Power Actions User's Guide

vmware®

You can find the most up-to-date technical documentation on the VMware website at:

https://docs.vmware.com/

VMware, Inc.
3401 Hillview Ave.
Palo Alto, CA 94304
www.vmware.com

Contents

Introduction to Power Actions

1

Power Actions is a remote plug-in that enables you to run PowerCLI scripts from the vSphere Client, and implement custom actions on the vSphere inventory objects.

Power Actions integrates the PowerCLI in the vSphere client to provide automation capabilities for the standard vSphere management client. Power Actions is a remote plug-in that enables you to leverage the PowerCLI capabilities from the vSphere Client. For example, as an administrator, you can define a new action for a virtual machine object by writing a PowerCLI script that accepts a virtual machine as a parameter and saving it into a script/content library. You can then share it with other users within your company who can execute the script as-is, without requiring PowerShell knowledge.

Power Actions is available for download as a virtual appliance from the VMware Flings website and can be installed through the vSphere client plug-in installer.

Installing Power Actions 2

Power Actions is available for download as a virtual appliance from the VMware Flings website and can be installed through the vSphere client plug-in installer.

Read the following topics next:

- Supported VMware Products
- Install Power Actions
- Uninstall Power Actions

Supported VMware Products

You can use Power Actions starting from vSphere 7.0 Update 3i and later versions.

Install Power Actions

Install Power Actions in your vSphere client.

Prerequisites

- vCenter Server 7.0 U3i or later
- An ESXi host with at least 4 CPUs
- A datastore with at least 100GB of free disk space (if you use thick provisioning)
- vCenter user with administrative priviledges

Procedure

1. Log in to the vSphere Client of the vCenter Server system on which you want to install Power Actions.
2. From the **Home** menu, select **Administration**.
3. Under **Solutions**, click **Client plug-ins**.
4. In the **Client plug-ins** pane, click **Add**.

   The **Install new solution** wizard opens.

5 On the **Select an OVF template** page, specify the location of the Power Actions OVA file. You can choose between two options:

Option	Description
URL	Enter the URL to the OVA file located on the Internet. The only supported URL sources are HTTPS.
Local file	Click Upload files and select the Power Actions OVA file.

6 Click **Next**

7 On the **Select a name and folder** page, enter a unique name for the virtual machine, select a deployment location, and click **Next**.

8 On the **Select a compute resource** page, select a resource on which to to run the deployed client plug-in VM, and click **Next**.

9 On the **Review details** page, verify the client plug-in details and click **Next**.

10 On the **License agreement** page, accept the end-user license agreements and click **Next**.

11 On the **Select storage** page, define where and how the files for the deployed client plug-in are stored.

   a Select a storage policy for the virtual machine. This option is available only if storage policies are enabled on the destination resource.

   b Select a datastore to store the deployed client plug-in. The plug-in manifest file and virtual disk files are stored on the datastore. Select a datastore large enough to accommodate the virtual machine and all associated virtual disk files.

   c Select a virtual disk format.

   d (Optional) Select the Disable Storage DRS for this virtual machine check box to deactivate Storage DRS for the virtual machine.

12 Click **Next**

13 On the **Select networks** page, select a source network and map it to a destination network. Click **Next**.

14 On the **Customize template** page, specify the following settings of the Power Actions plug-in.

   a In the vCenter server address field, enter the IP address or the FQDN of the vCenter Server system on which you want to use Power Actions.

   b In the vCenter username field, enter the user name you want to use for the Power Actions registration. This user must have administrative privileges.

   c In the vCenter password field, enter the password of the user from step b.

   d If your vCenter environment is using self-signed certificates, select the Disable vCenter Server Certificate Verification option.

e (Optional) If you have previously installed Power Actions on this vCenter environment, select the Clean vCenter Server option. It will remove the previous appliance registration from the vCenter Server if it hasn't been removed properly.

f In the Root password field, enter a password that you want to set for the default root account of the appliance. The password must be at least eight characters long. You can use uppercase, lowercase, and special characters but no common dictionary words. If a non-compliant password is provided the appliance password will be set to the default value - 'vmware'. Example: P@ssword123!

g In the Admin Username field, enter the user name that you want to set for the administrative account for the Power Actions API. Example: administrator

h In the Admin Password field, enter the password that you want to set for the administrative account for the Power Actions API. Example: P@ssword123!

i (Optional) To enable more detailed installation logging, select the Debugging option

j (Optional) In the Appliance networking section, enter the networking settings of the appliance.

k (Optional) In the Proxy settings section, enter the proxy settings of the appliance

15 Click **Next**

16 On the **Associate vCenter Servers** page, select the vCenter Server instances on which the client plug-in is deployed and click **Next**.

17 On the **Ready to complete** page, review your selections and click **Finish**.

As a result new tasks for downloading and installing the client plug-in appear in the Recent Tasks pane.

18 After successful completion of the tasks, refresh your browser to see the newly added client plug-in.

Uninstall Power Actions

If you decide you no longer need Power Actions, you can uninstall it from your vSphere Client.

Procedure

1 Unregister the Power Actions appliance from your vCenter Server by using the Power Actions Admin API.

- Invoke the API using PowerShell

```
$APPLIANCE_IP='<power-actions-appliance-ip>'
$APPLICANCE_ADMIN_USERNAME='<admin-api-administrator-username>'
$APPLICANCE_ADMIN_PASSWORD=ConvertTo-SecureString -String '<admin-api-administrator-
password>' -AsPlainText -Force
$VCENTER_ADDRESS='<vCenter-server-address>'
$VCENTER_THUMBPRINT='<the-thumbprint-of-the-vCenter-server-certificate>'
$VCENTER_USERNAME='<vCenter-server-administrator-username>'
```

```
$VCENTER_PASSWORD='<vCenter-server-administrator-password>'
$creds = New-Object
System.Management.Automation.PSCredential($APPLICANCE_ADMIN_USERNAME,
$APPLICANCE_ADMIN_PASSWORD)
Invoke-RestMethod -Method Delete `
-Uri "https://$APPLIANCE_IP/admin/vc-registration?true" `
-SkipCertificateCheck `
-ContentType "application/json" `
-Body "{`"address`": `"$VCENTER_ADDRESS`", `"thumbprint`": `"$VCENTER_THUMBPRINT`",
`"username`": `"$VCENTER_USERNAME`", `"password`": `"$VCENTER_PASSWORD`"}" `
-Credential $creds
```

- Invoke the API using bash

```
APPLIANCE_IP='<power-actions-appliance-ip>'
APPLICANCE_ADMIN_USERNAME='<admin-api-administrator-username>'
APPLICANCE_ADMIN_PASSWORD='<admin-api-administrator-password>'
VCENTER_ADDRESS='<vCenter-server-address>'
VCENTER_THUMBPRINT='<the-thumbprint-of-the-vCenter-server-certificate>'
VCENTER_USERNAME='<vCenter-server-administrator-username>'
VCENTER_PASSWORD='<vCenter-server-administrator-password>'
curl -k -i \
-X DELETE \
-H "Content-Type: application/json" \
-u "${APPLICANCE_ADMIN_USERNAME}:${APPLICANCE_ADMIN_PASSWORD}" \
"https://${APPLIANCE_IP}/admin/vc-registration?true" \
-d "{\"address\": \"${VCENTER_ADDRESS}\", \"thumbprint\": \"${VCENTER_THUMBPRINT}\",
\"username\": \"${VCENTER_USERNAME}\", \"password\": \"${VCENTER_PASSWORD}\"}"
```

You removed the Power Actions appliance registration from your vCenter Server system.

2 Remove the Power Actions plug-in from the vSphere Client.

a From the **Home** menu, select **Administration**.

b Under **Solutions**, click **Client plug-ins**.

c From the Name column in the **Client plug-ins** pane, select Power Actions.

The detailed client plug-in view opens.

d Check the Power Actions client plug-in instance and click **Remove**.

e In the confirmation dialog box, click **Yes**.

3 Remove the Power Actions appliance.

a Power-off the Power Actions appliance.

b Remove (Delete from disk) the Power Actions appliance.

Accessing Power Actions

3

After the installation, you can access Power Actions through the vSphere Developer Center.

Prerequisites

Power Actions is successfully installed

Procedure

1. From the **Home** menu, select **Developer Center**.
2. Select the **Power Actions** tab.

Creating a script library

4

You can create a script library to store your Power Actions scripts.

Script libraries are actually content libraries, so any content library can be used as a script library. If you don't have a content library at your disposal, you can create a new one following the vSphere product documentation .

Note It is recommended to use dedicated content libraries as Script libraries to mitigate the risk of running an arbitrary file as a script action.

Managing scripts in the script library

5

Power Actions enables you to manage the content of a Script library. You can add new scripts to the library and update/replace, export, or remove scripts or examine their content.

Read the following topics next:

- Add a new script
- Check script content
- Update a script
- Export a script from a script library
- Remove a script from a script library

Add a new script

To run scripts with Power Actions, you must first add them to your script library.

Prerequisites

- An existing script library (content library)
- A script file (.ps1) that you want to upload to the script library

Procedure

1. Open Power Actions.
2. From the Script Library dropdown menu, select the script library to which you want to add your scripts.
3. Click **Import**

   The **Import Library Item** dialog box opens.

4 In the Source section, choose the source of the script. Choose between the following options:

Option	Description
Import from a URL	Enter the path to the Web server where the script is.
Import from a Local File	Click Browse to navigate to a file that you want to import from your local system. Then after you select the file, click anywhere in the dialog box to populate the item name.

5 (Optional) In the **Destination** section, enter a name and a description for the item.

6 Click **Import**.

Results

In the **Recent Tasks** pane you see two tasks, one about creating a new item in the library, and the second about uploading the contents of the script to the library. After the task is complete, the script appears in the script library.

Check script content

With Power Actions, you can check the content of a script in your script library.

Procedure

1 Open the Power Actions script library.

2 Click the **>>** button of the script that you want to check.

The script content opens in the right pane.

3 To close the right pane click the **<<** button for the script that is displayed.

Update a script

With Power Actions, you can update/replace a script in your script library.

Procedure

1 Select the script that you want to update from the selected script library.

2 Click **Update**.

The **Import Library Item** dialog box opens.

3 In the Source section, choose the source of the script.

4 (Optional) In the Destination section, enter a name and a description for the item.

5 Click **Import**.

Export a script from a script library

With Power Actions, you can download a script from a script library by exporting it.

Procedure

1. Select the script that you want to export from the selected script library.
2. Click **Export**.

   The **Save File** dialog opens.
3. Select the location to export the script and click **Save**.

Remove a script from a script library

Procedure

1. Select the script that you want to remove from the selected script library.
2. Click **Delete**.

Running scripts

6

The main feature of Power Actions is to run pre-defined scripts from a script library. A script runs in an isolated PowerShell runspace within the Power Actions appliance.

You can think of a runspace as of a separate PowerShell session. After you run a script, you can monitor its progress on the **Script Runs** page. All recent scripts that are running or have been run by the user are listed there.

Read the following topics next:

- Run a script from a script library
- Script parameters
- Run scripts from object context menu
- Using advanced parameter mode

Run a script from a script library

With Power Actions, you can run PowerShell scripts from a script library.

Procedure

1. Select the script that you want to run from the selected script library.
2. Click **Run**.

   If the script has no parameters, it is executed immediately. If the script has parameters, the **Enter Parameters** dialog appears.
3. (Optional) Enter the script parameters and click **Ok**.

Results

The script is executed. You can monitor the progress on the **Script Runs** page.

Script parameters

The scripts that are added to the script library may have parameters. Power Actions provides integrated experience in the vSphere client for some predefined parameter types.

Scripts might or might not have parameters. Parameters are defined through the standard `param` statement. These can be of any valid PowerShell type. Power Actions provides special handling for some parameter types, for a better user experience in the vSphere Client.

The following example script filters the VMs in a data center that have at least one snapshot older than X days, where X is also a parameter:

```
param(
[VMware.VimAutomation.ViCore.Types.V1.Inventory.Datacenter]
$context,
[int]
$daysOld)

$context | Get-VM | Where { $_ | Get-Snapshot | Where { $_.Created.AddDays($daysOld) -lt (Get-
Date) } }
```

This script takes two parameters: a data center object, and an integer specifying how old the snapshot must be in days. You can use the parameters freely as variables in the script. Running the script brings up the **Enter Parameters** dialog. For “first-class” object types in the vSphere Client (e.g. VMs, hosts, data centers), an object selector control is displayed.

Power Actions does not pass any pipeline input to the script. The following table lists the mapping of the PowerCLI object types to vSphere Client object types for context menus and object selectors:

vSphere Client object type	PowerCLI type
Cluster	VMware.VimAutomation.ViCore.Types.V1.Inventory.Cluster
Datacenter	VMware.VimAutomation.ViCore.Types.V1.Inventory.Datacenter
Datastore	VMware.VimAutomation.ViCore.Types.V1.DatastoreManagement.Datastore
Datastore Cluster	VMware.VimAutomation.ViCore.Types.V1.DatastoreManagement.DatastoreCluster
Distributed Virtual Switch	VMware.VimAutomation.Vds.Types.V1.VDSwitch
Distributed Virtual Portgroup	VMware.VimAutomation.Vds.Types.V1.VDPortgroup
Folder	VMware.VimAutomation.ViCore.Types.V1.Inventory.Folder
Host	VMware.VimAutomation.ViCore.Types.V1.Inventory.VMHost
Host Profile	VMware.VimAutomation.ViCore.Types.V1.Host.Profile.VMHostProfile
Network	VMware.VimAutomation.ViCore.Types.V1.Network.Network
Opaque Network	VMware.VimAutomation.ViCore.Types.V1.Network.OpaqueNetwork

vSphere Client object type	PowerCLI type
Resource Pool	VMware.VimAutomation.ViCore.Types.V1.Inventory.ResourcePool
Virtual Machine	VMware.VimAutomation.ViCore.Types.V1.Inventory.VirtualMachine
vApp	VMware.VimAutomation.ViCore.Types.V1.Inventory.VApp
Content Library	VMware.VimAutomation.ViCore.Types.V1.ContentLibrary.ContentLibrary
Content Library Item	VMware.VimAutomation.ViCore.Types.V1.ContentLibrary.ContentLibraryItem

Run scripts from object context menu

With Power Actions, you can run a script from the context menu of an object in the vSphere Client.

When you run a script from the context menu, the first parameter that matches the selected context by type is automatically populated with the object.

For example, if you right-click on a datacenter, and run the example script from above, the datacenter object will be pre-populated with the object that you right-clicked on. The pre-populated context object can be subsequently changed in the **Enter Parameters** dialog.

Note When no parameter of the script matches the selected context by type you will get the following warning: "The selected script doesn't have a parameter that matches the type of the selected context object. The context is not relevant to the script and will be ignored."

Procedure

1. In the vSphere Client, right-click on the object that you want to run a script on.
2. From the context menu select **Power Actions** and then **Run script**.
3. Select the script that you want to run on the object.
4. If the script has other parameters than the object you're running it on, click the Enter parameters button and enter the rest of the parameters. The context object should be already pre-populated.
5. Click **Ok**.

Results

The script is executed and you can monitor its progress in the Script Runs window.

Using advanced parameter mode

With Power Actions, you can use the PowerCLI/PowerShell script output as an input to an action/script using the advanced parameter mode.

You can enter PowerCLI/PowerShell expressions as advanced mode parameter values. These expressions are evaluated in the runspace before the script invocation, and then the results are passed as parameter values to your script.

Procedure

1. Run a script with parameters by using the script library or the context menu.
2. In the **Enter parameters** dialog, turn on the **Advanced mode** toggle for the parameter for which you want to use advanced mode.
3. In the input text field, enter the PowerShell expression for the parameter.
4. Click **Ok**.

Results

The script is executed and you can monitor its progress in the **Script Runs** window.

Using the built-in PowerShell console

7

Power Actions features a fully functional PowerShell console with which you can run commands or scripts directly within your vSphere Client.

Procedure

- From the left pane of **Power Actions**, select **Console**.

  The PowerShell console loads and you can start running commands and scripts in it.

  Note Every time you navigate away from the console in the vSphere Client and then return, a new PowerShell session is started. The runspace however remains the same.

  If you want to reload the console with a completely new runspace click **Reload** and then confirm the reset console prompt

Monitor the progress of executed scripts

8

You can monitor the progress and check the output of the scripts that you executed in Power Actions by using the Script runs window.

Procedure

1 Open **Power Actions**.

2 Select the **Script runs** window.

You can see a list of all running and completed scripts.

Note The state of a completed script run is "success" if the script has completed without any terminating errors and "error" if a terminating error has occurred. Non-terminating errors don't set the state of a script run to "error".

Stop a running script

You can stop a running script from the **Script runs** window.

Procedure

1 Open the **Script runs** window.

2 Select a script that is in a running state.

3 Click **Stop**.

Check the output of script runs

You can check the output of script runs that have completed.

Procedure

1 Open the **Script runs** window.

2 Select a script run that has completed (the state can be either success or error).

3 Click on the >> button of the script run.

You will see the script output in the right pane.

4 To close the pane with the script output, click the << button of the script run.

Configuring Power Actions

9

You can use the Power Actions Admin API to modify the Power Actions settings.

Power Actions has two sets of settings:

- runspace-related settings
- script run retention settings.

You can modify these settings to improve the performance and scalability of the plugin. There is no user interface for updating these settings at the moment. You can only modify these settings by using the Power Actions Admin API.

Read the following topics next:

- Modify runspace-related settings
- Modify script run retention settings

Modify runspace-related settings

You can use the runspace-related settings to optimize Power Actions scalability and performance.

Every script that you run in Power Actions creates a separate runspace - this is the PowerShell instance in which the scripts run. Additionally, the built-in console also creates a runspace. Every runspace consumes system resources on the Power Actions appliance. Use the runspace-related settings to optimize Power Actions scalability and performance.

You can use the following runspace-related settings:

Name	Description	Default value
max_number_of_runspaces	The maximum number of active runspaces.	10
max_runspace_idle_time_minutes	The time (in minutes) after which Power Actions deletes idle runspaces.	2
max_runspace_active_time_minutes	The time (in minutes) after which Power Actions will delete a runspace even if it's still active	60

The following examples illustrate how to retrieve runspace-related settings by using the Power Actions Admin API.

- Retrieve the settings by using PowerShell

```
$APPLIANCE_IP='<power-actions-appliance-ip>'
$APPLICANCE_ADMIN_USERNAME='<admin-api-administrator-username>'
$APPLICANCE_ADMIN_PASSWORD=ConvertTo-SecureString -String '<admin-api-administrator-
password>' -AsPlainText -Force
$creds = New-Object System.Management.Automation.PSCredential($APPLICANCE_ADMIN_USERNAME,
$APPLICANCE_ADMIN_PASSWORD)
Invoke-RestMethod -Method Get `
-Uri "https://$APPLIANCE_IP/admin/runspaces/provider-settings" `
-SkipCertificateCheck ` -Authentication Basic ` -Credential $creds
```

- Retrieve the settings by using Bash

```
APPLIANCE_IP='<power-actions-appliance-ip>'
APPLICANCE_ADMIN_USERNAME='<admin-api-administrator-username>'
APPLICANCE_ADMIN_PASSWORD='<admin-api-administrator-password>'
curl -k -i \
-X PATCH \
-H "Content-Type: application/json" \
-u "${APPLICANCE_ADMIN_USERNAME}:${APPLICANCE_ADMIN_PASSWORD}" \
"https://${APPLIANCE_IP}/admin/runspaces/provider-settings"
```

The following examples illustrate how to use the Power Actions Admin API to modify the runspace-related settings.

- Modify the settings by using PowerShell

```
$APPLIANCE_IP='<power-actions-appliance-ip>'
$APPLICANCE_ADMIN_USERNAME='<admin-api-administrator-username>'
$APPLICANCE_ADMIN_PASSWORD=ConvertTo-SecureString -String '<admin-api-administrator-
password>' -AsPlainText -Force
$MAX_RUNSPACES_NUMBER = 5
$MAX_RUNSPACES_IDLE_TIME = 10 # In minutes
$MAX_RUNSPACES_ACTIVE_TIME = 60 # In minutes
$creds = New-Object System.Management.Automation.PSCredential($APPLICANCE_ADMIN_USERNAME,
$APPLICANCE_ADMIN_PASSWORD)
Invoke-RestMethod -Method Patch `
-Uri "https://$APPLIANCE_IP/admin/runspaces/provider-settings" `
-SkipCertificateCheck `
-Authentication Basic `
-Credential $creds `
-ContentType "application/json" `
-Body "{`"max_number_of_runspaces`": $MAX_RUNSPACES, `"max_runspace_idle_time_minutes`":
$MAX_RUNSPACES_IDLE_TIME, `"max_runspace_active_time_minutes`": $MAX_RUNSPACES_ACTIVE_TIME }"
```

- Modify the settings by using Bash

```
APPLIANCE_IP='<power-actions-appliance-ip>'
APPLICANCE_ADMIN_USERNAME='<admin-api-administrator-username>'
APPLICANCE_ADMIN_PASSWORD='<admin-api-administrator-password>'
```

```
MAX_RUNSPACES_NUMBER=5
MAX_RUNSPACES_IDLE_TIME=10 # In minutes
MAX_RUNSPACES_ACTIVE_TIME=60 # In minutes
curl -k -i \
-X PATCH \
-H "Content-Type: application/json" \
-u "${APPLICANCE_ADMIN_USERNAME}:${APPLICANCE_ADMIN_PASSWORD}" \
"https://${APPLIANCE_IP}/admin/runspaces/provider-settings" \
-d "{\"max_number_of_runspaces\": ${MAX_RUNSPACES_NUMBER},
\"max_runspace_idle_time_minutes\": ${MAX_RUNSPACES_IDLE_TIME},
\"max_runspace_active_time_minutes\": ${MAX_RUNSPACES_ACTIVE_TIME} }"
```

Modify script run retention settings

Use the script run retention settings to control how many script runs and for how long are stored on the Power Actions appliance.

You can use the following Power Actions script run retention settings:

Name	Description	Default value
max_number_of_scripts_per_user	The maximum number of script runs retained per user.	30
no_older_than_days	The time (in days) after which Power Actions deletes the script runs.	5

The following examples illustrate how to retrieve the script run retention settings by using the Power Actions Admin API.

- Retrieve the settings by using PowerShell

```
$APPLIANCE_IP='<power-actions-appliance-ip>'
$APPLICANCE_ADMIN_USERNAME='<admin-api-administrator-username>'
$APPLICANCE_ADMIN_PASSWORD=ConvertTo-SecureString -String '<admin-api-administrator-
password>' -AsPlainText -Force
$creds = New-Object System.Management.Automation.PSCredential($APPLICANCE_ADMIN_USERNAME,
$APPLICANCE_ADMIN_PASSWORD)
Invoke-RestMethod -Method Get `
-Uri "https://$APPLIANCE_IP/admin/script-executions/retention-policy" `
-SkipCertificateCheck `
-Authentication Basic `
-Credential $creds
```

- Retrieve the settings by using bash

```
APPLIANCE_IP='<power-actions-appliance-ip>'
APPLICANCE_ADMIN_USERNAME='<admin-api-administrator-username>'
APPLICANCE_ADMIN_PASSWORD='<admin-api-administrator-password>'
curl -k -i \
-X PATCH \
-H "Content-Type: application/json" \
-u "${APPLICANCE_ADMIN_USERNAME}:${APPLICANCE_ADMIN_PASSWORD}" \
"https://${APPLIANCE_IP}/admin/script-executions/retention-policy"
```

The following examples illustrate how to modify the script retention settings by using the Power Actions Admin API.

- Modify the settings by using PowerShell

```
$APPLIANCE_IP='<power-actions-appliance-ip>'
$APPLICANCE_ADMIN_USERNAME='<admin-api-administrator-username>'
$APPLICANCE_ADMIN_PASSWORD=ConvertTo-SecureString -String '<admin-api-administrator-
password>' -AsPlainText -Force
$MAX_SCRIPT_PER_USER = 30
$MAX_SCRIPT_AGE_DAYS = 10
$creds = New-Object System.Management.Automation.PSCredential($APPLICANCE_ADMIN_USERNAME,
$APPLICANCE_ADMIN_PASSWORD)
Invoke-RestMethod -Method Patch `
-Uri "https://$APPLIANCE_IP/admin/script-executions/retention-policy" `
-SkipCertificateCheck `
-Authentication Basic `
-Credential $creds `
-ContentType "application/json" `
-Body "{`"max_number_of_scripts_per_user`": $MAX_SCRIPT_PER_USER, `"no_older_than_days`":
$MAX_SCRIPT_AGE_DAYS }"
```

- Modify the settings by using Bash

```
APPLIANCE_IP='<power-actions-appliance-ip>'
APPLICANCE_ADMIN_USERNAME='<admin-api-administrator-username>'
APPLICANCE_ADMIN_PASSWORD='<admin-api-administrator-password>'
MAX_SCRIPT_PER_USER=30
MAX_SCRIPT_AGE_DAYS=10
curl -k -i \
-X PATCH \
-H "Content-Type: application/json" \
-u "${APPLICANCE_ADMIN_USERNAME}:${APPLICANCE_ADMIN_PASSWORD}" \
"https://${APPLIANCE_IP}/admin/script-executions/retention-policy" \
-d "{\"max_number_of_scripts_per_user\": ${MAX_SCRIPT_PER_USER}, \"no_older_than_days\": $
{MAX_SCRIPT_AGE_DAYS} }"
```

Troubleshooting Power Actions

10

Read the following topics next:

- Troubleshooting Power Actions installation

Troubleshooting Power Actions installation

When Power Actions installation fails, the status of the Power Actions plug-in at the Client plug-ins pane is set to Failed. Your first action must be to see the Power Actions appliance console. All errors that occurred during installation are colored in red and marked as ERROR. If the installation output has been pushed out by another output, you can collect the installation log. To get a more detailed installation log, we encourage you to enable the Debugging installation parameter.

Collecting the installation log

The installation log is located in /var/log/bootstrap.log on the Power Actions appliance.

Collecting the runtime logs

The runtime logs are located in /var/log/power-actions/*.log on the Power Actions appliance.