



Windows PowerShell Automations PowerPack

Version 105

Table of Contents

Introduction to the Windows PowerShell Automations PowerPack	3
What is the Windows PowerShell Automations PowerPack?	4
Installing the Windows PowerShell Automations PowerPack	4
Creating a Credential for Windows PowerShell	5
Configuring Windows PowerShell Automations	8
Windows PowerShell Automation Policies	9
User-initiated Automation Policies	13
Creating and Customizing Automation Policies	13
Removing an Automation Policy from a PowerPack	15
Configuring Windows PowerShell Run Book Actions	17
Windows PowerShell Run Book Actions	18
Authentication for Windows Devices with the Windows PowerShell Automations PowerPack	20
Creating a Custom Run Book Action Policy	20
Creating a Windows PowerShell Run Book Action	21
Run Book Variables	23
Run Book Variables	24

Chapter

1

Introduction to the Windows PowerShell Automations PowerPack

Introduction to the Windows PowerShell Automations PowerPack

This manual describes how to use the run book automation policies, run book actions, and custom action types found in the "Windows PowerShell Automations" PowerPack.

NOTE: ScienceLogic provides this documentation for the convenience of ScienceLogic customers. Some of the configuration information contained herein pertains to third-party vendor software that is subject to change without notice to ScienceLogic. ScienceLogic makes every attempt to maintain accurate technical information and cannot be held responsible for defects or changes in third-party vendor software. There is no written or implied guarantee that information contained herein will work for all third-party variants. See the End User License Agreement (EULA) for more information.

This chapter covers the following topics:

<i>What is the Windows PowerShell Automations PowerPack?</i>	4
<i>Installing the Windows PowerShell Automations PowerPack</i>	4
<i>Creating a Credential for Windows PowerShell</i>	5

What is the Windows PowerShell Automations PowerPack?

The "Windows PowerShell Automations" PowerPack includes:

- A custom action type for running PowerShell commands on remote devices
- A device group with rules that include only Windows devices
- A set of run book actions that run diagnostic commands on Windows systems via PowerShell
- A set of run book automation policies that tie events from monitoring PowerPacks to the run book actions

The run book automation actions for this PowerPack are executed on the SL1 All-In-One Appliance or Data Collector.

In addition to using the standard content, you can use the content in the "Windows PowerShell Automations" PowerPack to:

- Create your own run book automation policies that include the pre-defined actions that run different sets of diagnostic commands.
- Use the supplied "Execute PowerShell Request" custom action type to configure your own run book action by supplying a set of commands to be executed via PowerShell.

Version 105 and later of this PowerPack supports Python 3.6.

Installing the Windows PowerShell Automations PowerPack

Before completing the steps in this manual, you must import and install the latest version of the "Windows PowerShell Automations" PowerPack.

IMPORTANT: You must install the "Datacenter Automation Utilities" PowerPack version 201 or later before using this release of the "Windows PowerShell Automations" PowerPack.

NOTE: The "Windows PowerShell Automations" PowerPack requires SL1 version 12.1.2 or later. For details on upgrading SL1, see the appropriate SL1 [Release Notes](#).

TIP: By default, installing a new version of a PowerPack overwrites all content from a previous version of that PowerPack that has already been installed on the target system. You can use the **Enable Selective PowerPack Field Protection** setting in the **Behavior Settings** page (System > Settings > Behavior) to prevent new PowerPacks from overwriting local changes for some commonly customized fields. For more information, see the section on [Global Settings](#).

To download and install the PowerPack:

1. Search for and download the PowerPack from the **PowerPacks** page (Product Downloads > PowerPacks & SyncPacks) at the [ScienceLogic Support Site](#).

2. In SL1, go to the **PowerPacks** page (System > Manage > PowerPacks).
3. Click the **[Actions]** button and choose *Import PowerPack*. The **Import PowerPack** dialog box appears.
4. Click **[Browse]** and navigate to the PowerPack file from step 1.
5. Select the PowerPack file and click **[Import]**. The **PowerPack Installer** modal displays a list of the PowerPack contents.
6. Click **[Install]**. The PowerPack is added to the **PowerPacks** page.

NOTE: If you exit the **PowerPack Installer** modal without installing the imported PowerPack, the imported PowerPack will not appear in the **PowerPacks** page. However, the imported PowerPack will appear in the **Imported PowerPacks** modal. This page appears when you click the **[Actions]** menu and select *Install PowerPack*.

Creating a Credential for Windows PowerShell

If you do not have the "Microsoft: Windows Server" PowerPack installed, you must create a credential that includes the username and password to communicate with your Windows devices.

To prepare your Windows systems for monitoring, follow the instructions in [Configuring Windows Servers for Monitoring with PowerShell](#).

NOTE: If you have the "Microsoft: Windows Server" PowerPack installed and configured, you may skip this section.

To define a PowerShell credential in SL1:

1. Collect the information you need to create the credential:
 - The username and password for a user on the Windows device.
 - If the user is an Active Directory account, the hostname or IP address of the Active Directory server and the domain.
 - Determine if an encrypted connection should be used.
 - If you are using a Windows Management Proxy, the hostname or IP address of the proxy server.
2. Go to the **Credential Management** page (System > Manage > Credentials).
3. In the **Credential Management** page, click the **[Actions]** menu. Select **Create PowerShell Credential**.
4. The **Credential Editor** page appears, where you can define the following fields:
 - **Profile Name**. Name of the credential. Can be any combination of alphanumeric characters. This field is required.
 - **Hostname/IP**. Hostname or IP address of the device from which you want to retrieve data. This field is required.

- You can include the variable **%D** in this field. SL1 will replace the variable with the IP address of the device that is currently using the credential.
- You can include the variable **%N** in this field. SL1 will replace the variable with the hostname of the device that is currently using the credential. If SL1 cannot determine the hostname, SL1 will replace the variable with the primary, management IP address for the current device.
- You can include the prefix **HOST** or **WSMAN** before the variable **%D** in this field if the device you want to monitor uses a service principal name (for example, "HOST://%D" or "WSMAN://%D"). SL1 will use the WinRM service HOST or WSMAN instead of HTTP and replace the variable with the IP address of the device that is currently using the credential.
- **Username.** Type the username for an account on the Windows device to be monitored or on the proxy server. This field is required.

NOTE: The user should not include the domain name prefix in the username for Active Directory accounts. For example, use "em7admin" instead of "MSDOMAIN\em7admin".

- **Encrypted.** Select whether SL1 will communicate with the device using an encrypted connection. Choices are:
 - *yes.* When communicating with the Windows server, SL1 will use a local user account with authentication of type "Basic Auth". You must then use HTTPS and can use a Microsoft Certificate or a self-signed certificate.
 - *no.* When communicating with the Windows server, SL1 will not encrypt the connection.
- **Port.** Type the port number used by the WinRM service on the Windows device. This field is automatically populated with the default port based on the value you selected in the **Encrypted** field. This field is required.
- **Account Type.** Type of authentication for the username and password in this credential. Choices are:
 - *Active Directory.* On the Windows device, Active Directory will authenticate the username and password in this credential.
 - *Local.* Local security on the Windows device will authenticate the username and password in this credential.
- **Timeout (ms).** Type the time, in milliseconds, after which SL1 will stop trying to collect data from the authenticating server. For collection to be successful, SL1 must connect to the authenticating server, execute the PowerShell command, and receive a response within the amount of time specified in this field.
- **Password.** Type the password for the account on the Windows device to be monitored or on the proxy server. This field is required.

- **PowerShell Proxy Hostname/IP.** If you use a proxy server in front of the Windows devices you want to communicate with, type the fully-qualified domain name or the IP address of the proxy server in this field.
- **Active Directory Hostname/IP.** If you selected Active Directory in the **Account Type** field, type the hostname or IP address of the Active Directory server that will authenticate the credential.
- **Domain.** If you selected Active Directory in the **Account Type** field, type the domain where the monitored Windows device resides.

5. To save the credential, click the **[Save]** button. To clear the values you set, click the **[Reset]** button.

Configuring Windows PowerShell Automations

Overview

This chapter describes how to use the automation policies, run book actions, and custom action types found in the "Windows PowerShell Automations" PowerPack.

This chapter covers the following topics:

<i>Windows PowerShell Automation Policies</i>	9
<i>Creating and Customizing Automation Policies</i>	13

Windows PowerShell Automation Policies

The "Windows PowerShell Automations" PowerPack includes the following automation policies:

- Windows PowerShell: Run CPU & Memory Diagnostic Commands
- Windows PowerShell: Run CPU Diagnostic Commands
- Windows PowerShell: Run Disk I/O Diagnostic Commands
- Windows PowerShell: Run Disk Usage Diagnostic Commands
- Windows PowerShell: Run Memory Diagnostic Commands
- Windows PowerShell: Run Print Job Error Diagnostic Commands

Each policy triggers a single run book action that collects diagnostic data within a PowerShell session, and an action that formats the output as HTML for events associated with devices in the "Windows Automation" device group, and an action that formats the output. The "Windows Automation" device group is included in this PowerPack. You will need to manually add the Windows devices you want to monitor to the "Windows Automation" device group.

All of the run book actions use the same custom action type, "Execute PowerShell Request", which is supplied in the PowerPack.

All of the automation policies are tied to included ScienceLogic SL1 events generated by the Dynamic Applications from the "Microsoft Windows Server" PowerPack.

Several of the run book actions use the substitution character feature of the "Execute PowerShell Request" custom action type. If an event variable is included in a command (such as "%Y" for the sub-entity name), the custom action type automatically replaces that variable with the value from the triggering event.

The following table shows the standard automation policies, their aligned events, and the run book actions that runs in response to the events.

NOTE: The aligned events are included as part of the "Microsoft Windows Server" PowerPack and are not installed with the SL1 platform. You must install the "Microsoft Windows Server" PowerPack to obtain these events.

Automation Policy Name	Aligned Device Group	Aligned Events	Aligned Run Book Actions
Windows PowerShell: Run CPU & Memory Diagnostic Commands	Windows Automation	<ul style="list-style-type: none"> • Minor: Microsoft: Windows Disk Transfer Time (Physical Disk) exceeded threshold 	<ul style="list-style-type: none"> • Automation Utilities: Calculate Memory Size for Each Action • Windows CPU and Memory Diagnostic Commands

Automation Policy Name	Aligned Device Group	Aligned Events	Aligned Run Book Actions
			<ul style="list-style-type: none"> Datacenter Automation: Format Output as HTML
Windows PowerShell: Run CPU Diagnostic Commands	Windows Automation	<ul style="list-style-type: none"> Minor: Microsoft: Windows CPU Utilization has exceeded the threshold Minor: Microsoft: Windows Processor Queue Length exceeded the threshold 	<ul style="list-style-type: none"> Automation Utilities: Calculate Memory Size for Each Action Windows CPU and Memory Diagnostic Commands Datacenter Automation: Format Output as HTML
Windows PowerShell: Run Disk I/O Diagnostic Commands	Windows Automation	<ul style="list-style-type: none"> Minor: Microsoft: Windows % Disk Time (Logical Disk) exceeded threshold Minor: Microsoft: Windows % Disk Time (Physical Disk) exceeded threshold Minor: Microsoft: Windows Current Disk QueueLength (Physical Disk)exceeded threshold 	<ul style="list-style-type: none"> Automation Utilities: Calculate Memory Size for Each Action Windows Disk I/O Diagnostic Commands Datacenter Automation: Format Output as HTML
Windows PowerShell: Run Disk Usage Diagnostic Commands	Windows Automation	<ul style="list-style-type: none"> Poller: File system usage exceeded (major) threshold Poller: File system usage exceeded (critical) threshold 	<ul style="list-style-type: none"> Automation Utilities: Calculate Memory Size for Each Action Windows Get Largest Event Log Files Windows Get Largest Files on Disk Windows Disk I/O Diagnostic Commands

Automation Policy Name	Aligned Device Group	Aligned Events	Aligned Run Book Actions
			<ul style="list-style-type: none"> Datacenter Automation: Format Output as HTML
Windows PowerShell: Run Memory Diagnostic Commands	Windows Automation	<ul style="list-style-type: none"> Major: Microsoft: Windows Available Memory below threshold Major: Microsoft: Windows Pages per Second has exceeded threshold Minor: Microsoft: Windows Paging File Usage has exceeded threshold 	<ul style="list-style-type: none"> Automation Utilities: Calculate Memory Size for Each Action Windows Memory Diagnostic Commands Datacenter Automation: Format Output as HTML
Windows PowerShell: Run Print Job Error Diagnostic Commands	Windows Automation	<ul style="list-style-type: none"> Minor: Microsoft: Windows: PowerShell: Print Job Errors exceeded threshold 	<ul style="list-style-type: none"> Automation Utilities: Calculate Memory Size for Each Action Windows Print Job Error Diagnostic Commands Datacenter Automation: Format Output as HTML

The following figure shows a memory event with a classification of "Major" appears on the **Events** page. Click the **[Actions]** button (⚙️) for an event, and select *View run book actions* to see the run book (automation) actions triggered by the events.

ORGANIZATION	SEVERIT...	NAME	MESSAGE	AGE	TICKET L...	CO...	EVENT NOT...	MASKED EVENTS	ACKNOWLEDGE	CLEAR
Windows Devices	Major	10.2.24.56	Example Windows CPU Event	8 days	1				Acknowledge	Clear
Windows Devices	Major	10.2.24.56	Example Windows Memory Event	8 days	1				Acknowledge	Clear
Windows Devices	Major	10.2.24.56	Example Windows Disk IO Event	8 days	1				Acknowledge	Clear
Windows Devices	Major	10.2.24.56	Example Windows Disk Performa...	8 days	1				Acknowledge	Clear
Example Devices	Major	netscaler	Device Failed Availability Check: U...	7 days 15 l		2195		Masked	Acknowledge	Clear
Example Devices	Minor	ec2-34-200...	Network latency exceeded thresh...	3 days 15 l		1044			Acknowledge	Clear
Linux Devices	Major	10.2.24.30	/: File system usage exceeded maj...	2 days 19 l 5		268			Acknowledge	Clear
System	Major	ec2-18-217...	Linux File System /dev/loop0 : /sn...	2 days 1 hr		491			Acknowledge	Clear
System	Major	System	EM7 major event: Dynamic applic...	19 hours 3		20			Acknowledge	Clear
System	Major	cscscol26	Swap memory utilization has exce...	14 hours 5 9		157			Acknowledge	Clear
System	Major	System	Process time exceeded: Process D...	1 hour 23 l		2			Acknowledge	Clear
System	Minor	cscscol26	App: 91, Snippet: 112 reported a c...	30 minutes		1			Acknowledge	Clear
System	Minor	System	Process time exceeded: Process E...	29 minutes		1			Acknowledge	Clear

The results shown for this event, in the **Event Actions Log** modal, include the automation policy that ran (shown at the top of the following figure), along with the run book actions (commands) that ran. Results for each command are also displayed. The following figure shows an example of this HTML output:

```

Event Actions Log | For Event [18263]
2019-12-05 16:29:57
Automation Policy Windows PowerShell: Run CPU & Memory Diagnostic Commands action Windows CPU and Memory Diagnostic Commands with HTML Output ran Successfully
Message CustomActionType (448) executed without incident
Result: Enrichment Command Output

Command: Get-Process | Sort CPU -descending | Select -first 20 | Format-Table -AutoSize
Handles NPM(K) PM(K) WS(K) CPU(s) Id SI ProcessName
-----
967480 91 2405648 204260 217,886.92 876 0 svchost
614 67 272260 183468 182,272.64 1728 0 MshpEng
900 0 124 136 159,606.53 4 0 System
554 62 238672 72732 60,299.03 2216 0 pdlservr
323 21 10072 22432 29,722.22 1840 0 vmtoolsd
93 8 4988 9280 22,232.44 4980 0 conhost
683 39 159316 63144 21,815.34 464 0 svchost
1394 71 28240 41604 20,762.52 1040 0 svchost
93 8 4988 9284 11,400.88 3372 0 conhost
93 8 5012 9672 11,078.36 3916 0 conhost
441 17 16092 19708 10,572.31 928 0 svchost
286 10 5804 9896 10,396.91 596 0 services
93 8 4988 9652 10,348.58 5076 0 conhost
1385 23 8980 17376 9,967.13 604 0 lsass
381 23 30040 41552 7,418.63 4360 0 WmiPrvSE
451 21 14460 26528 7,064.05 2276 0 WmiPrvSE
590 18 29704 35092 5,240.50 736 0 svchost
431 17 1980 4336 3,117.50 384 0 csrss
514 18 5728 14688 2,909.52 692 0 svchost
93 8 4984 9644 2,736.05 7164 0 conhost

Command: Get-Process | Select-Object Name, ID, @(Name='ThreadCount'; Expression = {$_ .Threads.Count}) | Sort-Object -Property ThreadCount -Descending | Select -first 20
Name Id ThreadCount
----
System 4 109
svchost 876 58
sqlservr 2216 55
svchost 1040 43
svchost 464 33
svchost 884 29
MshpEng 1728 25
svchost 1116 22
powershell 4476 20
powershell 5060 18
powershell 5484 17
svchost 928 16
svchost 940 16
WmiPrvSE 2276 16
svchost 3044 15
svchost 1632 14
svchost 692 13
spoolsv 1572 13
svchost 736 13
tsddis 3052 13

Command: Get-Process | Sort WS -descending | Select -first 20 | Format-Table -AutoSize
Handles NPM(K) PM(K) WS(K) CPU(s) Id SI ProcessName
-----
967475 90 2405432 204096 217,886.97 876 0 svchost
627 67 275096 183568 182,273.25 1728 0 MshpEng
654 63 238672 72732 60,299.03 2216 0 pdlservr

```

To learn more about which commands are executed by default for a given run book action, see [Configuring Windows PowerShell Run Book Actions](#).

TIP: Although you can edit the run book actions described in this section, it is a best practice to use "Save As" to create a new run book action, rather than to customize the standard automation policies.

User-initiated Automation Policies

All Windows PowerShell automation policies have a **Policy Type** of "Active Events/User Initiated", which enables all of the features of the "Active Events" and the "User Initiated" policy types. As a result, these automation policies can be triggered by active events that meet the criteria in the policy, or you can manually trigger the automation.

You can run these automation policies as needed from the **Devices** page, the **Events** page, and the **Service Investigator** page. If there is an event policy specified in the automation policy, that event must be active for the policy to be run manually, and the policy can only be run on that event type. The same applies for the device groups list.

For these automation policies to be visible from the **Tools** panel in the **Device Summary** modal, the following three bullets must be true between the event and the automation policy configuration:

- **Organization.** The organization associated with the event must match the organization configured in the automation policy. Policies in the "System" organization match all organizations.
- **Aligned Devices.** The device for which the event is triggered must be configured as a Aligned Device in the automation policy.
- **Aligned Event.** The event must match one of the Aligned Events configured in the automation policy.

In most situations, you would run a user-initiated automation in response to an event that just occurred. If you have Automation PowerPacks installed on your SL1 system, the **Event Actions Log** window for that event might contain diagnostic information from other automations that have already run, including information that helps you determine which user-initiated automation you should run next to address the cause of the event.

To run a user-initiated automation policy, click the open icon (🔗) to open the **Device Summary** modal for the event and click in the **Tools** section. Any available user-initiated automation policy will be listed there, available to run on-demand.

Creating and Customizing Automation Policies

You can use the default run book automation policies in this PowerPack, or you can create and customize the policies as needed.

TIP: You might need to configure a run book action policy before you can add it to the automation policy. For more information, see [Customizing Windows PowerShell Run Book Actions](#).

Before you create an automation policy using the run book actions in this PowerPack, you must determine:

- Which set of commands you want to run on a monitored device when an event occurs. There are ten automation actions in the PowerPack that run the "Execute PowerShell Request" action type with different commands. You can also create your own automation actions using the custom action type supplied in the PowerPack.
- What event criteria you want to use to determine when the automation actions will trigger, or the set of rules that an event must match before the automation is executed. This can include matching only specific event policies, event severity, associated devices, and so on. For a description of all the options that are available in Automation Policies, see the **Run Book Automation** manual.

To create or customize an automation policy that uses the run book actions in the "Windows PowerShell Automations" PowerPack:

1. Go to the **Automation Policy Manager** page (Registry > Run Book > Automation).
2. Click the **[Create]** button to create an automation policy, or search for an existing automation policy that you want to edit and click the wrench icon (🔧) for that policy. The **Automation Policy Editor** page appears:

3. Complete the following fields:
 - **Policy Name.** Enter a name for the automation policy.
 - **Policy Type.** Select whether the automation policy will match events that are active, match when events are cleared, or run on a scheduled basis. Typically, you would select *Active Events* in this field.

- **Policy State.** Specifies whether the policy will be evaluated against the events in the system. If you want this policy to begin matching events immediately, select *Enabled*. The default is *Disabled*.
- **Policy Priority.** Specifies whether the policy is high-priority or default priority. These options determine how the policy is queued.
- **Organization.** Select one or more organizations to associate with the automation policy. The automation policy will execute only for devices in the selected organizations (that also match the other criteria in the policy). To configure a policy to execute for all organizations, select *System* without specifying individual devices to align to.
- **Align With.** Select *Device Groups*.
- **Aligned Device Groups.** The "Windows Automation" device group needs to be aligned. To add the device group to the **Aligned Device Groups** field, select the "Windows Automation" device group in the **Available Device Groups** field and click the right arrow (>>).
- **Aligned Actions.** This field includes the actions from the "Windows PowerShell Automations" PowerPack. To add an action to the **Aligned Actions** field, select the action in the **Available Actions** field and click the right arrow (>>). To re-order the actions in the **Aligned Actions** field, select an action and use the up arrow or down arrow buttons to change that action's position in the sequence.

NOTE: You must have at least two **Aligned Actions**: one that runs the run book action and one that provides the output format. The actions providing the output formats are contained in the "Datacenter Automation Utilities" PowerPack, which is a prerequisite for running automations in this PowerPack.



NOTE: If you are selecting multiple collection actions that use the "Execute Shell Commands" action type, you may want to include the "Calculate Memory Size for Each Action" run book action, found in the "Datacenter Automation Utilities" PowerPack, in your automation policy.

4. Optionally, supply values in the other fields on this page to refine when the automation will trigger.
5. Click **[Save]** for a new policy, or click **[Save As]** if you are customizing an existing policy. If you modify one of the included automation policies and save it with the original name, any customizations you made to that policy will be overwritten when you upgrade the PowerPack.

Removing an Automation Policy from a PowerPack

If you have customized an automation policy from the PowerPack, you might want to remove that policy from that PowerPack to prevent your changes from being overwritten if you update the PowerPack later. If you have the license key with author's privileges for a PowerPack or if you have owner or administrator privileges with your license key, you can remove content from a PowerPack.

To remove content from a PowerPack:

1. Go to the **PowerPack Manager** page (System > Manage > PowerPacks).
2. Find the "Windows PowerShell Automations" PowerPack. Click its wrench icon ()
3. In the **PowerPack Properties** page, in the navigation bar on the left side, click **Run Book Policies**.
4. In the **Embedded Run Book Polices** pane, locate the policy you updated, and click the bomb icon () for that policy. The policy will be removed from the PowerPack and will now appear in the bottom pane.

Chapter

3

Configuring Windows PowerShell Run Book Actions

Overview

This manual describes how to customize the run book actions embedded in the "Windows PowerShell Automations" PowerPack to create run book actions to meet your organization's specific requirements.

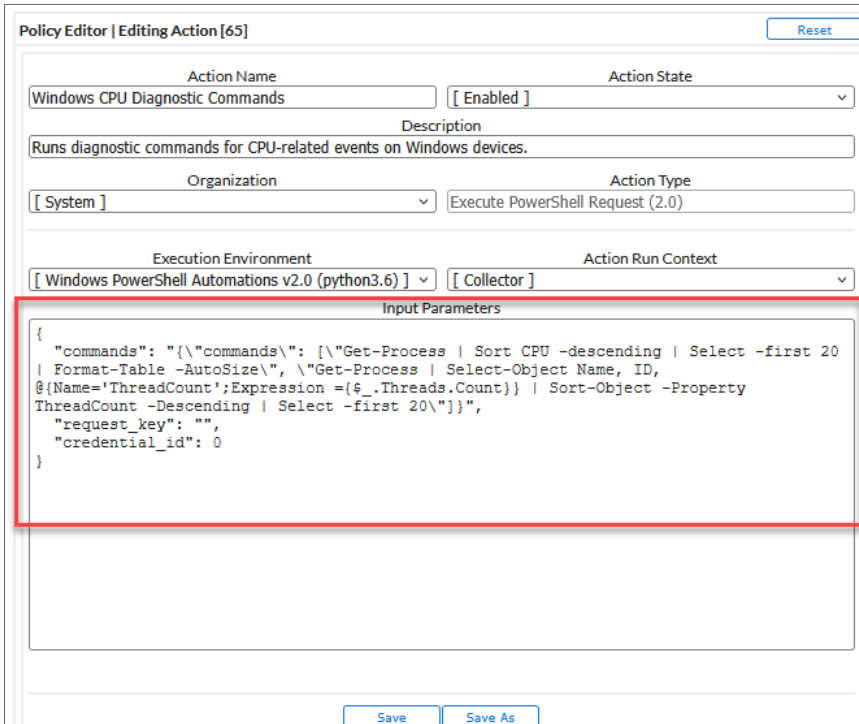
For more information about creating automation policies using custom action types, see [Configuring Windows PowerShell Automations](#).

This chapter covers the following topics:

Windows PowerShell Run Book Actions	18
Authentication for Windows Devices with the Windows PowerShell Automations PowerPack	20
Creating a Custom Run Book Action Policy	20
Creating a Windows PowerShell Run Book Action	21

Windows PowerShell Run Book Actions

The "Windows PowerShell Automations" PowerPack includes run book actions that execute the "Execute PowerShell Request" action type to request diagnostic information or remediate an issue. You can specify the host and the options in a JSON structure that you enter in the **Input Parameters** field in the **Action Policy Editor** modal.



The following automation actions that use the "Execute PowerShell Request" action type are included in the "Windows PowerShell Automations" PowerPack. Compare the commands run with the example in the image above. For more information about input parameter fields, see the table in [Creating a Windows PowerShell Run Book Action](#).

Action Name	Description	Commands Run
Windows CPU and Memory Diagnostic Commands	Runs diagnostic commands for CPU and Memory events on Windows devices.	<ul style="list-style-type: none"> Get-Process Sort CPU -descending Select -first 20 Get-Process Select-Object Name, ID, @ {Name='ThreadCount';Expression = {\$_ .Threads.Count}} Sort-Object -Property ThreadCount -Descending Select -first 20 Get-Process Sort WS -descending Select -first 20

Action Name	Description	Commands Run
		<ul style="list-style-type: none"> <code>Get-CimInstance -Class Win32_PageFileUsage Format-Table -Property Caption,Name,Status,Description,InstallDate,AllocatedBaseSize,PeakUsage,TempPageFile</code> A command that collects the memory usage of running processes, where the memory usage is aggregated across all instances of each named process. The command is not listed here for clarity.
Windows CPU Diagnostic Commands	Runs diagnostic commands for CPU-related events on Windows devices.	<ul style="list-style-type: none"> <code>Get-Process Sort CPU -descending Select -first 20</code> <code>Get-Process Select-Object Name, ID, @{Name='ThreadCount';Expression={\$_.Threads.Count}} Sort-Object -Property ThreadCount -Descending Select -first 20</code>
Windows Disk I/O Diagnostic Commands	Runs diagnostic commands for Disk I/O events on Windows devices.	<ul style="list-style-type: none"> A command that collects the "IO Data Bytes per second" counter for each running process. The command takes 10 samples at 1-second intervals and returns the average of all samples for each process. The command is not listed here for clarity. A command that collects the "IO Data Operations per second" counter for each running process. The command takes 10 samples at 1-second intervals and returns the average of all samples for each process. The command is not listed here for clarity.
Windows Get Largest Event Log Files	Gets the 20 largest Windows Event Log files.	<ul style="list-style-type: none"> <code>Get-ChildItem C:\Windows\System32\winevt\Logs Sort -Descending -Property length Select -first 20</code>
Windows Get Largest Files on Disk	Gets the 20 largest files on the disk specified in the event.	<ul style="list-style-type: none"> <code>Get-ChildItem %Y -r -erroraction 'silentlyContinue' Sort -Descending -Property length Select -first 20 Select-Object FullName,@{Name='SizeMB';Expression={[math]::Round(\$_.Length / 1MB,2)}</code>
Windows Memory Diagnostic Commands	Runs diagnostic commands for Memory-related events on Windows devices.	<ul style="list-style-type: none"> <code>Get-Process Sort WS -descending Select -first 20</code> <code>Get-CimInstance -Class Win32_PageFileUsage Format-Table -Property Caption,Name,Status,Description,InstallDate,AllocatedBaseSize,PeakUsage,TempPageFile</code> A command that collects the memory usage of running processes, where the memory usage is aggregated across all instances of each named process. The command is not listed here for clarity.
Windows Print Job Error	Runs diagnostic commands for	<ul style="list-style-type: none"> <code>Get-Printer Get-PrintJob Where-Object JobStatus -like '*error*'</code>

Action Name	Description	Commands Run
Diagnostic Commands	Print Job Error events on Windows devices.	

Authentication for Windows Devices with the Windows PowerShell Automations PowerPack

The "Execute PowerShell Request" custom action type supports hard-coded credentials where you specify the ID of a credential in the run book action. Alternately, the custom action type can dynamically determine the credential to use.

By default, the automation actions in this PowerPack use the dynamic method, which uses the default value of `"credential_id": 0` in the **Input Parameters** for the run book actions aligned with the "Execute PowerShell Request" custom action type.

The dynamic method uses the first credential that matches the following rules:

- If the "Microsoft: Windows Server Configuration Cache" Dynamic Application (from the "Microsoft: Windows Server" PowerPack) is aligned to the device associated with the triggering event, the credential aligned to that Dynamic Application is used.
- If the "Microsoft: Windows Server Performance Cache" Dynamic Application (from the "Microsoft: Windows Server" PowerPack) is aligned to the device associated with the triggering event, the credential aligned to that Dynamic Application is used.
- If the "Microsoft: Windows Server OS Configuration" Dynamic Application (from the "Microsoft: Windows Server" PowerPack) is aligned to the device associated with the triggering event, the credential aligned to that Dynamic Application is used.
- If none of the listed Dynamic Applications are aligned to the device associated with the triggering event, the first available credential aligned to the device as a secondary credential is used.

Creating a Custom Run Book Action Policy

You can use the "Execute PowerShell Request" action type included with the "Windows PowerShell Automations" PowerPack to create custom automation actions that you can then use to build custom automation policies.

To create a custom action policy using the "Execute PowerShell Request (2.0)" action type:

1. Navigate to the **Action Policy Manager** page (Registry > Run Book > Actions).
2. In the **Action Policy Manager** page, click the **[Create]** button. The **Action Policy Editor** modal appears.
3. In the **Action Policy Editor** page, supply a value in each field.
 - **Action Name**. Specify the name for the action policy.
 - **Action State**. Specifies whether the policy can be executed by an automation policy (enabled) or cannot be executed (disabled).
 - **Description**. Allows you to enter a detailed description of the action.

- **Organization.** Organization to associate with the action policy.
- **Action Type.** Type of action that will be executed. Select the "Execute PowerShell Request (2.0)" action type.
- **Execution Environment.** Select from the list of available Execution Environments. The default execution environment is *System*.
- **Action Run Context.** Select *Database* or *Collector* as the context in which the action policy will run.
- **Input Parameters.** A JSON structure that specifies each input parameter. Each parameter definition includes its name, data type, and whether the input is optional or required for this Custom Action Type. For more information about the available input parameters, see the table in [Creating a Windows PowerShell Run Book Action](#).

NOTE: Input parameters must be defined as a JSON structure.

6. Click **[Save]**. If you are modifying an existing action policy, click **[Save As]**. Supply a new value in the **Action Name** field, and save the current action policy, including any edits, as a new policy.

TIP: For more information about substitution variables, see [Appendix A: Run Book Variables](#).

Creating a Windows PowerShell Run Book Action

You can create a new automation action that runs remote PowerShell requests using the supplied "Execute PowerShell Request" custom action type. To do this, select "Execute PowerShell Request" in the Action Type drop-down list when you create a new automation action. You can also use the existing automation actions in the PowerPack as a template by using the **[Save As]** option.

The Windows PowerShell automation actions accept the following parameters in JSON:

Parameter	Input type	Description
commands	string	Specifies a single command or a list of commands, in JSON format, to execute. You can use substitution variables in the commands.
request_key	string	<p>(Optional field)</p> <p>Default value: empty</p> <p>The unique key for each instance (row) returned by the request. This unique key must be a property name, and the request must include that property (column) and return values from that property name (column).</p> <p>Example: Suppose you want to get the ID, number of cores, name, and maximum clock speed of every CPU installed on a Windows system, run the following command, where "DeviceID" is the request key.</p>

Parameter	Input type	Description
		<code>Get-WmiObject -Class Win32_Processor -Property DeviceID, NumberOfCores, Name, MaxClockSpeed Format-List DeviceID, NumberOfCores, Name, MaxClockSpeed</code>
credential_id	integer	<p>Default value: 0</p> <p>Specifies the credential_id to use for the connection.</p> <ul style="list-style-type: none"> • If set to 0 (false), the custom action type will dynamically determine the credential. • If set to an ID number, it maps to the credential ID specified. You can find credential IDs by going to System > Manage > Credentials.

Using Substitution Values. The commands input can contain substitution values that match the keys in EM7_VALUES.

TIP: For more information about substitution variables, see [Appendix A: Run Book Variables](#).

For a description of all options that are available in Automation Policies, see the *Run Book Automation* manual.

Appendix



A

Run Book Variables

Overview

This appendix defines the different variables you can use when creating an action policy.

Use the following menu options to navigate the SL1 user interface:

- To view a pop-out list of menu options, click the menu icon (.
- To view a page containing all of the menu options, click the Advanced menu icon ().

This appendix covers the following topics:

This chapter covers the following topics:

<i>Run Book Variables</i>	24
---------------------------------	----

Run Book Variables

You can include variables when creating an action policy. These variables are listed in the table below.

- In an action policy of type **Send an Email Notification**, you can include one or more of these variables in the fields **Email Subject** and **Email Body**.
- In an action policy of type **Send an SNMP Trap**, you can include one or more of these variables in the **Trap OID** field, **Varbind OID** field, and the **Varbind Value** field.
- In an action policy of type **Create a New Ticket**, you can include one or more of these variables in the **Description** field or the **Note** field of the related Ticket Template.
- In an action policy of type **Send an SNMP Set**, you can include one or more of these variables in the **SNMP OID** field and the **SNMP Value** field.
- In an action policy of type **Run A Snippet**, you can access variables from the global dictionary **EM7_VALUES**.
- In a policy of type **Execute an SQL Query**, you can include one or more of these variables in the **SQL Query** field.

Variable	Source	Description
%A	Account	Username
%a	Entity	IP address
%F	Dynamic Alert	Alert ID for a Dynamic Application Alert
%g	Asset	Asset serial
%h	Asset	Device ID associated with the asset
%l (uppercase "eye")	Dynamic Alert	For events with a source of "dynamic", this variable contains the index value from SNMP. For events with a source of "syslog" or "trap", this variable contains the value that matches the Identifier Pattern field in the event definition.
%i (lowercase "eye")	Asset	Asset Location
%K	Asset	Asset Floor
%k	Asset	Asset Room
%L	Dynamic Alert	Value returned by the label variable in a Dynamic Application Alert.
%m	Automation	Automation policy note
%N	Action	Automation action name
%n	Automation	Automation policy name
%P	Asset	Asset plate
%p	Asset	Asset panel
%Q	Asset	Asset punch
%q	Asset	Asset zone
%T	Dynamic Alert	Value returned by the Threshold function in a Dynamic Application Alert.

Variable	Source	Description
%U	Asset	Asset rack
%u	Asset	Asset shelf
%v	Asset	Asset tag
%W	Asset	Asset make
%w	Asset	Asset model
%V	Dynamic Alert	Value returned by the Result function in a Dynamic Application Alert.
_%category_id	Entity	Device category ID associated with the entity in the event.
_%category_name	Entity	Device category name associated with the entity in the event.
_%class_id	Entity	Device class ID associated with the entity in the event.
_%class_name	Entity	Device class description associated with the entity in the event.
_%parent_id	Entity	For component devices, the device ID of the parent device.
_%parent_name	Entity	For component devices, the name of the parent device.
_%root_id	Entity	For component devices, the device ID of the root device.
_%root_name	Entity	For component devices, the name of the root device.
_%service_investigator_url	Entity	The URL of the Business Service Investigator page for the event that triggered the automation (for run book actions that run against events aligned with business services).
%1 (one)	Event	Entity type. Possible values are: <ul style="list-style-type: none"> • 0. Organization • 1. Device • 2. Asset • 4. IP Network • 5. Interface • 6. Vendor • 7. Account • 8. Virtual Interface • 9. Device Group • 10. IT Service • 11. Ticket
%2	Event	Sub-entity type. Possible values for organizations are: <ul style="list-style-type: none"> • 9. News feed Possible values for devices are:

Variable	Source	Description
		<ul style="list-style-type: none"> • 1. CPU • 2. Disk • 3. File System • 4. Memory • 5. Swap • 6. Component • 7. Interface • 9. Process • 10. Port • 11. Service • 12. Content • 13. Email
%4	Event	Text string of the user name that cleared the event.
%5	Event	Date/time when event was deleted.
%6	Event	Date/time when event became active.
%7	Event	<p>Event severity (1-5), for compatibility with previous versions of SL1 . 1 =critical, 2=major, 3=minor, 4=notify, 5=healthy.</p> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p>NOTE: When referring to an event, %7 represents severity (for previous versions of SL1). When referring to a ticket, %7 represents the subject line of an email used to create a ticket.</p> </div>
%c	Event	Event counter
%d	Event	Date/time when last event occurred.
%D	Event	Date/time of first event occurrence.
%e	Event	Event ID
%H	Event	URL link to event
%M	Event	Event message
%s	Event	severity (0 - 4). 0=healthy, 1=notify, 2=minor, 3=major, 4=critical.
%S	Event	Severity (HEALTHY - CRITICAL)
%_user_note	Event	Current note about the event that is displayed on the Events page.
%x	Event	Entity ID
%X	Event	Entity name
%y	Event	Sub-entity ID
%Y	Event	Sub-entity name
%Z	Event	Event source (Syslog - Group)

Variable	Source	Description
%z	Event	Event source (1 - 8)
%_ext_ticket_ref	Event	For events associated with an external Ticket ID, this variable contains the external Ticket ID.
%3	Event Policy	Event policy ID
%E	Event Policy	External ID from event policy
%f	Event Policy	Specifies whether event is stateful, that is, has an associated event that will clear the current event. 1 (one)=stateful; 0 (zero)=not stateful.
%G	Event Policy	External Category
%R	Event Policy	Event policy cause/action text
%_event_policy_name	Event Policy	Name of the event policy that triggered the event.
%B	Organization	Organization billing ID
%b	Organization	Impacted organization
%C	Organization	Organization CRM ID
%o (lowercase "oh")	Organization	Organization ID
%O (uppercase "oh")	Organization	Organization name
%r	System	Unique ID / name for the current SL1 system
%7	Ticket	Subject of email used to create a ticket. If you specify this variable in a ticket template, SL1 will use the subject line of the email in the ticket description or note text when SL1 creates the ticket. NOTE: When referring to a ticket, %7 represents the subject line of an Email used to create a ticket. When referring to an event, %7 represents severity (for previous versions of SL1).
%t	Ticket	Ticket ID
%J	Ticket	Description field from the SL1 ticket.

© 2003 - 2025, ScienceLogic, Inc.

All rights reserved.

LIMITATION OF LIABILITY AND GENERAL DISCLAIMER

ALL INFORMATION AVAILABLE IN THIS GUIDE IS PROVIDED "AS IS," WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED. SCIENCELOGIC™ AND ITS SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT.

Although ScienceLogic™ has attempted to provide accurate information on this Site, information on this Site may contain inadvertent technical inaccuracies or typographical errors, and ScienceLogic™ assumes no responsibility for the accuracy of the information. Information may be changed or updated without notice. ScienceLogic™ may also make improvements and / or changes in the products or services described in this Site at any time without notice.

Copyrights and Trademarks

ScienceLogic, the ScienceLogic logo, and EM7 are trademarks of ScienceLogic, Inc. in the United States, other countries, or both.

Below is a list of trademarks and service marks that should be credited to ScienceLogic, Inc. The ® and ™ symbols reflect the trademark registration status in the U.S. Patent and Trademark Office and may not be appropriate for materials to be distributed outside the United States.

- ScienceLogic™
- EM7™ and em7™
- Simplify IT™
- Dynamic Application™
- Relational Infrastructure Management™

The absence of a product or service name, slogan or logo from this list does not constitute a waiver of ScienceLogic's trademark or other intellectual property rights concerning that name, slogan, or logo.

Please note that laws concerning use of trademarks or product names vary by country. Always consult a local attorney for additional guidance.

Other

If any provision of this agreement shall be unlawful, void, or for any reason unenforceable, then that provision shall be deemed severable from this agreement and shall not affect the validity and enforceability of any remaining provisions. This is the entire agreement between the parties relating to the matters contained herein.

In the U.S. and other jurisdictions, trademark owners have a duty to police the use of their marks. Therefore, if you become aware of any improper use of ScienceLogic Trademarks, including infringement or counterfeiting by third parties, report them to Science Logic's legal department immediately. Report as much detail as possible about the misuse, including the name of the party, contact information, and copies or photographs of the potential misuse to: legal@sciencelogic.com. For more information, see <https://sciencelogic.com/company/legal>.

ScienceLogic

800-SCI-LOGIC (1-800-724-5644)

International: +1-703-354-1010