Cisco UCS Director PowerShell Console Integration Guide, Release 6.5

First Published: 2017-07-11

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883
The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB's public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

NOTwithstanding any other warranty herein, all document files and software of these suppliers are provided "AS IS" with all faults. Cisco and the above-named suppliers disclaim all warranties, expressed or implied, including, without limitation, those of merchantability, fitness for a particular purpose and noninfringement or arising from a course of dealing, usage, or trade practice.

In no event shall Cisco or its suppliers be liable for any indirect, special, consequential, or incidental damages, including, without limitation, lost profits or loss or damage to data arising out of the use or inability to use this manual, even if Cisco or its suppliers have been advised of the possibility of such damages.

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

Cisco and the Cisco logo are trademarks or registered trademarks of Cisco and/or its affiliates in the U.S. and other countries. To view a list of Cisco trademarks, go to this URL: http://www.cisco.com/go/trademarks. Third-party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1110R)

© 2017 Cisco Systems, Inc. All rights reserved.
# CONTENTS

## Preface
Preface v
- Audience v
- Conventions v
- Related Documentation vii
- Documentation Feedback vii
- Obtaining Documentation and Submitting a Service Request vii

## Chapter 1
New and Changed Information 1
- New and Changed Information 1

## Chapter 2
Overview 3
- Cisco UCS Director PowerShell Console 3

## Chapter 3
Installing and Configuring 5
- System Requirements 5
- Downloading and Installing Cisco UCS Director PowerShell Console 5
- Cisco UCS Director PowerShell Console Configuration 6

## Chapter 4
Working with Cmdlets 7
- Cmdlets List 7
- Help for Cmdlets 8
- Cmdlets Inputs Definition 9
- Cmdlet Output Types 9
- Cmdlets for Viewing Reports 10
- Examples for Executing Cmdlets 11
  - Example: Obtaining Parameters of a Cmdlet using the Help Command 11
  - Example: Capturing and Filtering cmdlet Output 12
Example: Displaying a Tabular Report 12
Example: Canceling a Service Request 13

CHAPTER 5 Troubleshooting 15

Connection Exception 15
Running Script Disabled 16
Cmdlet Execution Failed 16
Preface

• Audience, page v
• Conventions, page v
• Related Documentation, page vii
• Documentation Feedback, page vii
• Obtaining Documentation and Submitting a Service Request, page vii

Audience

This guide is intended primarily for data center administrators who use Cisco UCS Director and who have responsibilities and expertise in one or more of the following:

• Server administration
• Storage administration
• Network administration
• Network security
• Virtualization and virtual machines

Conventions

<table>
<thead>
<tr>
<th>Text Type</th>
<th>Indication</th>
</tr>
</thead>
<tbody>
<tr>
<td>GUI elements</td>
<td>GUI elements such as tab titles, area names, and field labels appear in this font.</td>
</tr>
<tr>
<td></td>
<td>Main titles such as window, dialog box, and wizard titles appear in this font.</td>
</tr>
<tr>
<td>TUI elements</td>
<td>In a Text-based User Interface, text the system displays appears in this font.</td>
</tr>
<tr>
<td>Text Type</td>
<td>Indication</td>
</tr>
<tr>
<td>-----------</td>
<td>------------</td>
</tr>
<tr>
<td><strong>System output</strong></td>
<td>Terminal sessions and information that the system displays appear in this font.</td>
</tr>
</tbody>
</table>
| **CLI commands** | CLI command keywords appear in this font.  
Variables in a CLI command appear in this font. |
| [] | Elements in square brackets are optional. |
| {x | y | z} | Required alternative keywords are grouped in braces and separated by vertical bars. |
| [x | y | z] | Optional alternative keywords are grouped in brackets and separated by vertical bars. |
| **string** | A nonquoted set of characters. Do not use quotation marks around the string or the string will include the quotation marks. |
| <> | Nonprinting characters such as passwords are in angle brackets. |
| [] | Default responses to system prompts are in square brackets. |
| !, # | An exclamation point (!) or a pound sign (#) at the beginning of a line of code indicates a comment line. |

---

**Note**

Means *reader take note*. Notes contain helpful suggestions or references to material not covered in the document.

**Caution**

Means *reader be careful*. In this situation, you might perform an action that could result in equipment damage or loss of data.

**Tip**

Means *the following information will help you solve a problem*. The tips information might not be troubleshooting or even an action, but could be useful information, similar to a Timesaver.

**Timesaver**

Means *the described action saves time*. You can save time by performing the action described in the paragraph.
Related Documentation

Cisco UCS Director Documentation Roadmap

For a complete list of Cisco UCS Director documentation, see the Cisco UCS Director Documentation Roadmap available at the following URL: http://www.cisco.com/en/US/docs/unified_computing/ucs/ucs-director/doc-roadmap/b_UCSDirectorDocRoadmap.html.

Cisco UCS Documentation Roadmaps

For a complete list of all B-Series documentation, see the Cisco UCS B-Series Servers Documentation Roadmap available at the following URL:  http://www.cisco.com/go/unifiedcomputing/b-series-doc.

For a complete list of all C-Series documentation, see the Cisco UCS C-Series Servers Documentation Roadmap available at the following URL: http://www.cisco.com/go/unifiedcomputing/c-series-doc.

Note

The Cisco UCS B-Series Servers Documentation Roadmap includes links to documentation for Cisco UCS Manager and Cisco UCS Central. The Cisco UCS C-Series Servers Documentation Roadmap includes links to documentation for Cisco Integrated Management Controller.

Documentation Feedback

To provide technical feedback on this document, or to report an error or omission, please send your comments to ucs-director-docfeedback@cisco.com. We appreciate your feedback.

Obtaining Documentation and Submitting a Service Request

For information on obtaining documentation, using the Cisco Bug Search Tool (BST), submitting a service request, and gathering additional information, see What's New in Cisco Product Documentation.

To receive new and revised Cisco technical content directly to your desktop, you can subscribe to the What's New in Cisco Product Documentation RSS feed. RSS feeds are a free service.
New and Changed Information

The following table provides an overview of the significant changes to this guide for this current release.

Table 1: New and Changed Information for release 5.4

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
<th>Where Documented</th>
</tr>
</thead>
<tbody>
<tr>
<td>Additional version support</td>
<td>Additional version support has been added for this release. This support is backwards compatible. There are no changes in feature support from the previous release of Cisco UCS Director.</td>
<td>Cisco UCS Director Compatibility Matrix</td>
</tr>
</tbody>
</table>
Overview

Cisco UCS Director PowerShell Console

Cisco UCS Director offers REST APIs that enable applications to consume or manipulate the data stored in Cisco UCS Director. These APIs are used by any script capable of making HTTP or HTTPS requests to invoke operations on Cisco UCS Director.

Cisco UCS Director PowerShell Console is a Cisco developed application that provides a PowerShell interface to the Cisco UCS Director API. It is a connector that enables you to connect to Cisco UCS Director systems without a need to develop compiled code. The Console provides a set of PowerShell cmdlets wrapped in a module which then internally invokes REST APIs over HTTP. The console supports only JSON-based APIs.

A cmdlet runs natively in the PowerShell CLI session on a compatible Windows machine. Each cmdlet performs a single operation. Depending on the data returned by the JSON-based APIs, the cmdlet automatically interprets the data and converts it into Windows PowerShell object. You can chain multiple cmdlets together. A cmdlet supports common operations such as exporting to XML, searching, sorting, filtering, reporting, and more.

To view a list of available cmdlets, see Cmdlets List, on page 7. For more information about REST APIs, see Cisco UCS Director REST API Getting Started Guide.
Cisco UCS Director PowerShell Console is different from Cisco UCS Director PowerShell Agent. PowerShell Console provides cmdlet wrappers for the JSON-based APIs that enable you to submit workflow requests, examine workflow inputs and output schemas, and fetch reports. PowerShell Agent provides an interface between Cisco UCS Director and any Windows machine that is included in the WinRM configuration to add automation and orchestration functionality to Cisco UCS Director.
Installing and Configuring

This chapter contains the following sections:

- System Requirements, page 5
- Downloading and Installing Cisco UCS Director PowerShell Console, page 5
- Cisco UCS Director PowerShell Console Configuration, page 6

System Requirements

Windows PowerShell is built on top of the .NET Framework common language runtime (CLR) and the .NET Framework. It accepts and returns .NET Framework objects.

To work with PowerShell, you must install the Cisco UCS Director PowerShell console on any Windows-based system that supports the following:

- Microsoft .NET Framework 4.0 or higher
- Microsoft Windows PowerShell version 3.0 or higher

Downloading and Installing Cisco UCS Director PowerShell Console

The Cisco UCS Director PowerShell Console is an executable file that is installed as an application on the desktop. You can download the file from Cisco software download site. To install Cisco UCS Director PowerShell Console, do the following:

Step 1

Navigate to https://software.cisco.com/download/release.html to download the latest version of Cisco UCS Director PowerShell Console executable file to your Windows desktop. Only registered customers can download the software.
Step 2  Double-click the executable file to launch the installer and follow the instructions to install the PowerShell Console.

Step 3  Verify that Cisco UCS Director module has been installed by using the `Get-Module` command. The command displays the type, version, and name of the Cisco UCS Director module.

**Example:**

```
PS C:\Program Files (x86)\Cisco\Cisco UCS Director PowerShell Console\Modules\CiscoUcsdPS> Get-Module

ModuleType Version Name ExportedCommands
---------- ------- ---- ---------------
Binary 5.3.1.0 CiscoUcsdPS {Invoke-userAPICan...

PS C:\Program Files (x86)\Cisco\Cisco UCS Director PowerShell Console\Modules\CiscoUcsdPS>
```

**Note**  The example shows the installed Cisco UCS Director PowerShell module to be version 5.3.1.0. The version number may vary when you have a different version installed (for example, if you had upgraded to a newer version).

Step 4  Alternatively, you can download the PowerShell Console executable file directly from Cisco UCS Director by navigating to **Administration > Downloads**.

---

**Cisco UCS Director PowerShell Console Configuration**

You must configure the environment variables in the PowerShell console to run the cmdlets against the Cisco UCS Director server. All cmdlets accept the IP address of Cisco UCS Director and the REST Key as optional parameters. If you do not specify these parameters, the cmdlets choose their targets from the PowerShell environment variables UCSD_SERVER and UCSD_RESTKEY.

You can configure the environment variables using the `SetEnvironmentVariable` method as follows:

```
[Environment]::SetEnvironmentVariable("UCSD_SERVER","10.1.1.1","User")
[Environment]::SetEnvironmentVariable("UCSD_RESTKEY","562FDF763A384E78B9BAB7FE02CA13B6","User")
```

You can retrieve the configured environment variables using the `GetEnvironmentVariable` command as follows:

```
[Environment]::GetEnvironmentVariable("UCSD_RESTKEY","User")
```

**Note**  For environment variable changes to take effect, close the PowerShell console and open it again.
CHAPTER 4

Working with Cmdlets

This chapter contains the following sections:

- Cmdlets List, page 7
- Help for Cmdlets, page 8
- Cmdlets Inputs Definition, page 9
- Cmdlet Output Types, page 9
- Cmdlets for Viewing Reports, page 10
- Examples for Executing Cmdlets, page 11

Cmdlets List

You can view the list of available cmdlets by using `Get-Command`.

The following list of cmdlets was obtained by using the `Get-Command`:

```powershell
Get-Command -Module CiscoUcsdPS
```

<table>
<thead>
<tr>
<th>CommandType</th>
<th>Name</th>
<th>ModuleName</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cmdlet</td>
<td>Invoke-UserAPICancelServiceRequest</td>
<td>CiscoUcsdPS</td>
</tr>
<tr>
<td>Cmdlet</td>
<td>Invoke-UserAPIExecuteVMAction</td>
<td>CiscoUcsdPS</td>
</tr>
<tr>
<td>Cmdlet</td>
<td>Invoke-UserAPIGetAllCatalogs</td>
<td>CiscoUcsdPS</td>
</tr>
<tr>
<td>Cmdlet</td>
<td>Invoke-UserAPIGetAllGroups</td>
<td>CiscoUcsdPS</td>
</tr>
<tr>
<td>Cmdlet</td>
<td>Invoke-UserAPIGetAllVDCs</td>
<td>CiscoUcsdPS</td>
</tr>
<tr>
<td>Cmdlet</td>
<td>Invoke-UserAPIGetAvailableReports</td>
<td>CiscoUcsdPS</td>
</tr>
<tr>
<td>Cmdlet</td>
<td>Invoke-UserAPIGetHistoricalReport</td>
<td>CiscoUcsdPS</td>
</tr>
<tr>
<td>Cmdlet</td>
<td>Invoke-UserAPIGetInstantDataReport</td>
<td>CiscoUcsdPS</td>
</tr>
<tr>
<td>Cmdlet</td>
<td>Invoke-UserAPIGetPage</td>
<td>CiscoUcsdPS</td>
</tr>
<tr>
<td>Cmdlet</td>
<td>Invoke-UserAPIGetServiceRequestDetails</td>
<td>CiscoUcsdPS</td>
</tr>
<tr>
<td>Cmdlet</td>
<td>Invoke-UserAPIGetServiceRequests</td>
<td>CiscoUcsdPS</td>
</tr>
<tr>
<td>Cmdlet</td>
<td>Invoke-UserAPIGetServiceRequestWorkflow</td>
<td>CiscoUcsdPS</td>
</tr>
<tr>
<td>Cmdlet</td>
<td>Invoke-UserAPIGetTabularReport</td>
<td>CiscoUcsdPS</td>
</tr>
<tr>
<td>Cmdlet</td>
<td>Invoke-UserAPIGetWorkflowRequests</td>
<td>CiscoUcsdPS</td>
</tr>
<tr>
<td>Cmdlet</td>
<td>Invoke-UserAPIGetWorkflowInputs</td>
<td>CiscoUcsdPS</td>
</tr>
<tr>
<td>Cmdlet</td>
<td>Invoke-UserAPIGetWorkflowInputValue</td>
<td>CiscoUcsdPS</td>
</tr>
<tr>
<td>Cmdlet</td>
<td>Invoke-UserAPIGetWorkflowInputValues</td>
<td>CiscoUcsdPS</td>
</tr>
<tr>
<td>Cmdlet</td>
<td>Invoke-UserAPIGetWorkflowStatus</td>
<td>CiscoUcsdPS</td>
</tr>
<tr>
<td>Cmdlet</td>
<td>Invoke-UserAPIReconfigureVM</td>
<td>CiscoUcsdPS</td>
</tr>
<tr>
<td>Cmdlet</td>
<td>Invoke-UserAPIRollbackflow</td>
<td>CiscoUcsdPS</td>
</tr>
</tbody>
</table>
Help for Cmdlets

You can run cmdlets in Cisco UCS Director PowerShell Console. To get help for a cmdlet, use the `?` or `get-Help` command.

When you enter `?` with the cmdlet, you see the syntax, parameters, aliases, and remarks for the cmdlet.

This example shows how to get help for a cmdlet:

```
PS C:\Program Files (x86)\Cisco\Cisco UCS Director PowerShell Console\Modules\CiscoUcsdPS> Invoke-userAPISubmitServiceRequest -?
NAME
   Invoke-userAPISubmitServiceRequest
SYNTAX
   Invoke-userAPISubmitServiceRequest [-catalogName] <string> [-vdcName] <string>
   [-durationHours] <int> [-beginTime] <long> [-quantity] <int> [-comments] <string> [-server <string>] [-restkey <string>]
ALIASES
   None
REMARKS
   None
```

Each cmdlet has mandatory and optional parameters. For example, the `Invoke-userAPISubmitServiceRequest` cmdlet has `catalogName` as a mandatory parameter, while `server` and `restKey` are optional parameters.

You can get detailed help about an individual parameter of a cmdlet when you use the `get-Help` command, the cmdlet name, and the parameter:

```
PS C:\Program Files (x86)\Cisco\Cisco UCS Director PowerShell Console\Modules\CiscoUcsdPS> get-Help Invoke-userAPISubmitServiceRequest -Parameter catalogName
-catalogName <string>

   Required? true
   Position? 0
   Accept pipeline input? false
   Parameter set name <All>
   Aliases None
   Dynamic? None
```

```
PS C:\Program Files (x86)\Cisco\Cisco UCS Director PowerShell Console\Modules\CiscoUcsdPS> get-Help Invoke-userAPISubmitServiceRequest -Parameter server
-server <string>

   Required? false
   Position? Named
```
Cmdlets Inputs Definition

The cmdlets that submit a workflow take an array of name-value pairs as workflow inputs. You must specify the input as an array of colon-separated name-value pairs:

Name1:value1, Name2:value2

For example, to submit a workflow named user-add-test, specify the input in the following way:

Invoke-userAPISubmitWorkflowServiceRequest user-add-test -parameters user-type:Regular,group-id:1,login-name:cmdlettest1,password:test,confirm-password:test,email:user@mail.com

Cmdlet Output Types

You can connect commands with the pipe operator (\(\)), to execute common options on the command output. The output of each command is used as input for the next command.

The following table lists the types of objects that are written to the pipeline by cmdlets.

<table>
<thead>
<tr>
<th>Cmdlet</th>
<th>Return Object Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Invoke-userAPICancelServiceRequest</td>
<td>Boolean</td>
</tr>
<tr>
<td>Invoke-userAPIExecuteVMAction</td>
<td>String</td>
</tr>
<tr>
<td>Invoke-userAPIGetAllCatalogs</td>
<td>APITabularReport</td>
</tr>
<tr>
<td>Invoke-userAPIGetAllGroups</td>
<td>APITabularReport</td>
</tr>
<tr>
<td>Invoke-userAPIGetAllVDCs</td>
<td>APITabularReport</td>
</tr>
<tr>
<td>Invoke-userAPIGetAvailableReports</td>
<td>Array of APIReportDefinition</td>
</tr>
<tr>
<td>Invoke-userAPIGetHistoricalReport</td>
<td>Array of HistoricalDataSeries</td>
</tr>
<tr>
<td>Invoke-userAPIGetInstantDataReport</td>
<td>APISnapshotReport</td>
</tr>
<tr>
<td>Invoke-userAPIGetPage</td>
<td>VMDATAViewPaginated</td>
</tr>
<tr>
<td>Invoke-userAPIGetServiceRequestDetails</td>
<td>APIServiceRequestDetails</td>
</tr>
<tr>
<td>Invoke-userAPIGetServiceRequests</td>
<td>APITabularReport</td>
</tr>
<tr>
<td>Invoke-userAPIGetServiceRequestWorkflow</td>
<td>APIWorkflowStatus</td>
</tr>
<tr>
<td>Invoke-userAPIGetTabularReport</td>
<td>APITabularReport</td>
</tr>
<tr>
<td>Invoke-userAPIGetVMActionRequests</td>
<td>APITabularReport</td>
</tr>
<tr>
<td>Invoke-userAPIGetWorkflowInputs</td>
<td>Array of APIWorkflowInputDetail</td>
</tr>
<tr>
<td>Invoke-userAPIGetWorkflowInputValue</td>
<td>String</td>
</tr>
</tbody>
</table>
Cmdlets for Viewing Reports

You can view reports by using the following cmdlets:

- Invoke-UserAPIGetAllCatalogs
- Invoke-UserAPIGetAllGroups
- Invoke-UserAPIGetAvailableReports
- Invoke-UserAPIGetHistoricalReports
- Invoke-UserAPIGetInstantDataReport
- Invoke-UserAPIGetPage
- Invoke-UserAPIGetServiceRequestDetails
- Invoke-UserAPIGetServiceRequests
- Invoke-UserAPIGetServiceRequestWorkflow
- Invoke-UserAPIGetTabularReport
- Invoke-UserAPIGetVMActionRequests
- Invoke-UserAPIGetWorkflowInputs
- Invoke-UserAPIGetWorkflowInputValue
- Invoke-UserAPIGetWorkflowInputValues

<table>
<thead>
<tr>
<th>Cmdlet</th>
<th>Return Object Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Invoke-userAPIGetWorkflowInputValues</td>
<td>WorkflowInputValue</td>
</tr>
<tr>
<td>Invoke-userAPIGetWorkflows</td>
<td>Array of CustomActionDefinition</td>
</tr>
<tr>
<td>Invoke-userAPIGetWorkflowStatus</td>
<td>Integer</td>
</tr>
<tr>
<td>Invoke-userAPIReconfigureVM</td>
<td>String</td>
</tr>
<tr>
<td>Invoke-userAPIRollbackWorkflow</td>
<td>SR ID</td>
</tr>
<tr>
<td>Invoke-userAPISubmitServiceRequest</td>
<td>SR ID</td>
</tr>
<tr>
<td>Invoke-userAPISubmitServiceRequestCustom</td>
<td>SR ID</td>
</tr>
<tr>
<td>Invoke-userAPISubmitWorkflowServiceRequest</td>
<td>SR ID</td>
</tr>
<tr>
<td>Invoke-userAPISubmitWorkflowServiceRequestWithGroup</td>
<td>SR ID</td>
</tr>
<tr>
<td>Invoke-userAPISubmitWorkflowServiceRequestWithStartTimeAndDurationHours</td>
<td>SR ID</td>
</tr>
<tr>
<td>Invoke-userAPIValidateWorkflow</td>
<td>APIWFValidationResult</td>
</tr>
<tr>
<td>Invoke-userAPIWorkflowInputDetails</td>
<td>APIWorkflowInputDetails</td>
</tr>
</tbody>
</table>
• Invoke-UserAPIGetWorkflows
• Invoke-UserAPIGetWorkflowStatus
• Invoke-UserAPIGetAllVDCs

The userAPIGetPage API requires the name of the paginated report as one of the parameters.

The following table provides the report name and its context value:

<table>
<thead>
<tr>
<th>Report Name</th>
<th>Context Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHARGEBACK_DETAILS_SERVICES_GLOBAL_TABULAR</td>
<td>None</td>
</tr>
<tr>
<td>PER_CLOUD_ARCHIVED_HYPERV_VM_LIST_REPORT</td>
<td>&lt;cloudName&gt;</td>
</tr>
<tr>
<td>vms.paginated.report</td>
<td>&lt;cloudName&gt;</td>
</tr>
<tr>
<td>PER_CLOUD_ARCHIVED_VM_LIST_REPORT</td>
<td>&lt;cloudName&gt;</td>
</tr>
</tbody>
</table>

For example, specify the following input to fetch a page of the vmware-account account report:

```
Invoke-userAPIGetPage vms.paginated.report vmware-account 1 10
```

### Examples for Executing Cmdlets

Cmdlets are executed in the same way as any other PowerShell command. You can apply common operations, such as search and filtering, to the outputs of the commands by using the pipe operator (|). The following examples illustrate how to execute cmdlets.

#### Example: Obtaining Parameters of a Cmdlet using the Help Command

Cmdlets are executed in the same way as any other PowerShell command. You can apply common operations, such as search and filtering, to the outputs of the commands by using the pipe operator (|). The following example shows how to obtain the parameters of a cmdlet using the `?` command.

```
PS C:\Program Files (x86)\Cisco\Cisco UCS Director PowerShell Console\Modules\CiscoUcsdPS> Invoke-userAPIGetWorkflowInputValues --?
NAME
   Invoke-userAPIGetWorkflowInputValues
SYNTAX
   Invoke-userAPIGetWorkflowInputValues [-srId] <int> [-server <string>] [-restkey <string>] [<CommonParameters>]
ALIASES
   None
REMARKS
   None
PS C:\Program Files (x86)\Cisco\Cisco UCS Director PowerShell Console\Modules\CiscoUcsdPS>
```
Example: Capturing and Filtering cmdlet Output

When you know the syntax of a cmdlet, you can capture the cmdlet output as shown below:

```
PS C:\Program Files (x86)\Cisco\Cisco UCS Director PowerShell Console\Modules\CiscoUcsdPS> Invoke-userAPIGetWorkflowInputValues 429
```

```
<table>
<thead>
<tr>
<th>Input Set Id</th>
<th>Action Id</th>
<th>Field Id</th>
<th>Field Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>18</td>
<td>0</td>
<td>WF_EMPTY_INPUTS</td>
<td>Regular</td>
</tr>
<tr>
<td>18</td>
<td>156</td>
<td>input_0_user-type471</td>
<td>cmdlettest1</td>
</tr>
<tr>
<td>18</td>
<td>156</td>
<td>AddUser_231.OUTUT_USER_NAME</td>
<td>test</td>
</tr>
<tr>
<td>18</td>
<td>156</td>
<td>input_4_confirm-password361</td>
<td><a href="mailto:user@mail.com">user@mail.com</a></td>
</tr>
<tr>
<td>18</td>
<td>156</td>
<td>input_1_group-id135</td>
<td>1</td>
</tr>
<tr>
<td>18</td>
<td>156</td>
<td>input_2_login-name867</td>
<td>cmdlettest1</td>
</tr>
</tbody>
</table>
```

You can further filter the output as shown below:

```
PS C:\Program Files (x86)\Cisco\Cisco UCS Director PowerShell Console\Modules\CiscoUcsdPS> Invoke-userAPIGetWorkflowInputValues 429 | Select-Object 'Field Id', 'Field Value'
```

```
<table>
<thead>
<tr>
<th>Field Id</th>
<th>Field Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>WF_EMPTY_INPUTS</td>
<td>Regular</td>
</tr>
<tr>
<td>input_0_user-type471</td>
<td>cmdlettest1</td>
</tr>
<tr>
<td>AddUser_231.OUTUT_USER_NAME</td>
<td>test</td>
</tr>
<tr>
<td>input_4_confirm-password361</td>
<td><a href="mailto:user@mail.com">user@mail.com</a></td>
</tr>
<tr>
<td>input_1_group-id135</td>
<td>1</td>
</tr>
<tr>
<td>input_2_login-name867</td>
<td>cmdlettest1</td>
</tr>
</tbody>
</table>
```

Example: Displaying a Tabular Report

The following example displays the object value for the Invoke-userAPIGetAllGroups API.

```
PS C:\Program Files (x86)\Cisco\Cisco UCS Director PowerShell Console\Modules\CiscoUcsdPS> Invoke-userAPIGetAllGroups
```

```
com.cisco.cuic.api.client.APITabularReport@307e168
```

Adding "-verbose" to the same syntax displays the list of groups in the form of a tabular report.

```
PS C:\Program Files (x86)\Cisco\Cisco UCS Director PowerShell Console\Modules\CiscoUcsdPS> Invoke-userAPIGetAllGroups -verbose
```

```
VERBOSE: Connected to the UCSD at 172.22.234.237, Getting the List of Groups...
VERBOSE: List of Groups returned by the UCSD.
```

```
GROUP_ID GROUP_NAME GROUP_CODE GROUP_DESCRIPTION SOURCE COST_CENTER GROUP_CONTACT_NAME GROUP_CONTACT_EMAIL_ADDRESS BUDGET_REQUIRED RESOURCE_LIMITS_SET LDAP_ACCOUNT HOST TAGS GROUP_SHARE_POLICY ALLOWS_RESOURCES_TO_USER_TAG
```

```
GROUP_ID GROUP_NAME GROUP_CODE GROUP_DESCRIPTION SOURCE COST_CENTER GROUP_CONTACT_NAME GROUP_CONTACT_EMAIL_ADDRESS BUDGET_REQUIRED RESOURCE_LIMITS_SET LDAP_ACCOUNT HOST TAGS GROUP_SHARE_POLICY ALLOWS_RESOURCES_TO_USER_TAG
```
Example: Canceling a Service Request

This section explains how you can cancel a service request through the PowerShell Console.

Before You Begin

Ensure that you have configured the environment variables in the PowerShell Console to run the cmdlets against the Cisco UCS Director server. For more information, see Cisco UCS Director PowerShell Console Configuration, on page 6.

Step 1

View the list of cmdlets that are available for use by using Get-Command. The cmdlet for canceling a service request is Invoke–userAPICancelServiceRequest.

Step 2

Get the parameters of the Invoke–userAPICancelServiceRequest cmdlet using the ? command.

The input required for executing the Invoke–userAPICancelServiceRequest cmdlet are:

- requestld—Mandatory parameter
- server—Optional parameter
- restKey—Optional parameter

Note You can use the get-Help command to get detailed help about the individual parameters of a cmdlet. For more information, see Help for Cmdlets, on page 8.
Step 3: Execute the command by passing the service request ID (79 in this example) as follows:

```
PS C:\Program Files (x86)\Cisco\Cisco UCS Director PowerShell Console\Modules\CiscoUcsdPS> Invoke-userAPICancelServiceRequest 79
```

This command returns a Boolean value as output. On successful cancellation of the service request, `True` is returned as output.

```
True
PS C:\Program Files (x86)\Cisco\Cisco UCS Director PowerShell Console\Modules\CiscoUcsdPS>
```
Troubleshooting

This chapter contains the following sections:

- Connection Exception, page 15
- Running Script Disabled, page 16
- Cmdlet Execution Failed, page 16

Connection Exception

Problem
A connection exception error occurs when invoking a PowerShell API.

Description
When invoking a PowerShell API, the following message appears:

```
PS C:\Program Files (x86)\Cisco\Cisco UCS Director PowerShell Console\Modules\CiscoUcsdPS> Invoke -userAPIGetAllCatalogs
INFO: I/O exception <java.net.ConnectException> caught when processing request:
Connection Refused: connect
INFO: Retrying Request
Invoke -userAPIGetAllCatalogs : Connection Refused: connect
+ Invoke -userAPIGetAllCatalogs
+ CategoryInfo : InvalidResult: <Couldn't get the Catalogs:String> [Invoke -userAPIGetAllCatalogs], ConnectException
+ FullyQualifiedErrorId : Couldn't get the Catalogs,CiscoUcsdPS.userAPIGetAllCatalogsCmdlet
```

Solution
Before invoking any PowerShell API, make sure that all the Cisco UCS Director services are running. If a service is down, restart the service and invoke the PowerShell API again.
Running Script Disabled

Problem
PowerShell console cannot be started.

Description
When starting Cisco UCS Director PowerShell Console, the following error message appears:

PS C:\Program Files (x86)\Cisco\Cisco UCS Director PowerShell Console\Modules\CiscoUcsdPS> C:\Windows\System32\windowspowershell\v1.0\powershell.exe -NoExit -File -\StartUcsdPS.ps1

Windows PowerShell
Copyright <c> 2013 Microsoft Corporation. All rights reserved.

File C:\Program Files <x86>\Cisco\Cisco UCS Director PowerShell Console\Modules\CiscoUcsdPS\StartUcsdPS.ps1 cannot be loaded because running scripts is disabled on this system. For more information, see about_Exeuction_Policies at http://go.microsoft.com/fwlink/?LinkID=135170.

+ CategoryInfo : SecurityError: <>: {} , ParentContainsErrorRecord Exception
+ FullyQualifiedErrorId : UnauthorizedAccess

Solution
The execution policy determines whether you can load configuration files (including your PowerShell profile) and run scripts. It also determines which scripts must be digitally signed before they are run. The **Set-ExecutionPolicy** cmdlet changes the user preference for the PowerShell execution policy.

To set the execution policy to **By**, start the Cisco UCS Director PowerShell Console and use the following command:

Set-ExecutionPolicy Unrestricted

Cmdlet Execution Failed

Problem
During the execution of cmdlet, the PowerShell console throws one of the following errors:

- The Cisco UCS Director server could not be reached.
- The operation timed out error.
- IP/REST Key can't be Empty.

Description
Even after configuring the environment variables UCSD_RESTKEY and UCSD_SERVER, the cmdlet has failed to execute.

PS C:\UCSDPowerShellTest\Modules\CiscoUcsdPS> [Environment]::SetEnvironmentVariable("UCSD_SERVER","10.1.1.1","User")
PS C:\UCSDPowerShellTest\Modules\CiscoUcsdPS> [Environment]::SetEnvironmentVariable("UCSD_RESTKEY","5E8DA3924FDB4CC49213FCAAE2CBCEEB","User")
PS C:\UCSDPowerShellTest\Modules\CiscoUcsdPS> Invoke-userAPIGetAllGroups
Invoke-userAPIGetAllGroups : IP/REST Key can't be Empty
At line:1 char:1
+ Invoke-userAPIGetAllGroups
+ ~~~~~~~~~~~~~~~~~~~~~~~~~
Solution

Changes to environment variables take effect only after you close the PowerShell and open it again. Whenever you modify the environment variables, close the PowerShell console and open it again.
Cmdlet Execution Failed