrator: Windows PowerShell
PS C:\Program Files\WindowsPowerShell\Modules\TestModule> $UserCred = Get-Credential

cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:
Credential
PS C:\Program Files\WindowsPowerShell\Modules\TestModule> Enter-PSSession -ComputerName localhost -ConfigurationName End
pointTEST1 -Credential $UserCred
[localhost]: PS>_
```

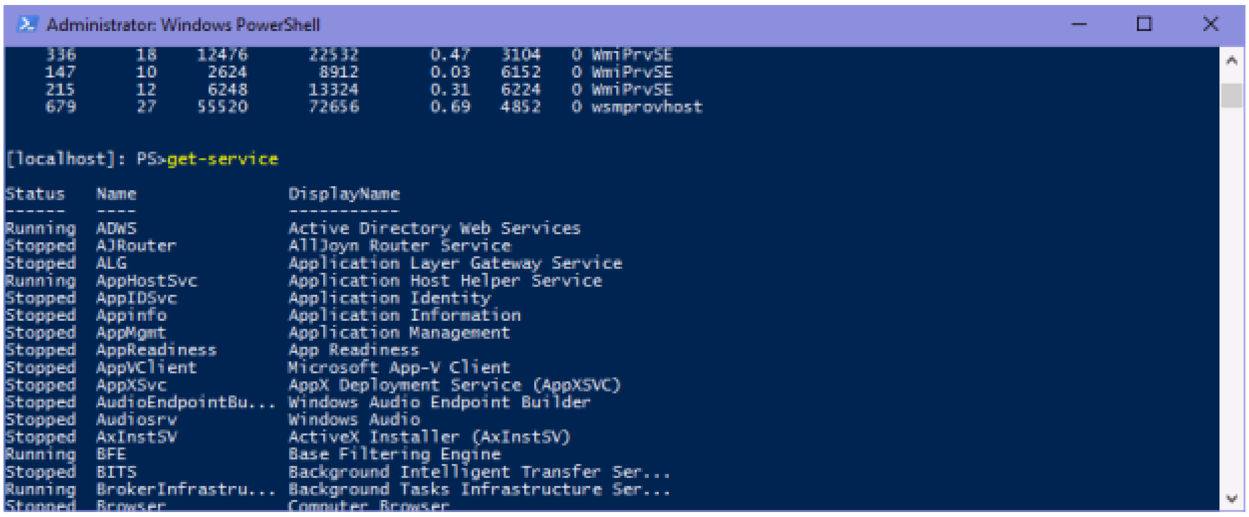
From now on, all cmdlets will be run under virtual administrative account.

Get-Process (It should succeed as it’s defined in the RoleCapabilities file)



```
Administrator: Windows PowerShell
PS C:\Program Files\WindowsPowerShell\Modules\TestModule> $UserCred = Get-Credential

cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:
Credential
PS C:\Program Files\WindowsPowerShell\Modules\TestModule> Enter-PSSession -ComputerName localhost -ConfigurationName End
pointTEST1 -Credential $UserCred
[localhost]: PS>Get-Process

Handles  NPM(K)    PM(K)      WS(K)     CPU(s)     Id  SI ProcessName
-------  ------    -----      -----     ------     --  -- -----------
    328      30    12176      19228       0.16   2272   0 certsrv
    207      11     3348       1340       0.14   5604   2 cmd
    164      11     3180       4252       1.02   3836   2 conhost
    167      11     5736       1680       0.23   5588   2 conhost
    319      13     1808       4296       1.42    332   0 csrss
    117       8     1352       4008       1.27    396   1 csrss
    347      18     2144       8580       0.97   1860   2 csrss
    357      31    13816      22720      11.86   2296   0 dfsrs
    167      12     2272       7388       0.61   2576   0 dfssvc
    165      16     3004      10512       0.09    132   2 dllhost
    165      24    15700      11380       0.03   6032   2 dllhost
   5360    3875    68480      69400       4.30   2324   0 dns
    307      19    17540      36244       0.17    828   1 dwm
```

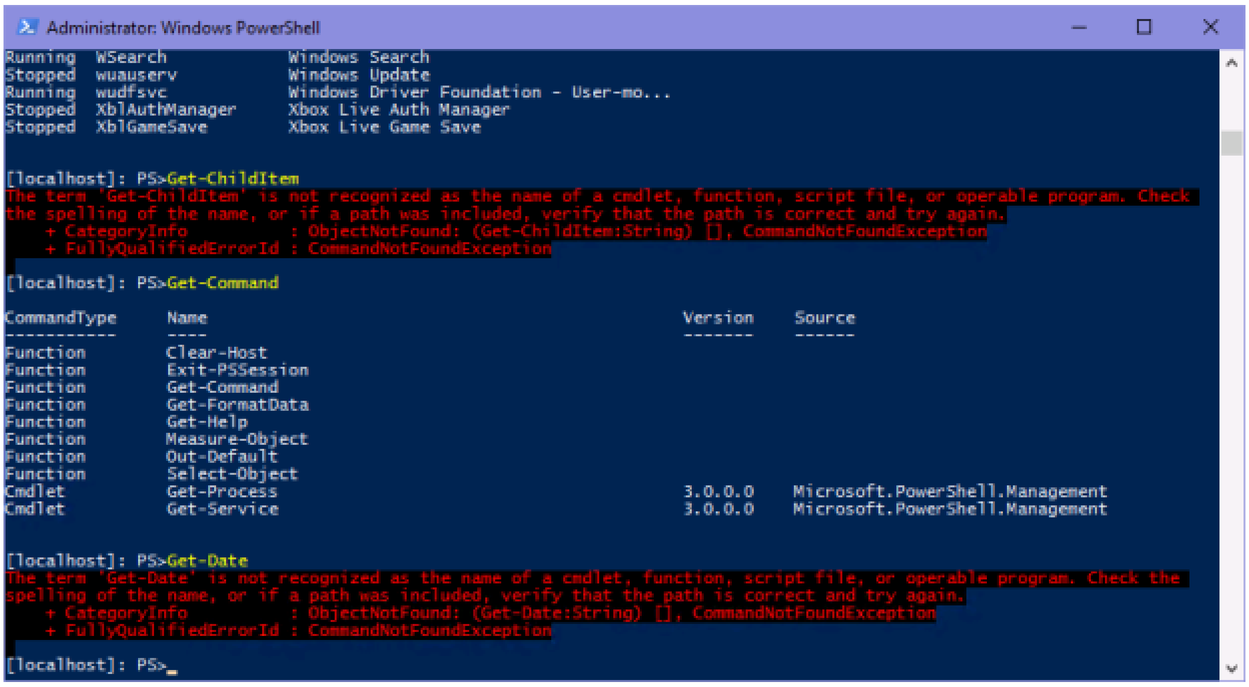
Get-Service (It should succeed as it’s defined in the RoleCapabilities file)



```
    336      18    12476      22532       0.47    3104   0 WmiPrvSE
    147      10     2624       8912       0.03    6152   0 WmiPrvSE
    215      12     6248      13324       0.31    6224   0 WmiPrvSE
    679      27    55520      72656       0.69    4852   0 wsmprovhost


[localhost]: PS>get-service

Status   Name               DisplayName
------   ----               -----------
Running  ADWS               Active Directory Web Services
Stopped  AJRouter           AllJoyn Router Service
Stopped  ALG                Application Layer Gateway Service
Running  AppHostSvc         Application Host Helper Service
Stopped  AppIDSvc           Application Identity
Stopped  Appinfo            Application Information
Stopped  AppMgmt            Application Management
Stopped  AppReadiness       App Readiness
Stopped  AppVClient         Microsoft App-V Client
Stopped  AppXSvc            AppX Deployment Service (AppXSVC)
Stopped  AudioEndpointBu... Windows Audio Endpoint Builder
Stopped  Audiosrv           Windows Audio
Stopped  AxInstSV           ActiveX Installer (AxInstSV)
Running  BFE                Base Filtering Engine
Stopped  BITS               Background Intelligent Transfer Ser...
Running  BrokerInfrastru... Background Tasks Infrastructure Ser...
Stopped  Browser            Computer Browser
```

Get-Childitem and Get-Date should NOT succeed because they are not listed in the RoleCapabilities file.

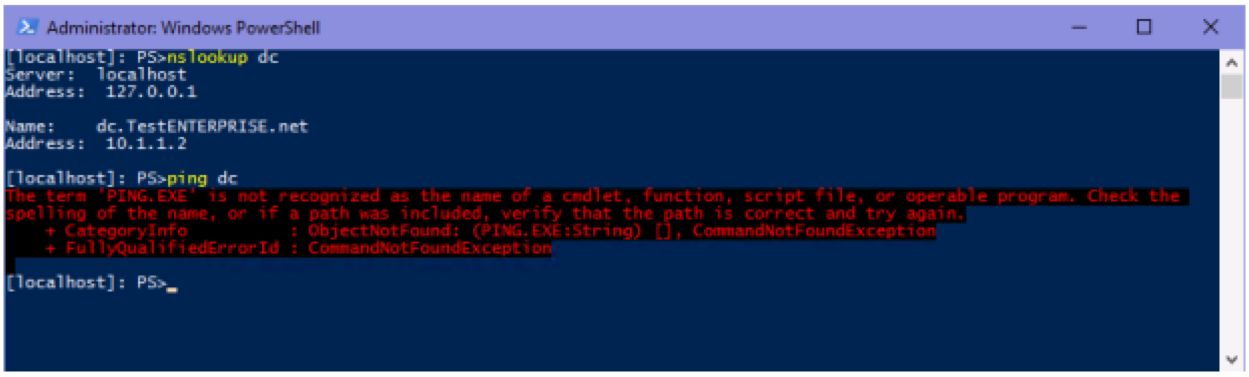
Get-Command will succeed as it’s one of the base cmdlets allowed in the restricted language mode:



```
Running  WSearch            Windows Search
Stopped  wuauserv           Windows Update
Running  wudfsvc            Windows Driver Foundation - User-mo...
Stopped  XblAuthManager     Xbox Live Auth Manager
Stopped  XblGameSave        Xbox Live Game Save


[localhost]: PS>Get-ChildItem
The term 'Get-ChildItem' is not recognized as the name of a cmdlet, function, script file, or operable program. Check
the spelling of the name, or if a path was included, verify that the path is correct and try again.
    + CategoryInfo          : ObjectNotFound: (Get-ChildItem:String) [], CommandNotFoundException
    + FullyQualifiedErrorId : CommandNotFoundException

[localhost]: PS>Get-Command

CommandType     Name                                               Version    Source
-----------     ----                                               -------    ------
Function        Clear-Host
Function        Exit-PSSession
Function        Get-Command
Function        Get-FormatData
Function        Get-Help
Function        Measure-Object
Function        Out-Default
Function        Select-Object
Cmdlet          Get-Process                                        3.0.0.0    Microsoft.PowerShell.Management
Cmdlet          Get-Service                                        3.0.0.0    Microsoft.PowerShell.Management


[localhost]: PS>Get-Date
The term 'Get-Date' is not recognized as the name of a cmdlet, function, script file, or operable program. Check the
spelling of the name, or if a path was included, verify that the path is correct and try again.
    + CategoryInfo          : ObjectNotFound: (Get-Date:String) [], CommandNotFoundException
    + FullyQualifiedErrorId : CommandNotFoundException

[localhost]: PS>_
```

External commands:

Nslookup is allowed while any others are not:



```
[localhost]: PS>nslookup dc
Server:  localhost
Address:  127.0.0.1

Name:    dc.TestENTERPRISE.net
Address:  10.1.1.2

[localhost]: PS>ping dc
The term 'PING.EXE' is not recognized as the name of a cmdlet, function, script file, or operable program. Check the
spelling of the name, or if a path was included, verify that the path is correct and try again.
    + CategoryInfo          : ObjectNotFound: (PING.EXE:String) [], CommandNotFoundException
    + FullyQualifiedErrorId : CommandNotFoundException

[localhost]: PS>_
```

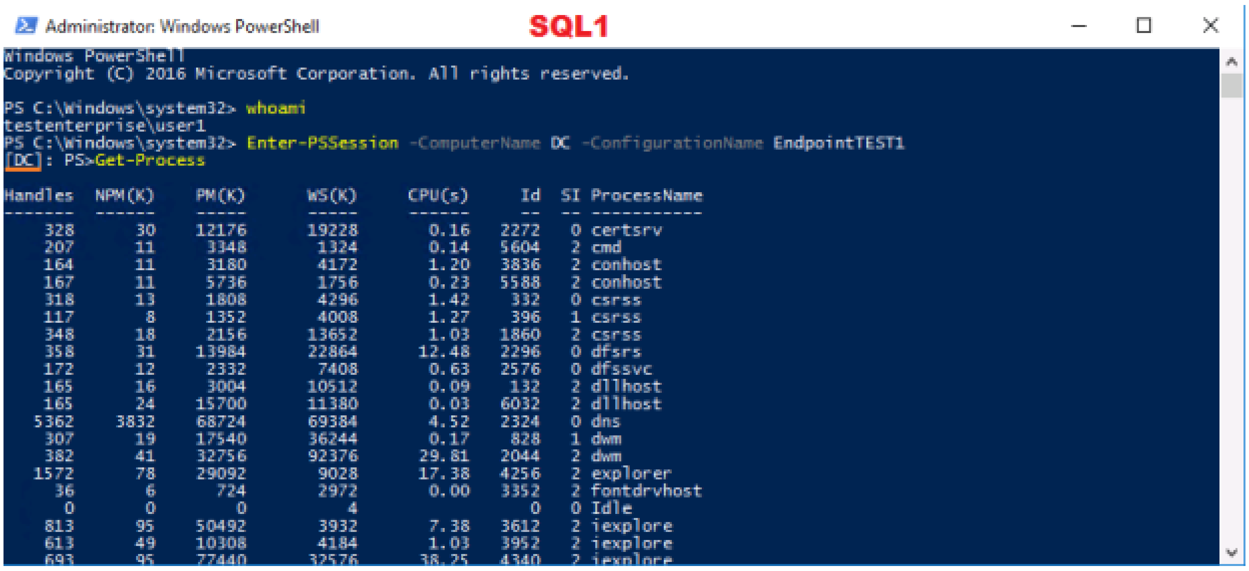
II Connecting to the remote server – SQL1

I think the more practical approach will be the case when an administrator wants to delegate control of some OS components/features to another administrator or a user. Suppose I want User1 to be able to see the the list of services and the list of processes on DC and run the single external command – *nslookup.exe* when connected to DC from another computer (in this case from SQL1). Once again: the new module – **TestModule1** – has been created on DC and we can connect to it (or to the corresponding endpoint) remotly from another computer (SQL1).

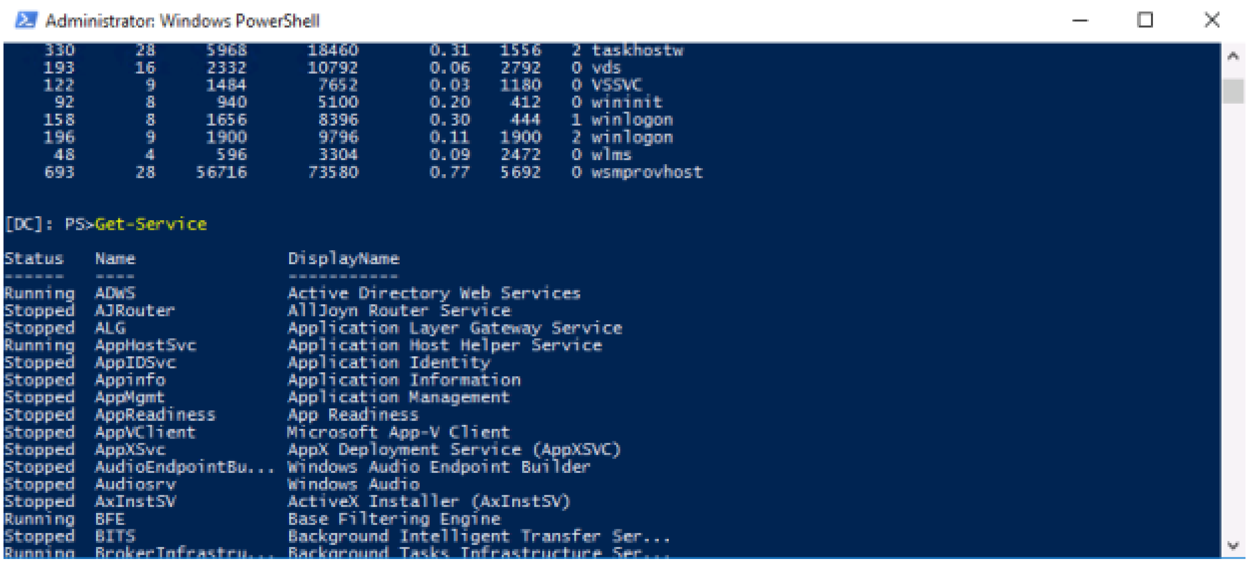
In this case there's no need to issue the Get-Credential command as I've already logged in as User1 onto SQL1:

Enter-PSSession -ComputerName DC -ConfigurationName EndpointTEST1 -Credential $UserCred

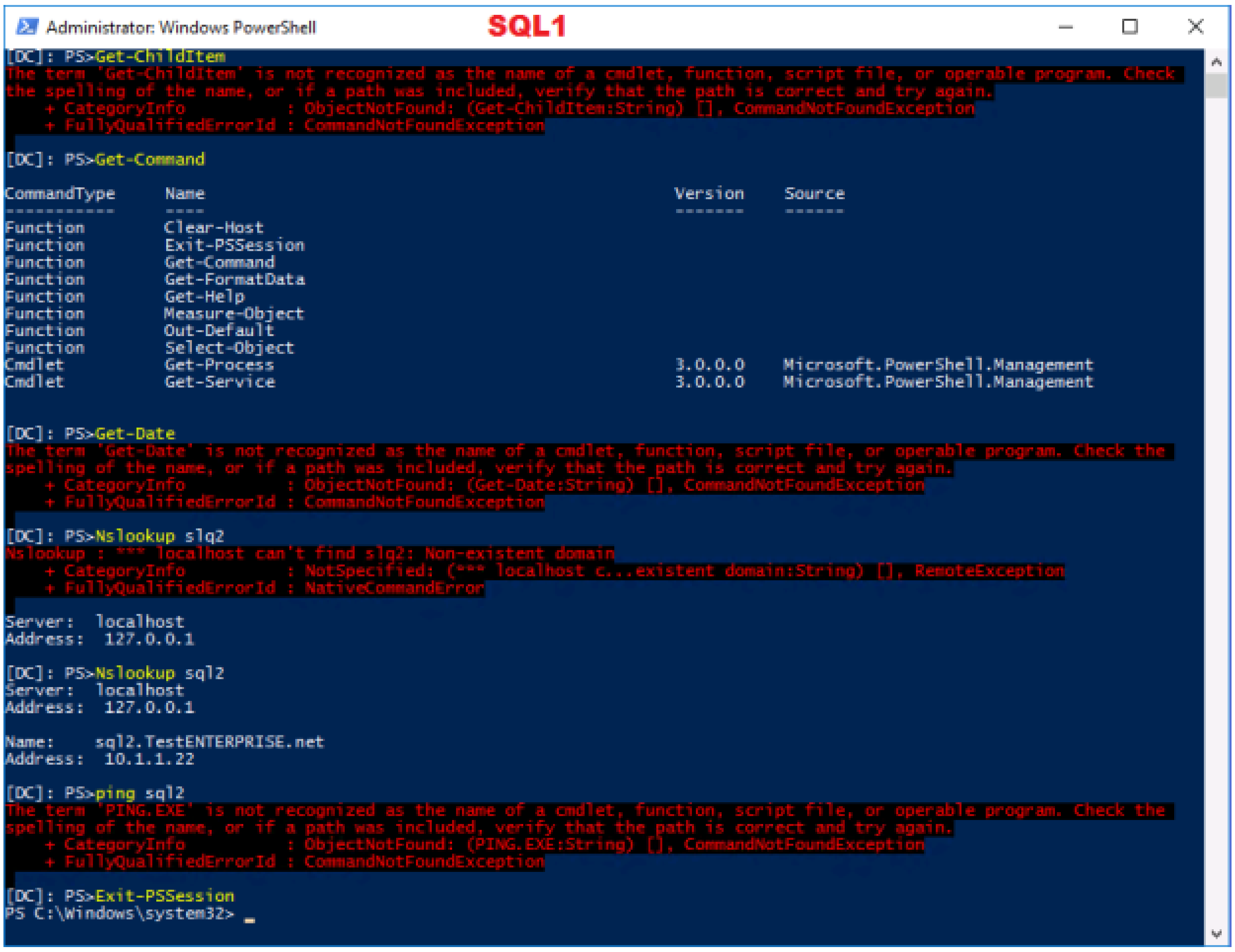
Please note the DC prefix when working with the remote server (DC).



```
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.

PS C:\Windows\system32> whoami
testenterprise\user1
PS C:\Windows\system32> Enter-PSSession -ComputerName DC -ConfigurationName EndpointTEST1
[DC]: PS>Get-Process

Handles  NPM(K)    PM(K)      WS(K)     CPU(s)     Id  SI ProcessName
-------  ------    -----      -----     ------     --  -- -----------
    328      30    12176      19228       0.16   2272   0 certsrv
    207      11     3348       1324       0.14   5604   2 cmd
    164      11     3180       4172       1.20   3836   2 conhost
    167      11     5736       1756       0.23   5588   2 conhost
    318      13     1808       4296       1.42    332   0 csrss
    117       8     1352       4008       1.27    396   1 csrss
    348      18     2156      13652       1.03   1860   2 csrss
    358      31    13984      22864      12.48   2296   0 dfsrs
    172      12     2332       7408       0.63   2576   0 dfssvc
    165      16     3004      10512       0.09    132   2 dllhost
    165      24    15700      11380       0.03   6032   2 dllhost
   5362    3832    68724      69384       4.52   2324   0 dns
    307      19    17540      36244       0.17    828   1 dwm
    382      41    32756      92376      29.81   2044   2 dwm
   1572      78    29092       9028      17.38   4256   2 explorer
     36       6      724       2972       0.00   3352   2 fontdrvhost
      0       0        0          4                 0   0 Idle
    813      95    50492       3932       7.38   3612   2 iexplore
    613      49    10308       4184       1.03   3952   2 iexplore
    693      95    77440      32576      38.25   4340   2 iexplore
```



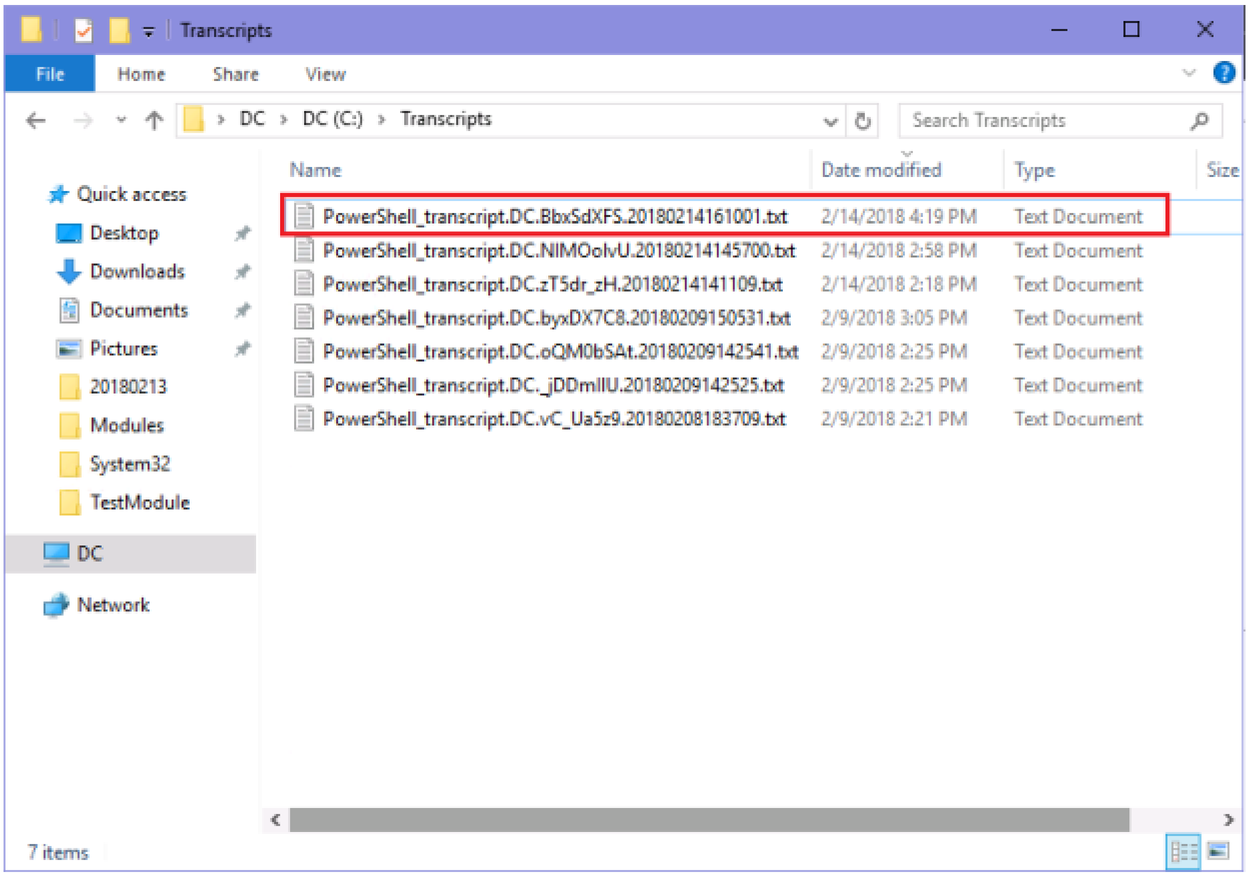
```
    330      28     5968      18460       0.31   1556   2 taskhostw
    193      16     2332      10792       0.06   2792   0 vds
    122       9     1484       7652       0.03   1180   0 VSSVC
     92       8      940       5100       0.20    412   0 wininit
    158       8     1656       8396       0.30    444   1 winlogon
    196       9     1900       9796       0.11   1900   2 winlogon
     48       4      596       3304       0.09   2472   0 wlms
    693      28    56716      73580       0.77   5692   0 wsmprovhost


[DC]: PS>Get-Service

Status   Name               DisplayName
------   ----               -----------
Running  ADWS               Active Directory Web Services
Stopped  AJRouter           AllJoyn Router Service
Stopped  ALG                Application Layer Gateway Service
Running  AppHostSvc         Application Host Helper Service
Stopped  AppIDSvc           Application Identity
Stopped  Appinfo            Application Information
Stopped  AppMgmt            Application Management
Stopped  AppReadiness       App Readiness
Stopped  AppVClient         Microsoft App-V Client
Stopped  AppXSvc            AppX Deployment Service (AppXSVC)
Stopped  AudioEndpointBu... Windows Audio Endpoint Builder
Stopped  Audiosrv           Windows Audio
Stopped  AxInstSV           ActiveX Installer (AxInstSV)
Running  BFE                Base Filtering Engine
Stopped  BITS               Background Intelligent Transfer Ser...
Running  BrokerInfrastru... Background Tasks Infrastructure Ser...
```



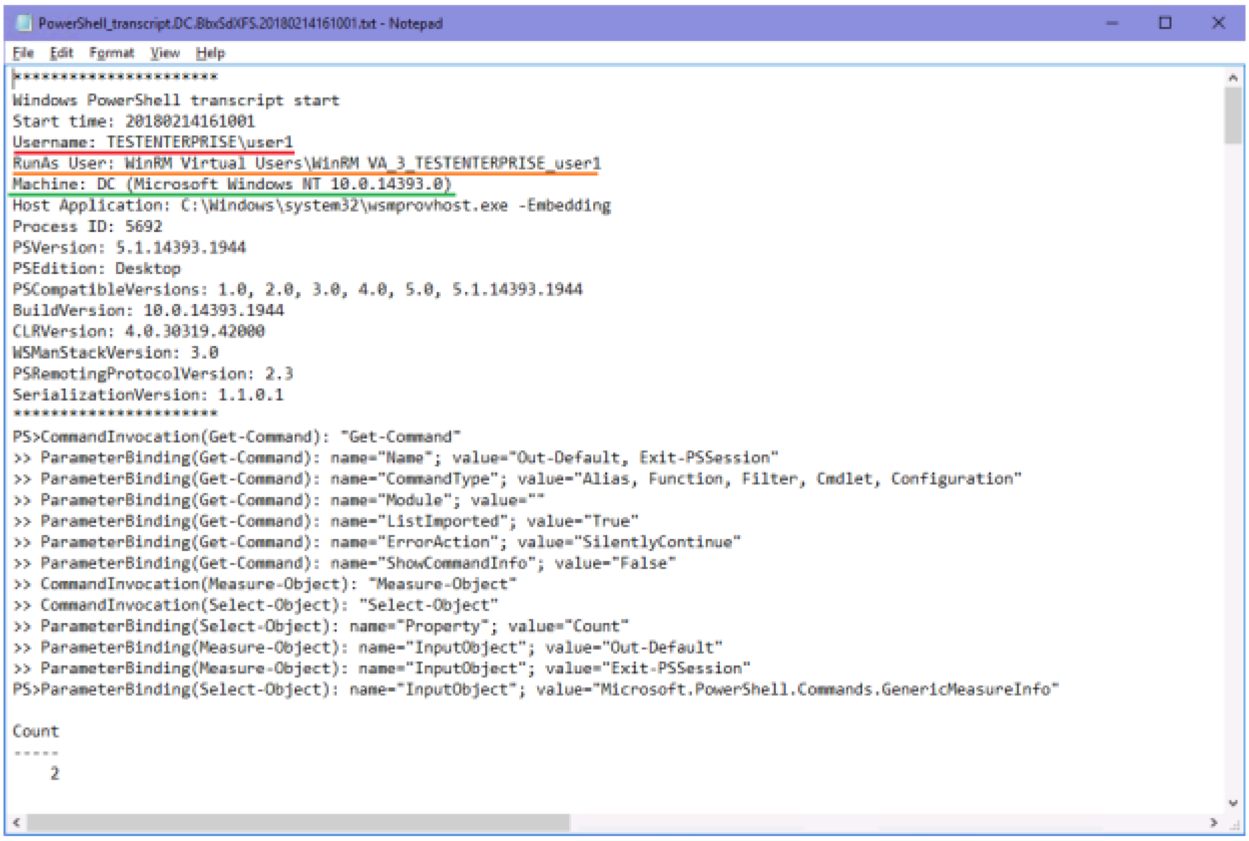
```
[DC]: PS>Get-ChildItem
The term 'Get-ChildItem' is not recognized as the name of a cmdlet, function, script file, or operable program. Check
the spelling of the name, or if a path was included, verify that the path is correct and try again.
    + CategoryInfo          : ObjectNotFound: (Get-ChildItem:String) [], CommandNotFoundException
    + FullyQualifiedErrorId : CommandNotFoundException

[DC]: PS>Get-Command

CommandType     Name                                               Version    Source
-----------     ----                                               -------    ------
Function        Clear-Host
Function        Exit-PSSession
Function        Get-Command
Function        Get-FormatData
Function        Get-Help
Function        Measure-Object
Function        Out-Default
Function        Select-Object
Cmdlet          Get-Process                                        3.0.0.0    Microsoft.PowerShell.Management
Cmdlet          Get-Service                                        3.0.0.0    Microsoft.PowerShell.Management


[DC]: PS>Get-Date
The term 'Get-Date' is not recognized as the name of a cmdlet, function, script file, or operable program. Check the
spelling of the name, or if a path was included, verify that the path is correct and try again.
    + CategoryInfo          : ObjectNotFound: (Get-Date:String) [], CommandNotFoundException
    + FullyQualifiedErrorId : CommandNotFoundException

[DC]: PS>Nslookup slq2
Nslookup : *** localhost can't find slq2: Non-existent domain
    + CategoryInfo          : NotSpecified: (*** localhost c...existent domain:String) [], RemoteException
    + FullyQualifiedErrorId : NativeCommandError

Server:  localhost
Address:  127.0.0.1

[DC]: PS>Nslookup sql2
Server:  localhost
Address:  127.0.0.1

Name:    sql2.TestENTERPRISE.net
Address:  10.1.1.22

[DC]: PS>ping sql2
The term 'PING.EXE' is not recognized as the name of a cmdlet, function, script file, or operable program. Check the
spelling of the name, or if a path was included, verify that the path is correct and try again.
    + CategoryInfo          : ObjectNotFound: (PING.EXE:String) [], CommandNotFoundException
    + FullyQualifiedErrorId : CommandNotFoundException

[DC]: PS>Exit-PSSession
PS C:\Windows\system32> _
```

Issuing the Exit-PSSession command will end the remote session: the DC prefix will change to PS C:\Windows\system32.

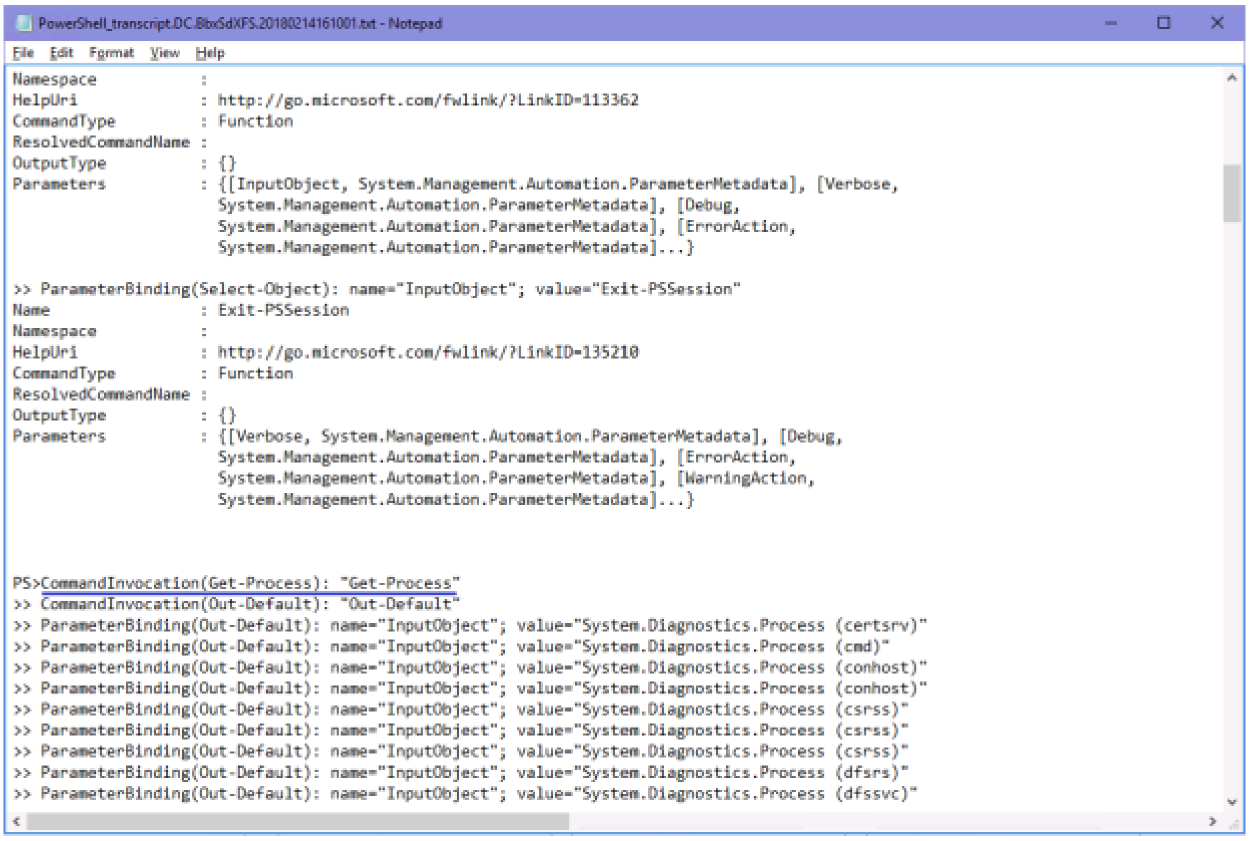
As you've already seen, the **EndpointTEST1** sessions have their transcript directory set to **C:\Transcripts** – let's parse the trascript of the session above (please note that logging occurs on the server where the user-defined PS module is installed):



Name	Date modified	Type	Size
PowerShell_transcript.DC.BbxSdXFS.20180214161001.txt	2/14/2018 4:19 PM	Text Document	
PowerShell_transcript.DC.NIMOolvU.20180214145700.txt	2/14/2018 2:58 PM	Text Document	
PowerShell_transcript.DC.zT5dr_zH.20180214141109.txt	2/14/2018 2:18 PM	Text Document	
PowerShell_transcript.DC.byxDX7C8.20180209150531.txt	2/9/2018 3:05 PM	Text Document	
PowerShell_transcript.DC.oQM0bSAt.20180209142541.txt	2/9/2018 2:25 PM	Text Document	
PowerShell_transcript.DC._jDDmIIU.20180209142525.txt	2/9/2018 2:25 PM	Text Document	
PowerShell_transcript.DC.vC_Ua5z9.20180208183709.txt	2/9/2018 2:21 PM	Text Document	


PowerShell_transcript.DC.BbxSdXFS.20180214161001.txt - Notepad
File Edit Format View Help

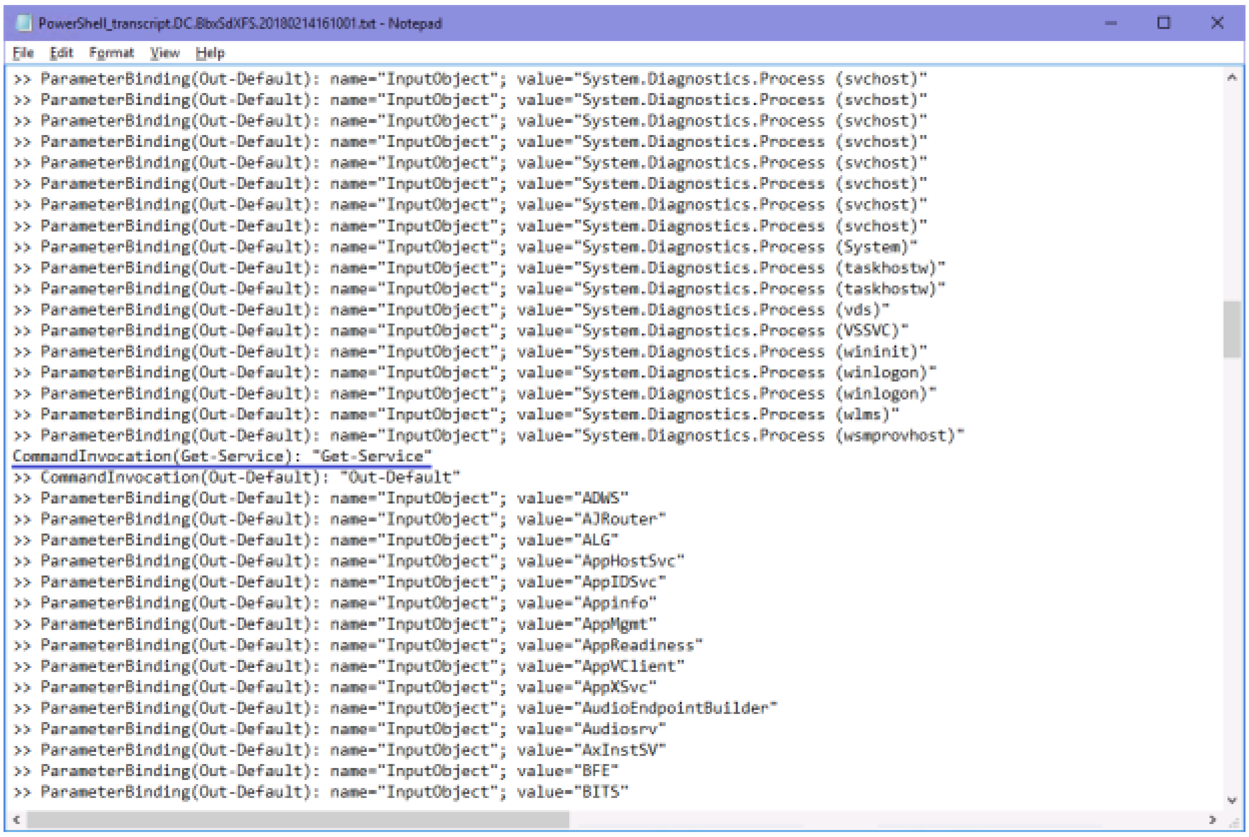
```
**********************
Windows PowerShell transcript start
Start time: 20180214161001
Username: TESTENTERPRISE\user1
RunAs User: WinRM Virtual Users\WinRM VA_3_TESTENTERPRISE_user1
Machine: DC (Microsoft Windows NT 10.0.14393.0)
Host Application: C:\Windows\system32\wsmprovhost.exe -Embedding
Process ID: 5692
PSVersion: 5.1.14393.1944
PSEdition: Desktop
PSCompatibleVersions: 1.0, 2.0, 3.0, 4.0, 5.0, 5.1.14393.1944
BuildVersion: 10.0.14393.1944
CLRVersion: 4.0.30319.42000
WSManStackVersion: 3.0
PSRemotingProtocolVersion: 2.3
SerializationVersion: 1.1.0.1
**********************
PS>CommandInvocation(Get-Command): "Get-Command"
>> ParameterBinding(Get-Command): name="Name"; value="Out-Default, Exit-PSSession"
>> ParameterBinding(Get-Command): name="CommandType"; value="Alias, Function, Filter, Cmdlet, Configuration"
>> ParameterBinding(Get-Command): name="Module"; value=""
>> ParameterBinding(Get-Command): name="ListImported"; value="True"
>> ParameterBinding(Get-Command): name="ErrorAction"; value="SilentlyContinue"
>> ParameterBinding(Get-Command): name="ShowCommandInfo"; value="False"
>> CommandInvocation(Measure-Object): "Measure-Object"
>> CommandInvocation(Select-Object): "Select-Object"
>> ParameterBinding(Select-Object): name="Property"; value="Count"
>> ParameterBinding(Measure-Object): name="InputObject"; value="Out-Default"
>> ParameterBinding(Measure-Object): name="InputObject"; value="Exit-PSSession"
PS>ParameterBinding(Select-Object): name="InputObject"; value="Microsoft.PowerShell.Commands.GenericMeasureInfo"

Count
-----
    2
```



PowerShell_transcript.DC.8bxSdXFS.20180214161001.txt - Notepad

File Edit Format View Help

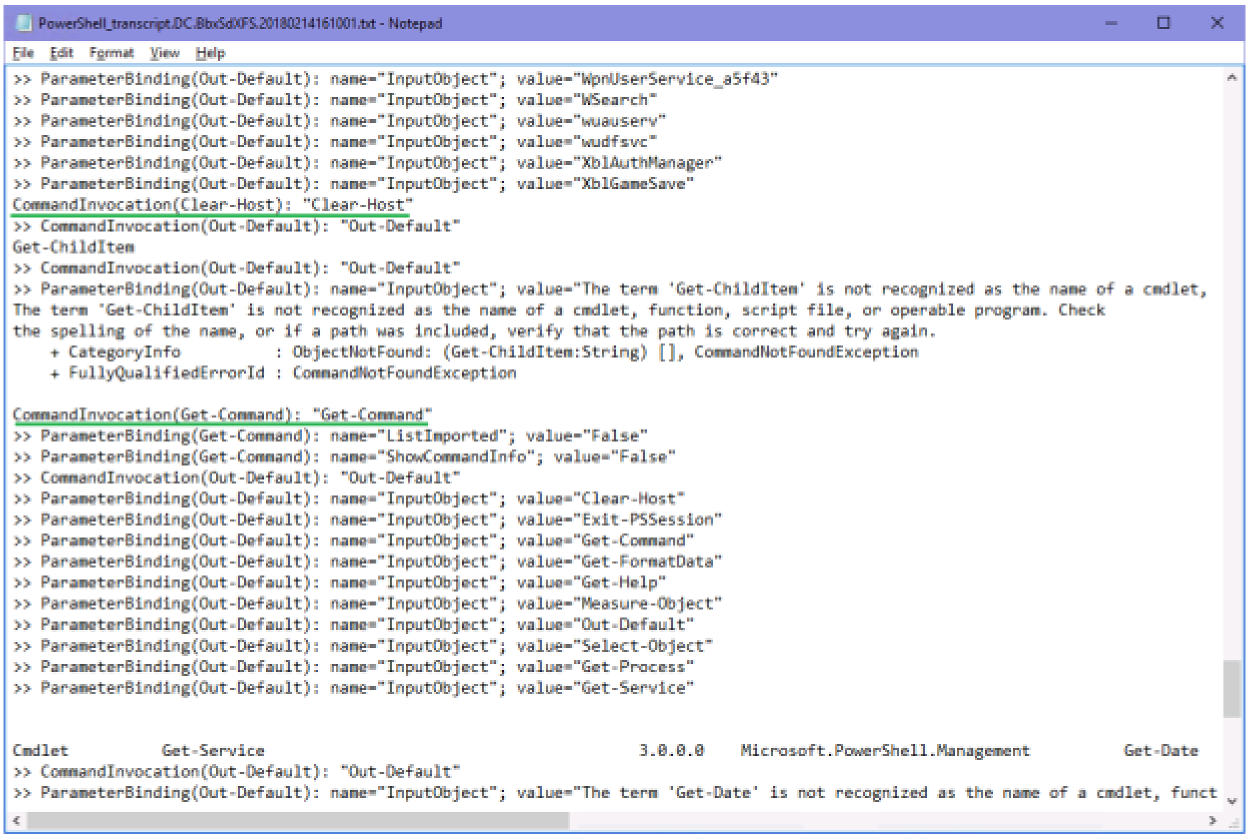
```
Namespace           :
HelpUri             : http://go.microsoft.com/fwlink/?LinkID=113362
CommandType         : Function
ResolvedCommandName :
OutputType          : {}
Parameters          : {[InputObject, System.Management.Automation.ParameterMetadata], [Verbose,
                      System.Management.Automation.ParameterMetadata], [Debug,
                      System.Management.Automation.ParameterMetadata], [ErrorAction,
                      System.Management.Automation.ParameterMetadata]...}

>> ParameterBinding(Select-Object): name="InputObject"; value="Exit-PSSession"
Name                : Exit-PSSession
Namespace           :
HelpUri             : http://go.microsoft.com/fwlink/?LinkID=135210
CommandType         : Function
ResolvedCommandName :
OutputType          : {}
Parameters          : {[Verbose, System.Management.Automation.ParameterMetadata], [Debug,
                      System.Management.Automation.ParameterMetadata], [ErrorAction,
                      System.Management.Automation.ParameterMetadata], [WarningAction,
                      System.Management.Automation.ParameterMetadata]...}


PS>CommandInvocation(Get-Process): "Get-Process"
>> CommandInvocation(Out-Default): "Out-Default"
>> ParameterBinding(Out-Default): name="InputObject"; value="System.Diagnostics.Process (certsrv)"
>> ParameterBinding(Out-Default): name="InputObject"; value="System.Diagnostics.Process (cmd)"
>> ParameterBinding(Out-Default): name="InputObject"; value="System.Diagnostics.Process (conhost)"
>> ParameterBinding(Out-Default): name="InputObject"; value="System.Diagnostics.Process (conhost)"
>> ParameterBinding(Out-Default): name="InputObject"; value="System.Diagnostics.Process (csrss)"
>> ParameterBinding(Out-Default): name="InputObject"; value="System.Diagnostics.Process (csrss)"
>> ParameterBinding(Out-Default): name="InputObject"; value="System.Diagnostics.Process (csrss)"
>> ParameterBinding(Out-Default): name="InputObject"; value="System.Diagnostics.Process (dfsrs)"
>> ParameterBinding(Out-Default): name="InputObject"; value="System.Diagnostics.Process (dfssvc)"
```



PowerShell_transcript.DC.8bxSdXFS.20180214161001.txt - Notepad

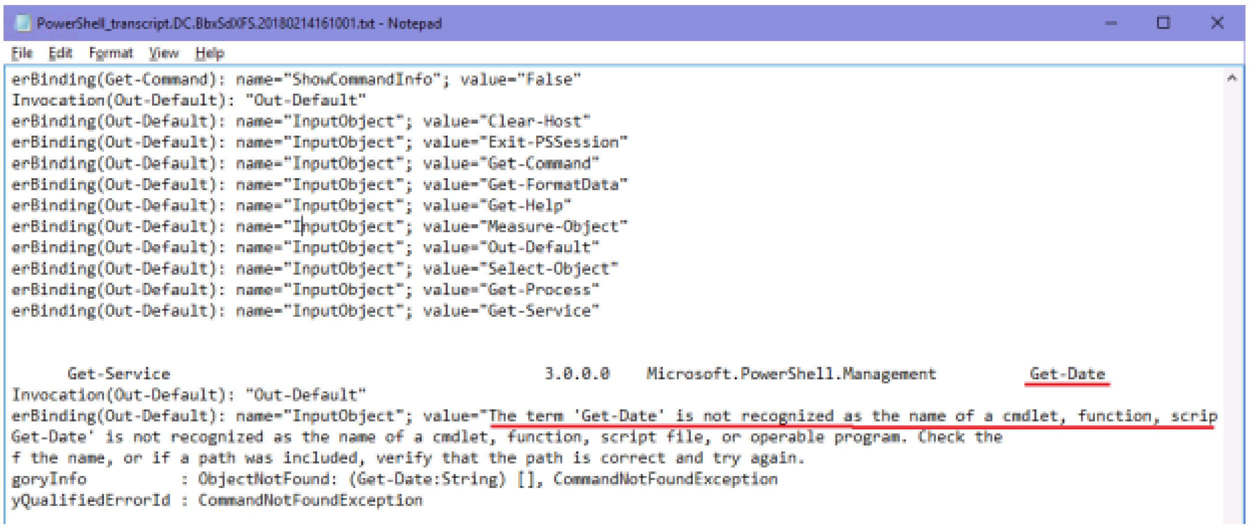
File Edit Format View Help

```
>> ParameterBinding(Out-Default): name="InputObject"; value="System.Diagnostics.Process (svchost)"
>> ParameterBinding(Out-Default): name="InputObject"; value="System.Diagnostics.Process (svchost)"
>> ParameterBinding(Out-Default): name="InputObject"; value="System.Diagnostics.Process (svchost)"
>> ParameterBinding(Out-Default): name="InputObject"; value="System.Diagnostics.Process (svchost)"
>> ParameterBinding(Out-Default): name="InputObject"; value="System.Diagnostics.Process (svchost)"
>> ParameterBinding(Out-Default): name="InputObject"; value="System.Diagnostics.Process (svchost)"
>> ParameterBinding(Out-Default): name="InputObject"; value="System.Diagnostics.Process (svchost)"
>> ParameterBinding(Out-Default): name="InputObject"; value="System.Diagnostics.Process (svchost)"
>> ParameterBinding(Out-Default): name="InputObject"; value="System.Diagnostics.Process (System)"
>> ParameterBinding(Out-Default): name="InputObject"; value="System.Diagnostics.Process (taskhostw)"
>> ParameterBinding(Out-Default): name="InputObject"; value="System.Diagnostics.Process (taskhostw)"
>> ParameterBinding(Out-Default): name="InputObject"; value="System.Diagnostics.Process (vds)"
>> ParameterBinding(Out-Default): name="InputObject"; value="System.Diagnostics.Process (VSSVC)"
>> ParameterBinding(Out-Default): name="InputObject"; value="System.Diagnostics.Process (wininit)"
>> ParameterBinding(Out-Default): name="InputObject"; value="System.Diagnostics.Process (winlogon)"
>> ParameterBinding(Out-Default): name="InputObject"; value="System.Diagnostics.Process (winlogon)"
>> ParameterBinding(Out-Default): name="InputObject"; value="System.Diagnostics.Process (wlms)"
>> ParameterBinding(Out-Default): name="InputObject"; value="System.Diagnostics.Process (wsmprovhost)"
CommandInvocation(Get-Service): "Get-Service"
>> CommandInvocation(Out-Default): "Out-Default"
>> ParameterBinding(Out-Default): name="InputObject"; value="ADWS"
>> ParameterBinding(Out-Default): name="InputObject"; value="AJRouter"
>> ParameterBinding(Out-Default): name="InputObject"; value="ALG"
>> ParameterBinding(Out-Default): name="InputObject"; value="AppHostSvc"
>> ParameterBinding(Out-Default): name="InputObject"; value="AppIDSvc"
>> ParameterBinding(Out-Default): name="InputObject"; value="Appinfo"
>> ParameterBinding(Out-Default): name="InputObject"; value="AppMgmt"
>> ParameterBinding(Out-Default): name="InputObject"; value="AppReadiness"
>> ParameterBinding(Out-Default): name="InputObject"; value="AppVClient"
>> ParameterBinding(Out-Default): name="InputObject"; value="AppXSvc"
>> ParameterBinding(Out-Default): name="InputObject"; value="AudioEndpointBuilder"
>> ParameterBinding(Out-Default): name="InputObject"; value="Audiosrv"
>> ParameterBinding(Out-Default): name="InputObject"; value="AxInstSV"
>> ParameterBinding(Out-Default): name="InputObject"; value="BFE"
>> ParameterBinding(Out-Default): name="InputObject"; value="BITS"
```


PowerShell_transcript.DC.BbxSdXFS.20180214161001.txt - Notepad

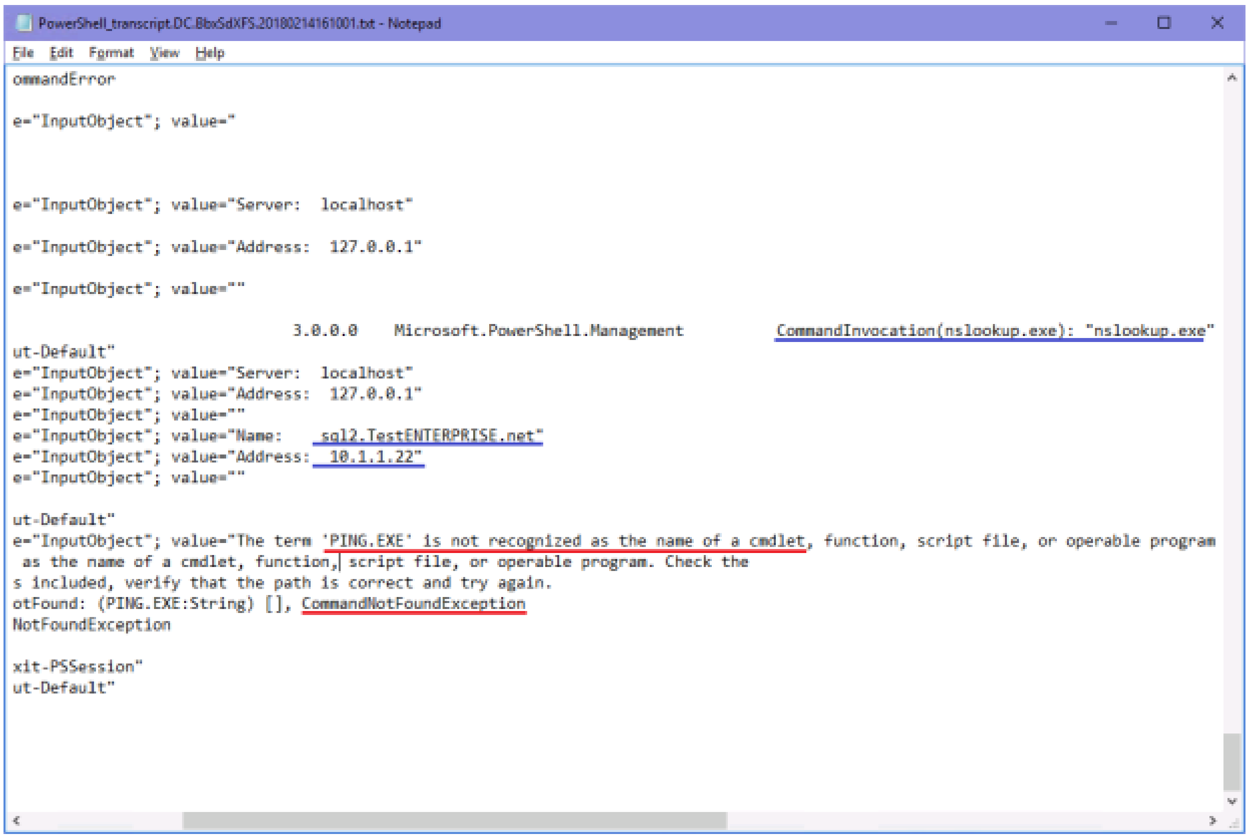
File Edit Format View Help

```
>> ParameterBinding(Out-Default): name="InputObject"; value="WpnUserService_a5f43"
>> ParameterBinding(Out-Default): name="InputObject"; value="WSearch"
>> ParameterBinding(Out-Default): name="InputObject"; value="wuauserv"
>> ParameterBinding(Out-Default): name="InputObject"; value="wudfsvc"
>> ParameterBinding(Out-Default): name="InputObject"; value="XblAuthManager"
>> ParameterBinding(Out-Default): name="InputObject"; value="XblGameSave"
CommandInvocation(Clear-Host): "Clear-Host"
>> CommandInvocation(Out-Default): "Out-Default"
Get-ChildItem
>> CommandInvocation(Out-Default): "Out-Default"
>> ParameterBinding(Out-Default): name="InputObject"; value="The term 'Get-ChildItem' is not recognized as the name of a cmdlet,
The term 'Get-ChildItem' is not recognized as the name of a cmdlet, function, script file, or operable program. Check
the spelling of the name, or if a path was included, verify that the path is correct and try again.
    + CategoryInfo          : ObjectNotFound: (Get-ChildItem:String) [], CommandNotFoundException
    + FullyQualifiedErrorId : CommandNotFoundException

CommandInvocation(Get-Command): "Get-Command"
>> ParameterBinding(Get-Command): name="ListImported"; value="False"
>> ParameterBinding(Get-Command): name="ShowCommandInfo"; value="False"
>> CommandInvocation(Out-Default): "Out-Default"
>> ParameterBinding(Out-Default): name="InputObject"; value="Clear-Host"
>> ParameterBinding(Out-Default): name="InputObject"; value="Exit-PSSession"
>> ParameterBinding(Out-Default): name="InputObject"; value="Get-Command"
>> ParameterBinding(Out-Default): name="InputObject"; value="Get-FormatData"
>> ParameterBinding(Out-Default): name="InputObject"; value="Get-Help"
>> ParameterBinding(Out-Default): name="InputObject"; value="Measure-Object"
>> ParameterBinding(Out-Default): name="InputObject"; value="Out-Default"
>> ParameterBinding(Out-Default): name="InputObject"; value="Select-Object"
>> ParameterBinding(Out-Default): name="InputObject"; value="Get-Process"
>> ParameterBinding(Out-Default): name="InputObject"; value="Get-Service"


Cmdlet          Get-Service                                        3.0.0.0    Microsoft.PowerShell.Management          Get-Date
>> CommandInvocation(Out-Default): "Out-Default"
>> ParameterBinding(Out-Default): name="InputObject"; value="The term 'Get-Date' is not recognized as the name of a cmdlet, funct
```


PowerShell_transcript.DC.BbxSdXFS.20180214161001.txt - Notepad

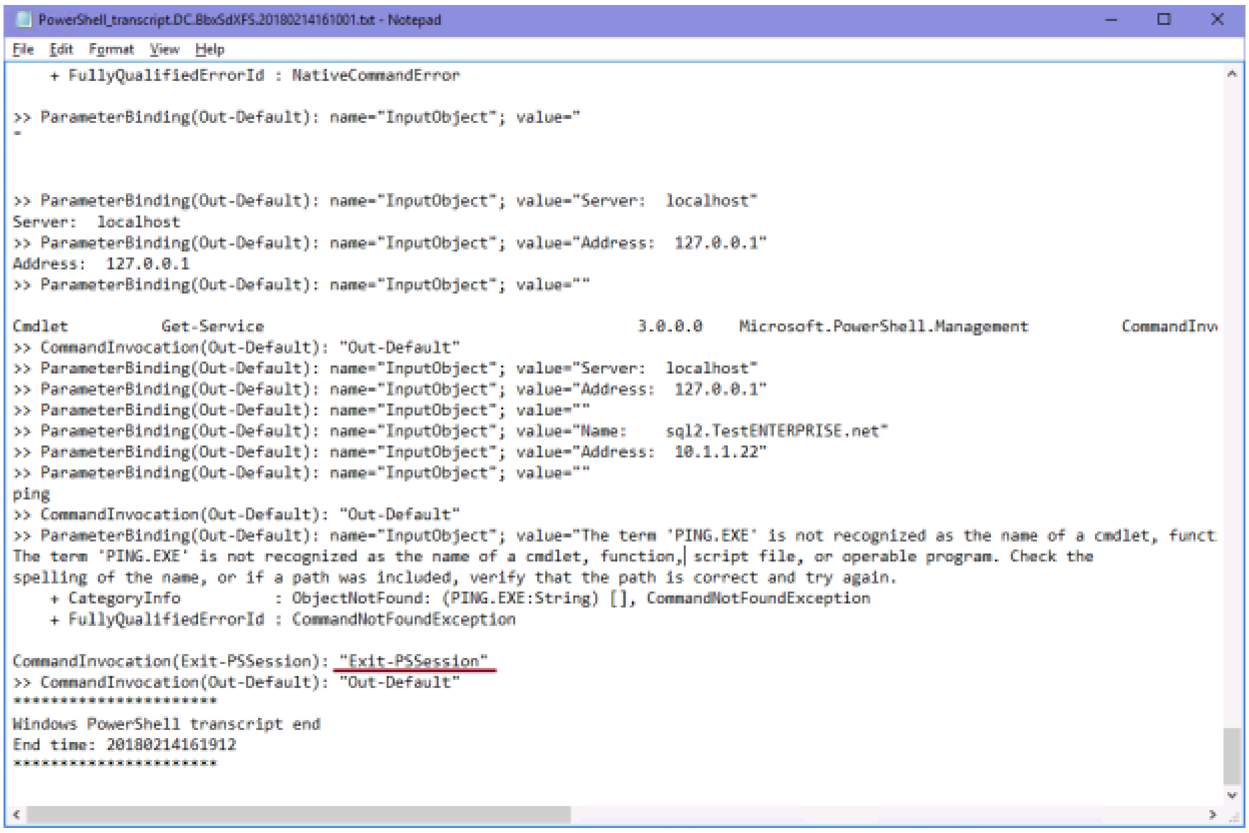
File Edit Format View Help

```
erBinding(Get-Command): name="ShowCommandInfo"; value="False"
Invocation(Out-Default): "Out-Default"
erBinding(Out-Default): name="InputObject"; value="Clear-Host"
erBinding(Out-Default): name="InputObject"; value="Exit-PSSession"
erBinding(Out-Default): name="InputObject"; value="Get-Command"
erBinding(Out-Default): name="InputObject"; value="Get-FormatData"
erBinding(Out-Default): name="InputObject"; value="Get-Help"
erBinding(Out-Default): name="InputObject"; value="Measure-Object"
erBinding(Out-Default): name="InputObject"; value="Out-Default"
erBinding(Out-Default): name="InputObject"; value="Select-Object"
erBinding(Out-Default): name="InputObject"; value="Get-Process"
erBinding(Out-Default): name="InputObject"; value="Get-Service"


      Get-Service                                        3.0.0.0    Microsoft.PowerShell.Management          Get-Date
Invocation(Out-Default): "Out-Default"
erBinding(Out-Default): name="InputObject"; value="The term 'Get-Date' is not recognized as the name of a cmdlet, function, scrip
Get-Date' is not recognized as the name of a cmdlet, function, script file, or operable program. Check the
f the name, or if a path was included, verify that the path is correct and try again.
goryInfo          : ObjectNotFound: (Get-Date:String) [], CommandNotFoundException
yQualifiedErrorId : CommandNotFoundException
```



PowerShell_transcript.DC.BbxSdXFS.20180214161001.txt - Notepad
File Edit Format View Help

```
ommandError

e="InputObject"; value="



e="InputObject"; value="Server:  localhost"

e="InputObject"; value="Address:  127.0.0.1"

e="InputObject"; value=""

                              3.0.0.0    Microsoft.PowerShell.Management         CommandInvocation(nslookup.exe): "nslookup.exe"
ut-Default"
e="InputObject"; value="Server:  localhost"
e="InputObject"; value="Address:  127.0.0.1"
e="InputObject"; value=""
e="InputObject"; value="Name:    sql2.TestENTERPRISE.net"
e="InputObject"; value="Address:  10.1.1.22"
e="InputObject"; value=""

ut-Default"
e="InputObject"; value="The term 'PING.EXE' is not recognized as the name of a cmdlet, function, script file, or operable program
 as the name of a cmdlet, function, script file, or operable program. Check the
s included, verify that the path is correct and try again.
otFound: (PING.EXE:String) [], CommandNotFoundException
NotFoundException

xit-PSSession"
ut-Default"
```



PowerShell_transcript.DC.BbxSdXFS.20180214161001.txt - Notepad
File Edit Format View Help

```
    + FullyQualifiedErrorId : NativeCommandError

>> ParameterBinding(Out-Default): name="InputObject"; value="
"



>> ParameterBinding(Out-Default): name="InputObject"; value="Server:  localhost"
Server:  localhost
>> ParameterBinding(Out-Default): name="InputObject"; value="Address:  127.0.0.1"
Address:  127.0.0.1
>> ParameterBinding(Out-Default): name="InputObject"; value=""

Cmdlet          Get-Service                                       3.0.0.0    Microsoft.PowerShell.Management         CommandInv
>> CommandInvocation(Out-Default): "Out-Default"
>> ParameterBinding(Out-Default): name="InputObject"; value="Server:  localhost"
>> ParameterBinding(Out-Default): name="InputObject"; value="Address:  127.0.0.1"
>> ParameterBinding(Out-Default): name="InputObject"; value=""
>> ParameterBinding(Out-Default): name="InputObject"; value="Name:    sql2.TestENTERPRISE.net"
>> ParameterBinding(Out-Default): name="InputObject"; value="Address:  10.1.1.22"
>> ParameterBinding(Out-Default): name="InputObject"; value=""
ping
>> CommandInvocation(Out-Default): "Out-Default"
>> ParameterBinding(Out-Default): name="InputObject"; value="The term 'PING.EXE' is not recognized as the name of a cmdlet, funct
The term 'PING.EXE' is not recognized as the name of a cmdlet, function, script file, or operable program. Check the
spelling of the name, or if a path was included, verify that the path is correct and try again.
    + CategoryInfo          : ObjectNotFound: (PING.EXE:String) [], CommandNotFoundException
    + FullyQualifiedErrorId : CommandNotFoundException

CommandInvocation(Exit-PSSession): "Exit-PSSession"
>> CommandInvocation(Out-Default): "Out-Default"
**********************
Windows PowerShell transcript end
End time: 20180214161912
**********************
```

Summary:

The new feature of Windows Server 2016 – Just Enough Administration – can help administrators control in the most specific way what administrative actions other administrators or users may perform on which systems: this level of control is not possible with Windows GUI.