Quali



Cisco IOS Shell

Release date: Jun, 2017

Shell version 5.0.0

Document version 1.0

*Qualisystems Ltd. Confidential and proprietary.*

*All Rights Reserved. No part of this software or material may be copied, reproduced, stored in or introduced into a retrieval system, distributed or displayed in any form or manner for any purpose whatsoever and no derivative works may be made without Qualisystems Ltd. advance written consent. All trademarks, brand names, product names and logos are trademarks or registered trademarks of Qualisystems Ltd. or applicable licensor [TestShell, CloudShell, the Qualisystems logo and the TestShell logo. The absence of a trademark from this list does not constitute a waiver of Qualisystems or applicable licensor's intellectual property rights concerning that trademark].*

*The above copyright and trademark notices shall be included in all such software and/or materials.*

*Copyright, 2016, Qualisystems Ltd. Software and materials are copyrighted and trademarked by Qualisystems Ltd. and any incorporated third party software is copyrighted by its respective licensor.*



Contents

[Overview 3](#_Toc27327)

[About Cisco IOS Shell 3](#_Toc27328)

[Standard version 3](#_Toc27329)

[Supported Cisco OSes 3](#_Toc27330)

[Certified models 3](#_Toc27331)

[Requirements 3](#_Toc27332)

[Downloading the Shell 3](#_Toc27333)

[Automation 4](#_Toc27334)

[Cisco iOS Switch 4](#_Toc27335)

[Import and Configure the Shell 6](#_Toc27336)

[Importing the Shell into CloudShell 6](#_Toc27337)

[Offline installation of a Shell 6](#_Toc27338)

[Configuring a new device 7](#_Toc27339)

[Upgrading a device that was defined using the Shell versions 2.0.0 – 2.0.4 7](#_Toc27340)

[Data Model 8](#_Toc27341)

[Families and Models 8](#_Toc27342)

[Attributes 8](#_Toc27343)

[Typical workflow and scenarios 12](#_Toc27344)

[Typical workflow 12](#_Toc27345)

[Use cases and scenarios 12](#_Toc27346)

[Scenario 1 - Autoload 12](#_Toc27347)

[Scenario 2 - Save configuration 12](#_Toc27348)

[Scenario 3 - Restore configuration 12](#_Toc27349)

[Scenario 4 – Load Firmware 13](#_Toc27350)

[Release notes 14](#_Toc27351)

[References 15](#_Toc27352)

2 | P a g e

# Overview

A Shell implements integration of a device model, application or other technology with CloudShell. A shell consists of a data-model that defines how the device and its properties are modeled in CloudShell along with an automation that enables interaction with the device via CloudShell.

# About Cisco IOS Shell

This Shell provides you with connectivity and management capabilities such as power management, save and restore configurations, structure autoload functionality and updating firmware, etc.

# Standard version

The Cisco iOS 5.0.0 Shell is based on the Networking Shell Standard Version 4.0.2

# Supported Cisco OSes

* Cisco IOS,
* Cisco IOS-XE,
* Cisco CatOS

# Certified models

 Catalyst 2950

# Requirements

 CloudShell version 7.0 and above

# Downloading the Shell

The Cisco iOS Shell is available from the [Quali Download Center.](https://support.qualisystems.com/entries/87063688-Solution-Pack-Download-Center) Download the files into a temporary location on your local machine.

**Note:** Registration to the Quali Support Portal is required. If you have not registered, click this link to register [New registration.](http://portal.qualisystems.com/entries/43187197)

The Shell comprises:

|  |  |
| --- | --- |
| **Cisco\_IOS\_Shell\_Package.zip** | iOS Shell |
| **cloudshell-networking-cisco-iosdependencies-package-5.0.X.zip** | Shell Python dependencies (**for offline installation only**) |
| **Cisco.iOS.Shell.5.0.0.pdf** | Documentation |

# Automation

This section describes the automation (drivers or scripts) associated with the data model. The automation code (either script or driver) is associated with the model and provided as part of the Shell package (in the .zip file).

The following commands are associated with a model inside the Shell:

## Cisco iOS Switch

|  |  |
| --- | --- |
| **Command** | **Description** |
| Autoload | Discovers the device, its hierarchy and attributes |
| Run Custom Command | Sends command to the device, and prints output. All commands will be executed in the enable mode, However it will not allow to enter configuration mode. |
| Run Custom Config  Command  (Hidden) | Sends command to the device in configuration mode, and prints output. All commands will be executed in the enable mode, accessible only via the API. |
| Load Firmware | Uploads and updates firmware |
| Save | Backs up running or startup configuration of the device |
| Restore | Restores running or startup configurations from file |
| orchestration\_save (Hidden) | Based on the Orchestration Save and Restore Standard - https://github.com/QualiSystems/sandbox\_orchestration\_standa rd/blob/master/save%20%26%20restore/save%20%26%20rest ore%20standard.md The command wraps the Save command with a standard interface that will be used by the Sandbox orchestration. This command will call the Save command which will create a configuration file. The command should be hidden from the UI. The backup type used is defined in the "Backup Type" attribute on the root resource (File System, FTP or TFTP), and the path in which to save the configuration file is defined in the "Backup Location" attribute on the root resource. In case the value of the attribute "Backup Location" is empty the command should throw an error. In case the backup type is FTP the user and password for the FTP server should be taken from the "Backup User" and "Backup Password" attributes on the root resource. |

|  |  |
| --- | --- |
| **Command** | **Description** |
| orchestration\_restore (Hidden) | Based on the Orchestration Save and Restore Standard - [https://github.com/QualiSystems/sandbox\_orchestration\_standa rd/blob/master/save%20%26%20restore/save%20%26%20rest ore%20standard.md](https://github.com/QualiSystems/sandbox_orchestration_standard/blob/master/save%20%26%20restore/save%20%26%20restore%20standard.md)  The command wraps the Restore command with a standard interface that will be used by the Sandbox orchestration. This command will call the Save command which will create a configuration file. The command should be hidden from the UI. In case the saved artifact type is FTP the FTP credentials should be available in the attributes "Backup User" and "Backup Password" on the root resource. |

# Import and Configure the Shell

This section describes how to import, configure and modify the Cisco iOS Shell.

# Importing the Shell into CloudShell

Use the following procedure to import the downloaded Shell:

**To import the Shell into CloudShell:**

1. Download the Shell from the [Quali Download Center.](https://support.qualisystems.com/entries/87063688-Solution-Pack-Download-Center)
2. Backup your database.
3. Log in to **CloudShell Portal** as administrator of the relevant domain.
4. In the **User** menu select **Import Package.**



1. Browse to the location of the downloaded Shell file, select the relevant .zip file and Click **Open**. Alternatively, drag the shell’s .zip file into CloudShell Portal.

# Offline installation of a Shell

**Note:** Offline installation instructions are relevant only if Cloudshell Execution Server has no access to PyPi. You can skip this section if your execution server has access to Pypi.

The Cisco iOS shell uses a variety of Python packages. To work in offline mode:

1. Download the **cloudshell-networking-cisco-ios-dependencies-package4.0.15** file (see *Downloading the Shell*).
2. Unzip it to a local repository. Make sure the execution server has access to this folder.
3. On the Execution Server, in the **customer.config** file, add the following key:

|  |  |
| --- | --- |
| <add key="PythonOfflineRepositoryPath" value="repository | |
| full path"/> |  |

Make sure to update the “repository full path” with path to the repository you unzipped the file to.

1. Restart the Execution Server.

# Configuring a new device

Use the following procedure to load a device, which will use this Shell, into CloudShell:

1. In the CloudShell Portal, in the **Inventory** dashboard, click **Add New**.
2. From the list, select the **Cisco iOS Switch** Shell.
3. Enter the Switch’s Name and IP address.
4. Click **Create**.
5. In the Resource Discovery information form, enter the all the fields relevant for the device. Make sure to fill in the device’s SNMP version and credentials.
6. Click **Start Discovery**.

This command discovers the device, fills in its attributes and creates the device’s structure in CloudShell (if such structure exists).

# Upgrading a device that was defined using the Shell versions 2.0.0 – 2.0.4

As the power-port address in the networking shell standard was change from PP<ID> to

PP<ContainerID><ID>, the flowing steps must be done to upgrade the device to work with the current version:

1. In CloudShell Resource Manager, right click the resource and select

**Configuration**. In the **Parameters** pane, in the **Driver** field, change the device driver to the new driver “Generic Cisco iOS Driver Version2”.

1. In the **Internal Resources** pane, right-click each and every power-port and select **Delete**.
2. Run the **Autoload** command (click the button at the bottom of the screen). This command will reload the power-ports and update the structure of the resource.

# Data Model

## Families and Models

|  |  |  |
| --- | --- | --- |
| **Family** | **Model** | **Description** |
| Switch | Cisco iOS Switch | Generic Cisco iOS switch |
| Router | Cisco iOS Router | Generic Cisco iOS Router |
| Chassis | Generic Chassis | Default Switch chassis |
| Module | Generic Module | Modules located on the chassis |
| Sub Module | Generic Sub Module | Sub modules |
| Port | Generic Port | Interface |
| Port Channel | Generic Port Channel | Group of interfaces |
| Power Port | Generic Power Port | Power Supply module |

## Attributes

The attribute names and types are listed in the following table:

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Default value** | **Description** |
| User | String |  | Username for Cisco CLI  (should be privileged user) |
| Password | Password |  | Password for Cisco CLI |
| Enable Password | Password |  | Enable Password for Cisco CLI |
| Sessions Concurrency Limit | Numeric | 1 | Number of sessions that can be opened to the device. Defines the number of commands that can run concurrently |
| System Name | String |  | Device hostname |
| Contact Name | String |  | Device contact name |
| OS Version | String |  | Operation system version |
| Vendor | String |  | Device manufacturer |

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Default value** | **Description** |
| Location | String |  | Device location |
| Model | String |  | Device model |
| Enable SNMP | Boolean | True | If set to True and SNMP isn’t enabled yet in the device the Shell will automatically enable  SNMP in the device when Autoload command is called, using the SNMP Read  Community value. If Value is  empty, relevant error will be raised. SNMP must be enabled on the device for the Autoload command to run successfully. |
| Disable SNMP | Boolean | False | If set to True SNMP will be disabled automatically by the Shell after the Autoload command execution is completed. |
| SNMP Read Community | String |  | Read Only SNMP community,  used for Autoload functionality |
| SNMP Write Community | String |  | Read Write SNMP community |
| SNMP V3 User | String |  | Snmp version 3 user name |
| SNMP V3 Password | String |  | SNMP version 3 password |
| SNMP V3 Private Key | String |  | SNMP version 3 private key |
| SNMP Version | String | v2 | Specifies version of SNMP, Autoload will use to load  attributes |
| Console Server IP Address | String |  | Shell allows to connect to the device through Console server. IP Address of Console server |
| Console User | String |  | User name for the Console server |
| Console Password | Password |  | Password for Console server |
| Console Port | Numeric |  | Port for the Console server |

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Default value** | **Description** |
| CLI Connection Type | Lookup | Auto | The protocol which the Shell will use to connect to the Device. Available methods:  Auto, Console, SSH, Telnet,  TCP |
| CLI TCP Port | Numeric |  | TCP Port to user for CLI connection. If kept empty a default CLI port will be used based on the chosen protocol, for example Telnet will use port 23. |
| Power Management | Boolean | False | Used by the power management service |
| Backup Type | String | File System | Supported protocols for saving and restoring of configuration and firmware files. Possible values are "File System", "FTP" and "TFTP". |
| Backup Location | String |  | Default Location, where files will be saved by Save method |
| Backup User | String |  | Username for the storage server used for saving and restoring of configuration and firmware files. |
| Backup Password | Password |  | Password for the storage server used for saving and restoring of configuration and firmware files. |
| VRF Management Name | String |  | The default VRF Management to use if configured in the network and no such input was passed to the Save, Restore or Load Firmware commands |
| Model | String |  | Element model (Module or Chassis, etc.) |
| Serial Number | String |  | Element serial number  (Module or Chassis, etc.) |
| **Attribute** | **Type** | **Default value** | **Description** |
| Version | String |  | Element version (Module or Chassis, etc.) |
| Mac Address | String |  | Interface mac address |
| L2 Protocol Type | String |  | Interface protocol type |
| IPv4 Address | String |  | Interface IPv4 address |
| IPv6 Address | String |  | Interface IPv6 address |
| Port Description | String |  | Interface description |
| Bandwidth | Numeric |  | Interface speed |
| MTU | Numeric |  | Interface mtu |
| Duplex | Lookup |  | Interface duplex (half or full) |
| Adjacent | String |  | If lldp/cdp is enabled on port, Adjacent shows connected device name and interface |
| Protocol Type | Lookup | Transparent | Attribute for internal usage |
| Auto Negotiation | Boolean | False | Shows if Auto negotiation is enabled on the interface |
| Associated Ports | String |  | Interfaces added to certain Port-channel |

# Typical workflow and scenarios

# Typical workflow

# Use cases and scenarios

## Scenario 1 - Autoload

See *Configuring a new device*.

## Scenario 2 - Save configuration

1. Login to CloudShell portal, reserve the Cisco iOS resource and run the **Save** command.
2. In the command input field, enter the following information:
   * **Folder Path:** For example, *tftp://ipaddress/shared folder*
   * **Configuration Type:** either ‘running’ or ‘startup’
   * **VRF Management Name*:*** provide the management VRF name if exists.

The startup or running configuration will be saved to a file named <ResourceName><startup/running-config>-<timestamp> which will be stored in the folder path you entered.

## Scenario 3 - Restore configuration

1. Login to CloudShell portal and reserve the Cisco iOS resource
2. Run the resource command **Restore**.
3. Enter the following input parameters:
   * **Path**: This is a mandatory input field. Enter the full path of the configuration file.
   * **Restore Method**: This is an optional input field. Can be ‘Append’ or ‘Override’. If nothing is entered in this input field, the Append method will be used.
   * **Configuration Type:** This is a mandatory input field. Possible values Startup or Running
   * **VRF Management Name*:*** This is an optional input field. Provide the management VRF name if exists.

## Scenario 4 – Load Firmware

1. Login to CloudShell portal and reserve the Cisco IOS resource
2. Run the resource command **Load Firmware**.
3. Enter the following input parameters:
   * **Remote Host** (mandatory input field). Enter the full path to the firmware file on remote host. For example:

tftp://10.1.1.1/ios12.SE1-smp-k8.bin

* + **File Path** (mandatory input field). Destination filename on the device itself, for example:

bootflash:/ios12.SE1-smp-k8.bin

# Release notes

**Version 3.0.1**

What’s new:

* Bugfix 154954: Save command input folder path on NxOS, iOS and ASA is mandatory when it should be optional
* Bugfix 154955: iOS shell - Restore command will fail with an error
* Bugfix 154979: iOS Shell - When running commands in telnet mode in the iOS shell command will fail
* Known issue 154025: Save command will fail if used when restore command is running.
* Known issue 154025: Restoring command will fail if some of the configuration failed and some succeeded.

**Version 4.0.0**

What’s new:

* Bugfix 156467: if there is no entity Alias Mapping table in the switch, the autoload command will not create all the ports.

Note that devices with the bug will need to be re-autoloaded. The first autoload will throw an error message that there is a conflict with some of the ports. In addition it will delete the ports form the resource. You will then need to re-autload to recreate the deleted ports. This will make the device ready.

**Version 4.0.1**

What’s new:

* Bugfix 156785: iOS shell - drivermetadata is set to standard version 4.0.0 not 3.2.0.
* Bugfix 156877: Restore command with override restore method, doesn’t fail when configuration file is invalid.

Known issue:

* Known issue: There are some limitations regarding use of QnQ for NXOS devices as described at Cisco website in the chapter on [Configuring Q-in-Q VLAN Tunnels.](http://www.cisco.com/c/en/us/td/docs/switches/datacenter/nexus5600/sw/interfaces/7x/b_5600_Interfaces_Config_Guide_Release_7x/b_6k_Interfaces_Config_Guide_Release_7x_chapter_0111.html#concept_7A305DDA71DE47A08C646239EAFE7137)
* Known issue 15248: Some Switches/routers don’t reflect port-channels members correctly in the IEEE 802.3ad Link Aggregation MIB.

Known issue 153734: Disconnecting VLAN will fail When the VLAN service is set as source when using the L2/L3 shell – **solved in CloudShell Patch 1**

**Version 5.0.0**

What’s new:

Built according to Networking Standard version 4.0.2

* Bugfix 156696: IOS Shell doesn't use VRF Management Name attribute from the resource configuration details
* Bugfix 158355: IOS Shell – add serial ports during autoload
* Bugfix 156604: After resource re-autoloading sometimes getting an error “conflict of address” in one of the sub resources.

Known issue:

* Known issue 15248: Some Switches/routers don’t reflect port-channels members correctly in the IEEE 802.3ad Link Aggregation MIB.

# References

Additional technical documentation is available in the [Quali Download Center.](https://support.qualisystems.com/entries/22858046-download-center)

For Quali discussion forums, you can access the [Quali Support Portal.](https://support.qualisystems.com/)