

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 SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

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

Text Type	Indication
GUI elements	GUI elements such as tab titles, area names, and field labels appear in **this font**. Main titles such as window, dialog box, and wizard titles appear in **this font**.
Document titles	Document titles appear in *this font*.
TUI elements	In a Text-based User Interface, text the system displays appears in `this font`.

Text Type	Indication
System output	Terminal sessions and information that the system displays appear in `this font`.
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.

Warning

IMPORTANT SAFETY INSTRUCTIONS

This warning symbol means danger. You are in a situation that could cause bodily injury. Before you work on any equipment, be aware of the hazards involved with electrical circuitry and be familiar with standard practices for preventing accidents. Use the statement number provided at the end of each warning to locate its translation in the translated safety warnings that accompanied this device.

SAVE THESE INSTRUCTIONS

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.

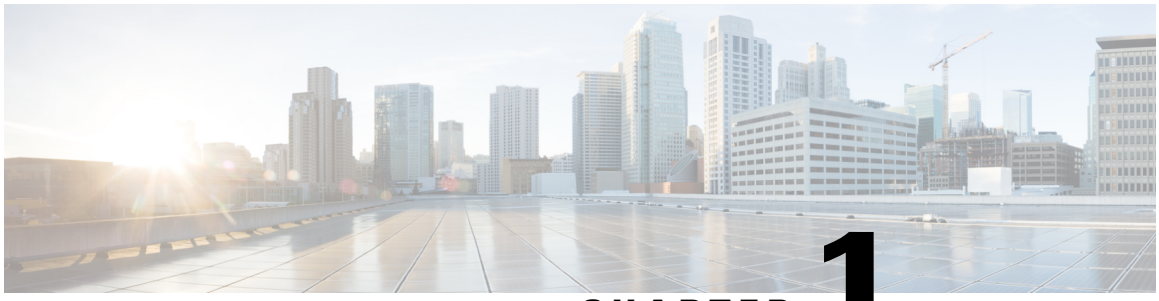
Documentation Feedback

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.



CHAPTER 1

New and Changed Information

- New and Changed Information, page 1

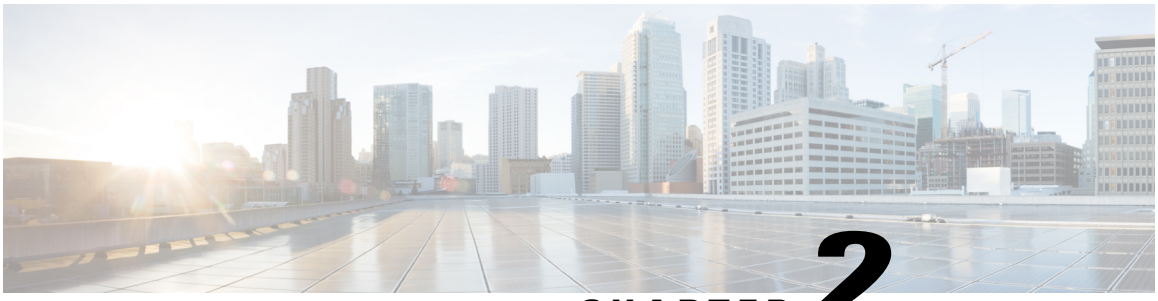
New and Changed Information

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

Feature	Description	Where Documented
Additional version support	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.	Cisco UCS Director Compatibility Matrix

New and Changed Information



CHAPTER 2

Overview

- Cisco UCS Director PowerShell Console, page 3

Cisco UCS Director PowerShell Console

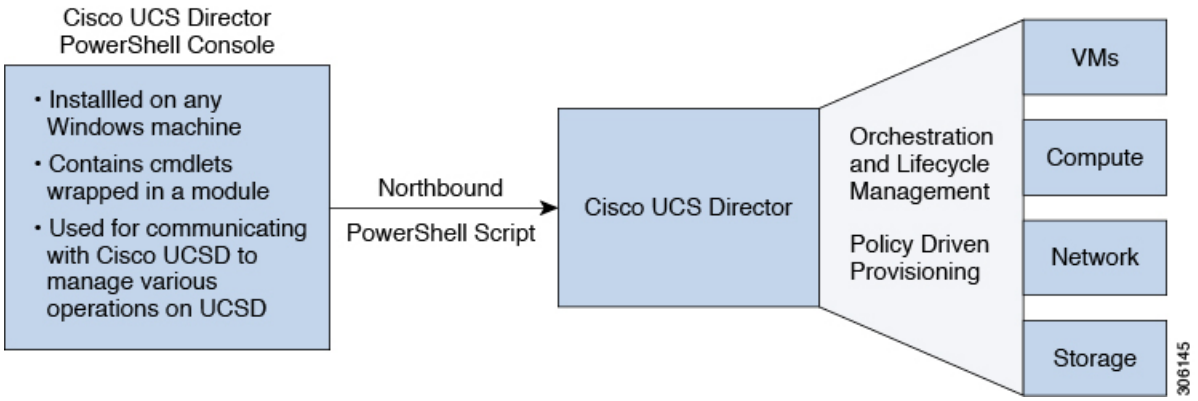
Cisco UCS Director offers REST APIs that enable applications to consume or manipulate the data stored in Cisco UCS Director. These APIs are used by any script capable of making HTTP or HTTPS requests to invoke operations on Cisco UCS Director.

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.

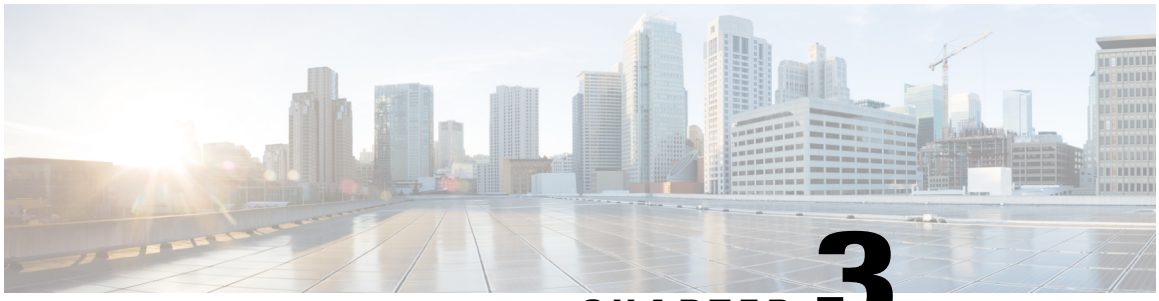
A cmdlet runs natively in the PowerShell CLI session on a compatible Windows machine. Each cmdlet performs a single operation. Depending on the data returned by the JSON-based APIs, the cmdlet automatically interprets the data and converts it into Windows PowerShell object. You can chain multiple cmdlets 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 .

Figure 1: Overview of Cisco UCS Director PowerShell Console



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.



CHAPTER 3

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")

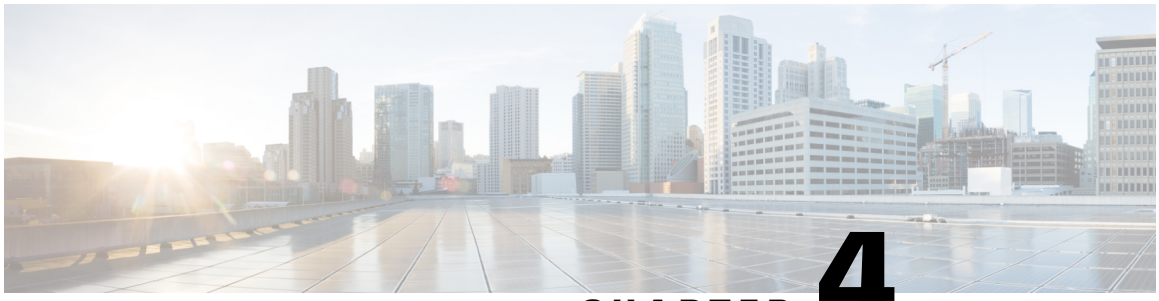
[Environment]::SetEnvironmentVariable("UCSD_RESTKEY","562FDF763A384E78B9BAB7FE02CA13B6","User")
```

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**:

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

CommandType     Name                                               ModuleName
-----------     ----                                               ----------
Cmdlet          Invoke-UserAPICancelServiceRequest                  CiscoUcsdPS
Cmdlet          Invoke-UserAPIExecuteVMAction                      CiscoUcsdPS
Cmdlet          Invoke-UserAPIGetAllCatalogs                       CiscoUcsdPS
Cmdlet          Invoke-UserAPIGetAllGroups                         CiscoUcsdPS
Cmdlet          Invoke-userAPIGetAllVDCs                           CiscoUcsdPS
Cmdlet          Invoke-UserAPIGetAvailableReports                  CiscoUcsdPS
Cmdlet          Invoke-UserAPIGetHistoricalReport                  CiscoUcsdPS
Cmdlet          Invoke-UserAPIGetInstantDataReport                 CiscoUcsdPS
Cmdlet          Invoke-UserAPIGetPage                              CiscoUcsdPS
Cmdlet          Invoke-UserAPIGetServiceRequestDetails              CiscoUcsdPS
Cmdlet          Invoke-UserAPIGetServiceRequests                    CiscoUcsdPS
Cmdlet          Invoke-UserAPIGetServiceRequestWorkflow             CiscoUcsdPS
Cmdlet          Invoke-UserAPIGetTabularReport                     CiscoUcsdPS
Cmdlet          Invoke-UserAPIGetVMActionRequests                  CiscoUcsdPS
Cmdlet          Invoke-UserAPIGetWorkflowInputs                    CiscoUcsdPS
Cmdlet          Invoke-UserAPIGetWorkflowInputValue                CiscoUcsdPS
Cmdlet          Invoke-UserAPIGetWorkflowInputValues               CiscoUcsdPS
Cmdlet          Invoke-UserAPIGetWorkflows                         CiscoUcsdPS
Cmdlet          Invoke-UserAPIGetWorkflowStatus                    CiscoUcsdPS
Cmdlet          Invoke-UserAPIReconfigureVM                        CiscoUcsdPS
Cmdlet          Invoke-UserAPIRollbackflow                         CiscoUcsdPS
```

```
Cmdlet          Invoke-UserAPISubmitServiceRequest                 CiscoUcsdPS
Cmdlet          Invoke-UserAPISubmitServiceRequestCustom           CiscoUcsdPS
Cmdlet          Invoke-UserAPISubmitWorkflowServiceRequest         CiscoUcsdPS
Cmdlet          Invoke-UserAPISubmitWorkflowServiceRequestWithG... CiscoUcsdPS
Cmdlet          Invoke-UserAPISubmitWorkflowServiceRequestWithS... CiscoUcsdPS
Cmdlet          Invoke-UserAPIValidateWorkflow                     CiscoUcsdPS
Cmdlet          Invoke-UserAPIWorkflowInputDetails                 CiscoUcsdPS


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

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


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

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
  Paramter set name            <All>
  Aliases                      None
  Dynamic?                     None


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

```
  Accept pipeline input?       false
  Paramter set name            <All>
  Aliases                      None
  Dynamic?                     false

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

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.

Cmdlet	Return Object Type
Invoke-userAPICancelServiceRequest	Boolean
Invoke-userAPIExecuteVMAction	String
Invoke-userAPIGetAllCatalogs	APITabularReport
Invoke-userAPIGetAllGroups	APITabularReport
Invoke-userAPIGetAllVDCs	APITabularReport
Invoke-userAPIGetAvailableReports	Array of APIReportDefinition
Invoke-userAPIGetHistoricalReport	Array of HistoricalDataSeries
Invoke-userAPIGetInstantDataReport	APISnapshotReport
Invoke-userAPIGetPage	VMDataViewPaginated
Invoke-userAPIGetServiceRequestDetails	APIServiceRequestDetails
Invoke-userAPIGetServiceRequests	APITabularReport
Invoke-userAPIGetServiceRequestWorkFlow	APIWorkflowStatus
Invoke-userAPIGetTabularReport	APITabularReport
Invoke-userAPIGetVMActionRequests	APITabularReport
Invoke-userAPIGetWorkflowInputs	Array of APIWorkflowInputDetail
Invoke-userAPIGetWorkflowInputValue	String

Cmdlet	Return Object Type
Invoke-userAPIGetWorkflowInputValues	WorkflowInputValue
Invoke-userAPIGetWorkflows	Array of CustomActionDefinition
Invoke-userAPIGetWorkflowStatus	Integer
Invoke-userAPIReconfigureVM	String
Invoke-userAPIRollbackWorkflow	SR ID
Invoke-userAPISubmitServiceRequest	SR ID
Invoke-userAPISubmitServiceRequestCustom	SR ID
Invoke-userAPISubmitWorkflowServiceRequest	SR ID
Invoke-userAPISubmitWorkflowServiceRequestWithGroup	SR ID
Invoke-userAPISubmitWorkflowServiceRequestWithStartTime AndDurationHours	SR ID
Invoke-userAPIValidateWorkFlow	APIWFValidationResult
Invoke-userAPIWorkflowInputDetails	APIWorkflowInputDetails

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-UserAPIGetSeviceRequestDetails
- Invoke-UserAPIGetSeviceRequests
- Invoke-UserAPIGetSeviceRequestWorkflow
- Invoke-UserAPIGetTabularReport
- Invoke-UserAPIGetVMActionRequests
- Invoke-UserAPIGetWorkflowInputs
- Invoke-UserAPIGetWorkflowInputValue
- Invoke-UserAPIGetWorkflowInputValues

- 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:

Report Name	Context Value
CHARGEBACK_DETAILS_SERVICES_GLOBAL_TABULAR	None
PER_CLOUD_ARCHIVED_HYPERV_VM_LIST_REPORT	<cloudName>
vms.paginated.report	<cloudName>
PER_CLOUD_ARCHIVED_VM_LIST_REPORT	<cloudName>

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

Input Set Id   Action Id    Field Id                      Field Value
------------   ---------    --------                      -----------
18             0            WF_EMPTY_INPUTS
18             156          input_0_user-type471          Regular
18             156          AddUser_231.OUTUT_USER_NAME   cmdlettest1
18             156          input_3_password453           test
18             156          input_4_confirm-password361   test
18             156          input_5_email1932             user@mail.com
18             156          input_1_group-id135           1
18             156          input_2_login-name867         cmdlettest1

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

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'

Field Id                                    Field Value
--------                                    -----------
WF_EMPTY_INPUTS
input_0_user-type471                        Regular
AddUser_231.OUTUT_USER_NAME                 cmdlettest1
input_3_password453                         test
input_4_confirm-password361                 test
input_5_email1932                           user@mail.com
input_1_group-id135                         1
input_2_login-name867                       cmdlettest1

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

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

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

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
```

```
1  Default Group  DEF  Default Group. All discovered VMs are placed in this group.  Local
   System Administrator   No  No   No

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

Note For all APIs that return tabular reports, add "-verbose" to the syntax to view tabular reports. When you do not add "-verbose", only the object type and its address are returned.

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.

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

NAME
    Invoke–userAPICancelServiceRequest

SYNTAX
    Invoke–userAPICancelServiceRequest [–requestId] <int> [-server <string>] [-restkey <string>]
[<CommonParameters>]

ALIASES
    None

REMARKS
    None

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

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

- requestId—Mandatory parameter
- server—Optional parameter
- restKey—Optional parameter

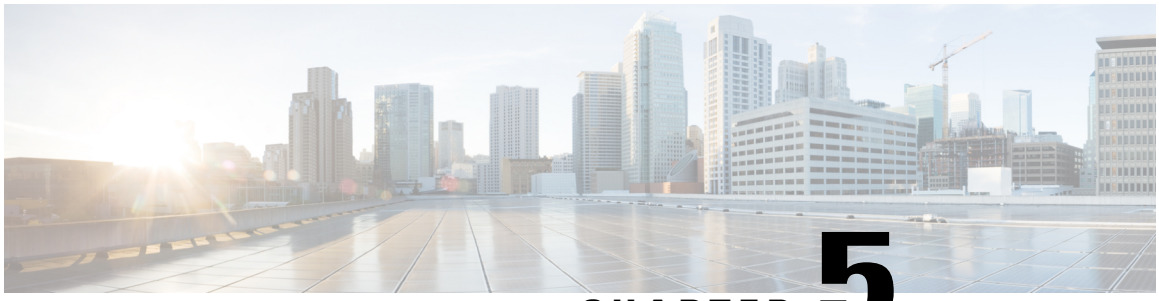
Note You can use the **get-Help** command to get detailed help about the individual parameters of a cmdlet. For more information, see Help for Cmdlets, on page 8.

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>
```



CHAPTER 5

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

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

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
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~
```

```
+ CategoryInfo          : InvalidArgument: (server/restKey:String) [Invoke
- userAPIGetAllGroups], ArgumentException
```

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.