

# Cisco UCS Director PowerShell API Getting Started Guide, Release 5.4

---

**First Published:** November 03, 2015

## Getting Started With Cisco UCS Director PowerShell Console

### Cisco UCS Director PowerShell Console

Cisco UCS Director offers JSON-based REST APIs that enable you to submit workflow requests, examine workflow inputs and output schemas, and fetch reports. You can integrate Cisco UCS Director APIs with Cisco UCS Director PowerShell Console for improved automation of datacenter management.

Cisco UCS Director PowerShell Console provides cmdlet wrappers for the JSON-based APIs. Each cmdlet performs a single operation. The cmdlets are executed in a Microsoft Windows server. Depending on the data returned by the JSON-based APIs, the cmdlets automatically interpret the data and convert it into Windows PowerShell objects. You can chain multiple cmdlets together. To view a list of available cmdlets, see [Cmdlets List, on page 3](#). For more information about REST APIs, see the [Cisco UCS Director REST API Getting Started Guide](#).

The PowerShell Console is different from the PowerShell Agent. The PowerShell Console provides cmdlet wrappers for the JSON-based APIs. The PowerShell Agent interfaces between Cisco UCS Director and Microsoft System Center Virtual Machine Manager (SCVMM) and is responsible for inventory collection and other management functions.

### Installing and Configuring

#### 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:

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

#### Determining Windows PowerShell Version

Use the `$PSVersionTable` command to view information about the Windows PowerShell version that is running on your system.

The following console output is an example of the **\$PSVersionTable** command:

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

Name                               Value
----                               -
PSVersion                           4.0
WSManStackVersion                   3.0
SerializationVersion                1.1.0.1
CLRVersion                          4.0.30319.34209
BuildVersion                        6.3.9600.16406
PSCompatibleVersions                {1.0, 2.0, 3.0, 4.0}
PSRemotingProtocolVersion           2.2

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

## Verifying the Cisco UCS Director PowerShell Console

Use the **Get-Module** command to verify that Cisco UCS Director PowerShell Console has been installed.

The **Get-Module** command displays the type, version, and the name of the module, as shown below.

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

ModuleType  Version  Name                               ExportedCommands
-----
Binary      5.4.0.0  CiscoUcsdPS                       {Invoke-userAPICan...}

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

To verify only the version after obtaining the name of the module, use the command as shown below:

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

Major Minor Build Revision
-----
5      4      0      0

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



### Note

The previous two examples show the installed Cisco UCS Director PowerShell Console module to be version 5.4.0.0. The version number might vary when you have a different version installed. For example, if you have upgraded to a newer version.

## 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 pick 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.

## Working with Cmdlets

### 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-UserAPICancelSeviceRequest	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-UserAPIGetSeviceRequestDetails	CiscoUcsdPS
Cmdlet	Invoke-UserAPIGetSeviceRequests	CiscoUcsdPS
Cmdlet	Invoke-UserAPIGetSeviceRequestWorkflow	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
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-userAPISubmitWorkflowServiceRequestWithStartTimeAndDurationHours	SR ID

Cmdlet	Return Object Type
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-UserAPIGetServiceRequestDetails
- Invoke-UserAPIGetServiceRequests
- Invoke-UserAPIGetServiceRequestWorkflow
- 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
```

### 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 2](#).

**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 3.

**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

### Connection Exception Error

**Problem**

A connection exception error occurs when invoking a PowerShell API.

**Description**

When invoking a PowerShell API, the following error 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 up and 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_Execution_Policies at
http://go.microsoft.com/fwlink/?LinkID=135170.
```

```
+ CategoryInfo          : SecurityError: <:> [], ParentContainsErrorRecord Exception
+ FullyQualifiedErrorId : UnauthorizedAccess
```

### Solution

The execution policy is part of the security strategy of Cisco UCS Director PowerShell. It 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 *Unrestricted*, 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 make changes to the environment variables, close the PowerShell console and open it again.



---

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)

© 2015 Cisco Systems, Inc. All rights reserved.