HP Operations Orchestration

for the Windows and Linux operating systems

Software Version: OO 10.x

PowerShell Wizard Guide

Document Release Date: May 2014 Software Release Date: May 2014



Legal Notices

Warranty

The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

The information contained herein is subject to change without notice.

Restricted Rights Legend

Confidential computer software. Valid license from HP required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

Copyright Notices

© Copyright 2009-2014 Hewlett-Packard Development Company, L.P.

Trademark Notices

For information on open-source and third-party software acknowledgements, see *Open-Source and Third-Party Software Acknowledgements* (HPOO_OpenSrc_3rd-PartyAcks.pdf) in the documentation set for this release.

Documentation Updates

The title page of this document contains the following identifying information:

- Software Version number, which indicates the software version.
- Document Release Date, which changes each time the document is updated.
- Software Release Date, which indicates the release date of this version of the software.

To check for recent updates or to verify that you are using the most recent edition of a document, go to:

https://hpln.hp.com/

This site requires that you register for an HP Passport and sign-in. To register for an HP Passport ID, go to:

http://h20229.www2.hp.com/passport-registration.html

Or click the New users - please register link on the HP Passport login page.

You will also receive updated or new editions if you subscribe to the appropriate product support service. Contact your HP sales representative for details.

Support

Visit the HP Software Support Web site at:

www.hp.com/go/hpsoftwaresupport

This Web site provides contact information and details about the products, services, and support that HP Software offers.

HP Software online support provides customer self-solve capabilities. It provides a fast and efficient way to access interactive technical support tools needed to manage your business. As a valued support customer, you can benefit by using the support Web site to:

- Search for knowledge documents of interest
- Submit and track support cases and enhancement requests
- Download software patches
- Manage support contracts
- Look up HP support contacts
- Review information about available services
- Enter into discussions with other software customers
- Research and register for software training

Most of the support areas require that you register as an HP Passport user and sign in. Many also require a support contract. To register for an HP Passport ID, go to:

http://h20229.www2.hp.com/passport-registration.html

To find more information about access levels, go to:

http://h20230.www2.hp.com/new_access_levels.jsp

Contents

1	Introduction	.7
	Purpose of the PowerShell Wizard Integration	7
	Supported Versions	7
	Supported Languages	7

2 Getting Started with the PowerShell Wizard Integration8

Download the PowerShell Wizard	.8
Install the PowerShell Wizard	.8
Configure Logging Settings	.8
Uninstall the PowerShell Wizard	.8
PowerShell Wizard Requirements	.9
System Requirements	.9
Target Host	.9
PowerShell Wizard Enhancements from 9.x	.9

3	The PowerShell Wizard Steps	10
	Step 1. Select the Repository	
	Step 2. Configure the PowerShell Connection	
	Step 3. Select the Modules	
	Step 4. Select Operations (Cmdlets)	13

PowerShell Wizard Integration Operations and Flows	14
Generated Flows	14
Inputs	15
Descriptions	16
PowerShell Script Operation	17
Inputs	
Connection Inputs	19
Session and Piping Inputs	20
Additional Modules and Snapins	21
PowerShell Script and cmdlet Inputs	21
Result Formatting	21
Run a PowerShell Script on a Localhost	22
Run PowerShell Scripts from a File	23
Load PowerShell Functions from Files	24
Run a PowerShell Script on a Remote Host	25
Format the Result	26

Run Multiple PowerShell Cmdlets Scripts in the Same PowerShell Session	28
Assign the Result of One Cmdlet as a Parameter to Another Cmdlet	31
Solution 1: Create a New PowerShell Script Step	34
Solution 2: Run a PowerShell Script in the Generated Flow Context	
Solution 3: Use Generated Flows Only and Minimize the User Effort	37

5	PowerShell Remoting	
	Overview	
	Enable Remoting Using GPO (Group Policy Objects)	
	Group Policy Configuration for a Single Host	
	Group Policy Configuration for a Group of Servers	
	Enable Remoting for Non-Administrative Users	
	Authentication Types	
	Basic	
	CredSSP	
	Default	
	Digest	
	Kerberos	
	Negotiate	
	NegotiateWithImplicitCredential	

6	Troubleshooting	.48
	Could not connect to the host	48
	The wizard fails to load modules on a x64 localhost.	

1 Introduction

With this integration, users can generate OO flows from the selected PowerShell cmdlets found in a list of modules/snapins.

Purpose of the PowerShell Wizard Integration

Its main advantages are:

- Automation. Avoid having to repeat the same time-consuming process of creating flows which execute PowerShell cmdlets. Perform the following steps as an alternative to using the PowerShell Wizard (multiply by the number of cmdlets):
 - Create an empty flow
 - Drag and drop the PowerShell Script operation
 - Search for the cmdlet description
 - Set the required input values
 - Set the description of the flow. The step inherits its description from the PowerShell Script operation, but this is not available for the flow.
- Authoring ease. The description of each flow contains the default description of the corresponding cmdlet which it executes. Therefore, the user is not forced to open the cmdlet description in a browser and switch between OO and the Internet.
- Module and cmdlet discovery. The wizard discovers the available modules and cmdlets from a target host.

Supported Versions

Table 1Supported Versions

Operations Orchestration Version	PowerShell Wizard Version		
OO 10.x with Base Content Pack	10.x		

Supported Languages

This release supports the following languages:

- en English
- fr French
- de German
- ja Japanese
- es Spanish
- zh_CN Simplified Chinese

2 Getting Started with the PowerShell Wizard Integration

Download the PowerShell Wizard

The PowerShell Wizard Installer is an executable file that can be downloaded from the HP Live Network page

From https://hpln.hp.com, click Operations Orchestration Community and log in.

The **Operations Orchestration Community** page contains links to announcements, discussions, downloads, documentation, help, and support.

- 2 On the left-hand side, click **Operations Orchestration Content Packs**.
- 3 In the **Operations Orchestration Content Packs** box, click **Content**. The HP Passport and sign-in page appears.
- 4 Enter your user ID and Password to access to continue.
- 5 Click **HP Operations Orchestration 10.x**, and then select the items that you want to download.

Install the PowerShell Wizard

The wizard is installed if Studio is selected from the Operation Orchestration installer.

Configure Logging Settings

The configure logging settings are no longer supported in the10x wizard.

Uninstall the PowerShell Wizard

The wizard is uninstalled when Studio is uninstalled.

PowerShell Wizard Requirements

System Requirements

The minimum software requirements for systems running the PowerShell Wizard for HP Operations Orchestration are:

- Java SE Runtime Environment (also known as JRE) 7
- .NET Framework 2 or a later version

Target Host

- PowerShell with remoting enabled.
- For the products providing PowerShell cmdlets, the target host must have the modules and snapins available. Run one of the following cmdlets to list the required modules and snapins. Otherwise, it means that the host does not have the cmdlets provided for that product.

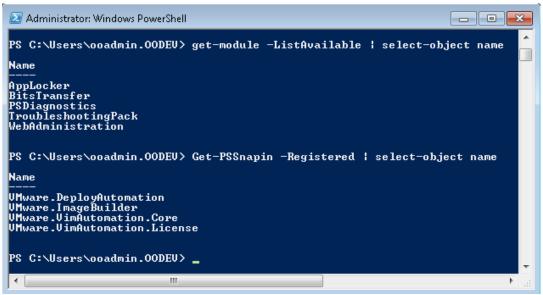


Figure 1: How to list the modules and snapins in the PowerShell console

PowerShell Wizard Enhancements from 9.x

- The wizard now has the version (10.x) displayed in the title.
- The wizard now appears in the taskbar and can be closed, minimized or brought to the front.

3 The PowerShell Wizard Steps

The PowerShell Wizard contains only a few steps. The Welcome page contains a short summary of the wizard. This section describes the steps that you have to perform.

Step 1. Select the Repository

PowerShell Wizard 10	0.0
Steps	Select repository
1. Welcome to the PowerShell Wizard	Enter the location of an existing studio project or a folder where a new project will be created:
2. Select repository	C:\Users\[username]\.oo\Workspace\New Project Browse
3. Connection settings	
 Select modules and snapins 	
5. Select cmdlets	
6. Finish	
	Back Next Cancel

Figure 2: PowerShell Wizard: Select the project location page

- In the **Enter the location**: field, type in the required project path or click **Browse** to locate the project location.
- The wizard generates a 10.x studio project, but not a content pack or a repository. The project has a default location: C:\Users\[username]\.oo\Workspace\New Project.

Step 2. Configure the PowerShell Connection

PowerShell Wizard	
Steps	PowerShell connection
 Welcome to the PowerShell Wizard Select repository PowerShell connection Modules page Select cmdlets Finish 	Host: Username: Password: Port: Authentication type: Default Use SSL
	< Back Next > Cancel

Figure 3: PowerShell Wizard: Connection page

Host

Type the name of the host that you want to connect to. If you leave the **Host** field empty, the PowerShell Wizard uses localhost as the default.

Note: If Host is empty then the authentication type would be

NegotiateWithImplicitCredential. If the host has been defined, the wizard considers the host definition provided by the user.

Username

Enter the user name to connect to the target host..

Password

Enter the password.

Port

The port values can be in the range of 1-65535. If you set the port value to 0, the wizard ignores it and uses the default port values. The default port values are: 5985 (HTTP) and 5986 (HTTPS).

Step 3. Select the Modules

Steps	Modules page		
 Welcome to the PowerShell Wizard Select repository PowerShell connection 		to be loaded. They will be displayed on the next page. Get-Host) will only be included if you select the checkbox	
4. Modules page	Available Modules and Snapins	Selected Modules and Snapins	
 Select cmdlets Finish 	ADRMS AppLocker BestPractices BitsTransfer PSDiagnostics ServerManager TroubleshootingPack WebAdministration	FailoverClusters SqlServerCndletSnapin100 SqlServerProviderSnapin100 <	
	Add default PowerShell cmdlets (i. e. the	nes that come with PowerShell installation)	

Figure 4: PowerShell Wizard: Modules page

The wizard detects all the available modules/snapins on the target host and displays them in a list as shown above. You can select/unselect any module and the wizard retrieves only those cmdlets contained in the selected modules.

Cmdlets such as Get-Process and Get-Service are not contained in the list of available modules. These are cmdlets which are available by default in PowerShell. To retrieve the list of default cmdlets, select the checkbox **Add default PowerShell cmdlets**.

Note: If you select the Add default PowerShell cmdlets checkbox, the lists are disabled.

Step 4. Select Operations (Cmdlets)

	Select cmdlets			
 Welcome to the PowerShell Wizard 	There are 74 PowerShell cmdlets that would like to import.	t matched your s	search on the given host. Please s	elect the ones you
2. Select repository	Search cmdlets:			
3. PowerShell connection				
4. Modules page	All available cmdlets		Selected cmdlets	
5. Select cmdlets 6. Finish	Add-ClusterDisk Add-ClusterFileServerRole Add-ClusterGenericApplicationRole Add-ClusterGenericScriptRole Add-ClusterGenericServiceRole Add-ClusterGenericServiceRole Add-ClusterPrintServerRole Add-ClusterPrintServerRole Add-ClusterResource Add-ClusterResourceDependency Add-ClusterResourceType		> < > > < < < < < < < < < < < < < < < <	•

Figure 5: PowerShell Wizard: Select cmdlets page

The selected modules are loaded to the PowerShell runspace, and the wizard retrieves the names of the cmdlets from those modules.

You can move the cmdlets from left to right or right to left. Use the search textbox if the list is very large, and you have difficulties finding the required cmdlet. The wizard searches the list for the cmdlets with names containing the search text. In addition, the wizard updates the list while you are typing.

4 Using the PowerShell Wizard – OO Integration

PowerShell Wizard Integration Operations and Flows

This section describes the operations and flows in the PowerShell integration.

Generated Flows

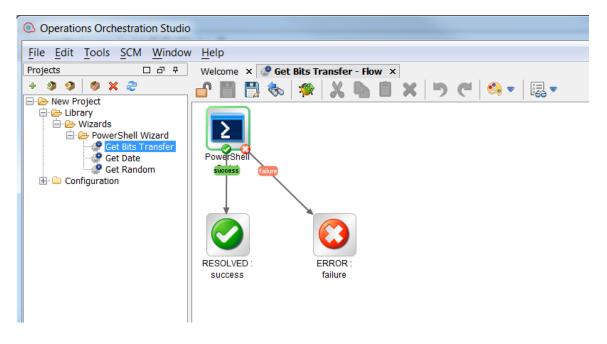


Figure 6: Example of a generated flow

The PowerShell wizard generates one flow for each selected cmdlet unless the project already contains the flow. For example, if the flow was generated in a previous run of the wizard.

The name of the flow is obtained from the name of the cmdlet by applying the following rules:

• Replace "-" with " ".

For example, the name of the flow for the cmdlet *Get-IScsiHbaTarget* is going to be *Get IScsi Hba Target*.

Inputs

Input	Required	Туре	From	
host		Single Value	Prompt User	کا کی کی
URI		Single Value	Prompt User	e
shellURI		Single Value 🔻	Prompt User	e s
username		Single Value 🔻	Prompt User	6
password		Single Value	Prompt User	6
configurationName		Single Value 🔻	Prompt User	e
authType		Single Value 🔻	Prompt User from List - Selection List	Solution
connectionTimeout		Single Value 🔻	Prompt User	e>
scriptTimeout		Single Value		e>
useSSL		Single Value 🔻	Prompt User from List - Selection List	S
skipCACheck		Single Value 🔻	Value:	6
skipCNCheck		Single Value 🔻	Value:	e>
port		Single Value	Prompt User	e e
keepSessionAlive		Single Value	Prompt User from List - Selection List	e s
enablePiping		Single Value 🔻	Prompt User from List - Selection List	6
runspaceID		Single Value	Prompt User	e>
modules		Single Value	Value: BitsTransfer	e s
snapins		Single Value 🔻	Prompt User	6
script		Single Value 🔻	Prompt User	e>
cmdlet		Single Value	Value: Get-BitsTransfer	e la companya de la compa
returnTable		Single Value 🔻	Prompt User from List - Selection List	6
delimiter			Prompt User	
colDelimiter		Single Value	Prompt User	0
rowDelimiter		Single Value	Prompt User	0

Figure 7: Inputs of a generated flow

Each flow has the following inputs:

• Common inputs. All the inputs of the PowerShell Script operation

Descriptions

D: f0123251-b45d-4e50-8b24-a7f62dc1d855	8
	120
ssign Categories:	1 20
puts Outputs Responses Description Scriptlet	
e>	
Get-BitSTransfer cmdlet retrieves a set of BITS transfer jobs. By default, the cmdlet returns the jobs that are owned by the current user. However, if you have administrative credentials, yo cify the AllUsers parameters so that the command returns jobs that are owned by all users. The returned jobs can be filtered by name or ID. The jobs are represented by BitsJob objects. http://go.microsoft.com/fwilnk/?LinkId=141429	iu can
its:	
ost - The hostname or ip address of the PowerShell host. This input is mutual exclusive with URI and shellURI. fault value: localhost	
IRI - Specifies a Uniform Resource Identifier (URI) that defines the connection endpoint for the interactive session.	
ue format: <transport>://<computername>:<port>/<applicationname></applicationname></port></computername></transport>	
mple: http://exch2010CAS1/Powershell?serializationLevel=Full	
hell/URI - Gets the Uniform Resource Identifier (URI) of the shell that is launched when the connection is made. This input is mutual exclusive with host.	
ample: http://schemas.microsoft.com/powershell/Microsoft.Exchange isername - The username to use when connecting to the server.	
semanie - The demanie to dee when connecting to the server.	
use format, deen annew woman work to the many sector and the server.	
automotion in plasmotion to dia maniformation by the second state of the new PSSession. If you specify only the configuration name, the following schema URI is prepended:	
(c) schemas.microsoft.com/powershell. Use Microsoft.PowerShell32 to force x32 PowerShell version on x64 machines.	
uthType - Specifies the mechanism that is used to authenticate the user's credentials.	
id values: Default, Basic, Negotiate, Credssp, Kerberos, NegotiateWithImplicitCredential. (case-insensitive)	
fault value: Default	
connectionTimeout - Determines how long the client computer waits for the remote session connection to be established. When the interval expires, the command to establish the connection	fails.
er a positive value in milliseconds.	
fault value:180000 (3 minutes).	
criptTimeout - Determines how long the client computer waits for the PowerShell script to be executed. When the interval expires, the operation fails. Enter a positive value in milliseconds. /	An
ty value means no time-out; the script execution continues indefinitely.	
seSSL - If true, the operation uses the Secure Sockets Layer (SSL) protocol to establish a connection to the remote computer. By default, SSL is not used. WS-Management encrypts all wind	
verShell content transmitted over the network. UseSSL is an additional protection that sends the data across an HTTPS connection instead of an HTTP connection. Default port for SSL is 598 fault value: false	5.
aur value: raise kipCACheck - Indicates whether, when connecting over HTTPS, the client does not validate that the server certificate is signed by a trusted certificate authority (CA).	
AppCACHECK = indicates whenever, when connecting over HTTPS, the cherk does not validate that the server certaincate is signed by a trusted certaincate additionity (CA). Tault value: true	
aun voice. use	
Applications and a second a second a second a second and a second a se	
ort - Specifies the network port on the remote computer that is used for this connection. To connect to a remote computer, the remote computer must be listening on the port that the connection.	ection
s. The default ports are 5985 (the WinRM port for HTTP) and 5986 (the WinRM port for HTTPS).	
eepSessionAlive - If true, the operation will not close the PowerShell runspace (i.e. the PSSession in case of remote connections) created during the execution and returns the runspaceID a	as the
It of the operation. Actually the PowerShell runspace is saved in the OO session and can be used in other operations using the runspaceID result of previous operation.	
fault value: false	
enablePiping - If true the operation enables piping the way PowerShell does. This input should be used only when keepSessionAlive input is true. First time the piped objects are null and the	scone

Figure 8: Description of a generated flow

The description of each generated flow contains the following items:

- A description of the cmdlet as found in its native documentation
- A link where the user can find a detailed description of the cmdlet written by the provider of the module
- The description of the common inputs copied from the PowerShell Script operation's description.
- The description of the common results

Most of the information included in the description can be obtained from the PowerShell console as shown below:

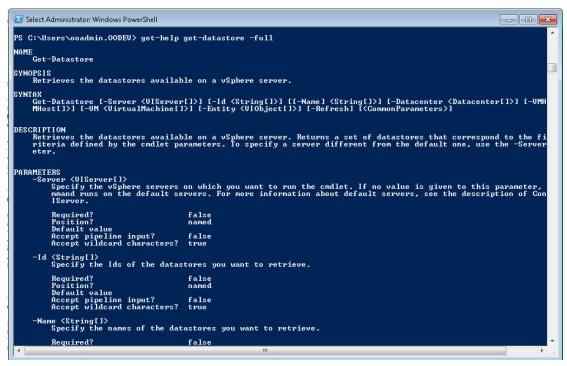


Figure 9: Obtaining the description from the PowerShell console

PowerShell Script Operation

The operation is used to execute a PowerShell script or cmdlet on a target host, either local or remote. If the operation executes a single cmdlet, the parameters of the cmdlet should be passed to the operation inputs.

Inputs

PowerShe	ll Scn - Properties $~\times~$					
(50					
Name: PowerSh	ell Script					
JUID: f0b2afd2	-5733-47e4-80ba-7f2387cc66d5		Version: 72	(11/23/11 18:11 admir	1)	
Assign Categ	iories:					<u> </u>
Inputs Outpu	its Responses Description Script	let				
▲ ▼ Inputs	Summary					4 F 0
RAS Operation fi	ields					
Action Class:	dotNETWebActions_PowerShell.IAc	tions.PowerShelli	BaseAction			
Archive:	dotNETWebActions_PowerShell.dll					
RAS:	/Configuration/Remote Action	Services/RAS_O	perator Path			
		2011/00000000_01	per den _r den			
Override RAS:	\${overrideNRAS}					
inputs ———						
inputs ———			Add Input	Remove Input	1	
nputs Input		Required		Remove Input	1	
		Required		Template	ŧ	· · · · ·
Input		Required	Туре	Template Prompt User	1	
Input		Required	Type Single Value	Template Prompt User Prompt User	1	
Input host URI		Required	Type Single Value - Single Value -	Template Prompt User Prompt User Prompt User	Î	
Input host URI shellURI		Required	Type Single Value Single Value Single Value	Template Prompt User Prompt User Prompt User Prompt User	Î	
Input host URI shellURI username		Required	Type Single Value Single Value Single Value Single Value Single Value	Template Prompt User Prompt User Prompt User Prompt User Prompt User Prompt User	1	
Input host URI shellURI username password		Required	Type Single Value Single Value Single Value Single Value Single Value Single Value	Template Prompt User	1	
Input host URI shellURI username password authType		Required	Type Single Value Single Value Single Value Single Value Single Value Single Value Single Value	Template Prompt User	1	
Input host URI shellURI username password authType useSSL		Required	Type Single Value Single Value Single Value Single Value Single Value Single Value Single Value Single Value	Template Prompt User Value:	1	
Input host URI shellURI username password authType useSSL skipCACheck		Required	Type Single Value Single Value	Template Prompt User Value: Value: Value:	^	
Input host URI shellURI username password authType use55L skipCACheck skipCNCheck	ve	Required	Type Single Value Single Value	Template Prompt User Prompt User Prompt User Prompt User Prompt User Prompt User Value: Value: Prompt User Value:		
Input host URI shellURI username password authType useSSL skipCACheck skipCNCheck port	<pre>//E</pre>	Required Image: Constraint of the second	Type Single Value Single Value	Template Prompt User Value: Value: Prompt User Prompt User Prompt User Prompt User Prompt User Prompt User		
Input host URI shellURI username password authType useSSL skipCACheck skipCACheck sport keepSessionAli ^k	/8	Required Image: Constraint of the second	Type Single Value Single Value	Template Prompt User Value: Value: Prompt User Prompt User Prompt User Prompt User Prompt User Prompt User		
Input host URI shellURI username password authType useSSL skipCACheck skipCNCheck skipCNCheck port keepSessionAliv enablePiping	ve	Required Image: Constraint of the second	Type Single Value Single Value	Template Prompt User Prompt User Prompt User Prompt User Prompt User Prompt User Value: Value: Prompt User		
Input host URI shellURI username password authType use55L skipCACheck skipCNCheck port keepSessionAliv enablePiping runspaceID	/e	Required Image: Image of the second secon	Type Single Value Single Value	Template Prompt User Value: Value: Value: Prompt User		
Input host URI shellURI username password authType useSSL skipCACheck skipCNCheck skipCNCheck port keepSessionAllik enablePiping runspaceID modules	/e	Required Image: Constraint of the second	Type Single Value Single Value	Template Prompt User Prompt User Prompt User Prompt User Prompt User Prompt User Value: Value: Value: Value: Prompt User Promp		
Input host URI shellURI username password authType use55L skipCACheck skipCACheck skipCNCheck port keep5essionAlit enablePiping runspaceID modules snapins	ve	Required Image: Image of the second secon	Type Single Value Single Value	Template Prompt User Prompt User Prompt User Prompt User Prompt User Prompt User Value: Value: Value: Value: Prompt User		
Input host URI shellURI username password authType use55L skipCACheck skipCNCheck skipCNCheck port keepSessionAlix enablePiping runspaceID modules snapins script	/8	Required Image: Image of the second secon	Type Single Value Single Value	Template Prompt User Prompt User Prompt User Prompt User Prompt User Prompt User Value: Value: Value: Value: Prompt User		
Input host URI shellURI username password authType useSSL skipCACheck skipCACheck skipCACheck skipCACheck port keepSessionAlik enablePiping runspaceID modules snapins snapins cmjet cmdlet	ve	Required Image: Image of the second secon	Type Single Value Single Value	Template Prompt User Prompt User Prompt User Prompt User Prompt User Prompt User Value: Value: Value: Prompt User		
Input host URI shellURI username password authType useSSL skipCACheck skipCACh	VE	Required Image: Imag	Type Single Value Single Value	Template Prompt User Value: Value: Value: Prompt User		

Figure 10: Inputs of the PowerShell Script operation

Connection Inputs

host

The hostname or ip address of the PowerShell host. This input is mutual exclusive with URI and shellURI

Default value: localhost

URI

Specifies a Uniform Resource Identifier (URI) that defines the connection endpoint for the interactive session.

Value format: <Transport>://<ComputerName>:<Port>/<ApplicationName>

Example: http://exch2010CAS1/Powershell?serializationLevel=Full

shellURI

Gets the Uniform Resource Identifier (URI) of the shell that is launched when the connection is made. This input is mutual exclusive with host.

Example: http://schemas.microsoft.com/powershell/Microsoft.Exchange

Username

The username to use when connecting to the server.

Value format: username@domain or domain\username

password

The password to use when connecting to the server

authType

Specifies the mechanism that is used to authenticate the user's credentials. Valid values: Default, Basic, Credssp, Digest, Kerberos, Negotiate, NegotiateWithImplicitCredential. (case-insensitive).

Default value: Default

useSSL

If true, the operation uses the Secure Sockets Layer (SSL) protocol to establish a connection to the remote computer. By default, SSL is not used. WS-Management encrypts all Windows PowerShell content transmitted over the network. UseSSL is an additional protection that sends the data across an HTTPS connection instead of an HTTP connection.

Default port for SSL: 5986.

Default value: false

skipCACheck

Indicates when connecting over HTTPS that the client does not validate that the server certificate is signed by a trusted certificate authority (CA).

Default value: true

skipCNCheck

Indicates whether the certificate common name (CN) of the server and the hostname of the server are not checked for being the same.

Default value: true

port

Specifies the network port on the remote computer that is used for this connection. To connect to a remote computer, the remote computer must be listening on the port that the connection uses. The default ports are 5985 (the WinRM port for HTTP) and 5986 (the WinRM port for HTTPS).

Session and Piping Inputs

keepSessionAlive

If true, the operation will not close the PowerShell runspace (i.e. the PSSession in case of remote connections) created during the execution and returns the runspaceID as the result of the operation. Actually the PowerShell runspace is saved in the OO session and can be used in other operations using the runspaceID result of previous operation.

Default value: false

enablePiping

If true, the operation enables piping the way PowerShell does. This input should be used only when keepSessionAlive input is true. First time the piped objects are null and the scope is to save the objects resulted from the PowerShell script execution. Next time the piped objects can be referred in the script as "\$_" objects.

Default value: false

Example: First, run "Get-Service" script with enablePiping=true and keepSessionAlive=true, then run the script "Select-Object -property name,displayname,status | where-object {\$_.Name -lt "C"}" with enablePiping=true keepSessionAlive=true. Observe the use of "\$_".

runspaceID

If this is not empty, the operation searches the OO session for keys which equal the runspaceID. If the operation finds the runspace specified by the runspaceID it tries to use it and does not create another PowerShell runspace with provided inputs. If the runspace exists but is broken or unavailable the operation uses its authentication parameters and tries to reconnect and recreate the PowerShell runspace.

Note: The operation uses the connection parameters of the runspace identified by runspaceID, not the values provided as user inputs. The operation processes the connection inputs, for example, host, username, password, authType, useSSL that is provided by the user only if the runspaceID does not exist in the OO session.

Additional Modules and Snapins

Modules

A list of PowerShell modules that is loaded after the PowerShell connection is established. Each value from the list specifies the name of the module to import. Enter the name of the module or the name of a file in the module, such as a .psd1, .psm1, .dll, or ps1 file. File paths are optional. Wildcards are not permitted. Specify only the module name whenever possible. When you specify a file name, only the members that are implemented in that file are imported. If the module contains other files, they are not imported, and you might be missing important members of the module. The list of modules should be separated by the comma "," delimiter.

Example: FailoverClusters

Snapins

A list of PowerShell snapins loaded after the PowerShell connection is established. Each value from the list specifies the name of a registered snapin, for example, the Name, not the AssemblyName or ModuleName.

Example: Microsoft.Exchange.Management.PowerShell.E2010

PowerShell Script and cmdlet Inputs

script

The script to execute on the PowerShell host. If you want to execute a script from a file just provide the file path.

Example: C:\PowerShellScripts\GetHost.ps1

Cmdlet

The name of the PowerShell cmdlet to invoke. If the cmdlet has additional parameters, please provide them as inputs to the operation. If the parameter has the same name as one of the operation's inputs just prefix it with "_". This input is intended to be used together with the PowerShell wizard so please use the script input whenever possible.

Result Formatting

returnTable

If true, the operation will return a table containing a row for each PSObject that the script emits. The table's columns represent the properties of these PSObjects, in the propertyName<delimiter>propertyValue format. If false the operation returns a string representation of the result similar to the output from the PowerShell console.

Default value: false.

delimiter

The delimiter used to separate each property name from the property value in the output table.

Default value: ":".

colDelimiter

The delimiter used to separate columns in the output table.

Default value: ",".

rowDelimiter

The delimiter used to separate rows in the output table.

Default value: newline.

Run a PowerShell Script on a Localhost

The only setting required to execute the PowerShell scripts on the localhost is that the ExecutionPolicy must be RemoteSigned. Use Get-ExecutionPolicy to display the current execution policy and Set-ExecutionPolicy to set the execution policy.

In addition, the required input is the script input.

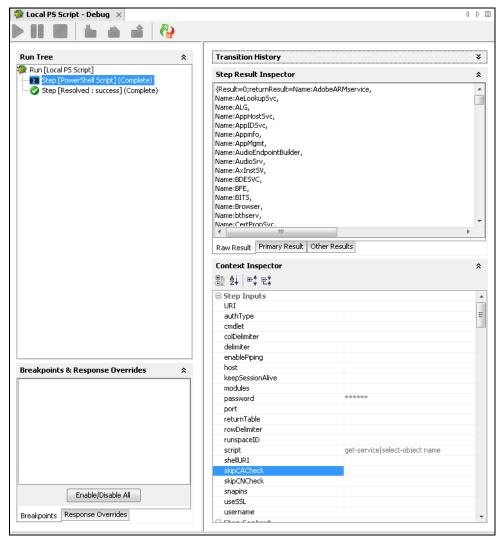


Figure 11: Running the PowerShell Script operation on localhost

If one script requires elevated rights, enter a username and a password.

Run PowerShell Scripts from a File

Run Tree	*	Transition History		¥
Run [Run PS Script from File]		Step Result Inspector		*
Step [Resolved : success] (Complete)		{Result=0;returnResult=		•
		Name : Default Host Version : 2.0		
		InstanceId : 6e68454c-2c98-4462-		
		UI : System.Management.Auto terface	omation.Internal.Host.InternalHostUserIn	E
		CurrentCulture : en-US		=
		CurrentUICulture : en-US		
		PrivateData : IsRunspacePushed :		
		Runspace :		
		returnCode=0:ohiertsCount=1:sessio Kaw Result Primary Result Other R	nId=7a72a3d4-0143-4771-8e33-3hh614e6880d:scrint=Followinn i esults	s the *
		Context Inspector		\$
		B 2↓ ■‡ EX		^
		Step Inputs		*
		URI		E
		authType cmdlet		=
		cmalec colDelimiter		
		delimiter		
		enablePiping		
		host		
Breakpoints & Response Overrides	*	keepSessionAlive		
reakpoints & Response Overrides 🛛 🕆		100pp0000101111110		
		modules		
		modules password	stokostok	
		password	ala ana ana ana ana ana ana ana ana ana	
			***** false	
		password port		
		password port returnTable rowDelimiter		
		password port returnTable rowDelimiter runspaceID	false	
		password port returnTable rowDelimiter		
		password port returnTable rowDelimiter runspaceID script	false	
		password port returnTable rowDelimiter runspaceID script shellURI	false	
Enable /Dirable All		password port returnTable rowDelimiter runspaceID script shellURI skipCACheck	false	
Enable/Disable All		password port returnTable rowDelimiter runspaceID script shellURI skipCACheck skipCNCheck	false	

Figure 12: Running the PowerShell Script operation from a File

To run scripts from a file, just provide the path to that file. This should work in most of the cases, when the path to the file contains whitespaces, the operation fails. To fix this provide the path to that file like in the picture above.

PowerShell Script Error :	E
Inspector	₽
Step Name: PowerShell Script	
Inputs Results Display Description Advanced Scriptlet	
▲ ▼ Inputs Summary > script	↔ X
Name: script Input Type: Single Value	
Name: script Input Type: Single Value	Otherwise: Use Constant' Configuration
Name: script Input Type: Single Value Input Data Flow Assign from Variable: script	Otherwise: Use Constant' Configuration Constant Value:
Name: script Input Type: Single Value Input Data Flow Assign from Variable: script Otherwise: Use Constant	Otherwise: Use Constant' Configuration
Name: script Input Type: Single Value Input Data Flow Assign from Variable: script	Otherwise: Use Constant' Configuration Constant Value: . "C:\Scripts\PS Tests\GetParameter.ps1"

Load PowerShell Functions from Files

Figure 13: Using a script to enable all functions from a local file

In certain cases, a PowerShell script depends on functions from other file from the disk. The script from the picture above might help to load this file and enable all functions and cmdlets from it. Get-Parameter cmdlet is defined in the file named Get-Parameter.ps1.

Run a PowerShell Script on a Remote Host

PS Script - Flow 🗙 🦃 PS Script - I			4
Run Tree	\$	Transition History	
2		Step Result Inspector	
Run [PS Script] Step [PowerShell Script] (Complete) Step [Resolved : success] (Complete)	Result=0;returnResult=Name:AeLookup5vc,RequiredServices;,CanPauseAndContinue:False,CanSh Name:ALG,RequiredServices;,CanPauseAndContinue:True Name:AppHost5vc,RequiredServices;System.ServiceProcess.ServiceC Name:AppID5vc,RequiredServices;System.ServiceProcess.ServiceC Name:AppID5vc,RequiredServices;CanPauseAndContinue:True Name:AppID5vc,RequiredServices;System.ServiceProcess.ServiceC Name:AppID5vc,RequiredServices;CanPauseAndContinue:False,C Name:AppID5vc,RequiredServices;CanPauseAndContinue:False,C Name:AudioEndpointBuilder,RequiredServices;System.ServiceProcess.ServiceCont Name:BEF_RequiredServices;System.ServiceProcess.ServiceCont Name:BITS,RequiredServices;System.ServiceProcess.ServiceCont Name:CertPropSvc,RequiredServices;CanPauseAndContinue:True,Ca Name:CertSvc,RequiredServices;CanPauseAndContinue:True,Ca Name:CertSvc,RequiredServices;CanPauseAndContinue:True,Ca Name:CertSvc,RequiredServices;CanPauseAndContinue:True,Ca Name:CertSvc,RequiredServices;CanPauseAndContinue:True,Ca Name:CertSvc,RequiredServices;CanPauseAndContinue:True,Ca Name:CertSvc,RequiredServices;CanPauseAndContinue:True,Ca Name:CertSvc,RequiredServices;CanPauseAndContinue:True,Ca Name:CertSvc,RequiredServices;CanPauseAndContinue:True,Ca Raw Result Primary Result Other Results	hutdown:False, Ca e, CanShutdown:T Scontroller System ontroller System. CanShutdown:Fal Ise, CanShutdown ress. ServiceContr Controller System Controller System. ServiceController, Ser controller System. viceController, Car inShutdown:True, SeAndContinue:F	
		🗆 Step Inputs	
		URI	
		authType	
		cmdlet	
		colDelimiter	
		colDelimiter delimiter	
reakpoints & Response Overrides	*	delimiter	
eakpoints & Response Overrides	*	delimiter enablePiping	
reakpoints & Response Overrides	*	delimiter enablePiping host exch2010CAS1	
eakpoints & Response Overrides	*	delimiter enablePiping host exch2010CA51 keepSessionAlive	
reakpoints & Response Overrides	*	delimiter enablePiping host exch2010CAS1 keepSessionAlive modules	
reakpoints & Response Overrides	*	delimiter enablePiping host exch2010CAS1 keepSessionAlive modules password *******	
reakpoints & Response Overrides	*	delimiter enablePiping host exch2010CAS1 keepSessionAlive modules password ****** port	
reakpoints & Response Overrides	*	delimiter enablePiping host exch2010CAS1 keepSessionAlive modules password port returnTable rowDelimiter	
reakpoints & Response Overrides	*	delimiter enablePiping host exch2010CAS1 keepSessionAlive modules password ******* port returnTable rowDelimiter runspaceID	
reakpoints & Response Overrides	*	delimiter enablePiping host exch2010CAS1 keepSessionAlive modules password ******* port returnTable rowDelimiter runspaceID	
reakpoints & Response Overrides	*	delimiter enablePiping host exch2010CAS1 keepSessionAlive modules password ************************************	
reakpoints & Response Overrides	*	delimiter enablePiping host exch2010CAS1 keepSessionAlive modules password ******* port returnTable rowDelimiter runspaceID script Get-Service skipCACheck	
	\$	delimiter enablePiping host exch2010CAS1 keepSessionAlive modules password password port returnTable rowDelimiter runspaceID script Get-Service skipCACheck skipCNCheck	
eakpoints & Response Overrides	*	delimiter enablePiping host exch2010CAS1 keepSessionAlive modules password password returnTable rowDelimiter runspaceID script Get-Service shellURI skipCACheck skipCNCheck snapins	
	*	delimiter enablePiping host exch2010CAS1 keepSessionAlive modules password password port returnTable rowDelimiter runspaceID script Get-Service skipCACheck skipCNCheck	

Figure 14: Running the PowerShell Script operation on a remote host

First, execute the script providing only the required inputs: host, username, password and the script.

Negotiate is the default authentication type if the host is provided as an IP address and Kerberos for host names. Please refer to PowerShell Remoting and Authentication Types if the connection fails.

Format the Result

The result can be formatted as a table or the same way it will be displayed in the PowerShell console. The format is decided by the returnTable input described above. If returnTable is set to false the operation will return the result as in the PowerShell console and like the old operation.

Step Result Inspector	5
{Result=0;returnResult=	•
Status Name DisplayName	٦
Running AdobeARMservice Adobe Acrobat Update Service	
Stopped AeLookupSvc Application Experience	
Stopped ALG Application Layer Gateway Service	-
Running AppHostSvc Application Host Helper Service	
Stopped AppIDSvc Application Identity	
Stopped Appinfo Application Information	
Stopped AppMgmt Application Management	T
Running AudioEndpointBu Windows Audio Endpoint Builder	
Running AudioSrv Windows Audio	
Stopped AxInstSV ActiveX Installer (AxInstSV)	
Stopped BDESVC BitLocker Drive Encryption Service	
Running BFE Base Filtering Engine	
Running BITS Background Intelligent Transfer Ser	
Running Browser Computer Browser	
Stopped bthserv Bluetooth Support Service	
Running CertPropSvc Certificate Propagation	
Stopped_clr_optimizatio Microsoft .NET Framework NGEN v2.0	
Stopped_clr_optimizatio Microsoft .NET Framework NGEN v2.0	
Stopped_clr_optimizatio Microsoft .NET Framework NGEN v4.0	
Stopped_clr_optimizatio Microsoft .NET Framework NGEN v4.0	
Stopped COMSysApp COM+ System Application	
Running CryptSvc Cryptographic Services	
Running CscService Offline Files	
Running DcomLaunch DCOM Server Process Launcher	
Stopped defragsvc Disk Defragmenter	
Running Dhcp DHCP Client	
Running Discache DNS Client	
Stopped_dot3svc Wired_AutoConfig	
Running DPS Diagnostic Policy Service	
Stopped EapHost Extensible Authentication Protocol	
Stopped EFS Encrypting File System (EFS)	
Stopped ehRecvr Windows Media Center Receiver Service	
Stopped_ehSched Windows Media Center Scheduler Service	
Running eventlog Windows Event Log	
Running EventSystem COM+ Event System	
Stopped Fax Fax	
Stopped fdPHost Function Discovery Provider Host	٣
4	
Raw Result Primary Result Other Results	

Figure 15: The result if returnTable=false

The result is human readable, but the problem is that it is very difficult to parse, and does not contain properties which could not be displayed on the screen.

The result can be displayed as a table.

PowerShell session considerations results are displayed as a table. Each PowerShell object (in this case each service) is displayed by default on a line. Each line contains different properties of the service (default delimiter is ",") and the key-value pairs are delimited by ":". All these delimiters can be changed, refer to the Inputs section for more information.

For example Get-Service returns the following result:

Step Result Inspector

Jame: AppMgmt, RequiredServices: System. ServiceProcess. ServiceController[], CanPauseAndContinue: False, CanShutdown: False, CanStop: False, DisplayName: Applica	tio -
Jame: AudioEndpointBuilder, RequiredServices: System. ServiceProcess. ServiceController[1, CanPauseAndContinue: False, CanShutdown: False, CanStop: True, DisplayN	ime
Jame: AudioSrv, RequiredServices: System. ServiceProcess, ServiceController[], CanPauseAndContinue: False, CanShutdown: False, CanStop: True, DisplayName: Window	s F
lame: AxInstSV, RequiredServices: System. ServiceProcess. ServiceController 🗍, CanPauseAndContinue: False, CanShutdown: False, CanStop: False, DisplayName: Active:	
ame:BDESVC,RequiredServices:System.ServiceProcess.ServiceController[],CanPauseAndContinue:False,CanShutdown:False,CanStop:False,DisplayName:BitLocke	
ame:BFE, RequiredServices:System.ServiceProcess.ServiceController[], CanPauseAndContinue:False, CanShutdown:False, CanStop:True, DisplayName:Base Filtering	
ame:BITS, RequiredServices:System.ServiceProcess.ServiceController].CanPauseAndContinue:False, CanShutdown:False, CanStop: True, DisplayName:Background	
ame:Browser, RequiredServices:System, ServiceProcess, ServiceController[], CanPauseAndContinue:False, CanShutdown:False, CanStop: True, DisplayName: Compute	
me: bthserv, RequiredServices: System. ServiceProcess. ServiceController []. CanPauseAndContinue: False, CanShutdown: False, CanShutdown; False; False; False; False; False; False; False; False; False; False	
ame:CertPropSvc,RequiredServices:System.ServiceProces.ServiceProrteller] CanPauseAndContinue:False,CanShutdown:True,CanStop:True,DisplayName:Cert	
name: chroppinization v2.0.50727 32.Required Services System. ServiceProcess. ServiceControllerTI. CanPauseAndContinue: False. CanShutdown: False.	
named optimization _20.050727_06, RequiredServices:System.ServiceProcess.ServiceController]. CanPauseAndContinue:False,CanShutdown:False	
nameda optimization _v2.05072_07,PequiedServices:System.ServiceProcess.ServiceController[],CanPauseAndContinue:False,CanShutdowim-False,CanShutdow	
aamada gopiinization 14.0.30319 6 (Required Services:System Service Controller]).Can PauseAnd Continue: False, Can Duddowin False, Can Doptor: Jamedro gopiinization 14.0.30319 6 (Required Services: System Service Controller]).Can PauseAnd Continue: False, Can Shutdowin False, Can Doptor: False	
aana compunization+ roadon s_on period a construction of the second o	
ialie: ConsysApp, Required arvice: System, Service Trocess, Service Controller(), CanPauseAnd Continue: Faise, CanShutdown Fraise, CanSouphymaine: ConsysApp, Required arvice: System, ServiceFroet, Service: System Controller(), CanPauseAnd Continue: Faise, CanShutdown, True, CanSbutdown, Service: System Service: System Service: System Service: System Service: Se	
iamis:crypt.sv, kequiredservices;system.serviceProcess, ServiceController[], CamadowindContinue;Faise, Cambriddown;Faise, Cambriddown;Faise	
iame:scaervice,requiredservices:system.servicervicess.serviceController[],cameuseAndContinue:raise,canshutdown:raise,c	
ame dor agrix, Required services: system, service rocess, service controller(), can addexinuc on interinase, can bruch with raise, can bought raise, big paymaine; controller(), can addexinuc on interinase, can bruch with raise, can bought raise, big paymaine; controller(), can addexinuc on interinase, can bruch with raise, can bought raise, big paymaine; big big paymaine; big big paymaine; big	
lame:Dhcp,RequiredServices:System.ServiceProcess.ServiceController[],CanPauseAndContinue:False,CanShutdown:True,CanStop:True,DisplayName:DHCP Cleared	
lame:Dnscache,RequiredServices:System.ServiceProcess.ServiceController[],CanPauseAndContinue:False,CanShutdown:Fa	
lame:dot3svc,RequiredServices:System.ServiceProcess.ServiceController[],CanPauseAndContinue:False,CanShutdown:False,CanStop:False,DisplayName:Wired A	
lame:DPS,RequiredServices:System.ServiceProcess.ServiceController[],CanPauseAndContinue:False,CanShutdown:True,CanStop:True,DisplayName:Diagnostic Po	
lame:EapHost, RequiredServices: System. ServiceProcess. ServiceController[], CanPauseAndContinue: False, CanShutdown: False, CanStop: False, DisplayName: Extensi	
lame:EFS,RequiredServices:System.ServiceProcess.ServiceController[],CanPauseAndContinue:False,CanShutdown:False,CanStop:False,DisplayName:Encrypting F	
lame:ehRecvr,RequiredServices:System.ServiceProcess.ServiceController[],CanPauseAndContinue:False,CanShutdown:False,CanStop:False,DisplayName:Window	
lame:ehSched,RequiredServices:System.ServiceProcess.ServiceController[],CanPauseAndContinue:False,CanShutdown:False,CanStop:False,DisplayName:Window	
ame:eventlog,RequiredServices:System.ServiceProcess.ServiceController[],CanPauseAndContinue:False,CanShutdown:True,CanStop:True,DisplayName:Window:	
ame:EventSystem, RequiredServices:System.ServiceProcess.ServiceController[], CanPauseAndContinue:False, CanShutdown:False, CanStop:True, DisplayName:CO	
ame:Fax,RequiredServices:System.ServiceProcess.ServiceController[],CanPauseAndContinue:False,CanShutdown:False,CanStop:False,DisplayName:Fax,Depenc	
lame:fdPHost,RequiredServices:System.ServiceProcess.ServiceController[],CanPauseAndContinue:False,CanShutdown:False,CanStop:False,DisplayName:Function	
lame:FDResPub,RequiredServices:System.ServiceProcess.ServiceController[],CanPauseAndContinue:False,CanShutdown:False,CanStop:False,DisplayName:Funct	
lame:FontCache,RequiredServices:System.ServiceProcess.ServiceController[],CanPauseAndContinue:False,CanShutdown:True,CanStop:True,DisplayName:Windo	
lame:FontCache3.0.0.0, RequiredServices:System.ServiceProcess.ServiceController[], CanPauseAndContinue:False,CanShutdown:False,CanStop:False,DisplayNam	
Jame:gpsvc,RequiredServices:System.ServiceProcess.ServiceController[],CanPauseAndContinue:False,CanShutdown:False,CanStop:True,DisplayName:Group Polic	y C
Iame:hidserv,RequiredServices:System.ServiceProcess.ServiceController[],CanPauseAndContinue:False,CanShutdown:False,CanStop:False,DisplayName:Human I	ite
lame:hkmsvc,RequiredServices:System.ServiceProcess.ServiceController[],CanPauseAndContinue:False,CanShutdown:False,CanStop:False,DisplayName:Health K	
Jame:HomeGroupListener,RequiredServices:System.ServiceProcess.ServiceController[],CanPauseAndContinue:False,CanShutdown:False,CanStop:False,DisplayNa	ne
Jame:HomeGroupProvider,RequiredServices:System.ServiceProcess.ServiceController[],CanPauseAndContinue:False,CanShutdown:False,CanStop:False,DisplayNa	me
Jame:idsvc,RequiredServices:System.ServiceProcess.ServiceController[],CanPauseAndContinue:False,CanShutdown:False,CanStop:False,DisplayName:Windows C	arc 🧃
	b

Raw Result Primary Result Other Results

Figure 16: Raw result of the Get-Service flow

The result contains a full list of properties without any additional PowerShell script, for example, Get-Service | fl, Get-Service | Select-Object Status). Therefore, the status of the services appears and can be parsed by writing two types filters on the result.

nspector	₽
tep Name: PowerShell Script	
Inputs Results Display Description Advanced Scriptlet	
Step Results > serviceNames	•
Add Remove 😭 🦺 No Filter Selected	
₩ RegEx: 'Name:[^,]*'	
Replace All Name:' with "	
Test Filter Input	uuuu
Test All Filters Test Selected Filters	📄 🔚 🦐 Clear Quick Command
Name: Adobe&RMservice, RequiredServices: System. Servic alse, CanShutdown: False, CanStop: True, DisplayName: Ado	
alse, CanShutdown: False, CanStop: True, DisplayName: Ado Service, DependentServices: System. ServiceProcess. Ser Mservice, ServicesDependedOn: System. ServiceProcess. S	be Acrobat Update viceController[],MachineName:.,ServiceName:AdobeAR erviceController[],ServiceHandle:SafeServiceHandle 💌
alse, CanShutdown: False, CanStop: True, DisplayName: Ado Service, DependentServices: System. ServiceProcess. Ser Hservice, ServicesDependedOn: System. ServiceProcess. S	be Acrobat Update viceController[],MachineName:.,ServiceName:AdobeAR
alse, CanShutdown: False, CanStop: True, DisplayName: Ado Service, DependentServices: System. ServiceProcess. Ser Mservice, ServicesDependedOn: System. ServiceProcess. S	be Acrobat Update viceController[],MachineName:.,ServiceName:AdobeAR erviceController[],ServiceHandle:SafeServiceHandle 🔻
alse, CanShutdown: False, CanStop: True, DisplayName: Ado Service, DependentServices: System. ServiceProcess. Ser Mservice, ServicesDependedOn: System. ServiceProcess. S Test Output	be Acrobat Update viceController[],MachineName:.,ServiceName:AdobeAR erviceController[],ServiceHandle:SafeServiceHandle 🔻
alse, CanShutdown: False, CanStop: True, DisplayName: Ado Service, DependentServices: System. ServiceProcess. Ser Mservice, ServicesDependedOn: System. ServiceProcess. S Test Output Winnamt	be Acrobat Update viceController[],MachineName:.,ServiceName:AdobeAR erviceController[],ServiceHandle:SafeServiceHandle 🔻
alse, CanShutdown: False, CanStop: True, DisplayName: Ado Service, DependentServices: System. ServiceProcess. Ser Mservice, ServicesDependedOn: System. ServiceProcess. S Test Output Winngmt WinnEM	be Acrobat Update viceController[],MachineName:.,ServiceName:AdobeAR erviceController[],ServiceHandle:SafeServiceHandle 🔻
alse, CanShutdown: False, CanStop: True, DisplayName: Ado Service, DependentServices: System. ServiceProcess. Ser Hservice, ServicesDependedOn: System. ServiceProcess. S Test Output Winngmt WinnfM WinnY	be Acrobat Update viceController[],MachineName:.,ServiceName:AdobeAR erviceController[],ServiceHandle:SafeServiceHandle 🔻
alse, CanShutdown: False, CanStop: True, DisplayName: Ado Service, DependentServices: System. ServiceProcess. Ser Nservice, ServicesDependedOn: System. ServiceProcess. S Test Output Winngmt Winngt Winstr Wilnstvc wmiApSrv WHNetworkSvc WPCSvc	be Acrobat Update viceController[],MachineName:.,ServiceName:AdobeAR erviceController[],ServiceHandle:SafeServiceHandle 🔻
alse, CanShutdown: False, CanStop: True, DisplayName: Ado Service, DependentServices: System. ServiceProcess. Ser Mservice, ServicesDependedOn: System. ServiceProcess. S Test Output Winmgmt Winmym Winmym Winmyc wmiApSrv WHPMetworkSvc	be Acrobat Update viceController[],MachineName:.,ServiceName:AdobeAR erviceController[],ServiceHandle:SafeServiceHandle 🔻
alse, CanShutdown: False, CanStop: True, DisplayName: Ado Service, DependentServices: System. ServiceProcess. Ser Mservice, ServicesDependedOn: System. ServiceProcess. S Test Output Winmgmt WinmPM WinmrM WinmrV wmiApStv WIPNetworkSvc WIPSvc WIPDstusRnum wscsvc	be Acrobat Update viceController[],MachineName:.,ServiceName:AdobeAR erviceController[],ServiceHandle:SafeServiceHandle 🔻
alse, CanShutdown: False, CanStop: True, DisplayName: Ado Service, DependentServices: System. ServiceProcess. Ser Nservice, ServicesDependedOn: System. ServiceProcess. S Test Output Winmgmt Winmymt Winmer WinPM WURSYC WPDetoorkSvc WPDetoorkSvc WPDStor WDStanum wscsvc WSearch	be Acrobat Update viceController[],MachineName:.,ServiceName:AdobeAR erviceController[],ServiceHandle:SafeServiceHandle 🔻
alse, CanShutdown: False, CanStop: True, DisplayName: Ado Service, DependentServices: System. ServiceProcess. Ser Nservice, ServicesDependedOn: System. ServiceProcess. S Test Output Winngmt Winngmt Winsvc wmiApSrv WHNetworkSvc WPCSvc WPDSusEnum wscsvc WSearch WSearch wuauserv	be Acrobat Update viceController[],MachineName:.,ServiceName:AdobeAR erviceController[],ServiceHandle:SafeServiceHandle 🔻
alse, CanShutdown: False, CanStop: True, DisplayName: Ado Service, DependentServices: System. ServiceProcess. Ser Mservice, ServicesDependedOn: System. ServiceProcess. S Test Output Winngmt WinnRM Winnsvc wmiApSxv WMPNetworkSvc WDPNetworkSvc	be Acrobat Update viceController[],MachineName:.,ServiceName:AdobeAR erviceController[],ServiceHandle:SafeServiceHandle 💌
alse, CanShutdown: False, CanStop: True, DisplayName: Ado Service, DependentServices: System. ServiceProcess. Ser Nservice, ServicesDependedOn: System. ServiceProcess. S Test Output Winngmt Winngmt Winsvc wmiApSrv WHNetworkSvc WPCSvc WPDSusEnum wscsvc WSearch WSearch wuauserv	be Acrobat Update viceController[],MachineName:.,ServiceName:AdobeAR erviceController[],ServiceHandle:SafeServiceHandle 🔻

Figure 17: Filtering the results

\$

Run Multiple PowerShell Cmdlets Scripts in the Same PowerShell Session

This section explains how to run multiple PowerShell Script steps in the same PowerShell session. When the PowerShell Script has to execute a single script on a remote server keeping the sessions alive is not necessary. The PowerShell Script connects to the remote host, creates a new PowerShell Runspace, for example, a new PowerShell session, runs the full script on the target and closes the runspace and the connection.

If you want to use the PowerShell Wizard and run the generated flows in a different sequence, you need to consider how much effort is required from the user and whether additional PowerShell Scripts need to be created in order to general flows. However, there are situations when you want to keep the PowerShell session alive.

For example, one uses the PowerShell Wizard, connects to a host which has PowerCLI installed on it and follows the wizard steps to generate OO flows for the PowerShell cmdlets to execute VMWare tasks. Suppose that after the wizard finishes, the user wants to execute one simple cmdlet like Get VM.

To run cmdlets and keep the sessions alive:

1 Run the OOTB Get Datacenter flow generated with the PowerShell Wizard.

Note: This cmdlet does not have any required inputs, however the flow fails to run as you need to run the **Connect VIServer** cmdlet. The PowerShell Wizard generates this flow to solve this problem.

- 2 Create a flow sequence as shown below.
 - Do not modify the generated flow.
 - The flow tries to execute Connect VIServer before Get VM.

In the flow below, the parameters specific to the cmdlet were added as flow inputs. The names appear in capital letters.

ository - CLEMENS - E:/WORKSPACES/CLEMENS/Repositories) 🔲 🗗 🍕		S PS Waard WWsample - Debug X	4
Get Alarm Definition			
Get Annotation	* 🔚 🖶 🇞 🦈 🍃 🗎	💼 🗶 🔊 🖓 🔊 🖉 🕶 🔄 🕶	
- Bet CODrive			
Get Ouster		- 214	
Get Custom Attribute	Sector Sector		
9 Gel Datacenter	Caller .	-307	
- P Get Datastore		2	Dist
- 🤔 Get Datastore Cluster	Connect ViServer	Databenter	Resolu
- 🧬 Get Deploy Rule	Viberver	Datapemer	succe
- 🧬 Get Deploy Rule Set			
Get Drs Recommendation	×		
- 🧬 Get Drs Rule			
- 🧐 Get Error Report	C.7	2.9	
- 🧬 Get Esx Cl			
Get Esx Image Profile	Error:	Error:	
Get Esx Software Channel	failure	failure	
- 🧟 Get Esx Software Package	1 x	17	
- P Get Esx Top - Get Floopy Drive	and the second se		
Get Folder	Inspector		
Get HAPrimary WHost	Step Name: Connect VIServer		
Get Hard Dick			
Get Inventory	Inputs Results Display Description Ad	vanced Scriptlet	
- B Get ISosi Hba Target	+ * Inputs Summary		
🧐 Get License Data Manager			
9 Get Log	Add Input	Remove brout 🛛 🛊 🕹	
- Get Log Type			
Get Network Adapter	Assign To Input	Required Type From	
- 🔗 Get Nic Teaming Policy	shell.RI	Single Value _ Prompt User	1
Get OSCustomization Nic Mapping	enablePiping	Single Value Prompt User	1
- 🧬 Get OSCustomization Spec	authType	Single Value 🖕 Prompt User	1
- 🧐 Get Passthrough Device	slipCACheck	Single Value 🖕 Prompt User	12
- 🧟 Get Power QLIConfiguration	modules	Single Value 🖕 Prompt User	
- 🧬 Get Power CLIVersion	cmdlet	Single Value 🖕 Value: Connect-VIServer	16
- 🧟 Get Resource Pool	keepSessionAllve	Single Value 🖕 Value: true	6
- 🧟 Get Scsi Controller	script	Single Value 🖕 Prompt User	19
- 🥵 Get Scollun	use59,	Single Value 🖕 Prompt User	8
- 🧬 Get Scsilun Path	deleniter	Single Value 🖌 Prompt User	6
- 🧬 Get Snapshot	Server	Single Value 🕌 Value: 16.53.236.24	5
Get Stat	Port	Single Yalue 🖕 Prompt User	
Get Stat Interval	Protocol	Single Value 🖕 Prompt User	
- 🚱 Get Stat Type - 🧐 Get Tark	Credential	Single Value Prompt User	1
Get Task	User	Single Value: Value: Administrator Single Value: Value: ************************************	6
Get Usb Device	Password		5
- 9 Get Usb Device	Session NotCet aut	Single Value 🖕 Prompt User	
- 9 Get ViAccourt	NotDefault SaveCredentials	Single Value	
Get VIAccourt Get VICredential Store Item	AlLinked	Single Value Prompt User	K
	Maru	Single Value Prompt User Single Value Prompt User	10
Get VIEvent			

Figure 18: Adding values to specific parameters of the cmdlet

The user runs the flow and provides inputs for the server, user name and password (Connect VIServer). No inputs are required for the Get Datacenter. The Connect VIServer succeeds, however, the Get Datacenter fails.

The Get Datacenter fails for the following reasons:

- Connect VIServer passes successfully and the connection to the VMWare server was established.
- Connect VIServer created a new PowerShell runspace, for example, PowerShell session and executed the cmdlet which established a valid connection to the server, however the runspace is closed after the flow runs and the connection is lost.
- Get Datacenter flow creates another PowerShell runspace which is different from the one created by Connect VIServer flow. Therefore, Get-Datacenter cmdlet fails.
- 3 The solution to the previous step is to keep the session alive during the execution of the 2 cmdlets.

To do this:

keepSessionAlive=true for the first flow which is Connect VIServer; the runspaceID must be added to the results of the Connect VIServer flow;

📕 🖶 🇞 🐲	🛶 🖻 💼 🏏		_	
	o 🔤 📖 🤛	VEN (74) 🐼 🔻	<u> </u>	
Connect VIServer	Success	et Data senter	success.	Resolved : success
failure		failure		
		III		•
nspector				₽
ep Name: Connect VIServe	er			
Inputs Results Display	Description Advanced Scri	ptlet		\leftrightarrow ×
			Add Result	Remove Result
Name	From	Assign To	Assignment Action	Filters
unspaceID	Result Field: runspaceID	Flow Variable	OVERWRITE	No Filters 🐼

Design Properties

Figure 19: Adding a result to the Connect VIServer flow

4 **runspaceID** of the **Get Datacenter** flow must get its value from the result of the **Connect VIServer**. This happens automatically because runspaceID input assigns its value from the flow variable.

At this point the flow completes successfully.

Run Tree 🏾 🕆	Transition History	¥
Run [PS Wizard VMWare sample]	Step Result Inspector	\$
Step [Connect VIServer] (Complete) Step [PowerShell Script] (Complete) Step [Resolved : success] (Complete) Step [Get Datacenter] (Complete) Step [PowerShell Script] (Complete) Step [Resolved : success] (Complete) Step [Resolved : success] (Complete)	{returnResult= Name Id OOdatacenter Datacenter-datacenter-21 OOdatacenter Datacenter-datacenter-21 ;objectsCount=1;script=Following is the PowerShell script executed on the target ho \$GetDatacenter=Get-Datacenter -Server 16.53.236.24 \$GetDatacenter;warnings=;FailureMessage=;TimedOut=;Result=	* III
	Name Id OOdatacenter Datacenter-datacenter-21 OOdatacenter Datacenter-21	•
	Raw Result Primary Result Other Results	
	Context Inspector : ﷺ ∱↓ □□☆ Ͳ≵	*
	Step Inputs Cluster Id Id Location Name NoRecursion Server 16.53.236.24	* III
Breakpoints & Response Overrides 🛛 🛠	URI VM VMHost authType cmdlet colDelimiter	
	delimiter enablePiping host exch2010CA51 keepSessionAlive modules	
Enable/Disable All Breakpoints Response Overrides	password ******* port	

Figure 20: The execution of the PSWizard VMWare sample flow

Assign the Result of One Cmdlet as a Parameter to Another Cmdlet

Most of the "get" flows generated through the PowerShell Wizard should work OOTB with minimum effort from the user. But there are some cmdlets, probably the "new" cmdlets, which require as parameters the result of another cmdlet. For example, one would like to create a new virtual machine using the generated flow New VM.

Even if the user follows all the steps described in the previous section, the flow can not be executed. The PowerShell Wizard generates the flow, but the user can run OOTB flows only if their parameters have a built-in type (e.g. strings or integers). In case of cmdlets like Get-Help the parameters can be passed as strings (e.g. the name of the cmdlet to search for help information). The New VM flow parameters are below:

Repository - CLEMENS - E:\WORKSPACES\CLEMENS\Repositories	\$ □ ∂ ₽	🧬 New VM - Flow 🛛 🗙				4 0 0
- 🥐 Mount Tools	*	📕 🛃 🇞 🗶				
		📕 🛃 🇞				
Move Datacenter		Name: New VM				-
		UUID: aa500a2d-5624-40ca-				
Move Inventory		UUID: aa500a2d-5624-40ca-	bd2r-be4dcrd3a	09d Version: 1	l (01/16/12 08:19 adm	un) (205
Move Resource Pool		Assign Categories:				5
Move Templace		Inputs Outputs Response		Scriptlet Läduppee	al	
Move VM			es Description	Scriptiet Advance	u	
Move VMHost		A 🔻 Inputs Summary				$ + + \times $
New Advanced Setting						
New Alarm Action		Inputs				
- 🖉 New Alarm Action Trigger			Add Input	Remove Input		
- Provide America - Provide - Provid						
🖉 New Cluster		Input	Required		Template	
🖉 New Custom Attribute		rowDelimiter		Single Value	Prompt User	-
- 🧬 New Custom Field		rowDelimiter		Single Value 🖕 Single Value 🖕		
🖉 New Datacenter		username		Single Value		
🖉 New Datastore		URI		Single Value 🖕		
		port		Single Value		
New Drs Rule		skipCNCheck		Single Value		
New Esx Image Profile New Floppy Drive		colDelimiter		Single Value		
Mew Floppy Drive		runspaceID		Single Value 🖕		E
New Hard Disk		host		Single Value 🛛 🚽	Prompt User	E
🖉 New IScsi Hba Target		snapins		Single Value 🛛 💂	Prompt User	E
New Network Adapter		shellURI		Single Value 🛛 🚽		E
New OSCustomization Nic Mapping		enablePiping		Single Value 🛛 🚽		<u>s</u>
New OSCustomization Spec		authType		Single Value 🖕		<u>ea</u> -
	-	skipCACheck		Single Value 🖕		<u>ee</u>
- 🛷 New Scsi Controller	-	modules		Single Value 🖕		<u>e</u>
🕐 New Snapshot		cmalet keepSessionAlive		Single Value 🖕		
🐨 New Stat Interval		script				
🛷 New Template		useSSL		Single Value 🚽		
		delimiter		Single Value 🖕		
🖉 New VICredential Store Item		AdvancedOption		Single Value	Prompt User	
New VIPermission		VMHost			Prompt User	
New VIProperty		Version			Prompt User	S
New VIRole New Virtual Port Group		Name		Single Value 🖕		
New Virtual Switch		ResourcePool		Single Value 🖕	Prompt User	E
New VM		VApp		Single Value 🛛 🖕	Prompt User	
New VMGuest Route		Location		Single Value 🛛 💂		<u>e</u>
New VMddest Rodde		Datastore		Single Value 🛛 🚽		<u>s</u>
New VMHost Network Adapter		DiskMB		Single Value 🛛 🚽		<u>2</u>
New VMHost Profile		DiskPath		Single Value 🖕		<u>s</u>
🖉 New VMHost Route		DiskStorageFormat		Single Value 🖕		
- Remove Advanced Setting		MemoryMB		Single Value 🖕		<u>e</u>
Remove Alarm Action		NumCpu		Single Value 🖕		
Remove Alarm Action Trigger		Floppy Design Properties		Single Value 🖕	Prompt User	E

Figure 21: The inputs of the New VM flow

New VM has an input named VMHost. What is the type of this parameter?

The description of the generated flows contains information about the PowerShell cmdlet, but from size reasons and other considerations we could not include the full description of the cmdlet as it is displayed when someone executes Get-Help New-VM –full. The description of the operation contains the original link where the user can find detailed information about the cmdlet.

Edit Tools Repository Window Help Repository - CLEMENS - E:\WORKSPACES\CLEMENS\Repositories\ 디 다	Д	New VM - Flow X	>
Mount Tools			
Move Cluster			
Move Datacenter			
- Move Folder		Name: New VM	3
Move Inventory		UUID: aa500a2d-5624-40ca-bd2f-be4dcfd3a09d Version: 1 (01/16/12 08:19 admin)	2
Move Resource Pool			5
Move Template		Assign Categories:	
Move VApp		Inputs Outputs Responses Description Scriptlet Advanced	
- 🖉 Move VM		<pre></pre>	T.
- 🔗 Move VMHost		Creates a new virtual machine with the provided parameters. The network adapter and the SCSI	É
- 🖉 New Advanced Setting		adapter of the new virtual machine are created of the recommended type for the OS that is specified by	L
- 🔗 New Alarm Action		the GuestId parameter. If the OSCustomizationSpec parameter is used, the virtual machine is	
- 🛷 New Alarm Action Trigger		customized according to the spec. You must specify values for at least one of the ResourcePool,	
- 🛷 New CDDrive		VMHost, and VApp parameters.	L
- 🖉 New Cluster		uri : http://www.vmware.com/support/developer/PowerCLI/PowerCLI50/html/New-VM.html	
- 🧬 New Custom Attribute		Inputs:	
- 🛷 New Custom Field		password - The password to use when connecting to the server.	
- 🕐 New Datacenter		rowDelimiter - The delimiter used to separate rows in the output table.Default value: newline.	L
🖅 New Datastore		returnTable - If true, the operation will return a table containing a row for each PSObject that the	1
- 🧬 New Deploy Rule		script emits. The table's columns represent the properties of these PSObjects, in the	L
- 🛷 New Drs Rule		propertyName <delimiter>propertyValue format. If false the operation returns a string representation of the result similar to the output from the PowerShell console.Default value: true</delimiter>	L
- 🛷 New Esx Image Profile		username - The username to use when connecting to the server. Value format: username@domain or	L
- 🧬 New Floppy Drive		domain/username.	L
- 🖑 New Folder		URI - Specifies a Uniform Resource Identifier (URI) that defines the connection endpoint for the	L
- 🧬 New Hard Disk		interactive session. Value format:	L
		<transport>://<computername>:<port>/<applicationname>Example:</applicationname></port></computername></transport>	L
		http://exch2010CA51/Powershell?serializationLevel=Full port - Specifies the network port on the remote computer that is used for this connection. To connect	L
- 🧬 New OSCustomization Nic Mapping		to a remote computer, the remote computer must be listening on the port that the connection uses. The	L
Wew OSCustomization Spec		default ports are 5985 (the WinRM port for HTTP) and 5986 (the WinRM port for HTTP5).	
	Ξ	skipCNCheck - Indicates whether the certificate common name (CN) of the server and the hostname	
		of the server are not checked for being the same.Default value: true	
- Providence - Pro		colDelimiter - The delimiter used to separate columns in the output table.Default value: ",".	
New Template		runspaceID - If not empty, the operation will search the OO session for keys which equal the runspaceID. If the operation founds the runspace specified by the runspaceID it tries to use it and does	
New Template		not create another PowerShell runspace with provided inputs. If the runspace exists but is broken or	
New VApp New VICredential Store Item		unavailable the operation uses its authentication parameters and tries to reconnect and recreate the	
New VICredential Store Item Mew VIPermission		PowerShell runspace. Please note that the operation will use the connection parameters of the runspace	
New VIPermission		identified by runspaceID, not the values provided as user inputs. The operation will take into	
		consideration the connection inputs (i.e. host, username, password, authType, useSSL, etc.) provided by user only if the runspaceID does not exist in the OO session.	
New Virtual Port Group		host - The hostname or ip address of the PowerShell host. This input is mutual exclusive with URI and	
New Virtual Switch		shellURI.Default value: localhost	
New VM		snapins - A list of PowerShell snapins to load after the PowerShell connection is established. Each	
New VMGuest Route		value from the list specifies the name of a registered snapin (i.e. the Name, not the AssemblyName or	
- Providest Route		ModuleName).Example: Microsoft.Exchange.Management.PowerShell.E2010	
New VMHost Account Mew VMHost Network Adapter		shellURI - Gets the Uniform Resource Identifier (URI) of the shell that is launched when the	
New VMHost Profile		connection is made. This input is mutual exclusive with host.Example: http://schemas.microsoft.com/powershell/Microsoft.Exchange	
Mew VMHost Profile		enablePiping - If true the operatin enables piping the way PowerShell does. This input should be used	
Premove Advanced Setting		only when keepSessionAlive input is true. First time the piped objects are null and the scope is to save	
Remove Advanced Secting		the objects resulted from the PowerShell script execution. Next time the piped objects can be refered in	
Premove Alarm Action Trigger		the script as "\$_" objects.Default value: falseExample: First, run "Get-Service" script with	1
Remove CDDrive	-	Design Properties	

Figure 22: The description of the New VM flow

If the user opens the link in a browser he can observe the types of the cmdlet parameters. In our case, VMHost is of the same type as the name suggests.

Parameters	
NAME	түре
VMHost	VMHost
Template	<u>Template</u>
AdvancedOption	<u>AdvancedOption[]</u>
AlternateGuestName	String
CD	SwitchParameter
Confirm	SwitchParameter
Datastore	StorageResource

Figure 23: The VMHost object: parameters

The problem is that VMHost is an object, not a built-in type which can be resolved as a string. Following the link provided for the VMHost type the user can find which cmdlets return VMHost objects as it can be observed below. In our case the type suggests that a cmdlet like Get-VMHost would return this kind of objects.



Figure 24: The VMHost object : additional information

In the previous section, we explained how to execute in the same PowerShell session multiple OO flows generated with the PowerShell wizard. At this point we can imagine the following chain of cmdlets which need to be executed to create a new vm:

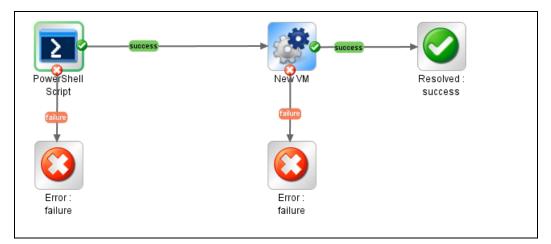
Connect-VIServer - this must be executed before any VMWare cmdlet;

Get-VMHost – we need the result of this cmdlet as parameter for the next cmdlet;

New-VM - this cmdlet actually creates a new virtual machine.

New VM has other parameters beside VMHost which are not built-in, but we are going to explain how to solve the VMHost parameter, because the process is the same for the other parameters, too.

You need to execute 3 cmdlets in the same PowerShell session. You have generated flows for each of the cmdlets and executed them in the same session. The next step is to take the result of the Get-VMHost cmdlet and pass it to the New-VM cmdlet? Select from one of the following solutions:



Solution 1: Create a New PowerShell Script Step

Figure 25: Add a PowerShell script step to the flow

Create a new PowerShell script in addition to the generated flow that you want to run. If you keep the session alive during the execution of the 2 PowerShell script steps, you can use the PowerShell script variables defined in the first step to pass them in the script of the second step or as parameters for the generated flow. In this case, you are not using the generated flows for Connect-VIServer and Get-VMHost, however you need to write the script.

1 Execute the following script, then save the result of the Get-VMHost cmdlet in the PowerShell variable named \$vmHost.

nspector		
tep Name: PowerSh Inputs Results D	nell Script Visplay Description Advanced Scriptlet	
 The second second	ary > script	() >
Name: script	Input Type: Single Value	•
Input Data Flow		'Otherwise: Use Constant' Configuration
Assign from Variable	e: script 👻	Constant Value:
y Otherwise	e: Use Constant 🔹	Connect-VIServer -Server server -User user -Password \${password} \$vmHost = Get-VMHost vmHostName
Assign to Variable:	script 👻	
Input Properties Validation Format:	Encrypted Required rot validated>	
Record Under:	<run history=""></run>	

Figure 26: Use a variable to save the result of the step

2 The next step is to assign the value of the VMHost input from the \$vmHost variable.

PowerShell Page 10	Example - Flow $ imes$				4 ▷ ≣
	🦃 ŀ 💼	🎽 🔊 🔍 🕺) ▼ 🔍 ▼		
sı C	success sipt	¢		Resolved : success	E
Inspector		E	101.		Ŧ
Step Name: New VM					
Inputs Results D	isplay Description Advanced	Scriptlet			
🔺 👻 Inputs Summ	ary > ¥MHost				4 ⊨ ×
Name: VMHost		Input Type: Single Value	•		
Input Data Flow			'Otherwise: Use Const	tant' Configuration	
Assign from Variabl	e: VMHost	•	Constant Value:		
🦕 Otherwis	e: Use Constant	▼	\$vmHost		
Assign to Variable:	VMHost	-			
Input Properties					
	Encryp	oted 📃 Required			
Validation Format:	<not validated=""></not>	-			
Record Under:	<run history=""></run>	▼]			

Figure 27: Assign the variable to the VMHost input of the flow

This way we managed to pass PowerShell cmdlets results between OO flows generated with the PowerShell wizard.

Note: The \$var refers to PowerShell variables and \${var} refers to OO flow variables.

Solution 2: Run a PowerShell Script in the Generated Flow Context

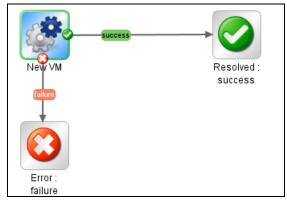


Figure 28: The NewVM Flow

The second solution is to execute the previous defined script in the context of the generated flow. The PowerShell script operation has two inputs which build the script that is going to be executed:

Script – a PowerShell script to execute on target host;

Cmdlet – the PowerShell cmdlet name. If the script input is not empty, than the PowerShell script defined by this input is going to be executed before the cmdlet. Although the script is executed in the same PowerShell runspace with no extra settings.

The solution is shown in the following flow:

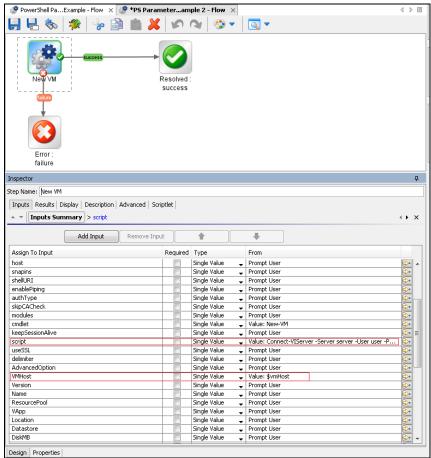


Figure 29: Add a script to the input of the flow

Solution 3: Use Generated Flows Only and Minimize the User Effort

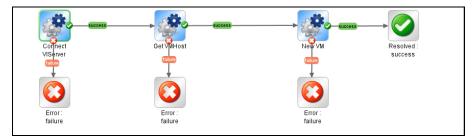


Figure 30: Using only generated flows

The result of each PowerShell cmdlet executed from a generated flow is saved in a PowerShell variable with the same name as the cmdlet, for example, for Get-VMHost the variable is \$GetVMHost.

Without writing any PowerShell script, the user can execute Get VMHost generated flow and know that the result of this cmdlet is saved in the \$GetVMHost variable. Pass the variable to the VMHost input of the New VM flow.

5 PowerShell Remoting

Overview

Enable PowerShell remoting, by running the following cmdlet: Enable-PSRemoting.

In workgroup environments, enable classic mode authentication for network logons. This can be performed by opening the "Local Security Policy" from Control Panel and selecting "Administrative Tools". Navigate to "Local Policies" \rightarrow "Security Options", double-click "Network Access: Sharing and Security Model for local accounts" and set it to classic.

Modify the WSMan trusted hosts setting, by adding the IP addresses of all remote clients to the list of trusted hosts. This can be done using one of the following commands:

Set-item wsman:localhost\client\trustedhosts -value * (adds all computers as trusted hosts)

Set-item wsman:localhost\client\trustedhosts -value Computer (only adds Computer to the trusted hosts)

*Set-item wsman:localhost\client\trustedhosts -value *.domain.com* (adds all computers in the specified domain)

Set-item wsman:localhost\client\trustedhosts -value 10.10.10.1 (adds the remote computer with the IP address 10.10.10.1 to the trusted hosts list).

Enable Remoting Using GPO (Group Policy Objects)

While remoting can be enabled manually using Enable-PSRemoting, it is recommended to use GPO management tools whenever it is possible. Use GPO to apply policies on a single host (i.e. the target PowerShell host) or a group of servers.

Group Policy Configuration for a Single Host

To enable PowerShell remoting for a single host execute the following steps:

- 1 Open Group Policy Management console, for example, gpedit.msc.
- 2 Go to Local Computer Policy Computer Configuration Administrative Templates Windows Components.

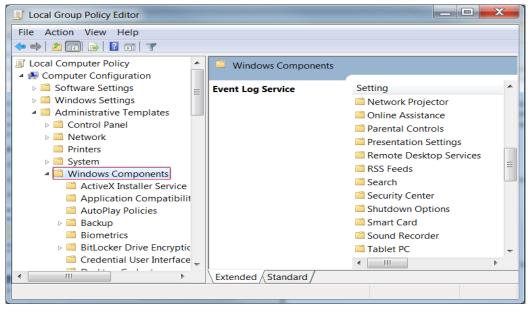


Figure 31: Group Policy Configuration : Windows components section

3 From the Windows Components listed in the right pane, we are interested in two of them. The first one is Windows Remote Management (WinRM) and the second one is Windows Remote Shell. The next step will refer to the first one, therefore browse the components and open Windows Remote Management (WinRM).

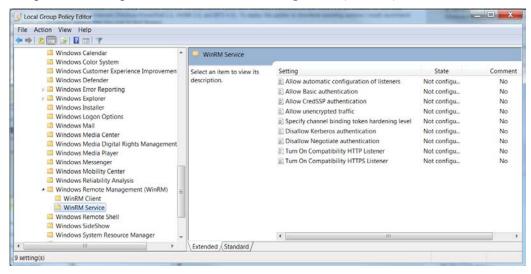


Figure 32: Group Policy Configuration : WinRM Service section

- 4 WinRM is the service which PowerShell uses for remote sessions. WinRM can be configured as client or service, depending on the role the host is going to have in a PowerShell connection (i.e. request access to execute scripts on other hosts or allow other hosts to execute scripts on the current host). At this point the user can enable different authentication types, specify the trusted hosts, enable HTTP or HTTPS listeners, etc.
- 5 There are some other default settings the user might want to change in a production environment. These settings can be found in Windows Remote Shell as shown below:

file Action View Help				
• • 200 3 1 0 1				
Windows Calendar Windows Color System	Windows Remote Shell	-		
Windows Customer Experience Improvemen	Select an item to view its	Setting	State	Commen
Windows Defender	description.	Allow Remote Shell Access	Not configu_	No
Windows Error Reporting		I Specify idle Timeout	Not configu_	No
Windows Explorer		MaxConcurrentUsers	Not configu_	No
Windows Installer		E Specify maximum amount of memory in MB pe	Not configu	No
Windows Logon Options		E Specify maximum number of processes per Shell	Not configu-	No
Windows Mail		Specify maximum number of remote shells per	Not configu	No
Windows Media Center			Not configu_	No
Windows Media Digital Rights Management Windows Media Player				
Windows Messenger				
Windows Messenger				
Windows Reliability Analysis				
Windows Remote Management (WinRM)				
Windows Remote Shell				
Windows SideShow				
Windows System Resource Manager				
Windows Update	1			
🖏 All Settings		۰ m		_
	Extended Standard /			

Figure 33: Group Policy Configuration : Windows Remote Shell section

- 6 You can specify the maximum number of remote shells per user (default is 2) or the maximum amount of memory in MB for shell (default is 150).
- 7 After configuring the GPO, you might need to restart the computer in order to apply the policies or try to run the command gpupdate.

Group Policy Configuration for a Group of Servers

Sometimes the GPO policies must be applied on multiple server hosts and repeating the above steps on every server might not be the best solution. Therefore, the user can create a new GPO policy, configure it and apply it on a list of servers.

- 1 Go to the domain controller or on a server where gpmc.msc is available and open it.
- 2 Right click the Group Policy Object item as in the following picture and choose New. Fill in the name for the new GPO and select the policy to inherit from, then go to the next step.

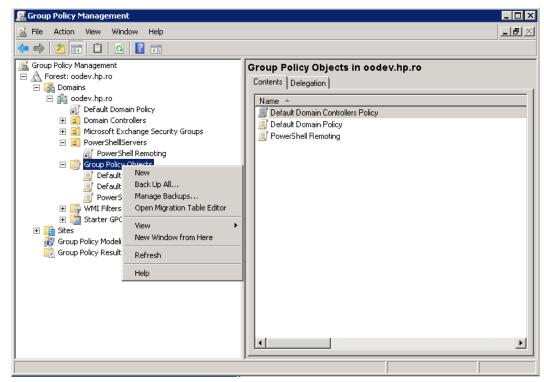


Figure 34: Using Group Policy Objects - step 1

3 Right-click the new GPO and select **Edit**.

🛃 Group Policy Management	
File Action View Window Help	_ B ×
🗢 🔿 💋 💼 😩 🗋 🗙 Q 🔽 🖬	
Group Policy Management Forest: oodev.hp.ro Domains Ocfault Domain Policy Domain Controllers Domain Controllers PowerShellServers PowerShellServers PowerShellServers PowerShellServers Default Domain Policy Default Domain Policy Policy Default Domain Policy Default Domain Policy Default Domain Policy Pol	PowerShell Remoting Scope Details Scope Details Display links in this location: oodev.hp.ro The following sites, domains, and OUs are linked to this GPO: Ink Enabled Location Enforced Link Enabled PowerShellRervers No Yes Security Filtering The settings in this GPO can only apply to the following groups, users, and computer: Name Add Remove Properties WMI Filtering This GPO is linked to the following WMI filter: Open
Help	

Figure 35: Using Group Policy Objects - step 2

- 4 Configure the GPO same way as for single hosts.
- 5 Now that there is a new GPO, you can apply it to a group of servers. The following example shows you how to link it on an existing OU from AD, however this can be applied to other groups either. The GPO interface displays the existing OU's from the domain controller AD. To link a GPO to an OU, go to that, right click it and select Link an Existing GPO.

🔜 Group Policy Management	
🛃 File Action View Window Help	<u>_8×</u>
🗢 🔿 🙍 📋 🔒 🖉 🖬	
Group Policy Management Forest: codev.hp.ro Domains codev.hp.ro Default Domain Policy Create a GPO in this domain, and Link it her Disc Interfance Group Policy Modeling Wizard New Organizational Unit New Window from Here Delete Rename Refresh Properties Help	Contents Delegation Delegation Delegation Delegation Delegation Delegation Delegation Default Domain Controllers Policy Default Domain Policy PowerShell Remoting
Select an existing GPO and link it to this container	

Figure 36: Using Group Policy Objects - step 3

6 The GPO settings will be applied to all servers contained in the selected OU. Although a GPO update might be required and also a reboot for the servers before the policies are actually applied.

Note: Local policies overwrite domain policies.

Enable Remoting for Non-Administrative Users

To establish a PSSession or run a command on a remote computer, the user must have permission to use the session configurations on the remote computer.

By default, only members of the Administrators group on a computer have permission to use the default session configurations. Therefore, only members of the Administrators group can connect to the computer remotely.

To allow other users to connect to the local computer, give the user Execute permissions to the default session configurations on the local computer.

The following command opens a property sheet that lets you change the security descriptor of the default Microsoft.PowerShell session configuration on the local computer.

Set-PSSessionConfiguration Microsoft.Powershell –ShowSecurityDescriptorUI

📜 Permissions for http://schema	s.microsoft.c	om/powers 🗙
http://schemas.microsoft.com/power	shell/Microsoft.F	Powershell
Group or user names:		
psuser (psuser@oodev.hp.ro) & Administrators (EXCH2010CAS)	1VAdministrators	\$)
1	Add	Remove
-		
Permissions for psuser	Allow	Deny
Full Control(All Operations)	V	
Read(Get,Enumerate,Subscribe)		
Write(Put,Delete,Create)		
Execute(Invoke)		
Special permissions		
For special permissions or advanced click Advanced.	d settings,	Advanced
Learn about access control and per	missions	
OK	Cancel	Apply

Figure 37: Adding a user to the PowerShell session configuration

Authentication Types

Basic

Client side steps

- Allow unencrypted communication for the client, by running the following PowerShell command: set-item wsman:\localhost\client\AllowUnencrypted -value true
- Enable Basic authentication for the client, by running the following PowerShell command: set-item wsman:\localhost\client\auth\Basic -value true

Server side steps

- Turn off encryption for the WinRM service, by running the following PowerShell command: set-item wsman:\localhost\service\AllowUnencrypted -value true
- Enable Basic authentication for the service, by running the following PowerShell command: set-item wsman:\localhost\service\auth\Basic -value true

Notes

- The client and server can be in different domains.
- When using Basic authentication, a local user account must be provided for authentication on the remote host.
- Basic can be used when the destination is an IP address.
- Basic can be used when the destination is one of the following: localhost, 127.0.0.1, [::1].
- The cluster name, as well as the hostnames of the cluster nodes can be used for the destination host.

CredSSP

Client side steps

- Enable CredSSP authentication for the client, by running the command: Enable-WSManCredSSP -Role Client -DelegateComputer WSMAN/*.
- Allow delegating fresh credentials by performing the following steps. Open gpedit.msc and go to Computer Configuration → Administrative Templates → System → Credentials Delegation. Enable "Allow Delegating Fresh Credentials" and add the wsman hosts to the server list. Run gpupdate /force from command line to force policy update.

Server side steps

- Enable CredSSP authentication, by running the following PowerShell command: Enable-WSManCredSSP -Role Server
- Create a new https listener by using the following commad: winrm create winrm/config/*Listener*?*Address*=*+*Transport*=*HTTPS*.

Domain Controller side steps

If the NETWORK SERVICE doesn't have "Validated write to service principal name", either try running the following command: dsacls
 "CN=AdminSDHolder,CN=System,DC=domain,DC=com" /G "Sn-1-5 20:WS;Validated write to service principal name" or open ADUC, go to Computers →
 DC object → Security, select Network Service and give it "Validated write to SPN".

Default

When Default authentication is used, the following situations can occur:

- Kerberos is the method of authentication used if the client is in the same domain as the destination host, and the value specified for that host is not one of the following: localhost, 127.0.0.1, [::1].
- Negotiate is the method of authentication used if the client is not in the same domain as the destination host, or the value specified for that host is one of the following: localhost, 127.0.0.1, [::1].

Digest

• Digest authentication is not supported for remote connections. It cannot be configured for the WinRM server component.

Kerberos

Client side steps

• Enable Kerberos authentication for the client, by running the following PowerShell command: set-item wsman:\localhost\client\auth\Kerberos -value true

Server side steps

• Enable Kerberos authentication for the service, by running the following PowerShell command: set-item wsman:\localhost\service\auth\Kerberos -value true

Notes:

- The client and server must be in the same domain.
- Either a local or a domain user account can be provided for authentication on the server host.
- Kerberos cannot be used when the destination is an IP address.
- Kerberos cannot be used when the destination is one of the following: localhost, 127.0.0.1, [::1].
- The cluster name cannot be used to specify the host. Only the hostnames of the cluster nodes can be used for the destination host.

Negotiate

Client side steps

• Enable Negotiate authentication for the client, by running the following PowerShell command: set-item wsman:\localhost\client\auth\Negotiate -value true

Server side steps

• Enable Negotiate authentication for the service, by running the following PowerShell command: set-item wsman:\localhost\service\auth\Negotiate -value true

Notes

- The client and server can be in different domains.
- Either a local or a domain user account can be provided for authentication on the server host. Local accounts can only be provided when connecting to the localhost.
- Negotiate can be used when the destination is an IP address.
- Negotiate can be used when the destination is one of the following: localhost, 127.0.0.1, [::1].
- The cluster name, as well as the hostnames of the cluster nodes can be used for the destination host.

NegotiateWithImplicitCredential

Notes

- When using NegotiateWithImplicitCredentials, no credentials should be provided. The current logged-on user account will be used for authentication. This can either be a local or a domain user account.
- NegotiateWithImplicitCredential can only be used when the destination is one of the following: localhost, 127.0.0.1, [::1].

6 Troubleshooting

This section provides troubleshooting procedures that you can use to solve problems you may encounter while using the wizard. It also includes an error message you may receive while using the integration and offers descriptions and possible fixes for the error.

Could not connect to the host

The possible reasons are:

- The user credentials are not correct.
- The user does not have permission to execute PowerShell scripts on the target host. Make sure the user has admin rights or refer to the section *Enable Remoting for Non-Administrative Users*.
- Authentication problems (most common). Refer to *Running a PowerShell Script on a Remote Host.*
- The WinRM service is stopped on the target host.
- WinRM default ports (5985 and 5986) were changed. You need to provide the correct port in the connection page of the wizard.

The wizard fails to load modules on a x64 localhost.

Some modules cannot be loaded using the wizard, but they are loaded from the PowerShell console. By default, the wizard runs in a x32 process (depends on the OO jre) which ends up calling x32 PowerShell. The x32 version of PowerShell cannot load some modules (e.g. FailoverClusters) therefore the wizard fails. In order to fix this, do not leave the host input empty. Instead, you need to provide the "localhost". This way, the wizard will try to authenticate the localhost like any other remote host. Note that remoting rules should be satisfied for localhost in this case. If user is left empty, the wizard will connect using the NegotiateWithImplicitCredential. Otherwise, you need to provide user credentials and authentication type as for any other remote host.

The user has exceeded the maximum allowed number of remote shells

The user has exceeded the maximum allowed number of remote shells. This error would probably occur if the user stresses the wizard with too many "back and next" actions without running the wizard from start to end. Refer to Group Policy Configuration sections in order to increase the allowed number of remote shells per user.

\mathbf{X}	Could not connect to target PowerShell host. Error:dotNETWebActions PowerShell.PowerShellExceptions
-	.SessionException: Could not connect to host> System
	Management.Automation.Remoting.PSRemotingTransportE
	xception: Connecting to remote server failed with the follow
	ing error message : The WS-Management service cannot pr
	ocess the request. This user is allowed a maximum number o f 5 concurrent shells, which has been exceeded. Close exist
	ing shells or raise the quota for this user. For more informati
	on, see the about Remote Troubleshooting Help topic.
	End of inner exception stack trace
	at PowerShellV2.PSRunspaceV2.NewRemoteSession(String
	host, PSCredential credential, AuthenticationMechanism au
	thentication)
	at PowerShellV2.PSRunspaceV2.CreateRunspace(String h ost, PSCredential credential, AuthenticationMechanism auth
	entication)
	at PowerShellV2.PSRunspaceV2.CreateNewRunspace()
	at PowerShellV2.PSRunspaceV2ctor(PSAuthenticationInf
	o authInfo)
	at PowerShellV2.PowerShellFactoryV2.CreateRemoteRun space(PSAuthenticationInfo psAuth, Boolean keepSessionAl
	ive, Boolean enablePiping)
	at PowerShellCommon.PowerShellFactory.CreateRunspac
	e(PSAuthenticationInfo authInfo, Boolean keepSessionAlive,
	Boolean enablePiping) at PowerShellCmdlets.Program.CreateRunspace(String ar
	gs)
	OK Details >>

Figure 38: Connection Error - number of remote shells has been exceeded