ADMINISTRATOR GUIDE: NETWORK CONFIGURATION MANAGER

© 2020 SolarWinds Worldwide, LLC. All rights reserved.

This document may not be reproduced by any means nor modified, decompiled, disassembled, published or distributed, in whole or in part, or translated to any electronic medium or other means without the prior written consent of SolarWinds. All right, title, and interest in and to the software, services, and documentation are and shall remain the exclusive property of SolarWinds, its affiliates, and/or its respective licensors.

SOLARWINDS DISCLAIMS ALL WARRANTIES, CONDITIONS, OR OTHER TERMS, EXPRESS OR IMPLIED, STATUTORY OR OTHERWISE, ON THE DOCUMENTATION, INCLUDING WITHOUT LIMITATION NONINFRINGEMENT, ACCURACY, COMPLETENESS, OR USEFULNESS OF ANY INFORMATION CONTAINED HEREIN. IN NO EVENT SHALL SOLARWINDS, ITS SUPPLIERS, NOR ITS LICENSORS BE LIABLE FOR ANY DAMAGES, WHETHER ARISING IN TORT, CONTRACT OR ANY OTHER LEGAL THEORY, EVEN IF SOLARWINDS HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

The SolarWinds, SolarWinds & Design, Orion, and THWACK trademarks are the exclusive property of SolarWinds Worldwide, LLC or its affiliates, are registered with the U.S. Patent and Trademark Office, and may be registered or pending registration in other countries. All other SolarWinds trademarks, service marks, and logos may be common law marks or are registered or pending registration. All other trademarks mentioned herein are used for identification purposes only and are trademarks of (and may be registered trademarks) of their respective companies.
# Table of Contents

**Introduction to NCM** ........................................................................................................... 9
- Scheduled configuration backups ....................................................................................... 9
- Policy management ........................................................................................................... 9
- Inventory ........................................................................................................................ 9
- Configuration change history .......................................................................................... 9
- Configuration viewing, tracking, and comparison ............................................................ 10
- Device configuration change management ....................................................................... 10
- Real-time change detection ............................................................................................... 10
- Firmware upgrades .......................................................................................................... 10

**NCM licensing model** ......................................................................................................... 11

**Orion Platform features in NCM** ..................................................................................... 12
- Pre-installation hints ....................................................................................................... 12
- Common features ............................................................................................................ 12

**Manage NCM user accounts and assign NCM roles** ........................................................ 15
- Grant access to NCM functionality .................................................................................. 15
- NCM roles ...................................................................................................................... 15
- Device access information ............................................................................................... 16

**Add nodes to NCM and specify connection information** .................................................. 17
- Add or remove NCM nodes ............................................................................................. 17
- Options for specifying NCM connection information ..................................................... 18
- Configure nodes to use device-level login credentials for NCM connections ............... 21
- Configure nodes to use global connection profiles for NCM connections ..................... 23
- Configure nodes to use user-level login credentials for NCM connections .................... 26
- Specify values for global variables .................................................................................. 29
- Protocols NCM uses to transfer configurations .............................................................. 30
Troubleshoot NCM device connections ................................................................. 32

**Manage nodes and collect node data in NCM** ..................................................... 33

Unmanage and remanage NCM nodes .................................................................. 33

Manage end of support and end of sales data in NCM ....................................... 34

Run an NCM inventory scan and view inventory data ....................................... 38

Find connected port for a host ............................................................................ 41

**Manage network configuration files** ................................................................. 43

NCM setup required for binary configs ............................................................... 43

Set NCM communication limits ......................................................................... 44

Specify Config Settings for NCM ....................................................................... 44

Download configuration files in NCM ................................................................. 46

Schedule daily backups of network config files ............................................... 47

Enable a new config type in NCM ....................................................................... 49

Enable the NCM config and policy caches ......................................................... 51

Edit network config files .................................................................................. 51

Upload a network config ................................................................................... 52

Import network config files ............................................................................... 53

Upload network configuration changes ................................................................ 54

Archive downloaded configuration files ............................................................. 55

Search network config files or node properties .................................................... 57

Delete network config files from the database ................................................. 58

Remove older NCM data from the database and config archive ..................... 59

Limit the number of simultaneous uploads or downloads of network configs ...... 61

Troubleshoot issues with NCM configuration files .......................................... 61

**Firmware vulnerability data** .............................................................................. 64

View firmware vulnerability data ....................................................................... 64

Access NCM firmware vulnerability settings ...................................................... 67
<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Import and export NCM config change templates</td>
<td>263</td>
</tr>
<tr>
<td>Execute an NCM config change template</td>
<td>265</td>
</tr>
<tr>
<td><strong>Approval system for device configuration changes</strong></td>
<td>267</td>
</tr>
<tr>
<td>Set up config change approval in NCM</td>
<td>267</td>
</tr>
<tr>
<td>Turn on the NCM configuration change approval system</td>
<td>268</td>
</tr>
<tr>
<td>View, approve, and decline network config change requests</td>
<td>269</td>
</tr>
<tr>
<td>Manage Cisco ACLs and Palo Alto policies in NCM</td>
<td>270</td>
</tr>
<tr>
<td>Manage ACLs on Cisco ASA and Nexus devices</td>
<td>270</td>
</tr>
<tr>
<td>Overlapping ACL rules</td>
<td>274</td>
</tr>
<tr>
<td>View Palo Alto policies</td>
<td>275</td>
</tr>
<tr>
<td><strong>NCM reports</strong></td>
<td>279</td>
</tr>
<tr>
<td>Run an NCM config change report once</td>
<td>279</td>
</tr>
<tr>
<td>Schedule a recurring network config change report</td>
<td>280</td>
</tr>
<tr>
<td>Access predefined NCM reports</td>
<td>280</td>
</tr>
<tr>
<td>Predefined NCM reports</td>
<td>281</td>
</tr>
<tr>
<td><strong>Alerts in NCM</strong></td>
<td>287</td>
</tr>
<tr>
<td>Types of NCM alert actions</td>
<td>287</td>
</tr>
<tr>
<td>Apply the default NCM alert</td>
<td>288</td>
</tr>
<tr>
<td>Add scripted commands to the default NCM alert</td>
<td>289</td>
</tr>
<tr>
<td><strong>Servers for NCM file transfers and email notifications</strong></td>
<td>291</td>
</tr>
<tr>
<td>Specify the SolarWinds TFTP server settings for NCM config transfers</td>
<td>291</td>
</tr>
<tr>
<td>Use the SolarWinds SFTP &amp; SCP server for NCM config transfers</td>
<td>294</td>
</tr>
<tr>
<td>Use a third-party SCP server for NCM config change transfers</td>
<td>298</td>
</tr>
<tr>
<td>Configure NCM email notifications</td>
<td>299</td>
</tr>
<tr>
<td><strong>EnergyWise devices</strong></td>
<td>300</td>
</tr>
<tr>
<td>What is EnergyWise?</td>
<td>300</td>
</tr>
<tr>
<td>Manage and enable EnergyWise nodes</td>
<td>300</td>
</tr>
</tbody>
</table>
Manage Power over Ethernet ports ................................................................. 301

Common NCM tasks ......................................................................................... 302
  Change the community string on multiple nodes ........................................... 302
  Block a MAC address on a wireless access point ........................................... 303
  Change an interface description .................................................................... 305
  Block all private addresses with an access control list .................................. 307
  Customize the login banner of a device ......................................................... 308

NCM macros (variables) .................................................................................... 310
  Global macros ............................................................................................... 310
  Node macros ............................................................................................... 311
  Transfer macros ......................................................................................... 312
  Command macros ....................................................................................... 312
  Date and time macros ................................................................................. 313
  Keystroke macros ....................................................................................... 315
  Prompt detection and delay macros ............................................................... 316
  Where can macros be used? ........................................................................... 318
  Custom macros (variables) in NCM ............................................................... 319
Introduction to NCM

Network Configuration Manager is a comprehensive, intuitive solution designed to streamline and automate network configuration management. NCM increases availability, saves time, improves security, and ensures policy adherence. NCM features automation capabilities that reduce the amount of time network engineers spend on mundane network tasks, allowing them to focus on business-critical network projects.

Use NCM to manage configurations on heterogeneous, multi-vendor networks. NCM provides a single point of management for routers, switches, firewalls, load balancers, and wireless access points. Supported vendors include Cisco, Cisco ASA, Cisco Nexus, Dell, Adtran, Arris, Aruba, Nortel, Nortel Alteon, Nortel BayStack, Extreme, Marconi, Radware, Netscreen, Motorola, HP, Netscaler, Juniper, Foundry, and F5.

NCM runs on the Orion Platform, which offers common alerting, reporting, and management features.

Scheduled configuration backups

Schedule configuration downloads, configuration uploads, device reboots, command script execution, and more. In addition, configuration backups are stored both in a relational database for archival history and as flat files in an intuitive folder structure for easy viewing.

Policy management

Use NCM policy reports to ensure device compliance with federal regulations and corporate standards. The Policy Reporting Manager includes policy reports such as SOX, HIPAA, CISP, and Cisco Security. You can also create custom reports.

Inventory

Use NCM to perform inventory scans on all of your nodes, node groups, or on a single node. The collected inventory statistics are displayed in the detail view of each device.

Configuration change history

Receive reports on what devices have had configuration changes over a specified time period. Configuration change reports can also compare current configurations with a baseline configuration, alerting you whenever a change is discovered.
Configuration viewing, tracking, and comparison

Use NCM to remotely view, track and make changes, and compare network device configurations without logging in to the physical NCM server. The Web Console offers these functions to the users you select.

Device configuration change management

Set up a request and approval system for processing the workflow of device configuration changes.

Real-time change detection

The Real Time Change Detection feature provides notification through email whenever a change to any of your device configurations occurs.

Firmware upgrades

Upgrade the firmware on many devices at one time. Because security fixes are distributed as new firmware revisions, upgrading firmware helps network administrators correct security vulnerabilities.
NCM licensing model

Network Configuration Manager can manage almost any network device, including routers, switches, and firewalls. Most CLI-capable devices with a command menu can provide configuration files to NCM.

Orion Platform products support both perpetual licenses and subscription licenses. See License types in the Orion Platform help for details.

NCM licensing is based on the number of nodes that NCM manages. A node is defined as an entire device, such as a router, switch, server, access point, or modem.

There are seven types of NCM licenses available:

- Up to 50 devices (DL50)
- Up to 100 devices (DL100)
- Up to 200 devices (DL200)
- Up to 500 devices (DL500)
- Up to 1000 devices (DL1000)
- Up to 3000 devices (DL3000)
- Unlimited devices (DLX)

Licensing keys for most products can be found by logging into the SolarWinds Customer Portal and clicking Licenses > Manage Licenses.
Orion Platform features in NCM

The Orion Platform is the core of the SolarWinds IT Management Portfolio. It provides a stable and scalable architecture that includes data collection, processing, storage, and presentation. The Orion Platform provides common features, such as user accounts and groups, views, dashboards, reporting, alerting, and more that you can use across all Orion Platform products and access from the Orion Web Console.

Pre-installation hints

Before you install your Orion Platform products, review the following details:

Orion Platform requirements

Hardware, software, and port requirements for the Orion Platform server and SolarWinds Orion database.

Licensing

Licensing differs among Orion Platform products. Activate, add, upgrade or assign licenses with the License Manager in the Orion Web Console.

Installation or upgrade

Use the SolarWinds Orion Installer to easily install or upgrade multiple Orion Platform products simultaneously.

While installing your Orion Platform products, you might need to configure SSL for the Orion Web Console, enable FIPS, or review directories to be excluded from antivirus protection.

Common features

The following features are available in Orion Platform products.

Learn Orion Platform basics

Log in to your Orion Platform product in a web browser and meet the Orion Web Console.

Review Events, syslogs, or SNMP traps to know what’s going on.

Get alerts about issues in your environment.

Generate reports to present the status of the monitored environment.

Review Performance Analysis dashboards, also known as PerfStack™.
Create, edit, and maintain Orion Web Console user accounts - set user rights, reset passwords, limit access to network segments, and enable authentication with Active Directory.

View monitored objects on Orion Maps in the Orion Web Console.

View other map options in the Orion Web Console - display objects with their location specified in the OpenStreet format in a widget, or create maps the Network Atlas tool and display them in the Orion Web Console.

Add devices for monitoring and manage monitored devices

Specify which devices to monitor and the information you need, then select the way you get this information. See Discover and add devices.

Add single nodes, use Active Directory domain controllers to add nodes, or discover devices on your network automatically.

Available polling methods include ICMP, WMI, SNMP, or agents deployed on Windows, Linux, and UIX devices.

Manage monitored devices - edit properties, set the polling method for monitored devices, toggle monitoring on and off, or mute alerts for nodes.

Customize your Orion Web Console

Customize Orion Web Console - customize dashboards, colors, logo, views, widgets and charts. Learn how to limit what objects users see on views, or specify what you want to see on views for specific device types.

Create custom properties - create custom fields to associate with monitored network objects and display custom information for monitored devices.

Create groups and dependencies - organize how monitored data is presented in the Orion Web Console. Set up dependencies to better represent the relationships between network objects and account for constraints on the network.

Set thresholds - specify thresholds for monitored metrics. Customize general thresholds or use baselines.

Monitor additional metrics and devices

Monitor hardware health - get insight into hardware issues on the network. Monitor hardware health based on hardware sensors, such as fan status, power supply status, or temperature.

Monitor virtual environments - monitor your virtual networks (VMware® ESX and ESXi servers, VMware vCenter®) in the Orion Web Console.

Quality of Experience - use packet analysis sensors to see packet-level traffic information about key devices and applications on your network.
Expand the Orion Platform functionality or scale your deployment

Use [SolarWinds High Availability](https://www.solarwinds.com/products/solarwinds-high-availability) (HA) to provide failover protection for your Orion server and additional polling engines to reduce data loss.

Do you need to scale your deployment? See [Scalability Engine Guidelines](https://www.solarwinds.com/products/solarwinds-high-availability).

Review the [tips for optimizing your deployment](https://www.solarwinds.com/products/solarwinds-high-availability).

Balance the load on polling engines by [specifying nodes to be polled by individual polling engines](https://www.solarwinds.com/products/solarwinds-high-availability).


Manage NCM user accounts and assign NCM roles

NCM roles determine what NCM functionality a user or group account can access. By default, user accounts in the Orion Platform do not have access to any NCM functionality.

Only a user with Administrator privileges can create or edit user accounts and assign NCM roles.

Grant access to NCM functionality

1. Create a new user account in the Orion Platform, or edit an existing account.
2. Expand the Network Configuration Manager Settings section.
3. Assign the NCM role that defines what NCM functionality is available to this user account. To be able to make configuration changes and upload them, the account must have at least the WebUploader role.
4. If you want to prevent the account from being able to access specific NCM views, select None for those views.
5. Click Submit.

NCM roles

The following roles are available.

<table>
<thead>
<tr>
<th>Role</th>
<th>Privileges</th>
</tr>
</thead>
<tbody>
<tr>
<td>Administrator</td>
<td>This role has unlimited access to NCM functionality, including device configuration management, user account management, and configuration change approvals.</td>
</tr>
<tr>
<td>Engineer</td>
<td>This role has Administrator privileges, but cannot view the device configuration transfer status for all users.</td>
</tr>
<tr>
<td>WebUploader</td>
<td>This role has read and write access on network devices. However, if an approval system is enabled, this role cannot change device configurations without Administrator approval.</td>
</tr>
<tr>
<td>WebDownloader</td>
<td>This role can read and download network device configurations.</td>
</tr>
</tbody>
</table>
### Role Privileges

<table>
<thead>
<tr>
<th>Role</th>
<th>Privileges</th>
</tr>
</thead>
<tbody>
<tr>
<td>WebViewer</td>
<td>This role can only read network device configurations.</td>
</tr>
<tr>
<td>None</td>
<td>For new accounts, this role is selected by default. This role cannot access NCM features and functions. This role will not see NCM resources on non-NCM views in the Orion Web Console. Even if the account has privileges to add nodes, the user cannot add nodes to NCM (the Add Node option does not appear), and NCM properties are hidden when the user edits a node.</td>
</tr>
</tbody>
</table>

### Device access information

An NCM user logs on directly at the network device with unencrypted credentials and can perform actions the NCM role gives permission for.

If the network administrator wants to use the same credentials for NCM to log on to all network devices, the NCM software provides a Global Login and an option to enable global login settings on all devices. See [Options for specifying NCM connection information](#).

### Third Party Authentication

If a network administrator sets up third-party authentication, such as a Diameter, RADIUS, or TACACS server, the admin should create valid accounts and permissions in the authentication server database for NCM users.

Only NCM interacts with network devices, not the authentication server. Though the network device must handle interaction with Diameter, RADIUS, TACACS, or any other authentication server, special logic in the relevant NCM component (SWTelnet9) handles the RADIUS authentication prompt, since devices connected to the RADIUS server may have a slightly different login flow.
Add nodes to NCM and specify connection information

After you discover nodes and add them to the Orion Platform for monitoring, you must complete the following tasks to manage nodes with NCM:

1. Determine which nodes you want to manage with NCM, and add the nodes to NCM.
2. Specify the connection information that NCM uses to log in to nodes and perform tasks such as downloading configurations, performing firmware upgrade operations, or running scripts. See Options for specifying NCM connection information.

Add or remove NCM nodes

After you discover devices and add them to the Orion Platform for monitoring, you also need to add the devices to NCM. You can perform NCM management operations (such as config backups or firmware upgrade operations) only on devices that have been added to NCM.

Network devices include switches, routers, firewalls, and Windows servers. They are known collectively as nodes.

Add nodes to NCM

1. Click Settings > Manage Nodes.

   The Manage Nodes view lists all of the nodes that have been added to the Orion Platform. Nodes that have been added to NCM have Yes in the NCM - Licensed column.

   - If the Manage Entities view opens instead of the Manage Nodes view, click Commands > Switch Back to Legacy page.
   - If the Manage Nodes view does not show the NCM - Licensed column, click the icon at the far right of the table header and add the NCM - Licensed column.

2. Locate the nodes you want to add to NCM. For example:
   - To search for a specific node, enter the IP address or host name in the Search field, and click Search.
   - Use the Group By list to filter the nodes. For example, you can show all nodes of a certain
machine type, or all nodes that are not currently managed by NCM.

3. Select the nodes you want to add to NCM.

4. Click More Actions > Add Nodes to NCM, and then click OK at the confirmation message.

After you add a node, you must specify credentials that NCM can use to log in to the device.

Remove nodes from NCM

If you no longer want to manage configs or perform other NCM actions on a node, you can remove the node from NCM.

⚠️ If you remove a node from NCM, all data associated with the node, such as configs and inventory data, are also removed.

1. Click Settings > Manage Nodes.

   The Manage Nodes view lists all of the nodes that have been added to the Orion Platform. Nodes that have been added to NCM have Yes in the NCM - Licensed column.

2. Select the nodes you want to remove from NCM.

3. Click Edit Properties.

4. Scroll down to the Manage Node(s) with NCM list, and select one of the following values:
   
   - Select **No** if you currently do not want the node managed with NCM. It can be added to NCM later through discovery.
   - Select **Never** if you never want the node managed by NCM. It will never be added to NCM through discovery, but it can be added manually.

5. Click Submit.

Options for specifying NCM connection information

After you add a node for NCM for monitoring, you must specify the connection information that NCM uses to log in to nodes and perform tasks such as downloading configurations, performing firmware upgrade operations, or running scripts. This information is stored in each device's connection profile.
NCM provides the following options for defining connection information:

- Device-level login credentials
- Global connection profiles
- User-level login credentials

### Device-level login credentials

The connection profile for each node specifies the values that NCM uses to connect to that node. Global variables can be used to define the default connection values for all nodes. They can be overridden for individual nodes.

<table>
<thead>
<tr>
<th>Connection Profile</th>
</tr>
</thead>
<tbody>
<tr>
<td>Global Connection Profile: &lt;No Profile&gt;</td>
</tr>
<tr>
<td>Login Credentials: Device</td>
</tr>
<tr>
<td>Username: ${GlobalUsername}</td>
</tr>
<tr>
<td>Password: ***************</td>
</tr>
<tr>
<td>Enable Level: ${GlobalEnableLevel}</td>
</tr>
<tr>
<td>Enable Password: ***************</td>
</tr>
<tr>
<td>Execute Scripts Using: ${GlobalExecProtocol}</td>
</tr>
<tr>
<td>Request Configs Using: ${GlobalConfigRequestProtocol}</td>
</tr>
<tr>
<td>Transfer Configs Using: ${GlobalConfigTransferProtocol}</td>
</tr>
<tr>
<td>Telnet Port: ${GlobalTelnetPort}</td>
</tr>
<tr>
<td>SSH Port: ${GlobalSSHPort}</td>
</tr>
</tbody>
</table>

Global variables define common connection values, but they can be overridden for a specific device.

For more information, see [Configure nodes to use device-level login credentials for NCM connections](#).

### Global connection profiles

A global connection profile defines a set of connection values that can be used by multiple nodes. You can define multiple connection profiles for different groups of devices. To apply a connection profile to a node, select it in the node’s connection profile.
For more information, see [Configure nodes to use global connection profiles for NCM connections](#).

### User-level login credentials

User-level login credentials define a set of credentials associated with an Orion user account instead of a node. When a user logs in with an Orion account and performs an action in NCM, NCM connects to the node with the user-level login credentials associated with that user's Orion account.

User-level login credentials define only the **Username, Password, Enable Level, and Enable Password**. You must specify the value of other fields either manually or through global variables.

For more information, see [Configure nodes to use user-level login credentials for NCM connections](#).
Configure nodes to use device-level login credentials for NCM connections

When a node is configured to use device-level login credentials, the connection profile for that node specifies the login credentials and other connection information NCM uses to connect to that node. By default, the connection profile for each NCM node uses a set of global variables to define the value of each field, as shown below. You can specify a different value for any field.

<table>
<thead>
<tr>
<th>Connection Profile</th>
</tr>
</thead>
<tbody>
<tr>
<td>Global Connection Profile:</td>
</tr>
<tr>
<td>Login Credentials:</td>
</tr>
<tr>
<td>Username:</td>
</tr>
<tr>
<td>Password:</td>
</tr>
<tr>
<td>Enable Level:</td>
</tr>
<tr>
<td>Enable Password:</td>
</tr>
<tr>
<td>Execute Scripts Using:</td>
</tr>
<tr>
<td>Request Configs Using:</td>
</tr>
<tr>
<td>Transfer Configs Using:</td>
</tr>
<tr>
<td>Telnet Port:</td>
</tr>
<tr>
<td>SSH Port:</td>
</tr>
<tr>
<td>Reset to ${GlobalUsername}</td>
</tr>
<tr>
<td>Reset to ${GlobalPassword}</td>
</tr>
<tr>
<td>Reset to ${GlobalEnableLevel}</td>
</tr>
<tr>
<td>Reset to ${GlobalEnablePassword}</td>
</tr>
<tr>
<td>Reset to ${GlobalSSHPort}</td>
</tr>
</tbody>
</table>

To configure devices to use device-level login credentials, complete the following tasks.

For information about other configuration options, see Options for specifying NCM connection information.

Gather information

Before you begin:

- Locate the login credentials for the node you want to add to NCM.
- Determine whether NCM communicates with the device by way of Telnet or SSH.
- Identify the Telnet or SSH ports used for communication.

Review values for global variables

Review the values for global variables and change them if necessary. See Specify values for global variables.
Override default settings for a specific node

If the value of a global variable is not appropriate for a specific node, edit that node's connection profile to specify a different value.

1. Click Settings > All Settings.
2. Under Node & Group Management, click Manage Nodes.
3. Use the Group By list to filter the nodes.
4. Select a node in the list, and click Edit Properties.
5. Scroll down to NCM Properties.
6. For the Global Connection Profile, select <No Profile>.
7. For Login Credentials, select Device.
8. Enter the values for any fields that should not use the global variable values.

For example, you can override any of the following:

a. Enter the Username and Password to log in to this device.

b. Under Enable Level, select enable if NCM must enter a command to provide a higher level of access on the network devices. On some devices, this command is required to allow NCM to back up configs, upload changed configs, and run certain scripts.
If a command is required, but the command is *not* enable, do the following:

a. Select enable in the connection profile.

b. In the **device template** that defines the device-specific commands used to perform actions on that device, include the following:

   ```xml
   <Command Name="EnableCommand" Value="commandForMyDevice"/>
   ```

   where *commandForMyDevice* is the command entered on your device (for example, `super`).

If the NCM account has the privileges it needs without having to enter a command such as *enable*, then make sure *enable* is *not* selected in the connection profile.

c. If you selected *enable*, enter the password for it.

d. For Execute Scripts Using, choose the protocol that NCM should use to execute commands and scripts.

e. For Request Configs Using, choose the protocol that NCM should use to log in to a device and request a config transfer.

   ```
   For more information about Config Request and Transfer Config options, see [Protocols NCM uses to transfer configurations](#).
   ```

f. For Transfer Configs using, choose the protocol that NCM should use to upload and download configs.

g. Set the Telnet and SSH ports to the ports allowed on your network.

9. Click Submit.

**Configure nodes to use global connection profiles for NCM connections**

A global connection profile specifies the information that NCM needs to connect to a node, including credentials, communication protocols, and ports. Each global connection profile can be applied to multiple nodes. You can define multiple connection profiles. When connection information changes (for example, a password is updated), you can update the value for all associated nodes in one place.

```
For information about other configuration options, see [Options for specifying NCM connection information](#).
```
Gather information

Before you begin:

- Locate the login credentials for the node you want to add to NCM.
- Determine whether NCM communicates with the device by way of Telnet or SSH.
- Identify the Telnet or SSH ports used for communication.

Create a global connection profile

1. Click Settings > All Settings.
2. Under Product Specific Settings, click NCM Settings.
4. Click Create New.
5. Enter the credentials to log in to the devices that will be associated with this profile.

See the following section for information about associating devices with a profile.

6. Under Enable Level, select enable if NCM must enter a command to provide a higher level of access on the network devices. On some devices, this command is required to allow NCM to back up configs, upload changed configs, and run certain scripts.

If a command is required, but the command is not enable, do the following:

a. Select enable in the connection profile.

b. In the device template that defines the device-specific commands used to perform actions on that device, include the following:

   <Command Name="EnableCommand" Value="commandForMyDevice"/>

   where commandForMyDevice is the command entered on your device (for example, super).

If the NCM account has the privileges it needs without having to enter a command such as enable, then make sure enable is not selected in the connection profile.

7. If you selected enable, enter the password for it.
8. Specify the communication protocol and ports:
a. For Execute Scripts Using, choose the protocol that NCM should use to execute commands and scripts.

b. For Request Configs Using, choose the protocol that NCM should use to log in to a device and request a config transfer.

For more information about Config Request and Transfer Config options, see Protocols NCM uses to transfer configurations.

c. For Transfer Configs using, choose the protocol that NCM should use to upload and download configs.

d. Set the Telnet and SSH ports to the ports allowed on your network.

9. If you want NCM to attempt to automatically assign this profile to applicable devices, select Automatically test this profile.

When this option is selected, NCM attempts to use this profile when <Auto Detect> is selected in a device's Global Connection Profile field.

If you have multiple global connection profiles with this option selected, NCM tests them sequentially when it needs to connect to a device with <Auto Detect> selected. The first global connection profile that works for the device is permanently assigned to the device.

10. Click Submit.

Assign a global connection profile to devices

1. Choose Settings > Manage Nodes.

2. Select one or more nodes.

3. Click Edit Properties.

4. Scroll down to Manage node(s) with NCM, and verify that Yes is selected.

5. If multiple nodes are selected, click the Connection Profile check box to enable editing.

6. For Global Connection Profile, select one of the following:
   - Select a global connection profile.
   - Select <Auto Detect> to have NCM determine which global connection profile to apply. (The global connection profile must have Automatically test this profile selected.)

All of the Connection Profile values are defined by the global connection profile. They cannot be overridden locally or defined by macros.
7. Click Submit.

Configure nodes to use user-level login credentials for NCM connections

When NCM uses user-level login credentials to connect to a node, the login credentials are associated with an Orion user account rather than the node. Each NCM user defines a set of credentials that NCM can connect with. Each set of login credentials is associated with the Orion account under which it was defined:

- When a user logs in with an Orion account and performs an action in NCM, NCM connects to the node with the user-level login credentials associated with that Orion account.
- When NCM processes a job, it connects to the node with the user-level login credentials associated with the Orion account of the user who created or most recently edited the job definition.

The access provided by user-level login credentials can vary.

For information about other configuration options, see Options for specifying NCM connection information.

To configure user-level login credentials, complete the following tasks.

Gather information

Before you begin:

- Locate the login credentials for the node you want to add to NCM.
- Determine whether NCM communicates with the device by way of Telnet or SSH.
Identify the Telnet or SSH ports used for communication.

Enable user-level login credentials in NCM

User-level login credentials are not enabled by default. Complete the following steps to enable them.

1. Click Settings > All Settings.
2. Under Product Specific Settings, click NCM Settings.
5. Click Submit.

Define user-level login credentials

Each NCM user must define the credentials that NCM will use to log in to devices. When credentials change (for example, when passwords are updated on the authentication server), each user must update the user-level login credentials in NCM.

1. Log in to the Orion Web Console with the Orion user account that will be associated with this set of user-level login credentials.
2. Click Settings > All Settings.
3. Under Product Specific Settings, click NCM Settings.
5. Enter the user name and password that NCM should use to access the network device.
6. Under Enable Level, select enable if NCM must enter a command to provide a higher level of access on the network devices. On some devices, this command is required to allow NCM to back up configs, upload changed configs, and run certain scripts.

If a command is required, but the command is not enable, do the following:

a. Select enable in the connection profile.

b. In the device template that defines the device-specific commands used to perform actions on that device, include the following:

<Command Name="EnableCommand" Value="commandForMyDevice"/>

where commandForMyDevice is the command entered on your device (for example, super).
If the NCM account has the privileges it needs without having to enter a command such as `enable`, then make sure `enable` is not selected in the connection profile.

7. If you selected `enable`, enter the password for it.

8. Click Submit.

Assign user-level login credentials to nodes

1. Choose Settings > Manage Nodes.

2. Select one or more nodes.

3. Click Edit Properties.

4. Scroll down to Manage node(s) with NCM, and verify that Yes is selected.

5. If multiple nodes are selected, click the Connection Profile check box to enable editing.

6. For Login Credentials, select User.

   The user name, password, enable level, and enable level password can no longer be specified because they are defined by the user account.

   ![Connection Profile Table]

   - Login credentials are determined by the user account.
   - Other values use global variables or are entered manually.

   ![Connection Profile Table]

7. If necessary, change the values selected for protocols and ports.

   If the connection profile uses global variables, you can review or change the value of the global variables.

   To override global variables, update any of the following:

   a. For Execute Scripts Using, choose the protocol that NCM should use to execute commands and scripts.

   b. For RequestConfigs Using, choose the protocol that NCM should use to log in to a device
and request a config transfer.

For more information about Config Request and Transfer Config options, see Protocols NCM uses to transfer configurations.

c. For Transfer Configs using, choose the protocol that NCM should use to upload and download configs.
d. Set the Telnet and SSH ports to the ports allowed on your network.

8. Click Submit.

Specify values for global variables

Global variables allow you to specify common values used to enable NCM to communicate with devices. Global variables can be used in scripts and they can be specified in a device’s connection profile.

<table>
<thead>
<tr>
<th>Connection Profile</th>
<th>&lt;No Profile&gt;</th>
<th>Device</th>
<th>$(GlobalUsername)</th>
<th>Password</th>
<th>$(GlobalEnableLevel)</th>
<th>Enable Password</th>
<th>$(GlobalExecProtocol)</th>
<th>$(GlobalConfigRequestProtocol)</th>
<th>$(GlobalConfigTransferProtocol)</th>
<th>$(GlobalTelnetPort)</th>
<th>$(GlobalSSHPort)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Global Connection Profile:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Login Credentials:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Username:</td>
<td></td>
<td></td>
<td>$(GlobalUsername)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Password:</td>
<td></td>
<td></td>
<td>* * * * * * * * *</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enable Level:</td>
<td></td>
<td></td>
<td>$(GlobalEnableLevel)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enable Password:</td>
<td></td>
<td></td>
<td>* * * * * * * * *</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Execute Scripts Using:</td>
<td></td>
<td></td>
<td>$(GlobalExecProtocol)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Request Configs Using:</td>
<td></td>
<td></td>
<td>$(GlobalConfigRequestProtocol)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Transfer Configs Using:</td>
<td></td>
<td></td>
<td>$(GlobalConfigTransferProtocol)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Telnet Port:</td>
<td></td>
<td></td>
<td>$(GlobalTelnetPort)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SSH Port:</td>
<td></td>
<td></td>
<td>$(GlobalSSHPort)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

If you use global variables in scripts and connection profiles, specify the appropriate values for your environment.

1. Click Settings > All Settings.
2. Under Product Specific Settings, click NCM Settings.
4. In the Device Login Information section:
   a. Specify the Username and Password that NCM should use to connect to devices.
   b. Under Enable Level, select enable if NCM must enter a command to provide a higher level of access on the network devices. On some devices, this command is required to allow
NCM to back up configs, upload changed configs, and run certain scripts.

If a command is required, but the command is not enable, do the following:

a. Select enable in the connection profile.

b. In the device template that defines the device-specific commands used to perform actions on that device, include the following:

```xml
<Command Name="EnableCommand" Value="commandForMyDevice"/>
```

where commandForMyDevice is the command entered on your device (for example, super).

If the NCM account has the privileges it needs without having to enter a command such as enable, then make sure enable is not selected in the connection profile.

c. If you selected enable, enter the password for it.

5. In the Communication Transfer Protocol section:

a. For Execute Scripts Using, choose the protocol that NCM should use to execute commands and scripts.

b. For Request Configs Using, choose the protocol that NCM should use to log in to a device and request a config transfer.

For more information about Config Request and Transfer Config options, see Protocols NCM uses to transfer configurations.

c. For Transfer Configs using, choose the protocol that NCM should use to upload and download configs.

6. In the Transfer Ports section, set the Telnet and SSH ports to the ports allowed on your network.

7. If device template fields have been set to override global variables and you want to use global variables instead of individually selected values, select Reset All devices to use Global Settings. Select this option for each group of values you want to reset.

8. Click Submit.

Protocols NCM uses to transfer configurations

You can specify the protocols that NCM uses to transfer configurations in the device's connection profile, a global profile, or in global variables. NCM can use the following protocols.
Telnet or SSH

When NCM downloads a configuration using Telnet or SSH (Secure Shell), it logs in to the device and then runs commands on the device to display the configuration. When the device displays the configuration, NCM parses the information and stores it.

To use Telnet or SSH, specify the following options in the connection profile, global profile, or global variables:

- Select the desired protocol (Telnet or SSH) as the value for the Request Config option.
- Select the same protocol as the value for the Transfer Config option.

Transfers using Telnet or SSH are direct transfers.

TFTP

When NCM downloads a configuration using TFTP (Trivial File Transfer Protocol), it uses Telnet or SSH to log in to the device and initiate the TFTP session. Then it uses TFTP to transfer the configuration file from the device to the TFTP server. The SolarWinds TFTP server is included with NCM. It is automatically installed on the Orion server and on any additional polling engines.

To use TFTP:

1. Specify the following options in the connection profile, global profile, or global variables:
   - Select Telnet, SSH, or SNMP as the value for the Request Config option.

   SNMP (Simple Network Management Protocol) can be used only with Cisco devices that support this protocol, and it requires additional configuration. You must configure the SNMP read-write community string on the device and ensure that NCM has been set up with the same string. The device must support the CISCO-CONFIG-COPY-MIB.

   - Select TFTP as the value for the Transfer Config option.

2. Configure the TFTP server as needed for your environment.

Transfers using TFTP are indirect transfers.
SCP

When NCM downloads a configuration using SCP (Secure Copy Protocol), it uses Telnet or SSH to log in to the device and initiate the SCP session. Then it uses SCP to transfer the configuration file from the device to the SCP server. The SolarWinds SCP server is included with NCM. It is automatically installed on the Orion server and on any additional polling engines. You can also choose to use a third-party SCP server.

Binary device configs can be transferred only with SCP.

To use SCP:
1. Specify the following options in the connection profile, global profile, or global variables:
   - Select Telnet or SSH as the value for the Request Config option.
   - SSH is typically used with SCP.
   - Select SCP as the value for the Transfer Config option.
2. Configure the SolarWinds SCP server or a third-party SCP server as needed for your environment.

Transfers using SCP are indirect transfers.

Troubleshoot NCM device connections

When you experience problems connecting to a device, you may need to perform a session trace to troubleshoot the issue. A session trace shows all communication sent between NCM and the network device to which you are connecting. The session trace log contains error messages and commands sent that generated the error.

1. Click Settings > All Settings.
2. Under Product Settings, click CLI Settings.
3. Select Enable Session Tracing.
4. Perform the steps to recreate the issue you are troubleshooting.
5. Open the session trace file.
6. Edit the device template and apply the necessary changes to resolve the issue.
7. Turn off Enable Session Tracing when the problem has been resolved.
Manage nodes and collect node data in NCM

Network Configuration Manager helps you manage, organize, and track changes to your network devices. These devices are known collectively as nodes and include switches, routers, firewalls, and Windows servers.

- **Unmanage nodes** during maintenance periods to prevent NCM from performing actions on the nodes.
- Manage [end of support and end of sales data](#)
- Collect [inventory data](#)
- Find the [connected port for a host](#)

**Unmanage and remanage NCM nodes**

NCM does not perform job actions on unmanaged nodes. When performing maintenance on nodes, unmanage the nodes to discontinue processing jobs against them. Maintenance includes upgrading firmware, installing new hardware, or updating security.

Configuration data for unmanaged nodes will remain in the Orion Platform database.

**Access the management page**

1. Click Settings > All Settings.
2. Under Node & Group Management, click Manage Nodes.

**Unmanage nodes**

Select the nodes you want to unmanage in the nodes list, and click Unmanage.

**Remanage nodes**

Select the nodes you want to remanage in the nodes list, and click Remanage.
Manage end of support and end of sales data in NCM

To avoid the risks associated with unsupported or out-dated devices, you can use NCM to track the manufacturer’s End of Sales and End of Support (EOS) dates for your devices. NCM imports EOS data for Cisco devices and recommends EOS dates for your devices when possible. For other devices, you can manually enter EOS dates from the manufacturer. When you assign EOS dates to devices, you can track this data as part of the device profile. You can also filter the data to show a specific set of devices and export the results to a spreadsheet or CSV file.

- Assign EOS dates to devices
- Refresh EOS dates
- Ignore devices in EOS Management
- View or re-add ignored devices
- Filter the list of devices by the values in a table column
- Export EOS information
- Delete EOS data

SolarWinds neither verifies nor supports EOS data provided by EOS Lookup. Consult your vendor with any data-related issues or questions.

Assign EOS dates to devices

NCM suggests EOS dates for some devices based on information in its databases. You can assign suggested dates, or you can enter EOS dates manually.

1. Click My Dashboards > Network Configuration > End of Support.
2. Select the devices for which you want to assign EOS dates.
3. Click Assign Dates.

The Assign EOS Dates page shows the number of nodes selected, and lists the machine type, OS version, or part number. Use this information to verify that all selected devices are the same type and are therefore likely to have the same EOS dates. If you selected devices with different machine names, OS versions, or part numbers, you can deselect devices by clicking the remove icon.
4. If suggested are provided for the selected nodes, search the table for the dates that correspond to your device model. If you find the appropriate dates, select that row.

The dates listed for a model or series have indications in the Reliability column:

- High indicates that the dates are unambiguous and such dates are automatically applied to the relevant devices.
- Medium indicates that the dates are classified as ambiguous due to other incomplete or conflicting information.
- Confirmed indicates that the dates were confirmed by an NCM user.

5. If no suggested dates were provided or you want to enter different dates, select Option 2 and enter the dates.

6. Add comments, and click Assign.

Refresh EOS dates

NCM maintains a database with EOS data for Cisco device models. Based on a schedule, NCM matches EOS data with the machine type of your NCM devices. What you see in the table of the EOS resource is the result of the matching. To make sure you are looking at the latest matches, use Refresh Suggested Dates.

1. Click My Dashboards > Network Configuration > End of Support.
2. Select the devices for which you want updated data, if available.
3. Click Refresh Suggested Dates.

Ignore devices in EOS Management

1. Click My Dashboards > Network Configuration > End of Support.
2. Select the devices you want to ignore.
3. Click Ignore Devices.

By default, the ignored devices are no longer shown in the End of Support & End of Sales Lookup list. If the Display ignored devices option is selected, the devices are shown with Ignored in each date column.
View or re-add ignored devices

1. To include ignored devices in the End of Support & End of Sales Lookup list, select Display ignored devices.

2. To stop ignoring a device, select the device and click Delete EOS Data. Then click Yes at the confirmation prompt.

*Ignored* is no longer shown in each date column for the selected device.

Filter the list of devices by the values in a table column

You can filter the list of devices by the data in any table column. For example, you can use filters to:

- List devices that reach End of Support or End of Sales within a certain time period.
- Display the EOS status of devices in the same subnet.
- List devices with No Suggestions in an EOS date column.

You can add multiple filters. Filters are applied in the order that they are created.

1. Click the Filter icon next to the table column title.
   
   A list of filters appropriate for that column is displayed.

2. Select a filter to apply it.

   The table is updated to list only devices that meet the filter condition. The column title is shown in bold and italics to indicate that a filter is applied to the column data, and each filter
condition is shown above the table.

| End of Support: Next year | End of Sales: Next 3 months |

To remove one filter, click the delete icon in the box above the table that represents the filter. To remove all filters, click Clear All Filters below the table.

Export EOS information

You can export EOS data to an Excel worksheet or a CSV (comma-separated values) file.

1. Click My Dashboards > Network Configuration > End of Support.

2. Filter the list to include only the devices whose EOS data you want to export.
   - Use the Group By option to show devices with a certain attribute. For example, include only devices that are from the same vendor or used by the same department.
   - Filter the list of devices by the values in a table column. For example, include only devices that reach End of Support in the next three months.

3. Click Export, and select the format.
   The data is exported to a file, and the file is saved in your Downloads folder.

Delete EOS data

1. Click My Dashboards > Network Configuration > End of Support.

2. Select the devices whose EOS data you want to delete.

3. Click Delete EOS Data, and then click Yes at the confirmation prompt.
Run an NCM inventory scan and view inventory data

Perform inventory scans to collect detailed information about your devices, including:

- Serial numbers
- Port details
- IP addresses
- Vendors
- End-of-life dates
- End-of-support dates
- Maintenance providers

The collected inventory statistics are displayed in NCM inventory reports and in the detail view of each device.

For example, you can run an inventory scan of all IP addresses inventoried on each device. A searchable list of IP addresses can help you locate a specific address in a large network.

NCM uses SNMP communication to collect inventory information. Properly configure SNMP on all devices from which you want to collect detailed information.

To collect and review inventory data, perform the following tasks:

- Manually run an inventory scan, or schedule an inventory scan to ensure that inventory data is up to date.
- If necessary, adjust inventory settings.
- Access NCM inventory reports to view the collected data.

Run an inventory scan

You can run a scan at any time to get the latest statistics on your full inventory of managed devices in NCM. You may also run a single-node scan.

Run a complete inventory scan

A full inventory scan can take anywhere from a few minutes to several hours to complete. The time period varies based on the number of nodes and the type of statistics you want to collect. For more information on how to establish what statistics are collected, see Adjust inventory settings.

1. Click My Dashboards > Network Configuration > Configuration Management.
2. Use the Group By list and select No Grouping.
3. Select all nodes.
4. Click Update Inventory.
5. Click Yes at the confirmation prompt.

Run an inventory scan on a single node
1. Click My Dashboards > Network Configuration > Configuration Management.
2. Use Search or the Group By list to find a node.
3. Select the node, and click Update Inventory.
4. Click Yes at the confirmation prompt.

View inventory status
After you run a scan, you can view the status.
1. Click My Dashboards > Network Configuration > Configuration Management.
2. Click the Inventory Status tab.
3. Click a column to change the sort order.

Schedule an inventory scan
To ensure that inventory information is always up to date, schedule inventory scans to run regularly.
1. Click My Dashboards > Network Configuration > Jobs.
2. Click Create New Job.
3. Name the job, and then select Update Inventory from Job Type.
4. Specify when the job runs:
   - To run the job once or on a simple schedule, select Basic. Click the tab that identifies how frequently the job runs, and then specify the start time and (if needed) the day(s).
   - To create a more complex schedule, select Advanced and then use the five fields to create a CRON expression.
5. Add a comment if this job relates to a business rule, and click Next.
6. Select the nodes to target with this job, and click Next.
7. Select an email notification option, and click Next. If you click Email Results, the default email notification and SMTP server settings are populated. These settings can be overridden in each job.
8. Select the information types to include in the Inventory job, and click Next.

9. Review the settings for the job, and click Finish.

Adjust inventory settings

Node Inventory settings allows you to specify the type of inventory information NCM collects, and provides other options to help you manage the inventory process. For example, it may be necessary to change the number of devices that NCM can inventory concurrently.

1. Click Settings > All Settings.

2. Under Product Specific Settings, click NCM Settings.

3. Under NCM Node Management, click Node Inventory.

4. Select the information types to collect.

5. Adjust the number of devices that should be inventoried concurrently. The default is five.

6. If you have VLANs extended across network trunks, and you want to inventory the relevant devices, select Extend VLANs inventory.

   SolarWinds recommends against this situation due to the slow performance of the inventory process.

7. If the inventory process causes the NCM server to hang or if the process takes too long, select Disable Inventory Lookup.

   Determine what caused the problem, for example, inadequate server capacity, before re-enabling Inventory Lookup.

8. Click Submit.

Access NCM inventory reports

1. Click Reports > All Reports.

2. Group by Report Category.

3. Select one of the NCM inventory report categories, such as:
   - NCM Brocade Inventory
   - NCM Cisco Inventory
   - NCM F5 Inventory
   - NCM Inventory
4. Click a report name to view it.

For more information about the available reports, see Predefined NCM reports.

Find connected port for a host

You can use NCM to search for currently connected ports on wired or wireless end hosts. This feature requires that you manage both nodes and interfaces involved in the connection.

The information returned is based on the data available in the Orion database as of the last scheduled discovery of network nodes. It is specified at the top of the table in which search results are presented in this form: As of last discovery [MM/DD/YYYY] [HH:MM:SS] [AM/PM].

You can search for connected ports by these node properties:

- IP Address
- DNS Hostname
- MAC Address
- Port Description

To search for connected ports, complete the following steps:

1. Click My Dashboards > Network Configuration > Config Summary.
2. Under Find Connected Port for End Host, select the Search By filter, and enter a value in the Find field.
3. Click Edit in the resource header to adjust the columns of data to include in your search results.
4. Click Find.

Results are presented as rows in a table, one for each connection within the reach of hop of the context node, with the following default columns:

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For Wired Devices</strong></td>
<td></td>
</tr>
<tr>
<td>Node</td>
<td>Vendor and model of context node</td>
</tr>
<tr>
<td>IP Address</td>
<td>Of the context node</td>
</tr>
<tr>
<td>MAC Address</td>
<td>Of the context node</td>
</tr>
<tr>
<td>Connected Via Interface</td>
<td>On the context node</td>
</tr>
<tr>
<td>To This Interface</td>
<td>On a connection point</td>
</tr>
<tr>
<td>On This Node</td>
<td>Vendor and model of device supporting the connection point</td>
</tr>
<tr>
<td>Column</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>IP Address</td>
<td>Of a connection point</td>
</tr>
<tr>
<td>MAC Address</td>
<td>Of a connection point</td>
</tr>
<tr>
<td><strong>For Wireless Devices</strong></td>
<td></td>
</tr>
<tr>
<td>Mapped Host Name</td>
<td>Vendor and model of context node</td>
</tr>
<tr>
<td>Mapped MAC Address</td>
<td>Of context node</td>
</tr>
<tr>
<td>Mapped Device Type</td>
<td>Of context node</td>
</tr>
<tr>
<td>Source Interface</td>
<td>Of the wireless access point</td>
</tr>
<tr>
<td>Controller Source IP Address</td>
<td>Of the wireless access point</td>
</tr>
<tr>
<td>Controller Description</td>
<td>Of the wireless access point</td>
</tr>
<tr>
<td>Controller Host Name Source SSID</td>
<td>Of the wireless access point</td>
</tr>
<tr>
<td>Source Channel</td>
<td>Of the wireless access point</td>
</tr>
<tr>
<td>Source Interface Alias</td>
<td>Of the wireless access point</td>
</tr>
<tr>
<td>Source Radio Type</td>
<td>Of the wireless access point</td>
</tr>
<tr>
<td>Source Host Name</td>
<td>Of the wireless access point</td>
</tr>
<tr>
<td>Source Device Type</td>
<td>Of the wireless access point</td>
</tr>
<tr>
<td>Source IP Address</td>
<td>Of the wireless access point</td>
</tr>
</tbody>
</table>
Manage network configuration files

When nodes are added to NCM, you can download and edit the configuration files of those devices, search configuration files, and create an archive of config files for backup. See the following topics:

- NCM setup required for binary configs
- Set NCM communication limits
- Specify Config Settings for NCM
- Download configuration files in NCM
- Schedule daily backups of network config files
- Enable a new config type in NCM
- Enable the NCM config and policy caches
- Edit network config files
- Upload a network config
- Import network config files
- Upload network configuration changes
- Archive downloaded configuration files
- Search network config files or node properties
- Delete network config files from the database
- Remove older NCM data from the database and config archive
- Limit the number of simultaneous uploads or downloads of network configs
- Troubleshoot issues with NCM configuration files

NCM setup required for binary configs

If you have devices whose configurations files are in a binary format instead of a plain text format (for example, F5 BIG-IP load balancers), you can use NCM to back up these binary config files. (Binary config files cannot be opened and read or edited in a text editor.)

⚠️ Check out this SolarWinds Lab bit about binary config files in NCM (0:1:47 long, published April 27, 2018).

NCM stores binary configuration files that you download on a network share. If you will be managing device configurations that are in binary format, you must first specify the network share that you want NCM to use.

ℹ️ NCM cannot be used to edit or compare binary configuration files.
1. Click Settings > All Settings in the menu bar.
2. In the Product Specific Settings section, select NCM Settings.
3. In the Configs section, select Binary Config Storage Settings.
4. Enter the path to the network share. For example: `\MYLAB-SERVERNAME-01\binarystorage`.
5. Enter the credentials required to access the network share.
6. To validate the connection before submitting your settings, click Validate.
7. When finished, click Submit.

- The NCM server must be in the same domain as the network share system.

**Set NCM communication limits**

Define timeout values and retry numbers for ICMP, SNMP, Telnet, and SSH communication.

1. Click Settings > All Settings.
2. Under Product Specific Settings, click NCM Settings.
4. Enter timeout settings for each protocol, and click Submit.

**Specify Config Settings for NCM**

Use the Config Settings page to specify how NCM manages your device configurations.

1. Click Settings > All Settings.
2. Under Product Specific Settings, click NCM Settings.
3. Under Config Settings, click Config Settings.
4. Adjust any of the values described below.
5. Click Submit.

**Config Transfer**

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simultaneous Downloads/Uploads</td>
<td>Use this setting to throttle the load that a big job could put on the network. By default, this value is set to 25 download/upload sessions.</td>
</tr>
<tr>
<td>SNMP Config Transfer Timeout</td>
<td>Specify the maximum amount of time before an SNMP configuration file transfer times out.</td>
</tr>
</tbody>
</table>
**Config Min Length**  
Specify the minimum number of lines that a configuration file must contain to be recognized by NCM as valid for download.

By default, NCM does not upload or download any configuration file with fewer than 11 lines. Change this default value if you plan to work with configuration files that have fewer lines.

---

**Config Comparisons**

Adjust these settings to change how the Compare Configs Of Selected Nodes displays comparisons when you **manually compare configurations**.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Comparison Output Width</strong></td>
<td>If a line in a configuration is longer than 250 characters, increase this value to prevent the line from being truncated on the Compare Configs page.</td>
</tr>
</tbody>
</table>
| **Display the full side-by-side comparison** | Specify this option to display all config lines, both changed and unchanged.  
   - Selecting this option increases load times for large configs. |
| **Display the number of lines** | Select this option to display only changed lines and the specified number of unchanged lines before and after each change. The default is 5. 
   - This option is selected by default. With this option, config changes are easier to identify and large configs load more quickly. |

---

**Config Archive**

To enable archiving, select Save a copy of each config into the archive directory, as it is downloaded.

To save space on your storage target, and if you only need the immediately past version in addition to the current config, select When configs are edited, only retain the last version.

By default, NCM sets the directory for your Config Archive as `C:\ProgramData\SolarWinds\NCM\Config-Archive`. As preparation for a growing archive, SolarWinds recommends that you move the Config Archive from this location. You can enter a local directory or network share for the Config Archive.

If you want to store your Config Archive on a network share instead of a local directory on the NCM server, you must set up custom credentials that give NCM write access. To do that, open NCM Advanced Settings (Settings > NCM Settings > Advanced Settings), select Use custom credentials to give NCM write access, provide a valid user name and password, and click Validate Credentials to verify your entries.

Modify the template for saving new config files or accept the default:
Download configuration files in NCM

Download configuration files to view the current configuration of your managed devices, compare current and previous configurations, or archive configuration files for backup. NCM can transfer files using direct and indirect transfers.

If a configuration includes the command prompt character, NCM stops downloading. NCM provides the `UseExactPromptMatch` device template command you can use to instruct NCM to act on the command prompt only if it appears on a separate line. For more details about using this command, see Confine recognition of the command prompt during download.

Downloading from a single node

1. Log in to the Orion Console using an account that has the WebDownloader, WebUploader, Engineer, or Administrator role.
2. Click My Dashboards > Network Configuration > Config Summary.
3. Under NCM Node List, click a node in the list.
4. ClickConfigs on the left.
5. Under Download Config, select the config type, and click Download.

By default, NCM does not download a config with fewer than 11 lines. You can adjust the minimum configuration length if necessary.

6. If you receive a connection error, click Fix Connection in Device Template, and follow the Device Template wizard.

Downloaded configuration files are stored on your server in an archive. The location is specified in NCM Settings > Config Settings > Config Archive Folder Locations.

Downloading from multiple nodes

1. Click My Dashboards > Network Configuration > Jobs.
2. Click Create New Job.
3. Name the job, and then select Download Configs from Devices from Job Type.

By default, NCM does not download a config with fewer than 11 lines. You can adjust the minimum configuration length if necessary.
Specify when the job runs:

- To run the job once or on a simple schedule, select Basic. Click the tab that identifies how frequently the job runs, and then specify the start time and (if needed) the day(s).

- To create a more complex schedule, select Advanced and then use the five fields to create a CRON expression.

4.

5. Add a comment if this job relates to a business rule, and click Next.

6. Select the nodes to target with this job, and click Next.

7. Select an email notification option, and click Next. If you click Email Results, the default email notification and SMTP server settings are populated. These settings can be overridden in each job.

8. Select the configuration types you want to download.

9. Set your notifications preference, and click Next.

10. Review the settings for the job, and click Finish.

Schedule daily backups of network config files

To ensure that network configs are backed up on a regular basis, NCM includes the Nightly Config Backup job, which downloads the config files for all devices every night. This job is enabled by default, so it runs automatically.

- Config backups are stored in the Orion database and (optionally) the config archive location.

Examine the default settings of the default job to make sure they meet your needs. If necessary, you can change the defaults.

- NCM also includes the Weekly Config Backup job. This job is not enabled by default. If you prefer to back up configs weekly instead of daily, you can enable this job and disable the Nightly Config Backup job. SolarWinds does not recommend enabling both jobs.

Default settings

By default, the Nightly Config Backup job:

- Runs at 2:00 AM every day

- Downloads the Running and Startup configs from all nodes
The job runs only on devices that support the selected config types. If your devices support different config types, you can configure multiple backup jobs that target different groups of devices. For example, one job could target only Palo Alto devices and back up the Device State.

- Saves the results to a job log
- Does not send email notification when it runs
- Saves configs whether or not they changed
- Does not send notifications if configs have changed

**Changing the default settings**

You can change the default settings by editing the default job or by creating a copy of the job and editing the copy. Because changes to the default job cannot be reverted, SolarWinds recommends that you create a copy and edit the copy.

1. Click My Dashboards > Network Configuration > Jobs.
2. Select the Nightly Config Backup job.
3. Click Duplicate Job.
4. Specify a unique name. For example, include the name of your organization in the job name.
5. Specify when the job runs:
   - To run the job once or on a simple schedule, select Basic. Click the tab that identifies how frequently the job runs, and then specify the start time and (if needed) the day(s).
   - To create a more complex schedule, select Advanced and then use the five fields to create a CRON expression.
6. In the Comments box, include the reason for your changes, and then click Next.
7. Specify which devices this job targets, and click Next.
   - All Nodes is selected by default. Leave this selection to back up configs from all devices added to NCM.
   - If you do not want this job to run on all devices, select Dynamic Selection and specify the criteria used to select the devices.
For example, to create a job that runs on all Palo Alto devices, select **Vendor is Palo Alto Networks**. You could also create a second job to run on all other devices by selecting **Vendor is not Palo Alto Networks**. Click **View Selected Nodes** to verify that your criteria targets the expected nodes.

- Alternatively, you can choose **Select Nodes** and add specific nodes. However, you must remember to update the job definition whenever devices are added to the network.

8. Select an email notification option, and click Next. If you click **Email Results**, the default email notification and SMTP server settings are populated. These settings can be overridden in each job.

9. Select the configuration types you want to download, and specify whether you want to back up configs only if they have changed since the last backup.

10. If you want to be notified when a config has changed, select the send config change notifications. Then click Next.

11. Review the settings for the job, and click Finish.

   The job is enabled by default.

12. To avoid running both your custom job and the default job, disable the default job:

   a. In the Jobs List, select the Nightly Config Backup job.

   b. Click Disable.

### Enable a new config type in NCM

NCM includes the following standard config types:

- Startup
- Running
- Device State (Palo Alto devices only)

If you have devices that use other config types, you can define custom config types to be used with NCM operations. In NCM 7.9 and later, if the config type is used only by specific vendors, you can associate the config type with those vendors.
Add a custom config type

To add a custom config type, you must define the new type and then modify the templates for devices from which you intend to download the new config type.

1. Click Settings > All Settings.
2. Under Product Specific Settings, click NCM Settings.
3. Under Config Settings, click Config Types.

The Config Types page lists NCM's standard config types and any custom config types you have added.

4. Enter a unique name in the box at the bottom of the list, and click Add New.

   The config type is added to the list, with Edit and Remove options available.

5. See the NCM communication process diagrams to modify your device template with a statement.

   NCM uses the show command to download configurations.

Associate a custom config type with one or more vendors (NCM 7.9 and later)

When you associate a config type with a vendor, the config type is available for selection only when you are performing an operation on a device from that vendor. For example, when you download a config, the Download drop-down menu lists Device State only if a Palo Alto device is selected.
You can edit or remove only custom config types. If you click Edit next to a standard config type, its vendor associations are displayed but you cannot change them.

1. On the Config Types page, click Edit next to a custom config type.

2. Under Vendor operation, select one of the following:
   - **Is** associates the config with the vendors you select.
   - **Is not** associates all vendors except the ones you select (for example, all vendors except Juniper).

   The operation applies to all the vendors you select.

   If the Vendor operation drop-down menu is not shown or cannot be opened, you are attempting to edit a standard NCM config type. Standard config types are not editable.

3. Under Vendor, select a vendor and click Add New to add it to the list.

   The Vendor drop-down menu includes vendors of devices managed by NCM.

4. Click OK.

**Enable the NCM config and policy caches**

When the config and policy caches are enabled, NCM automatically updates cached configuration information at the specified time. By default, the config and policy caches are enabled, and the policy cache is generated at 11:55 PM.

1. Click Settings > All Settings.

2. Under Product Specific Settings, click NCM Settings.


5. Specify what time to generate the policy cache.

6. Click Submit.

**Edit network config files**

To update access lists, modify community strings, or make other configuration changes, edit the configuration files you downloaded with NCM.
1. Click My Dashboards > Network Configuration > Config Summary. Use an account that has the WebDownloader, WebUploader, Engineer, or Administrator role.

2. Click a node in the node list on the left.

3. Click Configs on the left.

4. Under Config List, select a config, and click Edit Config.

5. Edit the Config Title if necessary.

6. Select Edit Config Text, and make changes.

7. Add comments, and click Submit.

The revision is saved in the Config List with the indication that it is an Edited config of its type.

Upload the edited config file

1. In the Upload Config widget, select the edited config file.

2. Click Upload.

Upload a network config

The Upload Config widget allows you to upload a configuration file you have previously downloaded from this node to NCM.

This widget helps you easily correct unauthorized or incorrect changes made to a device configuration. You can also write the uploaded configuration to NVRAM, essentially making it the startup configuration for the device.

In the case of a multi-node upload/download operation, the Simultaneous Downloads/Uploads setting can be used as a throttle. By default it runs 25 sessions simultaneously.

For IPv6, you can rediscover devices that were previously discovered with the engine using IPv4. You can do inventories for devices already discovered with IPv4 or rediscovered with IPv6. Otherwise, new IPv6 addresses can be added to NCM, though IPv6 addresses cannot be communicated with through SNMP. You can execute scripts, upload, and download configuration files on IPv6 addresses. Telnet and SSH communications are supported.

If you selected a binary configuration for upload, editing of the configuration is not an option from the Upload Config screen. Instead you will see the Binary Config notation in the editing pane.
Upload a config to one or more nodes

1. Click My Dashboards > Network Configuration > Configuration Management.
   
   If you do not have administrator privileges you might not be able to see some nodes. Your account must be a member of the WebUploader group to upload configs.

2. Select one or more nodes, and click Upload.

3. Expand the tree to locate the correct node.

4. Select the config you want to upload.

5. Click Advanced and select options, if desired.

6. Click Upload.

Import network config files

You can import configuration files that you have already downloaded from your devices into NCM. NCM imports Configuration files that use the following file formats:

- NCM Archive: `.Config`
- SolarWinds Cisco Config Downloader: `.CiscoConfig`
- Text File: `.txt`
- Configuration File: `.cfg`
- Any file in ASCII text

To import a config file, complete the following steps:

1. Click My Dashboards > Network Configuration > Config Summary.

2. In the NCM Node List widget, click a node name to open the Node Details page.

3. In the menu on the left, click the Configs icon.

4. In the Config List widget, click Import Config.

   The Import Config page opens.
5. (Optional.) Change the default Config Title and add comments.

6. Click Choose File, browse to the config file, and select it.

7. Click Submit.

The imported file is listed in the Config List widget.

**Upload network configuration changes**

For a multi-node upload or download operation, the Simultaneous Downloads/Uploads setting can be used as a throttle. By default it is set to run 25 Sessions simultaneously, but you can change this setting.
For IPv6, you can rediscover devices that were previously discovered with the engine using IPv4, and you can do inventories for devices already discovered with IPv4 or rediscovered with IPv6. Otherwise, new IPv6 addresses can be added to NCM, though IPv6 addresses cannot be communicated with through SNMP. You can execute scripts, upload, and download configuration files on IPv6 addresses, and Telnet and SSH communication is supported.

You can upload changes to a custom config type only to a single device, but through an indirect transfer protocol, such as Telnet or TFTP. As a result, the Write Config to NVRAM After Upload option is disabled.

Upload an entire configuration

1. Click My Dashboards > Network Configuration > Configuration Management.
2. Select one or more nodes.
3. Click Upload.
4. Select a config, and edit it if necessary.
5. Click Advanced at the bottom.
6. Select one or more advanced options, and then click Upload.

Archive downloaded configuration files

Network Configuration Manager stores all downloaded configurations in a database. It can also store a copy of them in the configuration archive directory. You can specify a local directory or a network storage location as the config archive.

Specify a local directory as the config archive

1. Click Settings > All Settings.
2. Under Product Specific Settings, click NCM Settings.
3. Under Config Settings, click Config Settings.
4. Under Config Archive, select Save a Copy of Each Config to the Config Archive Directory.
5. If you are low on storage space and only need the last and current configs, select WhenConfigs Are Edited, Only Retain the Last Version.
6. For the Orion server and each additional polling engine:
   a. Type the path of the local directory to store the NCM Config Archive.
      By default, NCM sets the directory for the config archive as %PROGRAMDATA%\SolarWinds\NCM\Config-Archive. To prepare for a growing archive, SolarWinds recommends moving the config archive to a different location outside of the SolarWinds installation folder.
   b. Type the template you want to use for naming the configuration files.
      You can use Date and Time, Node, and Global variables (also called macros). You can also use the ${CONFIGTYPE} variable.

7. Click Submit.

Specify a network storage location as the config archive

1. Specify credentials to give NCM write access to the network share:
   a. Click Settings > All Settings.
   b. Under Product Specific Settings, click NCM Settings.
   c. Under Advanced, click Advanced Settings.
   d. Under Network Share Settings, select Use Custom Credentials to Give NCM Writer Access.
   e. Enter the credentials, and click Validate Credentials.
      If the validation fails, verify that the NCM-related account has Windows permissions for the network storage.

2. Click Settings > All Settings.

3. Under Product Specific Settings, click NCM Settings.

4. Under Config Settings, click Config Settings.

5. Under Config Archive, select Save a Copy of Each Config to the Config Archive Directory.

6. If you are low on storage space and only need the last and current configs, select When Configs Are Edited, Only Retain the Last Version.
7. For the Orion server and each additional polling engine:
   
a. Type the path of the network storage for the config archive.

   You can use [Date and Time], [Node], and [Global] variables (also called macros). You can also use the \${CONFIGTYPE} variable.

b. Type the template you want to use for naming the configuration files.

8. Click Submit.

Search network config files or node properties

Use the Advanced Search function to search for a text string within configurations and node properties. Use this functionality to find a specific node or config, or all nodes or configs that share a common characteristic. For example, you can search for all configs that have TACACS security enabled, find an IP address that has been assigned to an interface, or view which configs are managed by a vendor.

Complete the following procedure to search for specific strings of text in the properties of managed nodes or in configuration files stored in the Orion Platform database.

- When a config is initially added to the database, a search might not find it for up to 10 minutes.

1. Click My Dashboards > Network Configuration > Config Summary.

2. Under Search NCM, click Advanced Search.

3. Enter the search string.

   Do not include wildcard characters or regular expressions. You can use policy reports to search configs using regular expressions.

4. Select a search target from the drop-down menu, and then specify which node properties or config files to search:

<table>
<thead>
<tr>
<th>Search Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>All</td>
<td>Searches all properties on all nodes, and the specified config files on all nodes</td>
</tr>
<tr>
<td>Nodes</td>
<td>Searches the specified properties on all nodes</td>
</tr>
<tr>
<td>Configs from All Nodes</td>
<td>Searches the specified configs on all nodes</td>
</tr>
<tr>
<td>Configs from Selected Nodes</td>
<td>Searches the specified config files on the selected nodes</td>
</tr>
</tbody>
</table>
5. Click Search.

6. To search within the returned results, select Search in Results, type a new string pattern, and click Search again.

7. Click a link in the search results to open the node or config details view.

Delete network config files from the database

If you do not need to keep historical configuration files and want to improve database performance, you can automate the removal of unnecessary configuration files. If your database is not stored on a high performance database server, ensure you regularly purge unused config history.

A scheduled job automatically deletes older configs from the database. You can also delete individual jobs manually.

Customize the Default Purge Configs job

NCM includes the Default Purge Configs job, which deletes older configs from the database. By default, this job is enabled and it runs once a week to prevent the database from becoming too large. If you need to make changes to this job, SolarWinds recommends that you create a copy of the default job and change the copy. Changes to the default job cannot be reverted.

1. Click My Dashboards > Network Configuration > Jobs.

2. Select the Default Purge Configs job.

3. Click Duplicate Job.

4. Specify a unique name. For example, include the name of your organization in the job name.

5. Specify when the job runs:

   - To run the job once or on a simple schedule, select Basic. Click the tab that identifies how frequently the job runs, and then specify the start time and (if needed) the day(s).

   - To create a more complex schedule, select Advanced and then use the five fields to create a CRON expression.

6. In the Comments box, include the reason for your changes, and then click Next.

7. Select the nodes to target with this job, and click Next. By default, the job targets all nodes.

8. Specify if you want to save the job log or results.

9. Select an email notification option, and click Next. If you click Email Results, the default email notification and SMTP server settings are populated. These settings can be overridden in each job. By default, no email notification is sent.
10. Select a Purge Configs option, and then click Next:
   - **Purge all configs that were downloaded before this date**
     This option can be used if you run the job manually instead of scheduling it to run regularly.
   - **Delete all configs EXCEPT for the last xx current configs**
     This option is recommended. It sets an absolute limit on the number of configs stored in the database.
   - **Purge all configs except for the last timePeriod**
     This option ensures that all configs from a certain time period are stored in the database.

11. Review the settings for the job, and click Finish.
    The job is enabled by default.

12. To avoid running both your custom job and the default job, disable the default job:
   a. In the Jobs List, select the Default Purge Configs job.
   b. Click Disable.

**Manually delete a config from the database**

1. Click My Dashboards > Network Configuration > Configuration Management.
2. Locate the node, and click the arrow to the left of the node name to display the list of configs from that node.
3. Identify the row that contains the config you want to delete.
4. In the Suggested Action column, click Delete on that row.

**Remove older NCM data from the database and config archive**

NCM includes the Default Database and Archive Maintenance job, which deletes older configs from the config archive on your file system. It also deletes other types of older data from the NCM database. If older, unused data is not removed periodically, the config archive and the database can require large amounts of storage space.

By default, the Default Database and Archive Maintenance job is enabled and it runs once a week. If you need to make changes to this job, SolarWinds recommends that you create a copy of the default job and change the copy. Changes to the default job cannot be reverted.
1. Click My Dashboards > Network Configuration > Jobs.

2. Select the Default Database and Archive Maintenance job.

3. Click Duplicate Job.

4. Specify a unique name. For example, include the name of your organization in the job name.

5. Specify when the job runs:
   - To run the job once or on a simple schedule, select Basic. Click the tab that identifies how frequently the job runs, and then specify the start time and (if needed) the day(s).
   - To create a more complex schedule, select Advanced and then use the five fields to create a CRON expression.

6. In the Comments box, include the reason for your changes, and then click Next.

7. This job does not apply to nodes. Click Next.

8. Specify if you want to save the job log or results.

9. Select an email notification option, and click Next. If you click Email Results, the default email notification and SMTP server settings are populated. These settings can be overridden in each job. By default, no email notification is sent.

10. Select the data you want to purge and the age threshold for each type of data, and then click Next:
   - **Purge configs from the config archive folder**
     Removes older config files from the config archive folder on the NCM server.
   - **Purge config cache data**
     Removes older config cache data from the database. This data is used to display information in certain NCM widgets that show config changes and conflicts. It is also used to generate comparison reports, such as the Baseline vs. Config Conflicts report.

     Purging older, unneeded data from the config cache saves space in the database and can improve the load time of widgets and reports that depend on calculations from cached config data.

     However, consider the type of reporting data your organization requires before you set the age threshold. For example if the age threshold is 2 Weeks, any config that conflicts with the baseline but that was last updated over two weeks ago would not be shown on the Baseline vs. Config Conflicts report.

   - **Purge completed Approval Requests**
Removes data about older completed approval requests from the database.

- **Purge job log**
  Removes the data you see when you click the History icon in the Jobs List to open the Job Report. You can truncate the data in the job log based on age or on the size of the job log.

- **Purge config transfer requests**
  Removes data about older config transfer requests from the database

11. Review the settings for the job, and click Finish.
   The job is enabled by default.

12. To avoid running both your custom job and the default job, disable the default job:
   a. In the Jobs List, select the Default Database and Archive Maintenance job.
   b. Click Disable.

**Limit the number of simultaneous uploads or downloads of network configs**

A large number of config uploads or downloads could overwhelm resources on your NCM server. For example, a large number of downloads can occur when you configure real-time change detection. If you experience issues, you can limit the number of simultaneous uploads or downloads.

1. Click Settings > All Settings.
2. Under Product Specific Settings, click NCM Settings.
3. Under Config Settings, click Config Settings.
4. Adjust the number of Simultaneous Download/Uploads. The default number of concurrent sessions is 25.
5. Click Submit.

**Troubleshoot issues with NCM configuration files**

Use the following topics to resolve issues with downloading and managing config files and viewing config file information.
The Interface Config widget is not displayed for some devices

The Interface Config widget is displayed only when **all** of the following conditions are met:

- The selected interface is on a Cisco or Palo Alto device
- Both NPM and NCM are installed
- The config file for this device has been downloaded to NCM
- The user’s NCM role is not “None”
- Interface data is available

If a device meets all of the criteria but the Interface Config widget still isn’t displayed, it could be because you upgraded from NCM 7.7 or earlier and the device’s configs have not been downloaded since the upgrade. To display interface information, download the config. See this KB article for details.

Confine recognition of the command prompt during download

By default, NCM stops downloading a config file that contains the device command prompt in the body of the config.

**Use the UseExactPromptMatch command to prevent NCM from recognizing the command prompt unless it appears on a separate line:**

1. Click Settings > All Settings.
2. Under Product Specific Settings, click NCM Settings.
4. Select a device template.
5. Click Edit > Using XML Editor.
6. Add this line to the Template XML:
   ```xml
   <Command Name="UseExactPromptMatch" Value="true"/>
   ```
7. Click Save.

Log additional information to help troubleshoot issues

**Enable verbose log information**

Select which log types to keep verbose log information for. This information aids in troubleshooting.

1. Click Settings > All Settings.
2. Under Product Specific Settings, click NCM Settings.
4. Select the log types.
5. Click Submit.

Enable session tracing
Enable session tracing to create a log file of each Telnet session to troubleshoot communication node by node.

1. Click Settings > All Settings.
2. Under Product Specific Settings, click CLI Settings.
3. Select Enable Session Tracing.
Firmware vulnerability data

The following topics provide information about viewing and managing firmware vulnerability data:

- **View firmware vulnerability data**
- **Access NCM firmware vulnerability settings**
- **Test the URLs for downloading firmware vulnerability data**
- **Manually download and add firmware vulnerability files**
- **Ensure HA servers have complete firmware vulnerability data**

NCM 2019.4 with Hotfix 1 (or later) is required to process current firmware vulnerability data from NIST.

**View firmware vulnerability data**

NCM helps identify risks to network security by detecting potential vulnerabilities in the following device types:

- Cisco IOS
- Cisco Adaptive Security Appliance (ASA)
- Cisco Nexus
- Juniper

NCM imports the firmware vulnerability warnings provided by National Institute of Standards and Technology (NIST), and correlates vulnerability data with nodes that it currently manages. If NCM finds a match, the Firmware Vulnerabilities resource on the Config Summary page displays information about the vulnerability and the number of affected nodes.

NCM 2019.4 with Hotfix 1 (or later) is required to process current firmware vulnerability data from NIST.

The firmware vulnerability feature is enabled by default. If necessary, you can enable or disable this feature or change other default settings.

- If you are on a closed network, you can manually import vulnerability data.
- If you have High Availability (HA) backup servers, make sure that your HA servers have complete firmware vulnerability data.
View firmware vulnerability details and update the remediation status

When a firmware vulnerability potentially affects one or more managed nodes, use the Vulnerability Summary page to get additional information and track the remediation status.

1. Click My Dashboards > Network Configuration > Config Summary.

   The Firmware Vulnerabilities widget lists vulnerabilities that could affect nodes managed by NCM.

<table>
<thead>
<tr>
<th>ENTRY ID</th>
<th>CVSS V2 BASE SCORE</th>
<th>SEVERITY</th>
<th>Target Node(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CVE-2018-0228</td>
<td>7.8</td>
<td>High</td>
<td>(2) Nodes</td>
</tr>
<tr>
<td>CVE-2018-0101</td>
<td>10</td>
<td>High</td>
<td>(2) Nodes</td>
</tr>
<tr>
<td>CVE-2018-0092</td>
<td>3.6</td>
<td>Low</td>
<td>(3) Nodes</td>
</tr>
</tbody>
</table>

2. Click a vulnerability’s Entry ID.

   The Vulnerability Summary page displays a summary and the current state. You can click the URL to open the National Vulnerability Database web page for detailed information and links to related advisories and solutions.

3. Optionally, add a comment to record findings, plans, or completed actions.

4. Select the state that reflects the current remediation status:

<table>
<thead>
<tr>
<th>State</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Potential vulnerability</td>
<td>The vulnerability has not yet been verified. (This is the default.)</td>
</tr>
<tr>
<td>Confirmed vulnerability</td>
<td>The vulnerability is confirmed but no remediation is planned.</td>
</tr>
<tr>
<td>Not applicable</td>
<td>The vulnerability does not apply to the selected nodes.</td>
</tr>
<tr>
<td>Remediation planned</td>
<td>Action to remediate the threat is planned but has not been taken.</td>
</tr>
<tr>
<td>Remediated</td>
<td>The vulnerability is confirmed and action to remediate the threat has been taken on the selected nodes.</td>
</tr>
<tr>
<td>Waiver</td>
<td>A waiver has been issued to exempt the selected nodes from remediation.</td>
</tr>
</tbody>
</table>
5. Apply the selected state to all nodes, or select specific nodes.

6. Click Submit.

View firmware vulnerability reports

Firmware vulnerability reports list vulnerabilities discovered in the last run of the vulnerability matching logic. That logic is based on data last downloaded from sources in Firmware Vulnerability Settings.

1. Click Reports > All Reports.

2. In the Group By list, select Report Category.

3. Click the NCM Security category.

4. Click the report name:

   - **Nodes for each Vulnerability** is organized by vulnerability. The associated nodes are listed below each vulnerability.
   - **Vulnerabilities for each Node** is organized by node. The associated vulnerabilities are listed below each node.
   - **Vulnerabilities for each Node - <stageName>** lists only the nodes and associated vulnerabilities in a specific remediation stage (for example, Confirmed or Remediation planned).

Each report includes the following information.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Caption/Entry ID</td>
<td>The Common Vulnerabilities and Exposures (CVE) identifier for a specific vulnerability.</td>
</tr>
<tr>
<td>IOS Version</td>
<td>The operating system software versions to which the CVE pertains.</td>
</tr>
<tr>
<td>IOS Image</td>
<td>The operating system software image to which the CVE pertains.</td>
</tr>
<tr>
<td>URL</td>
<td>The location of the CVE on the NIST website from which NCM obtained vulnerability data.</td>
</tr>
<tr>
<td>CVSS V2 Base Score</td>
<td>A score that reflects the severity of the vulnerability. This score is calculated using the Common Vulnerability Scoring System (CVSS). Use this information to prioritize remediation activities.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Severity</td>
<td>The severity of the vulnerability based on the CVSS score:</td>
</tr>
<tr>
<td></td>
<td>• Low (0-3.9)</td>
</tr>
<tr>
<td></td>
<td>• Medium (4.0-6.9)</td>
</tr>
<tr>
<td></td>
<td>• High (7.0-10.0)</td>
</tr>
<tr>
<td>State</td>
<td>The current status of remediation activities on the associated nodes.</td>
</tr>
<tr>
<td>Last State Change</td>
<td>The date on which the State last changed for the associated nodes.</td>
</tr>
</tbody>
</table>

Troubleshoot firmware vulnerability reports

If a node is not listed with others of its type in a vulnerability announcement, check for errors in the Vulnerability Log (${All Users Profile}\Application Data\SolarWinds\Logs\Orion\NCM\VulnLib.log).

Access NCM firmware vulnerability settings

NCM imports the firmware vulnerability warnings provided by National Institute of Standards and Technology (NIST) and correlates vulnerabilities with managed nodes. Use the firmware vulnerability settings to specify when the matching logic runs or change the default import locations or alert threshold.

1. Click Settings > All Settings.
2. Under Product Specific Settings, click NCM Settings.

If you add or change the URLs under Vulnerability Data Import Settings, you can test the URLs to verify that the correct data will be downloaded.

5. Click Submit.

Test the URLs for downloading firmware vulnerability data

If you add or change the URLs that NCM uses to download firmware vulnerability data from NIST, complete the following steps to verify that you entered the correct URLs.
1. Click Settings > All Settings.
2. Under Product Specific Settings, click NCM Settings.
4. In the Vulnerability Data Import Settings section, copy the URL you want to test.
5. Paste the URL into a browser address bar and click Enter.
6. Verify that a .zip file with the expected file name is downloaded, and that the .zip file contains JSON data.

Manually download and add firmware vulnerability files

NCM ships with initial firmware vulnerability data. However, firmware vulnerability data is routinely updated to reflect new threats. To ensure that you have the latest data, NCM automatically downloads updated files based on the settings you specify.

If NCM cannot automatically download firmware vulnerability data (for example, because you are on a closed network), you can import vulnerability data files from the National Institute of Standards and Technology (NIST) and then manually add them to your NCM server. NCM will use the information in these files to search for vulnerabilities that could affect your devices.

If you have High Availability (HA) backup servers configured, perform this procedure on both your main polling engine and your HA backup servers.

1. Download the following firmware vulnerability .zip files from NIST:
   - https://nvd.nist.gov/feeds/json/cve/1.1/nvdcve-1.1-recent.json.zip
   - https://nvd.nist.gov/feeds/json/cve/1.1/nvdcve-1.1-modified.json.zip
   - https://nvd.nist.gov/feeds/json/cpematch/1.0/nvdcpematch-1.0.json.zip
2. Log in to your NCM server.
3. Verify the location of the vulnerability announcements folder. (The default location is C:\ProgramData\SolarWinds\NCM\Vuln\Json.)
   a. Click Settings > All Settings.
   b. Under Product Specific Settings, click NCM Settings.
   c. Under Advanced, click Firmware Vulnerability Settings.
   d. Under Vulnerability Data Import Settings, verify the location of the folder with vulnerability announcements JSON data.
4. Extract the contents of the following .zip files to the location verified above:
   - nvdcve-1.1-recent.json.zip
   - nvdcve-1.1-modified.json.zip

5. Extract the contents of the nvdcpematch-1.0.json.zip file to the \CpeMatch subdirectory in the location verified above. For example:
   C:\ProgramData\SolarWinds\NCM\Vuln\Json\CpeMatch

6. Under Vulnerability Search Settings, click Run Now.

**Ensure HA servers have complete firmware vulnerability data**

If you have implemented High Availability (HA) backup servers in your SolarWinds deployment, you must make sure that all servers have comprehensive firmware vulnerability data. When a failover occurs, the new server will not have the latest VulnData.sdf file. When the nightly update occurs, by default NCM downloads only recently added or updated vulnerability data, so the data will still be incomplete.

To ensure that your HA backup servers have complete firmware vulnerability data when a failover occurs, SolarWinds recommends that you add the data feeds from this year and last year to your HA backup servers.

1. Find the URL of the NIST vulnerability feeds for this year and last year:
   a. Go to the JSON Vulnerability Feeds page on the NIST site: [https://nvd.nist.gov/vuln/data-feeds#JSON_FEED](https://nvd.nist.gov/vuln/data-feeds#JSON_FEED)
   b. In the row for the current year’s CVE file, hover over the ZIP file link in the NVD JSON 1.1
Schema column.

c. Right-click the link and choose Copy Link Address.

2. Add the link to the NCM Firmware Vulnerability Settings:
   a. Click Settings > All Settings.
   b. Under Product Specific Settings, click NCM Settings.
   c. Under Advanced, click Firmware Vulnerability Settings.
   d. In the Vulnerability Data Import Settings section, click Add New.
   e. Paste the link address that you copied in the previous step.

3. Repeat these steps to add the feed for the previous year.
4. **Test the URLs** you added.

5. Click Submit to save the updated settings.
Firmware upgrades

Firmware upgrades deliver the latest bug fixes and protect your devices from firmware vulnerabilities. When you perform a firmware upgrade operation, NCM upgrades the firmware on multiple devices. The devices targeted by each operation are upgraded sequentially, but you can run multiple firmware upgrade operations simultaneously. To prepare for and perform firmware upgrade operations, see the following topics:

- Set up the storage location and firmware repository
- Simultaneous firmware upgrade operations
- Perform a firmware upgrade operation

Firmware upgrade templates define the commands that are executed when a user performs a firmware upgrade operation. NCM provides a set of default firmware upgrade templates, and you can create new templates to enable firmware upgrades on other device types. You can also share firmware upgrade templates through the SolarWinds community website, THWACK. See the following topics for more information:

- Manage firmware upgrade templates
- Import and export firmware upgrade templates

Set up the storage location and firmware repository

You can use NCM to upgrade the firmware on many devices in one operation. Because security fixes are distributed as new firmware revisions, upgrading firmware helps network administrators correct security vulnerabilities.

Before you perform a firmware upgrade operation, you must complete the initial setup. To do this, complete the following tasks as described below:

- Task 1: Designate a storage location to contain the firmware images. This feeds the firmware repository in NCM, from which you select images for devices when upgrading.

- Task 2: Populate the storage location with firmware images through the file system. When it is populated, you can manage firmware upgrade images in NCM with the Firmware Repository.

**i** NCM currently supports only .bin files for firmware upgrades.
Task 1: Designate a storage location to contain the firmware images

1. Click Settings > All Settings.
2. Under Product Specific Settings, click NCM Settings.
3. In the Firmware Upgrade section, click Upgrade Settings.
4. Enter the path to the firmware upgrade storage location.
   - For basic installations with one polling engine, you can use a local path on the server where NCM is installed.
   - The storage location cannot be the same as the TFTP or SFTP/SCP root folders.
   - If you have more than one polling engine, use a shared network location. Make sure all polling engines have access to this location, and they are in the same domain.
5. Enter credentials to access the storage location, and click Submit.

Task 2: Populate the storage location with firmware images

Next, populate the storage location with firmware images through the file system. Now you can manage firmware upgrade images in NCM with the Firmware Repository:

1. Click Settings > All Settings.
2. Under Product Specific Settings, click NCM Settings.
3. In the Firmware Upgrade Storage section, click Firmware Repository.
4. Click Scan Firmware Repository to see the latest image files. By default, all files with the .bin extension are displayed.

⚠️ If you select one or more entries and click Delete Firmware Images, the files are deleted from the folder you set as the storage location.

Integrity of firmware images

You can use the MD5 Hash column in the firmware repository to verify the integrity of your images. This value is calculated automatically when you add an image to the storage location. Compare the image hashes between the firmware repository and the website of the manufacturer of your device.

By default, NCM verifies the integrity of backed up and uploaded firmware images. You can toggle these settings under Upgrade Options in the Firmware Upgrade Operation wizard.
Simultaneous firmware upgrade operations

When a firmware upgrade operation runs, the nodes targeted by that operation are upgraded one at a time, sequentially. But with NCM 2020.2 and later, you can run multiple firmware operations at the same time. To speed up the process of applying a firmware upgrade to a large number of nodes, you can create multiple firmware upgrade operations that each target a unique set of nodes, and then schedule those operations to run simultaneously.

Requirements

Firmware upgrade operations can run simultaneously if the following conditions are met:

- The firmware upgrade templates used in the operations must enable simultaneous firmware upgrade operations.

  All firmware upgrade templates shipped with NCM except the EtherWAN template enable simultaneous firmware upgrade operations.

  Custom firmware upgrade templates enable simultaneous operations if they include the \${SubFolder} macro in the following commands:

  - Upgrade firmware image command
  - Back up firmware image command (if included)

  If you attempt to run firmware operations simultaneously but the associated templates do not enable it, the firmware upgrade operations run sequentially.

- The operations must target different nodes.

  If NCM detects that simultaneous operations are attempting to upgrade one or more of the same nodes, the operation that started first continues to run. NCM stops all other operations and displays an error message. If this occurs:

    1. Click the ⬇️ icon in the History column to open the Firmware Upgrade Report for the failed operation. Use this report to identify the nodes that are included in multiple operations.

    2. Re-execute each operation and remove the redundant nodes.

- The number of firmware upgrade operations running simultaneously cannot exceed the maximum number specified in Upgrade Settings.

  If you attempt to run more than the maximum number of operations, NCM runs only the maximum number of operations. Other operations are queued to start after the first operations finish.
Specify the maximum number of firmware upgrade operations

By default, a maximum of five firmware upgrade operations can run simultaneously. You can increase or decrease this value based on your needs and the capacity of your server.

Running a large number of firmware upgrade operations simultaneously can strain server resources.

If you increase this value, make sure that the SCP or TFTP root folder has enough free space to temporarily store that many copies of the firmware upgrade image. For example, if the firmware upgrade image is 500 MB and you schedule 10 operations to run simultaneously, the SCP or TFTP root folder must have 5,000 MB of free space.

1. Click Settings > All Settings.
2. Under Product Specific Settings, click NCM Settings.
3. In the Firmware Upgrade section, click Upgrade Settings.
4. Under Upgrade Settings, change the value of the Maximum number of simultaneous upgrade operations setting.

Perform a firmware upgrade operation

After you have set up your storage location and firmware repository, and reviewed or defined firmware upgrade templates, you can perform an upgrade operation.

With NCM 2020.2 and later, you can run multiple firmware upgrade operations simultaneously.

Add a firmware upgrade operation

Create a firmware upgrade operation to upgrade the firmware on many devices. The devices targeted by a firmware upgrade operation are upgraded one at a time in sequential order, which you can customize.

You must have the NCM WebUploader role (or higher) to add and execute a firmware upgrade operation.

1. Click My Dashboards > Network Configuration > Firmware Upgrades.
2. Click Add.
3. Enter an operation name, and select a firmware upgrade template. Click Next.
4. On the Select Firmware Image panel, specify which firmware image this operation will apply to the selected nodes:
   - If a firmware image is preselected by the upgrade template, it is listed under Select Firmware Image. Click Next to use the preselected image.
   - To select an image, click Select New Image From Repository. This option overrides any existing selection.

5. Select the nodes you want to upgrade, and click Start Collecting Data.
   The data collection may take a few moments, so you are returned to the Firmware Upgrade Operations view.

6. When the data collection finishes, click Please Review to Continue in the Status/Details column of the operation.
   The wizard resumes at the Upgrade Options step.

7. In the Selected Nodes list, click each node and review the image path and upgrade options for that node. Make changes if needed. See the Upgrade Options section for more information.

8. Select all the nodes in the list and click Confirm to verify that you have confirmed the selections for all nodes.
   If required information is missing for any node, a warning icon ▲ is displayed next to the node name. Click the node and complete the missing information. Then click Confirm again.

   ⚠️ To remove a node from the firmware upgrade operation, click the remove icon and click Yes at the confirmation prompt.

   NCM displays a warning if there is not enough free space to continue the operation without removing the current firmware image. These upgrade options are automatically selected:
   - Back up existing firmware image
   - Delete existing firmware image

9. When all nodes are confirmed, click Next.

10. Specify whether you want the results emailed, and click Next.

11. On the Summary panel, you review the order in which devices will be upgraded. You can use the arrows on the right to change the order.

12. Under Schedule Options, select Run Immediately or schedule the operation for later.
13. Type **YES** to continue, and click Finish.

You are returned to the Firmware Upgrade Operations view. You can check the progress of the operation in the Status/Details column.

**Upgrade Options**

**Determine image paths**

- **Path to upload image**: A required field under the selected firmware image. This is the path NCM uses to upload the new firmware image. NCM attempts to detect the existing image path and populate this field with the same drive where the existing image is located. For example, `flash:`.

- **Back up existing firmware image**: An option under Upgrade Options with a required field to enter the image path. NCM attempts to automatically determine the path to the existing firmware image on the device. For example, `flash:c2951-universalk9-mz.SPA.155-3.M1.bin`.

If one or both of these fields are empty, refer to the documentation of your device for more information. You can also try to determine the path from the output of the `show version` command.

**Config comparison**

Select the following options to list a config comparison report in the log, or in an email notification if enabled, after the firmware upgrade operation is complete:

- Back up running and startup configs before upgrade
- Back up running and startup configs after upgrade

**Rebooting a device after the upgrade**

Select the **Reboot device after upgrade** option to execute the Reboot command after the upgrade operation.

**Operation statuses for firmware upgrades**

Check the status of an operation in the Status/Details column of the Firmware Upgrades Operations view.

<table>
<thead>
<tr>
<th>Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Collecting data</td>
<td>NCM is executing the collect info commands for the upgrade operations.</td>
</tr>
<tr>
<td>Need review</td>
<td>All collect info commands have executed, and the operation is waiting for user review and confirmation of the upgrade options for each node. If necessary, you can remove any node from the upgrade operation.</td>
</tr>
<tr>
<td>Status</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Queued</td>
<td>Another firmware upgrade operation is already in the &quot;Upgrading&quot; status, and NCM is waiting for it to finish before starting this operation.</td>
</tr>
<tr>
<td>Upgrading</td>
<td>NCM is currently executing the firmware upgrade scripts.</td>
</tr>
<tr>
<td>Complete</td>
<td>The firmware upgrade operation finished successfully.</td>
</tr>
<tr>
<td>Error</td>
<td>The firmware upgrade operation failed.</td>
</tr>
<tr>
<td></td>
<td>See Additional actions for firmware upgrade operations to review your options.</td>
</tr>
</tbody>
</table>

Additional actions for firmware upgrade operations

Based on the operation status of a firmware upgrade, you may need to cancel, rollback, or re-execute the operation.

Cancel a firmware upgrade operation

If you made a mistake when configuring an operation and it currently has the "Upgrading" status, you can cancel the active firmware upgrade operations.

⚠️ If you cancel a currently running operation, you do not have the option to rollback.

1. Click My Dashboards > Network Configuration > Firmware Upgrades.  
2. Select one or more entries, and click Cancel Selected.

Roll back a firmware upgrade operation

If there is a problem with the new firmware, you can restore each node in a firmware upgrade operation to its previous image, boot register, and configuration.

1. Click My Dashboards > Network Configuration > Firmware Upgrades.  
2. Select one or more entries, and click Rollback.

Re-execute a firmware upgrade operation

If some of the nodes in a firmware upgrade operation were not upgraded successfully, you can re-execute the operation on only the failed nodes.

1. Click My Dashboards > Network Configuration > Firmware Upgrades.  
2. Select one or more entries, and click Re-execute.
Review a firmware upgrade operation

If you started but did not complete or execute a firmware upgrade operation, you can review it and continue.

1. Click My Dashboards > Network Configuration > Firmware Upgrades.
2. Select an entry, and click Review.

You are returned to the Upgrade wizard.

Manage NCM firmware upgrade templates

Each firmware upgrade template defines a set of device-specific commands and options that NCM uses to upgrade the firmware on a device of that type. NCM provides a set of default firmware upgrade templates, and you can create new templates to enable firmware upgrades on other device types.

NCM default firmware upgrade templates

SolarWinds NCM provides firmware upgrade templates for the following device series:

- Cisco ASA 5512 Multi Context Mode
  Upgrades Cisco Adaptive Security Appliance (ASA) devices that have been partitioned into multiple contexts.

  ⚠ To monitor a Cisco ASA device with multiple contexts, you must add each context as a node to be monitored by NCM. To do this, add the admin context by IP address. All other contexts are automatically discovered, and you can add each one as a node.

- Cisco ASA 5512 Single Context Mode
- Cisco Catalyst 2960
- Cisco Catalyst 3560
- Cisco Catalyst 3750
- Cisco Catalyst 3850
- Cisco IOS Software 2800/2801
- Cisco IOS Software 2921
- Cisco Nexus 5548
This template performs an in-service software upgrade (ISSU) and then reboots the device. Rebooting the device ensures that the firmware upgrade is completed successfully. Use this template only for upgrades. The ISSU process does not support downgrades.

### Important Note

If a Cisco Nexus device has multiple virtual device contexts (VDCs), firmware updates apply to all VDCs on the physical device. You cannot upgrade the firmware for an individual VDC.

- **Cisco Nexus C7004 - ISSU**
  
  This template performs an ISSU and then reboots the device. Rebooting the device ensures that the firmware upgrade is completed successfully. Use this template only for upgrades. The ISSU process does not support downgrades.

- **Cisco Nexus C7004 - Traditional**
  
  This template performs a traditional upgrade or downgrade. Use this template when ISSU is not supported.

- **EtherWAN**

- **Juniper j2320 Junos**

- **Lenovo Campus NOS**

- **Lenovo RackSwitch CNOS version 10.7 and newer**

- **Lenovo RackSwitch CNOS versions before 10.7**

### Add a firmware upgrade template

1. Click Settings > All Settings.
2. Under Product Specific Settings, click NCM Settings.
3. In the Firmware Upgrade section, click Firmware Upgrade Templates.
4. Perform one of the following actions:
   - To create a template based on a predefined NCM template, select the template and click Edit. Then click Duplicate & Edit Firmware Upgrade Template.
   - To create a new template from scratch, click Add.
5. Enter a name and description.
6. In the Collect Info Commands section, specify the actions that NCM should perform to gather information from devices before they are upgraded with this template. To enable an action, select the check box and then enter the command to perform that action.
For information about how NCM uses the information returned by these commands, see Collect Info commands.

- **Get current firmware image command** (optional):
  a. In the command box, enter the command to retrieve the current firmware image path for the device.

    For example, the command for Cisco IOS devices is:
    
    ```
    show ver | include system image
    ```

  b. To identify what part of the returned value is the image path that NCM should capture, select Use capture results pattern. Then enter the string that the command returns with `${CaptureData}` as a placeholder for the image path.

    For example, if the command returns System image file is "imagePath", enter:

    ```
    System image file is "${CaptureData}"
    ```

    For devices with multiple images (such as Cisco Nexus devices, which have a system image and a kickstart image), enter multiple patterns separated by the `$;` macro. For example:

    ```
    system image file is "${CaptureData}"$;kickstart image file is:"${CaptureData}"
    ```

- **Get info on config register command** (optional):
  a. In the command box, enter the command to retrieve the current configuration register value.

  b. To identify what part of the returned value is the configuration register value that NCM should capture, select Use capture results pattern. Then enter the string that the command returns with `${CaptureData}` as a placeholder for the configuration register value.

  c. Enter the expected configuration register value in the Expected value field.
Different device models might require different config register values to allow the device to boot correctly after the upgrade. For example, the appropriate config register value for Cisco IOS is 0x2102. For Cisco ASA, it is 0x1. This command is not applicable for Cisco Catalyst and Cisco Nexus devices.

- **Collect Boot Variable Info** (required for Cisco ASA devices, not used for other devices)
  
  For Cisco ASA devices, enter the command to collect all existing boot paths.

- **Get free space command** (optional)
  
  If you want NCM to verify that sufficient space is available to perform the upgrade operation, enter the command to check the amount of free space on the device.

- **Other commands** (optional)
  
  Enter commands to collect any additional information to be displayed on the Upgrade Options panel during a firmware upgrade operation.

7. In the Upgrade Commands section, specify the firmware image type:

- For devices with only one image:
  a. Enter any description (for example, "Main") in the Firmware Image Type box.
  b. Click Add New.
For devices that have **multiple** images, such as Cisco Nexus devices:

a. Enter the first image type (for example, "System" or "Kickstart") in the Firmware Image Type box.

b. Click Add New.

c. Repeat to add other images.

8. (Optional) To associate this template with a specific firmware image, click Select New Image From Repository and then select the image.

   ![Image](image.png)

   If you do not associate this template with a firmware image, users can select one later when they **perform a firmware upgrade operation**.

9. Enter the Upgrade firmware image command. See **Upgrade firmware image command** below for examples and information about the macros that can be used with this command.

   ![Command](command.png)

10. Select the transfer protocol.

11. (Optional) To allow users to select additional actions to perform during the firmware upgrade operation, specify the actions in the Upgrade Commands section. To enable an action, select the check box and then enter the command to perform that action.

   ![Command](command.png)

   **See Upgrade Commands** for more information about the commands and the macros available for use with each command. SolarWinds recommends always including the **Update boot variable** command.

12. In the Upgrade Options section, select the tasks to run before and after the upgrade with this template. These options are selected by default when a user performs an upgrade operation.

   If you select the **Reboot device after upgrade option**, the following setting is available:

   Wait time between reboot and checking that the node is "Up"

   Use this setting to prevent errors for devices that have multiple up/down sequences after they reboot.

13. Click Save.
Collect Info commands

When a user performs a firmware upgrade operation with the template, NCM first runs the commands selected in the Collect Info section. These commands collect information from the devices, which the user can review before continuing with the upgrade.

Get current firmware image command

This command is optional. If it is included, NCM runs this command to determine which image the device currently uses. The capture results pattern enables NCM to capture the image path and file name from the command output.

NCM uses this information in the following ways:

- NCM automatically enters the image path and file name when the user selects the following Upgrade Options:
  - Backup existing firmware image
  - Delete existing firmware image

- NCM automatically enters the drive where the existing image is located in the Path to upload image field.

- If NCM determines that the device does not have enough free space to upload the new image, NCM uses this information to determine if deleting the existing image will free enough disk space to proceed.

If the template does not specify this command:

- To back up or delete the existing image, the user must manually specify the path on the Upgrade Options panel.
- If the device does not have enough free space, NCM will not be able to determine if deleting the existing image will free enough disk space to proceed.
Get info on config register

This command is **optional**. If it is included, NCM uses this information to determine if the configuration register needs to be updated. NCM retrieves the current configuration register value and compares it to the expected value. If the values do not match, NCM automatically selects the Update config register option on the Upgrade Options panel.

![Update config register]

If the template does **not** specify this command, the Update Config Register option is deselected by default and users must manually select this option if necessary.

Collect Boot Variable Info

This command is **required** for Cisco ASA devices. It is **not used** for other devices. If this command is not specified for a Cisco ASA device, the firmware upgrade operation cannot be performed.

When a Cisco ASA device updates the boot variable (which tells the device what image it should boot from after a reboot), it must remove all existing boot paths. This command collects all existing boot paths and stores them in the `${BootItem}` macro. This value is used by the Update boot variable command.

Get free space command

This command is **optional**. Include this command if you want NCM to check the amount of free space on the device. When you include this command:

- NCM displays the output of the "Get free space" command on the Upgrade Options page. This informs users of the amount of available space.
- NCM runs its own Free Space Detector over SNMP to verify that sufficient space is available to perform the operation.

The output of the "Get free space" command is not used to verify that sufficient space is available. However, including this command causes NPM to perform the SNMP free space verification.

Other commands

Other commands are **optional**. You can enter commands to collect additional information. This information is displayed on the Upgrade Options panel to help users determine whether to proceed with the firmware upgrade operation.

> On a Cisco Nexus device, you cannot include pre-upgrade check commands that require the new image to already be uploaded on the device. For example, you cannot include commands to verify compatibility (show incompatibility) or to identify the upgrade impact (show install all impact).
Upgrade commands

Upgrade firmware image command

This command is required. This command copies the firmware image from the TFTP or SCP server to the device file system.

The following example is a command for non-Nexus Cisco devices. The \${SubFolder} macro enables simultaneous firmware upgrade operations.

```
copy ${TransferProtocol}://${StorageAddress}/${SubFolder}/${NewImageName}
${NewImageSlot}${CRLF}dir ${NewImageSlot} ${SuccessRegEx:${NewImageName}}
```

The following example is a command for Nexus Cisco devices. The \${SubFolder} macro enables simultaneous firmware upgrade operations. The \${System.FileName} macro specifies the type of image.

```
copy ${TransferProtocol}://${StorageAddress}/${SubFolder}/${System.FileName}
${NewImageSlot} vrf management${CRLF}${CRLF}dir ${NewImageSlot} ${SuccessRegEx:${System.FileName}}
copy ${TransferProtocol}://${StorageAddress}/${SubFolder}/${Kickstart.FileName}
${NewImageSlot} vrf management${CRLF}${CRLF}dir ${NewImageSlot} ${SuccessRegEx:${Kickstart.FileName}}
```

The following macros can be used with this command.

<table>
<thead>
<tr>
<th>Macro</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>${TransferProtocol}</td>
<td>This macro resolves to the value selected for Transfer protocol. You can enter the value (TFTP or SCP) instead of this macro.</td>
</tr>
<tr>
<td>${StorageAddress}</td>
<td>(Applies only if you use TFTP to transfer the firmware image.) This macro resolves to the IP address specified in TFTP Server settings. You can enter the server IP address instead of this macro.</td>
</tr>
<tr>
<td>${SCPStorageAddress}</td>
<td>(Applies only if you use SCP to transfer the firmware image.) This macro resolves to the IP address specified in SCP Server settings. You can enter the server IP address instead of this macro.</td>
</tr>
<tr>
<td>${SCPServerUserName}</td>
<td>(Applies only if you use SCP to transfer the firmware image.) This macro resolves to the user name specified in SCP Server settings. You can enter the user name instead of this macro.</td>
</tr>
<tr>
<td>${SCPServerPassword}</td>
<td>(Applies only if you use SCP to transfer the firmware image.) This macro resolves to the password for the SCP user specified in SCP Server settings. You can enter the password instead of this macro.</td>
</tr>
<tr>
<td>Macro</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>${SubFolder}</td>
<td>This macro is required to enable firmware upgrade operations that use this template to run simultaneously with other firmware upgrade operations. The macro resolves to the name of a subfolder that is automatically created on the SCP or TFTP server. To enable multiple firmware operations to run simultaneously, NCM copies the files for each operation to a uniquely named subfolder instead of to the root directory.</td>
</tr>
<tr>
<td></td>
<td>You must use this macro to specify a subfolder. Do not enter a subfolder name.</td>
</tr>
<tr>
<td>${NewImageName}</td>
<td>(Required for non-Nexus devices.) This macro resolves to the name of the firmware image that is selected in the Firmware image(s) field during an upgrade operation.</td>
</tr>
<tr>
<td>${System.FileName}</td>
<td>(Required for Nexus devices.) This macro resolves to the name of the system image that is selected in the Firmware image(s) field during an upgrade operation.</td>
</tr>
<tr>
<td>${KickStart.FileName}</td>
<td>(Required for Nexus devices.) This macro resolves to the name of the kickstart image that is selected in the Firmware image(s) field during an upgrade operation.</td>
</tr>
<tr>
<td>${NewImageSlot}</td>
<td>(Required for all devices.) This macro resolves to the value entered in the Path to upload image field during an upgrade operation (for example, flash: or disk0:).</td>
</tr>
<tr>
<td>${CRLF}</td>
<td>Use this macro to send the Enter command when a device asks confirmation questions. Also, when more than one command must be executed as part of an operation, use this macro to separate the commands.</td>
</tr>
</tbody>
</table>
SolarWinds recommends adding commands to make sure the operation completed successfully. For instance, the non-Nexus example above includes the following commands to ensure that the new image file is on the device:

```
dir ${NewImageSlot} ${SuccessRegEx:${NewImageName}}
```

The `dir` command lists the files in the directory where the image was uploaded. The `${SuccessRegEx:${NewImageName}}` command searches the output of the `dir` command for the name of file defined in the `${NewImageName}` macro. If the file name is not found, the upgrade operation stops and displays an error.

The command for Nexus devices includes similar constructions using the `${System.FileName}` and `${Kickstart.FileName}` macros:

```
dir ${NewImageSlot} ${SuccessRegEx:${System.FileName}}
dir ${NewImageSlot} ${SuccessRegEx:${Kickstart.FileName}}
```

Delete firmware image command

This command is optional. If it is included, users performing an upgrade operation can select the Delete existing firmware image option, which deletes the existing image. An example of this command is:

```
delete ${ExistingImagePath}${CRLF}${CRLF}${CRLF}
```

The following macros can be used with this command.

<table>
<thead>
<tr>
<th>Macro</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>${ExistingImagePath}</code></td>
<td>(Required for all devices.) This macro resolves to the value entered in the Image path field for the Delete existing firmware image option during an upgrade operation.</td>
</tr>
<tr>
<td><code>${CRLF}</code></td>
<td>Use this macro to send the Enter command when a device asks confirmation questions. Also, when more than one command must be executed as part of an operation, use this macro to separate the commands.</td>
</tr>
</tbody>
</table>

Back up firmware image command

This command is optional. If it is included, users performing an upgrade operation can select the Delete existing firmware image option, which backs up the existing image into NCM firmware upgrade storage. An example of this command is:

```
copy ${ExistingImagePath} ${TransferProtocol}:/${StorageAddress}/${SubFolder}/vrf management${CRLF}${CRLF}${CRLF}
```
The following macros can be used with this command.

<table>
<thead>
<tr>
<th>Macro</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>${ExistingImagePath}</code></td>
<td><em>(Required for all devices.)</em> This macro resolves to the value entered in the Image path field for the Back up existing firmware image option during an upgrade operation.</td>
</tr>
<tr>
<td><code>${TransferProtocol}</code></td>
<td>This macro resolves to the value selected for Transfer protocol. You can enter the value <em>(TFTP or SCP)</em> instead of this macro.</td>
</tr>
<tr>
<td><code>${StorageAddress}</code></td>
<td><em>(Applies only if you use TFTP to transfer the firmware image.)</em> This macro resolves to the IP address specified in TFTP Server settings. You can enter the server IP address instead of this macro.</td>
</tr>
<tr>
<td><code>${SCPStorageAddress}</code></td>
<td><em>(Applies only if you use SCP to transfer the firmware image.)</em> This macro resolves to the IP address specified in SCP Server settings. You can enter the server IP address instead of this macro.</td>
</tr>
<tr>
<td><code>${SCPServerUserName}</code></td>
<td><em>(Applies only if you use SCP to transfer the firmware image.)</em> This macro resolves to the user name specified in SCP Server settings. You can enter the user name instead of this macro.</td>
</tr>
<tr>
<td><code>${SCPServerPassword}</code></td>
<td><em>(Applies only if you use SCP to transfer the firmware image.)</em> This macro resolves to the password for the SCP user specified in SCP Server settings. You can enter the password instead of this macro.</td>
</tr>
<tr>
<td><code>${SubFolder}</code></td>
<td>This macro is required to enable firmware upgrade operations that use this template to run simultaneously with other firmware upgrade operations. The macro resolves to the name of a subfolder that is automatically created on the SCP or TFTP server. To enable multiple firmware operations to run simultaneously, NCM copies the files for each operation to a uniquely named subfolder instead of to the root directory.</td>
</tr>
<tr>
<td><code>{CRLF}</code></td>
<td>Use this macro to send the Enter command when a device asks confirmation questions. Also, when more than one command must be executed as part of an operation, use this macro to separate the commands.</td>
</tr>
</tbody>
</table>

You must use this macro to specify a subfolder. Do not enter a subfolder name.
Update config register command

This command is optional. If it is included, users performing an upgrade operation to select the Update config register option. An example of this command is:

```
config terminal{CRLF}config-register 0x2102{CRLF}end{CRLF}write
memory{CRLF}{CRLF}{CRLF}
```

Update boot variable command

This command is optional, but SolarWinds recommends including it. If it is included, users performing an upgrade operation can select the Update boot variable option, which tells the device what image it should boot from after a reboot. If the boot variable is not updated, some devices might attempt to boot from the first image found on the device file system, and some devices might not boot at all.

An example of this command for non-ASA devices is:

```
config terminal{CRLF}no boot system{CRLF}boot system
${NewImageSlot}${NewImageName}${CRLF}end{CRLF}write
memory{CRLF}{CRLF}{CRLF}show startup | include boot
${SuccessRegEx:${NewImageName}}
```

An example of this command for Cisco ASA devices is:

```
config terminal[Repeat:no ${BootItem}]boot system
${NewImageSlot}${NewImageName}${CRLF}end{CRLF}write
memory{CRLF}{CRLF}{CRLF}show boot ${SuccessRegEx:${NewImageName}}
```

The following macros are used with this command.

<table>
<thead>
<tr>
<th>Macro</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>${NewImageName}</td>
<td><strong>(Required for all devices.)</strong> This macro resolves to the name of the image name that is selected for the Update boot variable option during an upgrade operation.</td>
</tr>
<tr>
<td>${BootItem}</td>
<td><strong>(Required for Cisco ASA devices.)</strong> For Cisco ASA devices, a separate no command must be sent for each boot path. The Collect Boot Variable Info command (issued during the information collection phase) saves all boot paths to the ${BootItem} macro, which is used in this command. The [Repeat:no ${BootItem}] construction repeats the command as many times as the number of values collected in the ${BootItem} macro. The Repeat statement works only with the ${BootItem} macro.</td>
</tr>
<tr>
<td>${NewImageSlot}</td>
<td><strong>(Required for all devices.)</strong> This macro resolves to the value entered in the Path to upload image field during an upgrade operation (for example, flash: or disk0:).</td>
</tr>
<tr>
<td>Macro</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>${CRLF}</td>
<td>Use this macro to send the Enter command when a device asks confirmation questions. Also, when more than one command must be executed as part of an operation, use this macro to separate the commands.</td>
</tr>
</tbody>
</table>

SolarWinds recommends including validation checks to ensure the boot variable is updated in the device startup config because the startup config is used after a reboot. The previous examples include the following additional command to accomplish this task:

```
show startup | include boot ${SuccessRegEx:${NewImageName}}
```

This command searches the startup config boot variable line. The ${SuccessRegEx:${NewImageName}} macro ensures that the output contains the new image name.

**Reboot command**

This command is **optional**. If included, users performing an upgrade operation can select the Reboot device after upgrade option. An example of this command is:

```
reload${CRLF}y${CRLF}y
```

**Verify uploaded firmware image integrity command**

This command is **optional**. If it is included, users performing an upgrade operation can select the Verify uploaded firmware image integrity option. If that option is selected, NCM performs the following actions to verify that the image was transferred to the device correctly and not corrupted:

1. NCM executes the specified command to calculate the MD5 hash of the image on the NCM server.
2. NCM executes the same command to calculate the MD5 hash of the image that was transferred to the device.
3. NCM compares the values to confirm that they are the same.

For non-Nexus devices, the following example verifies the hash on a single image:

```
verify /md5 ${NewImageSlot}${NewImageName} ${NewImageHash} ${SuccessRegEx:Verified}
```

For Cisco Nexus devices, the following example verifies the hash on the system image and the kickstarter image:

```
show file ${NewImageSlot}${System.FileName} md5sum ${SuccessRegEx:${System.Hash}}${CRLF}show file ${NewImageSlot}${Kickstart.FileName} md5sum ${SuccessRegEx:${Kickstart.Hash}}
```
The following macros are used with this command.

<table>
<thead>
<tr>
<th>Macro</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>${NewImageSlot}</td>
<td><em>(Required for all devices.)</em> This macro resolves to the value entered in the Path to upload image field during an upgrade operation <em>(for example, flash: or disk0:).</em></td>
</tr>
<tr>
<td>${NewImageName}</td>
<td><em>(Required for non-Nexus devices.)</em> This macro resolves to the name of the firmware image that is selected in the Firmware image(s) field during an upgrade operation.</td>
</tr>
<tr>
<td>${NewImageHash}</td>
<td><em>(Required for non-Nexus devices.)</em> This macro resolves to the MD5 hash of the image file on the NCM server.</td>
</tr>
<tr>
<td>${System.FileName}</td>
<td><em>(Required for Nexus devices.)</em> This macro resolves to the name of the system image that is selected in the Firmware image(s) field during an upgrade operation.</td>
</tr>
<tr>
<td>${System.Hash}</td>
<td><em>(Required for Nexus devices.)</em> This macro resolves to the MD5 hash of the system image file on the NCM server.</td>
</tr>
<tr>
<td>${Kickstart.FileName}</td>
<td><em>(Required for Nexus devices.)</em> This macro resolves to the name of the kickstart image that is selected in the Firmware image(s) field during an upgrade operation.</td>
</tr>
<tr>
<td>${Kickstart.Hash}</td>
<td><em>(Required for Nexus devices.)</em> This macro resolves to the MD5 hash of the kickstart image file on the NCM server.</td>
</tr>
<tr>
<td>${CRLF}</td>
<td>Use this macro to send the Enter command when a device asks confirmation questions. Also, when more than one command must be executed as part of an operation, use this macro to separate the commands.</td>
</tr>
<tr>
<td>${SuccessRegEx:Verified}</td>
<td><em>(Required to perform verification for non-Nexus devices.)</em> This macro determines if the command output (the hash value of the image uploaded to the device) matches the value of ${NewImageHash} (the hash value of the image on the server).</td>
</tr>
<tr>
<td>${SuccessRegEx:${System.Hash}}</td>
<td><em>(Required to perform verification for Nexus devices.)</em> This macro determines if the command output (the hash value of the system image uploaded to the device) matches the value of ${System.Hash} (the hash value of the system image on the server).</td>
</tr>
</tbody>
</table>
Macro | Description
--- | ---
${SuccessRegEx:${Kickstart.Hash}} | **Required** to perform verification for Nexus devices. This macro determines if the command output (the hash value of the kickstart image uploaded to the device) matches the value of ${Kickstart.Hash} (the hash value of the kickstart image on the server).

### Verify backed up firmware image integrity command

This command is **optional**. If it is included, users performing an upgrade operation can select the Verify backed up firmware image integrity option. If that option is selected, NCM performs the following actions to verify that the image was backed up to the NCM server correctly and not corrupted:

1. NCM executes the specified command to calculate the MD5 hash of the image that was backed up to the NCM server.
2. NCM executes the same command to calculate the MD5 hash of the image that was transferred to the device.
3. NCM compares the values to confirm that they are the same.

An example of this command is:

```
verify /md5 ${ExistingImagePath} ${BackupImageHash} ${SuccessRegEx:Verified}
```

The following macros are used with this command.

Macro | Description
--- | ---
${ExistingImagePath} | **Required** for all devices.) This macro resolves to the value entered in the Image path field for the Back up existing firmware image option during an upgrade operation.
${BackupImageHash} | **Required** for all devices.) This macro resolves to the MD5 hash of the backed up image file on the NCM server.
${SuccessRegEx:Verified} | **Required** to perform the verification.) This macro determines if the command output (the hash value of the image uploaded to the device) matches the value of ${BackupImageHash} (the hash value of the backed up image on the server).
Import and export firmware upgrade templates

If you have multiple Orion servers, you can share firmware upgrade templates between them. You can also share firmware upgrade templates on THWACK.

Share firmware upgrade templates between Orion servers

You can export a firmware upgrade template as an XML file, and then import it to another Orion server.

Export a firmware upgrade template

1. Click Settings > All Settings.
2. Under Product Specific Settings, click NCM Settings.
3. In the Firmware Upgrade section, click Firmware Upgrade Templates.
4. Select a template, and click Export As File.
   
   The XML file is saved in your \Downloads directory. The file name is based on the template name.

Import a firmware upgrade template

You can import a firmware upgrade template that was previously exported as a file.

1. Click Settings > All Settings.
2. Under Product Specific Settings, click NCM Settings.
3. In the Firmware Upgrade section, click Firmware Upgrade Templates.
4. Click Import.
5. Click Choose File, browse to the template, and select it.
6. Click Submit.

Share firmware upgrade templates with the THWACK community

You can share your firmware upgrade templates with other NCM users by exporting them to THWACK, and you can import firmware upgrade templates created by other NCM users.

Export a firmware upgrade template to THWACK

1. Click Settings > All Settings.
2. Under Product Specific Settings, click NCM Settings.
3. In the Firmware Upgrade section, click Firmware Upgrade Templates.

4. Select a template, and click Export to THWACK.

5. If prompted, enter your THWACK user name and password, and click Log In.

Import a firmware upgrade template from THWACK

1. Click Settings > All Settings.

2. Under Product Specific Settings, click NCM Settings.

3. In the Firmware Upgrade section, click Firmware Upgrade Templates.

4. Click the Shared Firmware Upgrade Templates on THWACK tab.

5. Locate the template you need:
   - Use the Search box to search for a string in the template name.
   - Click a tag in the left pane to filter the list.
   - Click the template name to display its description.

6. Select a template and click Import.

7. If prompted, enter your THWACK user name and password, and click Log In.
   - The template is copied to your Orion server.
Get notified when a network configuration file changes

Configure real-time change detection in NCM to be notified immediately whenever a configuration file changes. See the following topics:

- About real-time change detection in NCM
- Configure real-time change detection in NCM
- Examples of commands to send syslog or trap messages to the Orion server
- Limit the number of simultaneous uploads or downloads of network configs

About real-time change detection in NCM

When you configure real-time change detection (RTCD), NCM sends an email notification whenever a device configuration changes. Use RTCD notifications to quickly determine if the config change could potentially cause a problem. Immediate notification of changes helps you improve network security, prevent unexpected downtime, and resolve known errors faster.

Unlike the Config Change Report, RTCD detects changes only on the same configuration type. For example, if you download a startup configuration, make changes to it, and upload it as a running configuration, the change will be detected against the previous running configuration. RTCD does not compare running and startup configuration types.

How does real-time change detection work?

To enable real-time change detection, you configure your network devices to send syslog or trap messages to the Orion server, and then configure a rule to detect messages that indicate a config was changed. After you configure RTCD:

1. A user logs in to a device and changes the config.
2. The device sends a syslog or trap message to the Orion server.
3. A rule in your syslog or trap receiver detects a message that indicates a config change. The action in the rule launches the NCM executable RTNForwarder.exe.
4. RTNForwarder.exe downloads the updated config from the device.
5. NCM compares the updated config to a baseline to determine what changed.
6. NCM sends an email with details of the change to the designated recipient.
1. A user changes a device's config.

2. The device sends a syslog or trap message to the Orion server.

3. The action in a rule launches an NCM executable.

4. RTNForwarder.exe downloads the updated config.

5. NCM determines what changed.

6. NCM sends an email with details of the change.

Next steps

- Complete the steps to configure real time change detection.
- If a large number of config downloads is an issue after you enable RTCD, you can limit the number of simultaneous downloads.
Configure real-time change detection in NCM

When you configure real-time change detection, NCM sends an email notification whenever a config changes. To configure real-time change detection, complete the following steps.

**Step 1** Configure devices to send syslog or trap messages to the Orion server.

**Step 2** Configure rules to detect syslog or trap messages that indicate a config has been changed.

**Step 3** Create the account that NCM uses to access devices and download the updated config.

**Step 4** Specify config comparison and email notification options.

**Step 5** Enter NCM SMTP server details.

**Step 6** Enable real-time config change notifications.

---

**Step 1: Configure your devices to send syslog or trap messages to the Orion server**

Configure each device to send either syslog or trap data to the Orion server. You can configure multiple devices using a script, as described below. You can also use a config change template, such as the Enable Syslog - Cisco IOS template.

If your devices already send syslog or trap messages to Kiwi Syslog Server or a third-party syslog or trap receiver, you can skip this step and continue with **Step 2**.

⚠️ SolarWinds strongly recommends that you configure **Cisco** devices to send syslog messages, not trap messages. Cisco devices send trap messages when a user enters config mode but not when the user exits. Therefore, if you configure Cisco devices to send trap messages, you aren't notified of a change until the next time a user enters config mode.

1. Click My Dashboards > Network Configuration > Configuration Management.
2. Select the nodes, and then click Execute Script.
3. Enter the commands to forward syslog or trap messages to the Orion server.

See this topic for examples of the commands. For more information, see your device documentation.

In addition, consider disabling logging for logins from the NCM server, especially if NCM must open config mode to download the device configuration. Disabling logging for NCM logins ensures that changes made by NCM don't trigger RTCD, which would result in a loop.

4. Click Execute.

5. Click Transfer Status.

6. In the Action column, locate the most recent entry labeled Execute Script.

7. Click Show Script Results in the Status/Details column.

You can remove device configurations by running a command with no in front of it. For example, no set logging server {ip_address} removes that target from the remote logging stream.

Step 2: Configure rules to detect config change notifications

Configure rules to detect syslog or trap messages that indicate a config has been changed. You might need different rules for different device types. Each rule should include an action to execute RTNForwarder.exe, which downloads the config and determines what changed.

The message text used to trigger the rule should be from a message received after all the changes made in the session have been completed, for example when exiting configuration mode.

See the appropriate instructions below, depending on the message type and what product you use to manage syslog or trap messages:

- You have Cisco IOS and ASA devices that send syslog messages
- You have Log Analyzer or Orion Log Viewer (OLV)
- You do not have Log Analyzer or OLV, and your devices send syslog messages to the Orion server
- You do not have Log Analyzer or OLV, and your devices send trap messages to the Orion server
- Your devices send messages to Kiwi Syslog Server
- Your devices send messages to a third-party syslog or trap receiver
You have Cisco IOS and ASA devices that send syslog messages

Log Analyzer, Orion Log Viewer, and the SolarWinds Syslog Viewer all provide preconfigured alerts for Cisco IOS and Cisco ASA devices that send syslog messages to the Orion server. To use these alerts, you just enable them.

The SolarWinds Syslog Service account must have read-write access to the Orion Platform database. For example, if your Orion Platform database resides on the same server as NCM, consider using a local administrator account for the SolarWinds Syslog Service.

1. Access log rules:
   - If you have Log Analyzer or Orion Log Viewer:
     a. In the Orion Web Console, click My Dashboards > Logs > Log Viewer.
     b. In the upper-right corner, click Configure Rules.
     c. In the left pane, expand Syslog, and click NCM Rule: Realtime Change Notifications.
   - If you do not have Log Analyzer:
     a. In the SolarWinds Orion program folder, start Syslog Viewer.
     b. Click View > Alerts/Filter Rules.

2. Select the following rules and enable them:
   - Cisco IOS Realtime Change Notifications
   - Cisco ASA Realtime Change Notifications

You have Log Analyzer or Orion Log Viewer

You can use either Log Analyzer or Orion Log Viewer to configure rules for devices that send syslog messages or trap messages. If you have devices other than Cisco ASA and IOS devices, complete these steps to configure a custom rule.

1. In the Orion Web Console, click My Dashboards > Logs > Log Viewer.
2. In the upper-right corner, click Configure Rules.
3. In the left pane, expand either Syslog or Traps, and click My Custom Rules.
4. Click Create New Rule, enter a descriptive name, and click Next.
5. Under Log Entries, select Look for specific entries.
6. Select Message and Contains, and then enter a string from the syslog message that the device sends when a config has been changed. Click Next.
The message varies by device type. For example, when a change is made to a Cisco router config, the device sends a syslog message containing `Configured from console`. For more information about what messages a device sends, see the device documentation.

<table>
<thead>
<tr>
<th>LOG ENTRIES</th>
<th>IF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Message</td>
<td></td>
</tr>
<tr>
<td>Matches Regex</td>
<td>end configuration</td>
</tr>
<tr>
<td>OR</td>
<td>Matches Regex</td>
</tr>
</tbody>
</table>


8. Select Run External Program, and click Configure Action.

9. Under Program to run, enter:

   \texttt{path\Orion\SolarWinds.NCM.RTNForwarder.exe}

   where \texttt{path} is the location of the Orion folder. If the path contains, spaces, enclose the line in quotation marks.

10. Under Command line arguments, enter:

    \texttt{$\{IpAddress\},RealtimeNotification,$\{DateTime\},$\{Message\}}

    where:

    - \texttt{$\{IpAddress\}} is a variable that includes the IP address of the triggering device.
    - \texttt{RealtimeNotification} is the text is displayed as the user name value. The text is required if you include the Message variable.
    - \texttt{$\{DateTime\}} is a variable includes the current date and time.
    - \texttt{$\{Message\}} is a variable that includes the Syslog message in the real time detection notification. If your Syslog message contains the user making the change, the user name is included through the use of this variable.

    \textbf{Note:} You must include the commas and, if including Message, you must include placeholder text in the second comma delimited location and the \texttt{DateTime} variable.

11. Click Done to add the action.
12. Click Next, and then click Save.

**You do not have Log Analyzer, and your devices send syslog messages to the Orion server**

If you do not have Log Analyzer or Orion Log Viewer, use the Syslog Viewer to configure custom rules for devices (other than Cisco ASA and IOS devices) that send syslog messages.

1. In the SolarWinds Orion program folder, start Syslog Viewer.
2. Click View > Alerts/Filter Rules.
3. Click Add New Rule.
4. Enter information on the General and DNS Hostname tabs.
5. Click the Message tab.
6. Under Syslog Message Pattern, enter a string from the syslog message that the device sends when a config has been changed.
   
   The message varies by device type. For example, when a change is made to a Cisco router config, the device sends a syslog message containing **Configured from console**. For more information about what messages a device sends, see the device documentation.

   ```
   Syslog Message Pattern
   "Configured from console"
   ```

7. Click the Alert Actions tab, and click Add New Action.
8. Select Execute an External Program, and click OK.
9. In the Program to Execute field, enter the following:
   
   ```
   path\Orion\SolarWinds.NCM.RTNForwarder.exe ${IP_Address},RealtimeNotification,${DateTime},${Message}
   ```

   where:

   - **path** is the location of the Orion folder. If the path contains spaces, enclose the path section of the statement in quotation marks: "Path to executable".
   - **${IP_Address}** is a variable that includes the IP address of the triggering device.
- `RealtimeNotification` is the text displayed as the user name value. The text is required if you include the `Message` variable.
- `${DateTime}` is a variable that includes the current date and time.
- `${Message}` is a variable that includes the Syslog message in the real time detection notification. If your Syslog message contains the user making the change, the user name is included through the use of this variable.

You must include the commas and, if including `Message`, you must include placeholder text in the second comma delimited location and the `DateTime` variable.

```
Execute SolarWinds.NCM.RTNForwarder.exe ${IP_Address}, RealtimeNotification, ${DateTime}, ${Message}
```

10. Click OK.

11. Make sure the new rule is selected in the Alerts/Filter Rules tab of the Syslog Server Settings window, and click OK.

**You do not have Log Analyzer, and your devices send trap messages to the Orion server**

If you do not have Log Analyzer or Orion Log Viewer, use the SolarWinds Trap Viewer to configure custom rules for devices (other than Cisco ASA and IOS devices) that send SNMP trap messages.

- Trap Viewer does not include a predefined rule with filters for trap messages because SolarWinds strongly recommends using the syslog option instead.
- The SolarWinds Trap Service account must have read-write access to the Orion Platform database. For example, if your Orion Platform database resides on the same server as NCM, consider using a local administrator account for the SolarWinds Trap Service.

1. In the SolarWinds Orion program folder, start the Trap Viewer.
2. Click View > Alerts/Filter Rules.
3. Click Add New Rule.
4. Enter information on the General and DNS Hostname tabs.
5. Click the Conditions tab, and click Add a Condition.
6. Click SNMPv2-MIB:snmpTrapOID, and then browse to the MIB that contains the trap message.

   For example, browse to CISCO-CONFIG-MAN-MIB:ccmHistoryEventConfigDestination (1.3.6.1.4.1.9.9.43.1.1.6.1.5).
7. Click the asterisk, and type the message pattern to match.
   
   For example, when a change is made to the running config the HistoryEventMedium is 3. Changes to the startup config are designated by the integer 4.

8. If you need to match on more than one condition, click Browse next to your last condition, and then click the correct conjunction: And or Or.

   Repeat the previous steps for as many conditions as you need to match. For example, along with the change history event value, consider matching the command source CISCO-CONFIGMAN_MIB:ccmHistoryEventCommandSource (1.3.6.1.4.1.9.9.43.1.1.6.1.3) and select 1 (command line) or 2 (snmp) as the value. For more information about what messages are sent from your devices, see the device documentation.

9. Click the Alert Actions tab, and then click Add Action.

10. Select Execute an External Program, and click OK.

11. In the Program to Execute field, enter the following:

    `path\Orion\SolarWinds.NCM.RTNForwarder.exe $\{IP\}`

    where:

    - `path` is the location of the Orion folder. If the path contains spaces, enclose the path section of the statement in quotation marks: "Path to executable".
    - `$\{IP\}` is a variable that includes the IP address of the triggering device.

12. Click OK.

13. Ensure the new rule is selected in the Alerts/Filter Rules tab of the Trap Server Settings window, and click OK.

**Your devices send messages to Kiwi Syslog Server**

- Kiwi Syslog Server **must** be running on the Orion Platform server. Kiwi Syslog Server cannot be used to execute a program on a different server. If Kiwi Syslog Server runs on a different server, you can configure it to forward the syslog message to the Orion Platform, and then configure Log Analyzer or Syslog Viewer to execute the program.

1. Start the Kiwi Syslog Server Console.
2. Click File > Setup.
3. Right-click Rules and click Add Rule. Then right-click the new rule and rename it.

4. Add a filter to identify syslog messages sent when a config is changed:
   a. Under the rule, right-click Filters and click Add Filter.
   b. Right-click the new filter and rename it.
   c. In the Field drop-down, select Message text.
   d. In the Filter Type drop-down, select Simple.
   e. In the Include box, enter a string from the syslog message sent when a config is changed. Enter the string in quotes.
      The message varies by device type. For example, when a change is made to a Cisco router config, the device sends a syslog message containing Configured from console. For more information about what messages a device sends, see the device documentation.

5. Add the action to take when a message matches the filter criterion:
   a. Under the rule, right-click Actions and click Add Action.
   b. Right-click the new action and rename it.
   c. In the Action drop-down, select Run external program.
   d. In the Program File Name box, enter:
      
      ```
prompt\Orion\SolarWinds.NCM.RTNforwarder.exe
      ```
      where `path` is the location of the Orion folder. If the path contains spaces, enclose it in quotation marks.
   e. In the Command Line Options box, enter `%MsgIPAddr,RTN,%MsgText`.

6. Make sure that the rule, filter, and action are enabled (checked), and then click OK.

**Your devices send messages to a third-party syslog or trap receiver**

1. Start your third-party syslog or SNMP trap receiver.

2. Set up a rule that executes an external program when it receives a message indicating that a config changed.

3. Enter the following in the Program to execute field:

   ```
   \"prompt\Orion\SolarWinds.NCM.RTNforwarder.exe\" $IP
   ```
   where:
   - `path` is the location of the Orion folder
   - `$IP` is a variable that includes the IP address of the triggering device
4. Save the rule, and make sure it is enabled.

Step 3: Create the account that NCM uses to access devices and download configs

When a config change is detected, NCM must be able to access the device and download the latest config so that it can determine what changed. Use the Config Changes page to create the Windows account that NCM uses to create and run RTCD-related download jobs.

1. Open the Real-Time Change Detection page:
   a. In the Orion Web Console, click Settings > All Settings.
   b. Under Product Specific Settings, click NCM Settings.

2. Click the Config Changes link in Step 3.

   The Config Changes page opens.

3. Select Enable these account credentials to access all NCM-managed devices.

   If the check box is disabled, then the Device Login & User Account Credentials option is set to Global - Device Level. To change this option, click the Security link to open the Security page, and then select Individual - User Level.

4. Enter the credentials that NCM can use to access devices and run RTCD-related download jobs.

5. If you want the email message NCM generates to include the syslog or trap message that signaled a config change, select Include syslog/trap message.

6. Click Submit.

Step 4: Specify config comparison and email notification options

When a config change is detected, NCM accesses the device, downloads the modified config file, and compares it to an existing config file to determine what changed. NCM then emails the specified recipients to notify them of the changes. Complete the following steps to specify config comparison and email notification options.
1. **Open** the Real-Time Change Detection page.

2. Click the Config Downloads and Notification Settings link in Step 4.

   ```markdown
   **Step 4:** On the Config Downloads and Notifications Settings page:
   - Select a download option (running or startup)
   - Select a baseline config file (last downloaded or baseline)
   - Enter email address(es) for receiving notifications
   ```

3. Under Previously Downloaded Config File, select the type of config file that NCM should download for comparison when a config change is detected.

4. Under Baseline Config File, select the config file you want NCM to use for comparison when it determines what changed.

5. Select the desired Email Notification Options.

6. Enter the Sender Name and Subject for NCM to use in RTCD email notifications, and specify at least one recipient. The Reply Address is optional.

   Email notification fields can include the following types of variables (also called macros). For example, use `${DateTime}` to include the date and time in the Subject.

   - **Global**
   - **Node**
   - **Date and time**

7. Click Submit.

**Step 5: Enter NCM SMTP server details**

The email server settings you enter here are used to send notifications regarding RTCD, config change approvals, and running jobs. For information on config change approvals, see [Approval system for device configuration changes](#).

1. **Open** the Real-Time Change Detection page.

2. Click the NCM SMTP Server link in Step 5.

   ```markdown
   **Step 5:** Enter NCM SMTP Server details to specify which server to use for email notifications
   ```

3. Enter the fully qualified domain name (FQDN) or IP address of the mail server.

4. Enter the port number on which the mail server handles messages.

5. Select None or Password as the Authentication method.
6. Enter a user name and password.
7. Click Submit.

**Step 6: Enable real-time config change notification**

1. Open the Real-Time Change Detection page.
2. Under Enable Real-Time Config Change Notifications, click Enable.
3. Click Submit.

If a large number of config downloads is an issue after you configure RTCD, you can limit the number of simultaneous downloads.

**Examples of commands to send syslog or trap messages to the Orion server**

When you **configure real-time change detection**, you must configure devices to send either syslog or trap messages to the Orion server. The following sections provide examples of commands used for some device types. For more information, see your device documentation.

**Syslog (IOS)**

This example configures a Cisco router to send syslog events whose severity level is informational or greater (logging trap level 6) to the Orion server (IP address 10.199.3.43).

```bash
config terminal
logging 10.199.3.43
logging trap 6
end
```

**Syslog (CatOS)**

The example configures a Catalyst switch to send syslog events whose severity level is notification or greater to the Orion server (192.168.0.30).

```bash
set logging server 192.168.0.30
set logging server facility local4
set logging server severity 5
set logging server enable
```

The commands do the following:
- set logging timestamp adds timestamps to all syslog events.
- set logging server adds the Orion server (192.168.0.30) as an export target.
- set logging server severity limits syslog exports to events with a severity level of notification (level 5) or greater.
- set logging server specifically enables the switch to export relevant syslog data to the logging server you have already specified (the Orion server).

**Traps (IOS)**

This example configures a device to send trap alerts to the Orion server when a config-related event occurs on the device.

```bash
snmp-server 10.110.68.33 public config
snmp-server enable traps config
```

Here are the commands with variable names instead of sample values:

```bash
snmp-server {host_name|IP_address} community_string trap_type
snmp-server enable traps trap_type
```

**Traps (CatOS)**

This example configures a device to send trap alerts to the Orion server when a config-related event occurs on the device.

```bash
set snmp trap 10.110.68.33 public config
set snmp trap enable config
```

Here are the commands with variable names instead of sample values:

```bash
set snmp trap {host_name|IP_address} community_string trap_type
set snmp trap {enable|disable} trap_type
```
Ensure compliance to policy rules

In NCM, rules, policies, and policy reports work together to help you ensure compliance to internal policies and external regulations.

- About NCM policy reports
- Create and manage policy rules
- Regular expression pattern matching examples
- Create and manage policies
- Create and manage policy reports
- Find and remediate policy violations
- Import and export policy report definitions

About NCM policy reports

Use policy reports to verify that device configurations comply with internal policies and external regulations. Each policy report enforces one or more rules. When the policy report runs, NCM scans the specified configuration files and reports any rule violations. Policy reports can also include remediation scripts to bring the configuration file into compliance.

SolarWinds provides a set of example policy reports that you can modify to meet your needs. You can also create custom policy reports.

Policy reports cannot be run against configurations that are downloaded in XML format.

How rules, policies, and policy reports work together

Each policy report includes one or more policies, and each policy includes one or more rules.
A rule defines a condition that must or must not exist. Rules can also contain remediation scripts to be run if the rule is violated.

For example, a rule could specify that devices must have banners that include copyright information. And the rule could contain a script to add the copyright information if it is missing.

- A policy groups related rules, and specifies which nodes and config types the rules apply to.
- A policy report groups related policies. When the report runs, it scans the configs specified in the policies and reports any rule violations.

With this structure, you can include the same rule in multiple policies, and the same policy in multiple reports.

Common uses for policy reports

Regulatory compliance
Use policy reports to ensure that you are in compliance with federal regulations and other industry standards, including:

- Sarbanes-Oxley Act (SOX)
- Health Insurance Portability and Accountability (HIPAA)
- Computer Inventory of Survey Plans (CISP)
- Payment Card Industry (PCI) data security policies

Compliance with internal standards and policies
Standardization is a vital part of keeping the network running smoothly. Use policy reports to locate device configurations that do not comply with your organization’s standards or policies. For example:

- Enforce interface naming guidelines.
- Enforce Quality of Service (QoS) traffic shaping policies.
- Verify that the correct banner is present.
- Change the copyright date when the year changes.

Security
To proactively protect against hackers, malware, and other security threats, use policy reports to ensure that network device configurations comply with your organization’s security policies. For example:

- Ensure that the default password has been reset on all devices.
- For SNMP-enabled devices, verify that the default public community string is not present.
- Enforce standards for password length.
- Search access control lists (ACLs) for rules that must or must not be present.

**Tasks to create and run a policy report**

To create custom policy reports, complete the following tasks:

1. **Create the rules** that the policy report will enforce.
2. **Create policies** to group related rules and to define which device configs will be checked for compliance.
3. **Create the policy report** to group related policies.

You can then run the policy report to **find and remediate policy violations**. You can also **share policy reports** between Orion servers or with other SolarWinds users.

**Create and manage policy rules**

**Rules** are used to ensure that device configurations comply with policies. Each rule defines a search string that either must or must **not** be present in a configuration. For example, a rule can specify that configurations cannot include the read-only community string "public."

Each rule applies to a specific set of device configurations, which are selected at the policy level.

![Policy Report Diagram]

To learn how rules, policies, and policy reports work together, see **Policy compliance**.

**Create a rule**

Complete the following tasks to create a rule:

1. **Add and identify the rule**.
2. Do one of the following:
   - **Define the search string using basic matching**.
     Use basic matching if you want to search the entire config for one string.
   - **Define search conditions using Advanced Config Search**.
     Use Advanced Config Search if you need to build conditions using multiple strings, or limit the search to a block of the config.
3. **(Optional) Define a remediation script.**

4. **Test and save the rule.**

**Task 1: Add and identify the rule**

1. Click My Dashboards > Network Configuration > Compliance.
2. Click Manage Policy Reports.
3. Click the Manage Rules tab.
4. Click Add New Rule.
5. Name the rule, and enter a description.
6. Select the Alert Level that indicates the severity of violating this rule.

   You can [customize alert level descriptions](#).

7. Select an existing folder to store the rule in, or select New folder and enter a folder name.

**Task 2: Define the search string**

Define the search string in either of the following ways:

- Use **basic string matching** to search the entire config for one string. You can enter a simple string or a regular expression.

  Choose **Advanced Config Search** if you want to:

  - Define a search string that includes conditions (multiple lines connected by **AND** or **OR** operators).
  - Specify a block of the config to search, instead of searching the entire config.

   See [Examples of search string conditions](#) for information about defining conditions.

**Define the search string using basic string matching**

1. Specify whether the rule triggers an alert when the string is found or when the string is **not** found.

2. Specify whether the string should be evaluated as a regular expression or a simple string.

   A simple string can contain the variables ? and *.

3. Enter the string to search for.
NCM uses the Microsoft .NET RegEx engine to evaluate regular expressions. See Regular expression pattern matching examples for some examples of use in NCM. For detailed information about the syntax, see Microsoft’s Regular Expression Language Quick Reference.

Define search conditions using Advanced Config Search

1. Specify whether the rule triggers an alert when the string is found or when the string is not found.

   If the search string includes lines that must be included and lines that must not be included, choose the alerting option that is appropriate for the first line. See Example 2.

2. Select Advanced Config Search.

   The Advanced Config Search and Search Config File/Block sections are displayed.

3. In the Advanced Config Search section, define the first search string.

   Must/Must Not Contain

   String Type Specify whether the string should be evaluated as a regular expression or a simple string.

   A simple string can contain the variables ? and *.

   String Enter the string to search for.

4. To build conditions into your search:

   a. Click Add Another String.

   b. Enter the string and specify the relationship between the lines. (See Examples of search string conditions.)

   And/Or Select an operator to define the relationships between strings.

   Parens Optionally, use parentheses to group strings into conditional relationships.
c. Repeat to add as many conditions as you need to define the rule.

5. To search only part of the config:
   a. Under Search Config File/Block, select Config block.
   b. Enter strings or regular expressions to define the beginning and end of the block.

   |⚠️ Do not use a multiline regular expression to define the beginning of the block.|

   c. Specify the string type.

(Optional) Task 3: Define a remediation script

Each rule can include a script to add or remove lines of a configuration to remediate the policy violation.

1. Enter or load a script that makes the required changes to the config.

   The remediation script must include CLI statements that run on the relevant devices. When executed, the script runs through the default communication protocol, such as Telnet or SSH.

   The script should perform the following tasks in order:
   a. Put the device into configuration mode, if needed.
   b. Issue a series of config commands.
   c. Exit config mode.

2. Select a Remediation Script Type. Your selection determines how NCM executes commands against targeted devices to remediate a policy rule violation.

   - CLI allows NCM to use the commands in the script to change the config.
   - Config Change Template launches the Config Change Template wizard to guide you through executing the script.

3. For a CLI script:

   - If you want NCM to execute the script automatically when a policy violation is found, select Automatically execute this script.

   Otherwise, you can choose to execute the script when you view the policy report that lists the rule violation.

   |ℹ️ Be sure to test the script thoroughly before you choose to execute it automatically.|

   - If you want NCM to put the device in config mode before executing the script, select Execute Script in Config Mode.
Task 4: Test and save the rule

Test the rule to verify that it finds policy violations as expected.

If the rule includes multiple lines containing regular expressions, SolarWinds recommends testing each regular expression individually before you test the rule as a whole. You can:

- Use an online RegEx tester to test each expression.
- Include only one of the regular expressions in a rule, and use NCM to test that rule against your configs.

1. Click Test to validate the rule against a device configuration.
2. Do one of the following:
   - To paste the config contents you want to test the rule against, choose Paste a config block.
   - To test against an uploaded config file, choose Select a config (selected by default).
3. Paste a config block or select a config to test the rule against.
4. Click Test Rule Against Selected Config.

If the config violates the rule, the Results panel indicates which search strings violated the rule and whether the violation occurred because the string was found or not found.

If the violation occurred because the string was found, you can click the arrow to display the line number in the config file.

5. Click Select Different Config to test the rule against another config.

SolarWinds recommends testing the rule against at least two configurations: one known to comply with the rule, and another known to violate it.

If you are testing a regular expression that uses the $ operator, be aware that the downloaded config can contain special non-printable characters that prevent the $ operator from matching the line end. To determine if there are non-printable characters at line ends, paste lines from the config into a plain text file in a text editor. If you see extra, empty lines in the text file that are not visible in the config, those lines probably contain non-printable characters.
6. When you have finished testing, click Close.

7. Click Submit to save the rule in the selected folder.

Examples of search string conditions

The following sections show how to define some simple conditions, and how NCM evaluates each example.

Example 1: Config must contain at least one of multiple strings

In this example, a config must contain at least one of the following strings:

Must contain string1
OR must contain string2
OR must contain string3

NCM reports a rule violation if the config does not contain **any** of the strings.

Example 2: A config must contain one string and must not contain another

In this example, the first string is required, but the second cannot be included.

Must contain string1
AND must not contain string2

NCM reports a rule violation if:

- A config does **not** contain string1.
- A config contains **both** string1 and string2.
Example 3: Config must contain at least one string in a group and a string outside the group

This example uses parentheses to group the first two strings. A config must contain at least one of the strings in the group, as well as the third string.

(Must contain string1
OR must contain string2)
AND must contain string3

NCM reports a rule violation if:

- A config contains either **string1** or **string2**, but does not contain **string3**.
- A config contains **string3**, but does not contain either **string1** or **string2**.
- A config does not contain **any** of the strings.
Edit a rule

Edit a rule to reflect policy changes or to refine the search string or remediation script. For example, if policy reports return false positives on violations, edit the rule to improve the search string.

1. Click My Dashboards > Network Configuration > Compliance.
2. Click Manage Policy Reports.
3. Click the Manage Rules tab.
4. Select a rule, and click Edit.
5. Edit values, and click Submit.

Delete a rule

1. Click My Dashboards > Network Configuration > Compliance.
2. Click Manage Policy Reports.
3. Click the Manage Rules tab.
4. Select a rule, and click Delete.
5. Click Yes.

Customize alert level descriptions for rules

NCM provides three levels to specify the severity of a rule violation. By default, these levels have the following descriptions.

<table>
<thead>
<tr>
<th>Level</th>
<th>Default Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Informational</td>
</tr>
<tr>
<td>2</td>
<td>Warning</td>
</tr>
<tr>
<td>3</td>
<td>Critical</td>
</tr>
</tbody>
</table>

You can customize the description of each level.

1. Click Settings > All Settings.
2. Under Product Specific Settings, click NCM Settings.
4. In the text field for the level that you are changing, replace the word describing that level.

5. Click Submit.

Learn more

After you define rules, you can create a policy and then create a policy report.

Regular expression pattern matching examples

NCM uses the Microsoft .NET RegEx engine to evaluate regular expressions. For detailed information about the syntax, see Microsoft’s Regular Expression Language Quick Reference.

The following examples illustrate some of the uses of Regular Expression patterns in NCM:

```
ncm-server community public
```

Finds any line that includes the text `ncm-server community public`. There can be text before and/or after the string on the same line.

```
service tcp-keepalives-in.*\n(.*\n)*.*service tcp-keepalives-out
```

Finds the first line `service tcp-keepalives-in` and then looks for `service tcp-keepalives-out` on any line after that. The regular expression string `.*\n(.*\n)*.*` is used to search any number of lines between strings.

```
access-list 105 deny.*tcp any any eq 139 log
```

Finds the line with `access-list 105 deny`, followed by any number of characters of any type, followed by `tcp any any eq 139 log` on the same line. The regular expression string `.*` finds any character, and any number of characters on the same line. So, this could be used to find spaces, tabs, numbers, letters, or special characters.

```
ntp clock-period \d*
```

Finds any line that includes `ntp clock-period` followed by any number. The regular expression string `\d*` will find any number at any length, such as 3, 48, or 2394887.

```
user \x2a
```

Finds any line that includes `user` *. The regular expression string `\x` followed by a hexadecimal value specifies an individual character. In this example, `\x2a` represents the asterisk character, which has a hexadecimal value of 2a.

```
\baccess-list\b.*\bjoeb\b
```
Finds any line that includes access-list and joe. The regular expression \b specifies a word boundary, so the pattern will not find (for example) access-lists or joel. The . matches any character except line breaks, and the * matches 0 or more of the preceding character. Together, the .* allow any number of characters between the two strings, but not line breaks.

Create and manage policies

A **policy** is a collection of one or more **rules** used to ensure policy compliance. For example, you could create a password policy that includes all of your organization’s password rules. Each policy also identifies which nodes and config types will be checked for compliance to the associated rules.

You can include the same rule in multiple policies.

ℹ️ To learn how rules, policies, and policy reports work together, see [Policy compliance](#).

Create a policy

1. Click My Dashboards > Network Configuration > Compliance.
2. Click Manage Policy Reports.
3. Click the Manage Policies tab, and click Add New Policy.
4. Name the policy, and enter a description.
5. Select an existing folder to store the policy in, or select New folder and enter a folder name.
6. Select the nodes that must comply to this policy.
   - Select only nodes that you want to report on.
   - You can use custom properties to identify specific nodes.
   - Policy reports cannot be run against configurations that are downloaded in XML format (for example, configurations from Palo Alto devices).
7. Select the type of config to search for policy violations.
   - The config type "Favorite" identifies configs that were used as baselines in NCM 7.8 or earlier.
8. Under All Policy Rules, select the rules to include and click Add. If the rule you need is not listed, you can [create a rule](#).
9. Click Submit.
Edit a policy

1. Click My Dashboards > Network Configuration > Compliance.
2. Click Manage Policy Reports.
3. Click the Manage Policies tab.
4. Select a policy, and click Edit.
5. Edit values, and click Submit.

Delete a policy

1. Click My Dashboards > Network Configuration > Compliance.
2. Click Manage Policy Reports.
3. Click the Manage Policies tab.
4. Select a policy, and click Delete.
5. Click Yes.

Learn more

After you create your policies, you can create a policy report.
Create and manage policy reports

Policy reports provide a way to group policies, either by the devices that they will be executed against or by the type of rules they enforce. When a policy report runs, NCM checks device configurations to determine if they comply with the policies.

You can include the same policy in multiple policy reports.

To learn how rules, policies, and policy reports work together, see Policy compliance.

Create a policy report

1. Click My Dashboards > Network Configuration > Compliance.
2. Click Manage Policy Reports.
4. Name the report, and enter a description.
5. Select settings for Save in Folder and Display Settings.
6. Under All Policies, select the policies to include and click Add. If the policy you need is not listed, you can Create a policy.
7. Click Submit.

Edit a policy report

1. Click My Dashboards > Network Configuration > Compliance.
2. Click Manage Policy Reports.
3. Select a report, and click Edit.
4. Edit values, and click Submit.

Delete a policy report

1. Click My Dashboards > Network Configuration > Compliance.
2. Click Manage Policy Reports.
3. Select a report, and click Delete.

4. Click Yes.

Configure DISA STIG compliance reports

NCM provides default rules, policies, and policy reports you can use to determine if your devices comply with DISA standards. The Defense Information Systems Agency Security Technical Implementation Guides (DISA STIG) reports are available in your NCM installation. To run the reports that specify the policies for your devices, make sure that you have first defined the node "Device_Type" in custom properties for your device type.

For example, for Cisco devices, specify the Device_Type and OS in custom properties with the following options:

**Device Types:**

PRTR, PL3S, IRTR, IL3S, L2SW, FW

**OS types supported:**

IOS, NX, XE, ASA

Likewise, for F5 devices, specify the Device_Type in custom properties.

Create a DISA STIG dashboard

You can set up a dashboard specifically to monitor compliance. To create a DISA STIG compliance dashboard, see the complete details in [How to Create a DISA STIG Dashboard](#) on THWACK.

Learn more

After you define policy reports, you can use the reports to find and remediate policy violations. You can also share policy reports between Orion servers or with other SolarWinds users.

**Find and remediate policy violations**

You can use policy reports to find device configurations that do not comply to policies. To view information about policy violations, you must:

1. Make sure the [cached policy compliance information is up-to-date](#).
2. View a policy report to display the cached information.

After you have viewed a report and remediated one or more violations, you can [verify that the violations were successfully remediated](#).
Update cached policy compliance information

NCM runs policy reports against the selected nodes to locate any policy violations, and then caches the results so that the data can be accessed quickly. You can update cached policy compliance information in any of the following ways:

- Enable the policy cache to automatically update information each day. (This option is enabled by default.)
- Manually update the policy cache.
- Schedule a policy report job to update cached information for that report and send emails about violations.

Automatically update cached policy compliance information

When the policy cache is enabled, NCM automatically runs policy reports at the specified time to check for policy violations. By default, the policy cache is updated daily at 11:55 PM.

1. Click Settings > All Settings.
2. Under Product Specific Settings, click NCM Settings.
5. Specify what time to generate the policy cache.
6. Click Submit.

Manually update cached policy compliance information

After you modify policy rules or download updated configuration files, you can manually update cached policy compliance information to reflect the changes.

1. Click My Dashboards > Network Configuration > Compliance.
2. To manually update the cached information:
   - To update all reports, click Update All.
   - To update one or more reports, select the reports and click Update Selected.

Schedule a policy report job

You can configure a policy report job to send emails each time the job runs or only if it finds policy violations. When the job runs, it updates the cached policy compliance information so that the report provides a snapshot of current policy compliance.
1. Click My Dashboards > Network Configuration > Jobs.

2. Click Create New Job.

3. Name the job, and select Generate a Policy Report from Job Type.

4. Specify when the job runs:
   - To run the job once or on a simple schedule, select Basic. Click the tab that identifies how frequently the job runs, and then specify the start time and (if needed) the day(s).
   - To create a more complex schedule, select Advanced and then use the five fields to create a CRON expression.

5. Add a comment if this job relates to a business rule, and click Next.

6. On the Choose Nodes tab, click Next.

7. Select an email notification option, and click Next. If you click Email Results, the default email notification and SMTP server settings are populated. These settings can be overridden in each job.

8. Select the policy report to generate as part of the job.

9. If you want to suppress notifications when no violations are found, select Send Notification Only When There Are Policy Violations.

10. Click Next.

11. Review the settings for the job, and click Finish.

**View a policy report**

When you view a report, it displays the latest cached policy compliance information. Use this information to investigate and remediate policy violations.

If compliance information for a report is not current, you can manually update the information.

1. Click My Dashboards > Network Configuration > Config Summary.
   - The Policy Violations resource lists the policy violations found the last time each report ran.

2. Click a report name to open the Report Details page.
   - The upper left corner shows the Last Updated date and time.
   - Icons indicate which rules were violated on each node:
- A green check mark ✅ indicates that the rule was **not** violated.
- Any other icon indicates that the rule was violated, and reflects the severity of violating that rule.

<table>
<thead>
<tr>
<th>Policy</th>
<th>Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>CISP Cisco Password Security (searched 4 configs)</td>
<td>Enable Password Encryption (2 violations)</td>
</tr>
<tr>
<td>Node Name</td>
<td>IP Address</td>
</tr>
<tr>
<td>CiscoASA</td>
<td>Violation icon</td>
</tr>
<tr>
<td>N5K-BOTTOM-B</td>
<td>10.199.3.41</td>
</tr>
</tbody>
</table>

3. Click a violation icon.

The Violation Details dialog shows the rule that was violated and indicates if a remediation script is available.

If the violation occurred because the string was found, you can click the arrow to display the line number in the config file.

4. (Optional) Click View Config to open the Config Details view in a different tab.

5. (Optional) If a remediation script is available, click a Management option to execute the script on this node or on all nodes in violation.

**Verify that a policy violation was remediated**

When you discover a policy violation, complete the following steps to resolve the issue and verify that the remediation was successful.
1. Update the configuration file to resolve the policy violation, either by editing the file or by running a remediation script.

2. Download the updated configuration file.

3. Update the cached policy compliance information for the report that detected the policy violation.

4. View the report to verify that the policy violation is not found.

**Import and export policy report definitions**

If you have multiple Orion servers, you can share policy report definitions between them. You can also share policy report definitions on THWACK.

**Share policy reports between Orion servers**

You can export a policy report definition as an XML file, and then import it to another Orion server. The exported file also includes the underlying policy and rule definitions.

**Export a policy report definition as a file**

1. Click My Dashboards > Network Configuration > Compliance.

2. Click Manage Policy Reports.


   The XML file is saved in your \Downloads directory. The file name is based on the report name.

**Import a policy report definition**

1. Click My Dashboards > Network Configuration > Compliance.

2. Click Manage Policy Reports.

3. Click Import.

4. Click Choose File, and select a policy report previously exported as a file.

5. Click Submit.

   The policy report and underlying policies and rules are created on the Orion server. By default, the policy report is not assigned to a folder. You can edit the report to assign it to a folder.
Share policy report definitions with the THWACK community

You can share your policy report definitions with other NCM users by exporting them to THWACK, and you can import policy report definitions created by other NCM users.

Export a policy report definition to THWACK

1. Click My Dashboards > Network Configuration > Compliance.
2. Click Manage Policy Reports.
3. Select a report, and click Export to THWACK.
4. If prompted, enter your THWACK user name and password, and click Log In.

Import a policy report definition from THWACK

1. Click My Dashboards > Network Configuration > Compliance.
2. Click Manage Policy Reports.
3. Click the Shared on THWACK tab.
4. Locate the policy report definition you need:
   - Use the Search box to search for a string in the policy report definition name.
   - Click a tag in the left pane to filter the list.
   - Click the policy report definition name to display its description.
5. Select a policy report definition and click Import.
6. If prompted, enter your THWACK user name and password, and click Log In.

The report and the associated rules and policies are copied to your Orion server.
NCM device templates

To learn more about creating and managing NCM device templates, see the following topics:

- About NCM device templates
- Gather the information you need for an NCM device template
- Best practices for NCM device templates
- NCM communication process diagrams
- Create and manage device templates
- F5 device template behavior
- NCM device template commands
- Examples of device templates
- Import and export device templates
- How device templates are assigned to nodes

About NCM device templates

Device templates contain device-specific commands that enable NCM to perform actions on the associated device type. For example, one device template could enable NCM to execute scripts and download configuration files from Cisco Pix Firewall 535 devices, while another device template could enable those operations on Juniper devices.

NCM provides a set of device templates to perform actions on common device types. You can create custom device templates to support additional device types. You cannot change the default NCM device templates, but you can copy them to create a custom device template that meets your organization’s needs.

NCM uses device templates for the following operations:

- NCM uses device templates when it **uploads and downloads configs** on a device. The device template must define the commands used to perform those operations on the associated device type.

- The commands defined in a device template can also be included in **scripts**. To include a device template command in a script, format the command `Name` as a macro (for example, `${EnterConfigMode}`).
When scripts use device template command macros instead of CLI commands, you can use the same script to perform an operation on different device types that require different CLI commands. For example, the value of \texttt{EnterConfigMode} is \texttt{config terminal} for Cisco IOS devices, but it is \texttt{configure} for Juniper devices. When you use a command macro in a script instead of the actual command, you can use the same script for multiple device types.

\begin{figure}
\centering
\includegraphics[width=\textwidth]{device_template_diagram.png}
\caption{Diagram illustrating how device templates use command macros.}
\end{figure}

**Gather the information you need for an NCM device template**

Before attempting to modify or create a new device template, be sure you know the commands required for \texttt{CLI-based devices} and the keystrokes for \texttt{menu-based devices}.

**Command line interface commands for CLI-based devices**

The following command examples are for Cisco IOS. Identify the commands for your device type.

- What are the \texttt{Machine Type} and \texttt{System OID} values displayed in the Device Details tab of the node properties?

  Use this information to save the device template with a unique name that NCM recognizes, ensuring that the template is used when NCM connects to the device.

- What command is used to \texttt{disable pagination}?

  This command is the value used in the template \texttt{RESET} command.

  Example: \texttt{terminal pager 0}

- What command is used to \texttt{reboot the device}?
This command is the value used in the template Reboot command.

Example: `reload noconfirm`

- What command is used to enter configuration mode?
  
  This command is the value used in the template EnterConfigMode command.

  Example: `config terminal`

- What command is used to exit configuration mode?
  
  This command is the value used in the template ExitConfigMode command.

  Example: `quit`

- What command is used to specify the startup configuration?
  
  This command is the value used in the template Startup command.

  Example: `startup`

- What command is used to specify the running configuration?
  
  This command is the value used in the template Running command.

  Example: `running`

- What command sequence is used to directly download the configuration using Telnet or SSH?
  
  This command sequence is the value used in the template DownloadConfig command.

  Example: `show ${ConfigType}`

- What command sequence is used to upload the configuration using Telnet or SSH?

  This command sequence is the value used in the template UploadConfig command.

  Example:`${EnterConfigMode}${CRLF}${ConfigText}${CRLF}${ExitConfigMode}`

- What command sequence is used to download the configuration using SNMP, that is, indirect transfer?

  This command sequence is used in the template DownloadConfigIndirect command.

  Example: `copy ${TransferProtocol}://${StorageAddress}/${StorageFilename}$ConfigType`${CRLF}${CRLF}`
What command sequence is used to **erase the configuration**?

This command sequence is used in the template `EraseConfig` command.

Example: `write erase ${CRLF}Yes`

What command sequence is used to **commit a configuration to memory**?

This command sequence is used in the template `SaveConfig` command.

Example: `write memory`

What command sequence is used to **show the version** information?

This command sequence is used in the template `Version` command.

Example: `show version`

### Keystrokes for menu-based devices

Network Configuration Manager supports the upload and download of configs on menu-based devices that do not have command line interfaces. However, NCM does not support execution of command scripts on exclusively menu-based devices. To create or edit a device template for a menu-based device, you must determine what keystrokes are required to perform upload and download commands and format these as device template command values.

On some menu-based devices, such as Cisco SF300 LAN switches, menu item numbers can be used instead of arrow moves. For example, instead of assigning:

```
Value="${ENTER}${DownArrow}${DownArrow}${DownArrow}${DownArrow}${DownArrow}
                  ${DownArrow}}${DownArrow}
```

You could instead assign:

```
Value="1${ENTER}7${ENTER}"
```

To gather the information you need for a menu-based device command template:

1. Manually Telnet to your device to discover the pre-commands you must send before the device presents the login screen. Pre-commands are used for any device that requires input before prompting for credentials. For example, when you connect to a router and before you are asked for password, you must press Enter to wake up the connection. Add the following line to the template:

   `<Command Name="PreCommand" Value="${CRLF}"/>`
2. NCM also sends a **Version** command during the validate login action. To set this command value, complete the following:

   a. To determine this command, find the option in the menu which shows device version information. For example, if the System Information menu shows device version information and to access this menu item you press the down arrow key (↓) twice and then press Enter, type the following line into the device command template:

   ```xml
   <Command Name="Version" Value="${DownArrow}${DownArrow}${CRLF}">
   ```

   b. Find the string that is received when the command is complete. For example, if the command is complete when the device responds with System Characteristic, then you must add the following attribute to the command:

   ```xml
   RegEx="System Characteristic"
   ```

   c. Add a delay between keystrokes by adding the following attribute:

   ```xml
   Delay="300"
   ```

   d. The complete command line for the Version command is now:

   ```xml
   <Command Name="Version" Value="${DownArrow}${DownArrow}${CRLF}" RegEx="System Characteristic" Delay="300" />
   ```

3. Access the configuration file menu, and then download a configuration manually. During this operation, note the keys you press to complete this process. For example, on a Nortel Baystack 552048T you would press the following keys to download a configuration:

   - Down arrow (↓) nine times – Highlights Configuration file menu item
   - Enter – Opens Configuration file menu
   - Enter – Opens file Download/Upload menu
   - ConfigName + Enter – Sets the name of configuration file
   - Down arrow (↓) + TFTP IP Address + Enter – Sets the TFTP server address
   - Down arrow (↓) + Space + Enter – Starts the downloading process

4. Translate all these command into NCM variables. In this example, the following commands are used:

   ```xml
   - ${Downarrow}${Downarrow}${Downarrow}${Downarrow}${Downarrow}
   - ${Downarrow}${Downarrow}${CRLF}
   - ${CRLF}
   ```
For a list of commands and their descriptions, see NCM device template commands.

5. Find the string that is received when the command is complete. For example, the command is complete when the device responds with "written". In this case, you must add the following attribute to the command: RegEx="written".

6. Add a delay between keystrokes by adding the following attribute: Delay="300".

7. The complete download command is as follows:

   <Command Name="DownloadConfigIndirect" Value="{DownArrow}{Downarrow}{Downarrow}{Downarrow}{Downarrow}{Downarrow}{Downarrow}{Downarrow}{CRLF}{CRLF}{StorageFilename}{CRLF}{DownArrow}{StorageAddress}{CRLF}{DownArrow}" Delay="300" RegEx="written"/>

Click here for an example of a menu-based device template.

Best practices for NCM device templates

Review the following best practices before modifying device templates.

- Review several device templates and familiarize yourself with the command syntax before creating a new template.

- Write down a list of all the commands you need to include in the new device template, including whether or not you have to press Enter after you type the command to ensure the device recognizes the command.

- Telnet to your device to find the pre-commands you need. A pre-command can be used for any device which requires input before prompting for credentials. A pre-command is used before logging in. For example, when you connect to a router and before you are asked for a password, you must press Enter to wake up the connection. Add the following line to the template:

  <Command Name="PreCommand" Value="{CRLF}"/>

- Create a new device template by modifying an existing device template.

- Before modifying a device template, make a copy of the original.

- If you have a device that indicates enable mode with any character other than the # character, add the following line to the template: <Command Name="EnableIdentifier" Value="*"/>,
where * is the character used to indicate the enable privilege level.

- Ensure that you do not have two command templates with the same System OID available for automatic assignment.

- If the value for the Command Device Template field within the Node Details view is set to Auto Determine, NCM chooses the command template with the System OID value that is closest to the system OID of the device. For example, if the System OID for the device is 1.3.6.1.9.25.5.4, then NCM starts the search for a template that includes 1.3.6.1.9.25.5.4 as the System OID. If no template is found, NCM looks for a template with 1.3.6.1.9.25.5, and then 1.3.6.1.9.25, and so on. To be safe, use the full System OID when building templates.

- To declare the ready prompt for your device, use the VirtualPrompt command to designate the prompt: `<Command Name="VirtualPrompt" Value="unc-dsf%"/>`, where unc-dsf% is the prompt used by the device to designate it is ready for commands to be sent. You can use the Virtual Prompt to avoid an issue with special characters in banners. For example, to avoid NCM recognizing the # character as an enable prompt. Ensure you use the MenuBased command when using the VirtualPrompt command: `<Command Name="MenuBased" Value="false"/>` or `<Command Name="MenuBased" Value="true"/>`.

- Device such as VPN concentrators might require a null value for the Reset command to function properly. If you receive an Out of Range error, change the value of the Reset command from 0 to blank (\). For example, `<Command Name="RESET" Value=""/>`.

NCM communication process diagrams

The following diagram shows the process that NCM uses to establish an SSH or a Telnet session with a device and when it accesses commands from the device template during this process.
The following diagram shows the process that NCM uses to send upload or download commands to a device and when it accesses commands from the device template.
Create and manage device templates

NCM provides default device templates for many commonly used devices. You cannot edit the default templates. If you want to change a default template, or you need a template for a different type of device, you can create a custom template.
Use either of the following methods to create or edit device templates in NCM:

- Use the interactive wizard to guide you through the options. The interactive wizard can be used to create device templates for CLI-based devices. You can use it to specify the commonly used commands.

- Use the XML editor. You can use the XML editor to create device templates for menu-based devices and to include less common device template commands.

See the following sections:

- Before you start
- Create a device template using the interactive wizard
- Create a device command template using the XML editor
- How to specify CLI commands and keystrokes
- About specifying the system OID
- Edit a custom device template
- Confirm the device template is working
- Delete a custom device template

Before you start

1. If you plan to create a device template for a different device type, first explore the NCM content exchange page on THWACK to see if someone else has already created a template for that device type. If so, you might be able to use that device template as-is or edit it to meet your needs. You can import device templates from THWACK.

2. Gather the information you need for the device type.

3. Review the best practices for device templates.

4. If you are creating a device template for an F5 device, review F5 device template behavior.

Create a device template using the interactive wizard

1. Click Settings > All Settings.

2. Under Node & Group Management, click Manage Device Templates.
3. Perform one of the following actions:

- To create a new device template based on an existing template, complete the following steps.
  a. Find an existing template that is similar to the one you need.
  b. Select the existing device template and click Copy.

> You must copy and rename SolarWinds-provided templates because they are not editable. SolarWinds recommends copying and renaming any template that you are using as the basis for a new template because the existing template might be applied to nodes.

  c. Give the template a unique name and click Save.
  d. Select the new template and click Edit > Using Interactive Wizard.

- To create a new device template from scratch, click Add New > Using Interactive Wizard.

4. Select a node to test the operations against. (Use the Level and Group By lists to organize the node list.) Click Next.

5. Choose the scope of operations for this device template:

  - **Execute scripts only:** Requires a valid connection profile. The device template must define the commands to turn off paging and show the version. If the device uses unrecognized prompts, these must be specified.

  - **Execute scripts and download config from the device:** The device template must define the commands to download the config and to specify each type of config. If the device requires an additional password to access higher privileges, the connection profile must include that password.

  - **Execute scripts, download config from the device, and upload config to the device:** The device template must define the commands to enter config mode, upload the config, and exit config mode.

Then click Next.

The Connection Profile section displays the connection profile for the selected device. The connection profile specifies the what credentials NCM uses to log in to the device and what protocols are used to transfer configs and execute scripts.

6. If the connection profile needs to be updated, make the required changes.
7. In the Device Template section:
   a. In the Reset Terminal Size Command field, enter the command to turn off paging for the device.

   For more information about all available NCM device template commands, see NCM device template commands. For information about entering commands, see How to specify CLI commands and keystrokes.

   b. In the Show Version Command, enter the command to display the software version.

   c. If either of the following are true, click Show advanced options:
      - The device uses nonstandard prompts that NCM does not detect by default.
      - The command to provide a higher level of access is something other than enable.

      For each non-standard prompt or command, click Custom and enter value that NCM should use when communicating with the device.

   d. Click Test to verify that Connection Profile and Device Template settings work against the node you selected.

      The test logs in to the device, executes the Reset and Show Version commands, and exits the session.

   e. After a successful test, click Next.

      If the scope includes downloading configs, the Config Download panel is displayed. If not, continue with step 11.

8. On the Config Download panel:
   a. Enter the command to download the config using the transfer protocol specified in the device template.

   b. Enter the commands used by this device to identify each config type.

   The Config Types section lists all default and custom config types associated with the selected device type. If the device does not use standard config types, you can enable a custom config type.

   c. Set the config type to use in the download test, and click Perform Download Test.

   d. After a successful test, click Next.

      If the scope includes downloading configs, the Config Upload panel is displayed. If not, continue with step 10.
9. On the Config Upload panel:
   a. Enter the commands to enter and exit config mode, write to memory, and reboot the device.
   b. Enter the command to upload the config using the transfer protocol specified in the device template.
   c. Click Next.
10. On the Save Device Template Panel:
    a. In the Save Device Template field, specify a name to identify this device template.
       In most cases, the name includes the vendor and the device type. If you created a copy of a default NCM device template, add something to make the name unique, such as your organization’s name. The name should be meaningful to NCM administrators.
    b. Enter a system OID that identifies the device types that can use this device template. See About specifying the system OID for more information.
    c. Specify whether this template can be automatically assigned to devices:
       - If you to not want the template to be automatically assigned, select No from the Use for auto-detect drop-down menu.
       - If you want the template to be automatically assigned, select Yes from the Use for auto-detect drop-down menu. Then specify whether to use the system OID or the description for automatic assignment.
    d. If you want to manually assign this template to devices, select the devices in the Assign Device Template section. You can also choose to assign the same connection profile to the selected devices and download configs.
       By default, the template is manually assigned to the device you chose to test the template against. If you do not want to manually assign the template to that device, click Remove above the device list.
11. Click Finish to save the device template.

If both this template and another template are available for automatic assignment and target the same devices, disable automatic detection on one of the templates. For example, if you copied a default template to customize it and both the default template and the custom template specify the same system OID, be sure to disable automatic detection on the default template.
Create a device command template using the XML editor

1. Click Settings > All Settings.
2. Under Node & Group Management, click Manage Device Templates.
3. Perform one of the following actions:
   - To create a new device template based on an existing template, complete the following steps.
     a. Find an existing template that is similar to the one you need.
     b. Select the existing device template and click Copy.
     You must copy and rename SolarWinds-provided templates because they are not editable. SolarWinds recommends copying and renaming any template that you are using as the basis for a new template because the existing template might be applied to nodes.
     c. Give the template a unique name and click Save.
     d. Select the new template and click Edit > Using XML Editor.
   - To create a new device template from scratch, click Add New > Using XML Editor.
4. Edit the value of any command, or add other commands.
   For more information about all available NCM device template commands, see NCM device template commands. For information about entering commands, see How to specify CLI commands and keystrokes.
5. Specify the template name, system OID, and whether this template can be automatically assigned to devices. For more information about these fields, see step 10 in the previous section.
7. Click Save.
Login credentials for menu-based devices

For menu-based devices, the login user name and password must be sent as pre-command values instead of from the connection profile. To do this:

1. Edit the device's connection profile:
   a. Clear the Username and Password fields.
   b. Set Enable to <No Enable Login>.

2. Enter the following pre-commands in the device template:

   ```xml
   <Command Name="PreCommand" Value="username${DownArrow}"/>
   <Command Name="PreCommand" Value="password${ENTER}"/>
   ```

How to specify CLI commands and keystrokes

For some device template commands, you must specify the CLI commands and keystrokes that are executed on the device when that device template command is called. For example, the device template command to write to memory might have the value write memory.

In addition to strings, device template commands can include the following elements:

- **Keystroke macros**, such as `$\{CRLF}\` which can simulate pressing Enter between commands:
  
  ```
  reload$\{CRLF\}Yes
  ```

- **Node macros**, such as `$\{StorageAddress}\`, which supply information stored for a specific node:
  
  ```
  copy \$\{TransferProtocol\}:\$\{StorageAddress\}\$\{StorageFilename\} \$\{ConfigType\}
  ```

- **Transfer macros**, such as `$\{ConfigText}\`, which used in commands to upload or download a config:
  
  ```
  \$\{ConfigText\}$\{CRLF\}exit$\{CRLF\}y`
  ```

- **Date and time macros**, which can be used with commands such as `RebootAt` to specify a reboot at a specific date and time:
  
  ```
  reload at $\{HH\}:$\{NN\}$\{CRLF\}Yes
  ```

- **Command macros**, which are other device template commands formatted as macros. For example, the following device template command runs the commands specified for EnterConfigMode and for ExitConfigMode:
  
  ```
  \$\{EnterConfigMode\}$\{CRLF\}$\{ConfigText\}$\{CRLF\}$\{ExitConfigMode\}
  ```
In the XML editor, CLI commands are entered as the Value of the command. For example:

```xml
<Command Name="DownloadConfig" Value="Show ${ConfigType}" />
```

In the wizard, CLI commands are entered in the text box (without quotes):

```
Download Config Command: Show ${ConfigType}
```

### About specifying the system OID

If the device template can be used by multiple device types, remove enough numbers from the end of the system OID to match all device types that should use the template, but leave in enough numbers to exclude device types that should not use it.

You can set up cascading templates by creating a series that targets slightly different OIDs. For example:

- 1.3.6.1.4.1.9 = Cisco (All)
- 1.3.6.1.4.1.9.1.23 = Cisco 2507

You can specify a specific device with a more exact OID, and NCM will try to find the closest match. In this example, if NCM is communicating with a device with a system OID of 1.3.6.1.4.1.9.1.25, it uses the Cisco (All) template, but if the system OID is 1.3.6.1.4.1.9.1.23, it uses the Cisco 2507 template.

> In some cases, removing enough numbers to include all desired device types also includes device types that should not use the template. If so, you can use the system description for automatic assignment or you can manually assign the template to devices.

### Edit a custom device template

You can edit custom templates, but you cannot edit the default templates shipped with NCM. To change a default device template, make a copy of the template and [create a new device template](#) based on the copy.

1. Click Settings > All Settings.
2. Under Node & Group Management, click Manage Device Templates.
3. Select a custom device template.
4. Click Edit and select either Using Interactive Wizard or Using XML Editor.
5. Edit values, and click Finish (in the wizard) or Save (in the XML editor).
Confirm the device template is working

1. Click My Dashboards > Network Configuration > Config Summary.
2. Under NCM Node List, click a node to which you assigned the device template.
3. On the node details page, click Configs on the left.
4. Under Download Config, select a config type.
5. Click Download.
6. If the downloads succeeds, the device template is working as expected.

Delete a custom device template

You can delete only custom device templates.

1. Click Settings > All Settings.
2. Under Node & Group Management, click Manage Device Templates.
3. Select one or more custom device templates.
4. Click Delete and click Yes at the confirmation prompt.

If you select one of the default templates shipped with NCM, the Delete button is not available.

F5 device template behavior

When using device templates for F5 devices, consider the following behavior.

Device setup prerequisites

- Set the Terminal Access user on the device to Advanced Shell. Do not use Traffic Management Shell (TMSH).
- Verify the device can reach the NCM server for configuration transfers using SCP.
- If you plan to upload binary user configuration set (UCS) files, the upload can take a long time. Increase the NCM timeouts to have a complete upload report.

1. Click Settings > All Settings.
2. Under Product Specific Settings, choose CLI settings.
3. Set Telnet/SSH Connection Timeout: 360 seconds
4. Telnet/SSH Prompt Timeout: 300 seconds
Template details

If you set up NCM to transfer configurations using SSH:

- The template can download the current running config.
- You cannot download the startup config.
- You cannot upload the running or startup config.

If you set up NCM to transfer configurations using SCP:

- The template can download and upload a text single configuration file (SCF) as a running config.
- The template can download and upload a binary UCS as startup config.
- You cannot upload the running config to startup, or startup to running config.

Downloading or uploading configs using TFTP is not supported.

NCM device template commands

Device templates can define the following commands for a specific type of device. These commands specify what keystrokes or CLI commands NCM sends when it performs an action on that type of device. They can also be used to change NCM’s default behavior. For example, you can change the text that identifies the password prompt or you can change the line feed characters that NCM sends by default.

These commands modify the interaction between NCM and your network devices. Ensure you fully understand what modifications will do before modifying a device in production using these commands.

In the XML editor, every device template command includes the following attributes:

- Command Name identifies the command.
- Value defines what that command does when it is executed.

Not all commands are supported on all devices.

Download commands

DownloadConfig

Use the DownloadConfig command to download a config directly using Telnet or SSH. The Value defines the CLI commands to display the config. When the device displays the configuration, NCM parses the information and downloads the config. Examples include:

<Command Name="DownloadConfig" Value="Show ${ConfigType}" />
DownloadConfigIndirect

Use the DownloadConfigIndirect command to download the config indirectly using TFTP. The config is transferred from the device to the SolarWinds TFTP server and then downloaded. The Value defines the commands to transfer the config to the TFTP server. For example:

```xml
<Command Name="DownloadConfigIndirect" Value="copy ${ConfigType} ${TransferProtocol}:/${StorageAddress}/${StorageFilename}${CRLF}${StorageAddress}${CRLF}y${CRLF}"/>
```

DownloadConfigIndirectSCP

Use the DownloadConfigIndirectSCP command to download the config indirectly using SCP. The config is transferred from the device to the SCP server and then downloaded. The Value defines the commands to transfer the config to the SCP server.

```xml
<Command Name="DownloadConfigIndirectSCP" Value="${EnterConfigMode}${CRLF}save ${TransferProtocol}:/${SCPServerUserName}@${SCPStorageAddress}:${StorageFilename}${CRLF}$SCPServerPassword${CRLF}${ExitConfigMode}"/>
```

ConfigType attribute for download commands

The ConfigType attribute identifies the config type (Running, Startup, Device Type, or a custom type). Include this attribute if different commands are issued for the same action depending on the config type. For example, Palo Alto devices can have a different DownloadConfigIndirectSCP command for each device type:

```xml
<Command Name="DownloadConfigIndirectSCP" ConfigType="Running" Value="${TransferProtocol} export configuration from running-config.xml to ${SCPServerUserName}@${SCPStorageAddress}:${StorageFilename}${CRLF}yes${CRLF}${SCPServerPassword}"/>
```

```xml
<Command Name="DownloadConfigIndirectSCP" ConfigType="Device State" Value="${EnterConfigMode}${CRLF}save device-state${CRLF}exit${CRLF}${TransferProtocol} export device-state to ${SCPServerUserName}@${SCPStorageAddress}:${StorageFilename}${CRLF}yes${CRLF}${SCPServerPassword} Regex="/\[edit\]" "/>
```
Multiple commands with the same name and config type

If a device template contains multiple commands with the same name and the same config type, NCM uses the value for the first command listed. For example, if a device template contained the following entries, NCM would use the first value, Show ${ConfigType}:

```xml
<Command Name="DownloadConfig" ConfigType="Running" Value="Show ${ConfigType}"/>
<Command Name="DownloadConfig" ConfigType="Running" Value="display ${ConfigType}"/>
```

Commands with the same name, with and without a config type

If a command does not specify a config type, it is the default command used for all config types not explicitly specified. For example, if a device template contained the following entries, NCM would use the value Show ${ConfigType} for any config type other than Running:

```xml
<Command Name="DownloadConfig" ConfigType="Running" Value="Show running"/>
<Command Name="DownloadConfig" Value="Show ${ConfigType}"/>
```

Upload commands

UploadConfig

Use the UploadConfig command to upload the config directly to a device using Telnet or SSH. The Value defines the CLI commands to upload the config. For example:

```xml
<Command Name="UploadConfig"
Value="${EnterConfigMode}$CRLF${ConfigText}$CRLF${ExitConfigMode}" />
```

UploadConfigIndirect

Use the UploadConfigIndirect command to upload the config indirectly using TFTP. The config is transferred to the SolarWinds TFTP server and then uploaded to a device. The Value defines the commands to transfer the config to the TFTP server. For example:

```xml
<Command Name="UploadConfigIndirect" Value="copy ${TransferProtocol}:/${StorageAddress}/${StorageFilename} ${ConfigType} vrf management${CRLF}$CRLF" />
```

UploadConfigIndirectSCP

Use the UploadConfigIndirectSCP command to upload the config indirectly using SCP. The config is transferred to the SCP server and then uploaded to a device. The Value defines the commands to transfer the config to the SCP server. For example:
ConfigType attribute for upload commands

The ConfigType attribute identifies the config type (Running, Startup, Device Type, or a custom type). Include this attribute if different commands are issued for the same action depending on the config type. For example, Palo Alto devices can have a different UploadConfigIndirectSCP command for each device type:

```xml
<Command Name="UploadConfigIndirectSCP" ConfigType="Running"
Value="${TransferProtocol}://${SCPServerUserName}@${SCPStorageAddress}/${StorageFilename} ${ConfigType} vrf management${CRLF}${SCPServerPassword}" />
```

```xml
<Command Name="UploadConfigIndirectSCP" ConfigType="Device State"
Value="${TransferProtocol}://${SCPServerUserName}@${SCPStorageAddress}/${StorageFilename} ${ConfigType} vrf management${CRLF}${SCPServerPassword}" />
```

For more information, see [Multiple commands with the same name and config type](#) and [Commands with the same name](#)

Commands to perform other actions

<configType> commands

Include a command for each config type (default or custom) that you manage on the device. The Name of the command specifies the config type (for example, Running, Startup, Device State, or a custom config type). The Value indicates the commands to specify that config type on the device. For example:

```xml
<Command Name="Startup" Value="saved-configuration"/>
<Command Name="Running" Value="current-configuration"/>
```

If the config is in XML and you want to maintain the formatting, use the Format attribute to specify XML:

```xml
<Command Name="Running" Value="running" Format="XML" />
```

Use the IsBinary attribute to identify a binary config:

```xml
<Command Name="Device State" Value="Device State" IsBinary="true" />
```
Disconnect

The value of this command defines the commands used to terminate the session when the operation is complete. The following example would issue the command exit:

<Command Name="Disconnect" Value="exit"/>

EnterConfigMode

The value of this command defines the commands used to enter the configuration mode of a device. The following example would issue the command config terminal:

<Command Name="EnterConfigMode" Value="config terminal"/>

ExitConfigMode

The value of this command defines the commands used to exit the configuration mode of a device. The following example would issue the command end:

<Command Name="ExitConfigMode" Value="end"/>

PreCommand

Use pre-commands for devices that require input before prompting for credentials, and for devices that prompt for interaction after an action is performed. The value attribute specifies the action to take when a device has not responded for more than three seconds (by default) and, optionally, when the device sends a specific prompt (such as Press Any Key).

For example, when you connect to a certain type of router, you must press Enter to wake up the connection before you are asked for a password. The following example sends a carriage return and line feed combination after a three second delay:

<Command Name="PreCommand" Value="${CRLF}"/>

The following example presses Ctrl+Y when the device does not respond for three seconds:

<Command Name="PreCommand" Value="${CTRL+Y}"/>

Use the Delay attribute to add a number of seconds to the default delay of three seconds. For example, the following command presses Ctrl+Y when the device does not respond for five seconds (the default three plus two added by the Delay attribute):

<Command Name="PreCommand" Value="${CTRL+Y}" Delay="2"/>

Use the RegEx attribute to respond to a specific prompt. For example, the following command presses Ctrl+Y when the device does not respond for four seconds and the last data received is Press Any Key:

<Command Name="PreCommand" Value="${CTRL+Y}" Delay="1" RegEx="Press Any Key"/>
Reboot

The Reboot command identifies the commands and keystrokes required to reboot the device. For example:

<Command Name="Reboot" Value="reload${CRLF}Yes"/>

RebootAt

The RebootAt command identifies the commands and keystrokes required to reboot the device at a specified time. Use the available date and time variables to assign the date and time. For example:

<Command Name="RebootAt" Value="reload at ${HH}:${NN}${CRLF}Yes"/>

RESET

The RESET command identifies the commands and keystrokes required to turn off paging. NCM attempts to run this command after it has logged in to the device, detected the enable prompt, and determined that the device is ready to accept commands. For example:

<Command Name="RESET" Value="terminal width 0${CRLF}terminal length 0"/>

Some devices, such as VPN concentrators, might require a null value for the RESET command to function properly. If you receive an Out of Range error, change the value of the RESET command from 0 to blank. For example:

<Command Name="RESET" Value=""/>

You can also use the RESET command if a device’s command mode prompt changes after the execution of a specific command. For example, some devices require you to switch user context before executing a command, resulting in a different command prompt. To handle this type of prompt changing, include the MenuBased command to specify the template logic should run in CLI mode, followed by the RESET command:

<Command Name="MenuBased" Value="false"/>
<Command Name="RESET" Value="switchContextCommand" RegEx="newPrompt"/>

The Value attribute identifies the command that changes the command mode prompt, and the RegEx attribute specifies the new prompt. For example:

<Command Name="RESET" Value="run /util bash" RegEx="#"/>

SaveConfig

The SaveConfig command identifies the commands and keystrokes required to write the configuration to the device’s memory. For example:

<Command Name="SaveConfig" Value="write memory"/>
Version

The `Version` command identifies the commands and keystrokes required to display the software version of the device. For example:

```xml
<Command Name="Version" Value="show version"/>
```

Commands that specify custom prompts or change the default behavior

AllocatePty

If your device does not support pseudoterminal device pairs, use the following command to prevent Telnet from attempting to negotiate a pseudoterminal setup:

```xml
<Command Name="AllocatePty" Value="false"/>
```

CommandCharKey

By default, NCM sends the following characters after each command to simulate pressing Enter:

- After pre-commands, NCM sends CRLF.
- After commands sent using SSH, NCM sends CR.
- After commands sent using Telnet, NCM sends CRLF.

For devices that require different characters, include this command to override the defaults.

```xml
<Command Name="CommandCharKey" Value="xx"/>
```

The value is one of the following:

- CR - NCM sends CR after each pre-command and command.
- CRLF - NCM sends CRLF after each pre-command and command.
- None - NCM does not send any characters after pre-commands, and it sends the default characters after each command.

ConfigStartsFromRegex

Use this command to exclude echo and specify the start of the configuration:

```xml
<Command Name="ConfigStartsFromRegex" Value="startOfConfig" />
```

The value is a regular expression that identifies the start of the config. For example:

```xml
<Command Name="ConfigStartsFromRegex" Value="&lt;\S+&gt;" />
```
If a device template includes the command above and a device returns the following:

echo 1
echo 2
<Config>
...
</Config>

NCM saves this as the configuration:

<Config>
...
</Config>

CustomMorePromptBehaviour

If NCM needs to send a CRLF to the device immediately when a More prompt is detected, add the following line to the top of the device command template:

<Command Name="CustomMorePromptBehaviour" Value="True"/>

CustomPasswordPrompt

By default, NCM recognizes the password prompt when it includes the words `password` or `passcode` and ends with a colon (:). Include this command for devices that use password prompts that NCM does not recognize:

<Command Name="CustomPasswordPrompt" Value="passwordPrompt"/>

The Value is a regular expression that identifies the custom password prompt. For example, the following identifies `key:` as the password prompt.

<Command Name="CustomPasswordPrompt" Value="key:"/>

CustomQuestionPrompt

If a device sends a question prompt that NCM does not recognize, use this command to identify the question prompt:

<Command Name="CustomQuestionPrompt" Value="promptText" />

The Value is a regular expression that matches the prompt text. For example:

<Command Name="CustomQuestionPrompt" Value="please confirm" />

When you include a CustomQuestionPrompt command, NCM’s default prompt detection is disabled. Therefore, you can use a CustomQuestionPrompt command for situations where you don’t want NCM to detect a question prompt by mistake. For example, a Huawei device sends a line similar to the following before the config and waits for Enter to continue:

When you include a CustomQuestionPrompt command, NCM’s default prompt detection is disabled. Therefore, you can use a CustomQuestionPrompt command for situations where you don’t want NCM to detect a question prompt by mistake. For example, a Huawei device sends a line similar to the following before the config and waits for Enter to continue:
With its default prompt detection logic, NCM mistakenly identifies this line as a question prompt and attempts to send the next command. To prevent this, specify any custom question prompt to disable the default prompt detection logic, and also include a pre-command to press Enter:

```xml
<Command Name="CustomQuestionPrompt" Value="/?" />
<Command Name="PreCommand" Value="${ENTER}" />
```

**CustomUserNamePrompt**

Use this command if the device has a non-default user name prompt that NCM does not recognize:

```xml
<Command Name="CustomUserNamePrompt" Value="userNamePrompt"/>
```

The `Value` attribute is a regular expression that identifies the user name prompt. For example:

```xml
<Command Name="CustomUserNamePrompt" Value="Name:"/>
```

**EnableCommand**

If NCM must enter a command to provide a higher level of access on a device but that command is **not** `Enable`, use this command to specify the command to use:

```xml
<Command Name="EnableCommand" Value="commandForHigherAccess"/>
```

For example:

```xml
<Command Name="EnableCommand" Value="super"/>
```

**EnableIdentifier**

If a device does not return the `#` character at the end of a prompt to indicate enable mode, use this command to identify the prompt that it uses:

```xml
<Command Name="EnableCommand" Value="promptUsedInEnableMode"/>
```

For example:

```xml
<Command Name="EnableIdentifier" Value="(enable)"/>
```

**EnterCLI**

Use this command to specify the keystrokes required to enter CLI mode on a menu-based device. NCM sends these commands after it receives the `VirtualPrompt` command:

```xml
<Command Name="EnterCLI" Value="keystrokesToEnterClimode" Delay="milliseconds" RegEx=""/>
```
The value defines the keystrokes that NCM sends. The delay attribute specifies the number of milliseconds to pause between keystrokes. The regex attribute should always be empty.

```
<Command Name="EnterCLI" Value="$\{Downarrow\}$\{Downarrow\}$\{CRLF\}" Delay="300" RegEx=""/>
```

**FreezeLoginForPreCommand**

Use this command to send all pre-commands to a device before starting the login process.

For example, if a device prompts for credentials twice, the following pre-commands send the first pair of credentials.

```
<Command Name="FreezeLoginForPreCommand" Value="True" />
<Command Name="PreCommand" Value="userName" />
<Command Name="PreCommand" Value="password" />
```

The commands above can also be used when NCM is unable to read the login prompts on a device.

*FreezeLoginForPreCommand cannot be used with SSH.*

**LineBreakChar**

By default, NCM uses CRLF to identify line breaks. For devices that use different characters, use this command to specify what characters NCM should use when uploading a config:

```
<Command Name="LineBreakChar" Value="lineBreakCharacters"/>
```

The value is one of the following:

- LF
- CRLF
- CR

For example:

```
<Command Name="LineBreakChar" Value="LF"/>
```

**MenuBased**

Use this command to specify whether the device is menu- or CLI-based. Valid values are true (for menu-based devices) or false (for CLI-based devices). For example:

```
<Command Name="MenuBased" Value="true"/>
```

If a device is menu-based and you can switch it to CLI from the menu, use the VirtualPrompt and EnterCLI commands to do so.
MenuDrivenConfigStart

For menu-based devices, use this command if you need to declare a value after which the transmitted data is considered the requested config. In the following example from the Cisco VPN Concentrator device template, the Value is #######. The information sent after the ten hash signs is saved as the requested configuration file.

```xml
<Command Name="MenuDrivenConfigStart" Value="###########"/>
```

More

If paging is not turned off on a device, the configuration could be split into multiple pages and NCM must send a Space to go to the next page. In most cases, the More prompt is automatically recognized. If not, include the More command to identify the More prompt used by the device.

```xml
<Command Name="More" Value="unrecognizedMorePrompt"/>
```

For example:

```xml
<Command Name="More" Value="-Press Any Key For More-"/>
```

Do not specify this command unless you are experiencing issues with paging.

MorePromptKey

By default, NCM sends a Space in response to a More prompt. For devices that require a different response, use this command to specify what keystroke should be sent.

```xml
<Command Name="MorePromptKey" Value="responseToMorePrompt"/>
```

For example, use the following command for devices that require pressing Enter in response to a More prompt:

```xml
<Command Name="MorePromptKey" Value="${Enter}"/>
```

PromptIgnoreSequences

On some devices, the command mode prompt changes after certain script lines are executed. For example, on Huawei devices, after `sys` is sent, the command prompt changes from `<prompt>` to `[prompt]`. For these devices, use this command to define the sequences of characters to be ignored during prompt comparison:

```xml
<Command Name="PromptIgnoreSequences" Value="charactersToIgnore"/>
```

The Value specifies the characters to ignore. Multiple characters are separated with this macro: `${;}$. For example, the following command specifies that parentheses () and square brackets [] should be ignored during prompt comparison:

```xml
<Command Name="PromptIgnoreSequences" Value="[${;}${;})${;]}("/>
```
If you include this command, be careful not to specify characters that, if ignored, would cause NCM to match other data sent by the device to the command prompt.

UseExactPromptMatch

When a config contains the command mode prompt as part of the config text, by default NCM stops capturing the config at that point and the downloaded config is incomplete. To prevent this, use the following command to cause NCM to detect the command mode prompt only if it appears on a separate line with no additional data:

```xml
<Command Name="UseExactPromptMatch" Value="true" />
```

UseMultipleDownloadCommands

By default, for direct transfers, NCM captures the output from only the first command in the DownloadConfig value. If you need to capture the output from other commands to download a config from a device, set UseMultipleDownloadCommands to True:

```xml
<Command Name="UseMultipleDownloadCommands" Value="true" />
<Command Name="DownloadConfig" Value="command1${CRLF}command2${CRLF}command3" />
```

For example:

```xml
<Command Name="UseMultipleDownloadCommands" Value="true" />
<Command Name="DownloadConfig" ConfigType="Running" Value="set cli op-command-xml-output on${CRLF}show config running="/>
```

You can also use UseMultipleDownloadCommands with a Cisco ASA device to download both the running and startup configs from the same context with only one device template. To do this:

1. Create a custom config type for each type of config in each context (for example, Context1Running and Context1Startup).
2. In your device template, include commands to switch into each context and download the config for each config type you created.
3. In the DownloadConfig command:
   - Enter `${ConfigType}` as the Value.
   - Specify the command prompt symbol as the RegEx attribute (for example, #, >, or ]).
For example:

```
<Command Name="Context1Running" Value="changeto context Context1{CRLF}show running" />
<Command Name="Context1Startup" Value="changeto context Context1{CRLF}show startup" />
<Command Name="DownloadConfig" Value="${ConfigType}" RegEx="#" />
<Command Name="UseMultipleDownloadCommands" Value="True" />
```

**UseVirtualPromptForCommands**

By default, the `VirtualPrompt` value is used only to recognize a nonstandard command mode prompt during the login stage. However, some devices change the command mode prompt frequently, especially during script execution. This causes NCM to execute scripts slowly. For these devices, you can include `UseVirtualPromptForCommands` to specify that NCM should use the `VirtualPrompt` value during the entire NCM session.

This command is useful when it is not possible to use `PromptIgnoreSequences` to define all possible character sequences in the command prompt, but the prompt always includes a common string. You can specify that common string as the `VirtualPrompt` value.

For example, during script execution, a device's command prompt changes after each command:

```
[Prompt#]
[Prompt#-Interface1]
[Prompt#-Interface2]
```

The common string is `Prompt#`, so the device template would include the following commands:

```
<Command Name="UseVirtualPromptForCommands" Value="True"/>
<Command Name="VirtualPrompt" Value="Prompt#"/>
<Command Name="MenuBased" Value="false"/>
```

⚠️ If you include this command, be careful not to specify a `VirtualPrompt` value that would cause NCM to match other data sent by the device to the command prompt.

**VirtualPrompt**

When the device does not have a standard command mode prompt that ends in `>` or `#`, use this command to identify the command prompt used to designate that the device is ready for commands to be sent. Include the `MenuBased` command to specify whether the template logic should run in CLI mode:

```
<Command Name="MenuBased" Value="trueOrFalse"/>
<Command Name="VirtualPrompt" Value="nonstandardPrompt"/>
```
The **Value** attribute is a string or regular expression that identifies the prompt that NCM should receive before sending commands. CLI-based example:

```xml
<Command Name="MenuBased" Value="false"/>
<Command Name="VirtualPrompt" Value="(enable)"/>
```

**Menu-based example:**

```xml
<Command Name="MenuBased" Value="true"/>
<Command Name="VirtualPrompt" Value="Welcome"/>
```

**VirtualEnablePrompt**

Use the **VirtualEnablePrompt** command to search the device’s entire response for a string or regular expression that identifies the enable prompt. This command is often used with menu-based devices, allowing you to locate a specific phrase returned by the device. If you use the **VirtualEnablePrompt** command, also include the **MenuBased** to specify whether the template logic should run in CLI mode. For example:

```xml
<Command Name="MenuBased" Value="true"/>
<Command Name="VirtualEnablePrompt" Value="device_edge#"/>
```

**Examples of device templates**

The following sections provide examples of NCM **device templates**:

- [Cisco IOS CLI device template example](#)
- [Nortel BayStack 380 CLI device template example](#)
- [Cisco VPN concentrator menu-based device template example](#)

For a list of commands and their descriptions, see [NCM device template commands](#).

**Cisco IOS CLI device template example**

**File Name**

Cisco IOS-1.3.6.1.4.1.9.ConfigMgmt-Commands

**Contents**

```xml
<!--SolarWinds Network Management Tools-->  
<!--Copyright 2008 SolarWinds.Net All rights reserved-->  
<Configuration-Management Device="Cisco Devices" SystemOID="1.3.6.1.4.1.9">
<Commands>
  <Command Name="RESET" Value="terminal width 0${CRLF}terminal length 0"/>
  <Command Name="Reboot" Value="reload${CRLF}y${CRLF}y"/>
  <Command Name="EnterConfigMode" Value="config terminal"/>
  <Command Name="ExitConfigMode" Value="end"/>
  <Command Name="Startup" Value="startup"/>
  <Command Name="Running" Value="running"/>
  <Command Name="DownloadConfig" Value="Show ${ConfigType}"/>
  <Command Name="UploadConfig" Value="${EnterConfigMode}${CRLF}${ConfigText}${CRLF}${ExitConfigMode}"/>
  <Command Name="DownloadConfigIndirect" Value="copy ${ConfigType}${TransferProtocol}://${StorageAddress}/${StorageFilename}${CRLF}${CRLF}"
        Value="copy ${TransferProtocol}://${StorageAddress}/${StorageFilename}${ConfigType}${CRLF}"/>
  <Command Name="EraseConfig" Value="write erase${CRLF}Y"/>
  <Command Name="SaveConfig" Value="write memory"/>
  <Command Name="Version" Value="show version"/>
</Commands>
</Configuration-Management>

Nortel BayStack 380 CLI device template example

File Name
Nortel Baystack380-1.3.6.1.4.1.45.3.45.ConfigMgmt-Commands

Contents

<!--SolarWinds Network Management Tools-->  
<!--Copyright 2008 SolarWinds.Net All rights reserved-->  
<Configuration-Management Device="Nortel BayStack 380 Devices" SystemOID="1.3.6.1.4.1.45.3.45">  
<Commands>
  <Command Name="RESET" Value="terminal length 0"/>
  <Command Name="Reboot" Value="reload${CRLF}Yes"/>
  <Command Name="EnterConfigMode" Value="config terminal"/>
  <Command Name="ExitConfigMode" Value="end"/>
Cisco VPN concentrator menu-based device template example

The following example provides the values declared for a menu-driven indirect transfer.

File Name

Cisco VPN concentrator-1.3.6.1.4.1.3076.ConfigMgmt-Commands

Contents

```xml
<Configuration-Management Device="Cisco VPN concentrator" SystemOID="1.3.6.1.4.1.3076">
  <Commands>
    <Command Name="RESET" Value=""/>
    <Command Name="Reboot" Value="2${ENTER}3${ENTER}"/>
    <Command Name="EnterConfigMode" Value=""/>
    <Command Name="ExitConfigMode" Value=""/>
    <Command Name="Startup" Value=""/>
    <Command Name="Running" Value=""/>
    <Command Name="DownloadConfig" Value="2${ENTER}8${ENTER}3${ENTER}config${ENTER}" Delay="500" RegEx="\[End\]"/>
    <Command Name="UploadConfig" Value=""/>
  </Commands>
</Configuration-Management>
```
Import and export device templates

If you have multiple Orion servers, you can share device templates between them. You can also share device templates on THWACK.

Share device templates between Orion servers

You can export a device template as an XML file, and then import it to another Orion server.

Export a device template

1. Click Settings > All Settings.
2. Under Node & Group Management, click Manage Device Templates.
3. Select a template, and click Export As File.

   The XML file is saved in your Downloads directory with the .ConfigMgmt-Commands extension.

Import a device template

You can import a device template that was previously exported as a file.

1. Click Settings > All Settings.
2. Under Node & Group Management, click Manage Device Templates.
3. Click Import.
4. Click Choose File, browse to the template, and select it.
5. Click Submit.

**Share device templates with the THWACK community**

You can share your device templates with other NCM users by exporting them to THWACK, and you can import device templates created by other NCM users.

**Export a device template to THWACK**

1. Click Settings > All Settings.
2. Under Node & Group Management, click Manage Device Templates.
3. Select a template, and click Export to THWACK.
4. If prompted, enter your THWACK user name and password, and click Log In.

**Import a device template from THWACK**

1. Click Settings > All Settings.
2. Under Node & Group Management, click Manage Device Templates.
3. Click the Shared Templates on THWACK tab.
4. Locate the template you need:
   - Use the Search box to search for a string in the template name.
   - Click a tag in the left pane to filter the list.
   - Click the template name to display its description.
5. Select a template and click Import.
6. If prompted, enter your THWACK user name and password, and click Log In.

The template is copied to your Orion server.

**How device templates are assigned to nodes**

You can manually assign a device template to a node or allow NCM to assign templates to nodes automatically.
Requirements for automatic assignment

To enable NCM to determine which device template to use for a node:

- The node must be configured to allow automatic assignment. (Automatic assignment is selected by default.)
- SNMP communication must be configured on the node. NCM uses SNMP to retrieve the system OID.
- The device template that you want the node to use must be available for automatic assignment.

How does NCM use OIDs to automatically assign device templates?

When a device template is automatically assigned to a node, NCM typically uses the device’s system OID to find the device template that best matches the node. Device templates specify a partial system OID to identify the target device type.

To assign a device template to a device, NCM searches the device templates that are available for automatic assignment. First, it attempts to find a device template that matches the entire system OID. If no match is found, NCM ignores the last number in the system OID and searches again. NCM repeats this process until it finds a match or no numbers are left.

For example, if a device’s OID is 1.3.6.1.4.1.9.1.516:

1. NCM first searches for a device template that specifies 1.3.6.1.4.1.9.1.516 as the OID.
2. If it does not find a match, it searches for a device template that specifies 1.3.6.1.4.1.9.1.
3. Next, it searches for a device template that specifies 1.3.6.1.4.1.9. This matches the NCM device template named Cisco IOS, and so that device template is assigned to the node.

Enable automatic assignment for a node or manually assign a device template to a node

1. Click Settings > All Settings.
2. Under Node & Group Management, click Manage Nodes.
3. Select a node, and then click Edit Properties.
4. Scroll to the bottom and make sure Manage Node(s) with NCM is set to Yes.
5. Under NCM Properties > CLI Device Template, select one of the following from the Device Template list:
   
   - To enable automatic assignment, select Auto Determine.

   ![Device Template: Auto Determine]

   - To manually assign a device template, select the device template.

   ![Device Template: PaloAlto]

6. Click Submit.

**Specify whether a template is available for automatic assignment**

You can specify which device templates NCM chooses from when it automatically assigns a template. For example, if you create a custom device template to use in place of a template that is shipped with NCM, you can make the custom template available for automatic assignment, and turn that feature off for the standard template.

When the Use for auto-detect option is set to No, the template is used only if it is manually assigned to a device.

1. Click Settings > All Settings.

2. Under Node & Group Management, click Manage Device Templates.

   The Use for auto-detect column contains Yes or No to indicate whether each template will be used for auto-detection.

   ![PaloAlto: Yes](1.3.6.1.4.1.25461.2)
   ![PaloAlto5050: No](1.3.6.1.4.1.25461.2)

3. Locate the device template you want to enable or disable:

   - To disable auto-detection, click the Yes button and then click Yes at the confirmation prompt. The button changes to No.
   - To enable auto-detection, click the No button. The button changes to Yes.
Compare a network config to a baseline or another config

To compare configurations to a baseline or another config, see the following topics:

- When you establish baselines as the approved configuration for a device, NCM automatically compares the device configuration to the baseline and reports any differences.
- Use NCM to find and review differences between a device’s config and the baseline.
- You can also compare two config files to each other. For example, you can compare different versions of the same config, or configs on different devices.
- To avoid flagging insignificant differences, such as the hostname, exclude expected differences from comparison.

Establish baselines as a comparison point for network config changes

The following sections provide information about creating and managing baselines:

- What is a baseline?
- Permissions required for baseline management
- Create a new baseline
- Edit a baseline
- Assign an existing baseline to devices

What is a baseline?

A baseline is a template that defines the approved configuration (or part of the approved configuration) for a device. After you define the baseline and assign it to one or more devices, NCM compares the baseline to downloaded configs and reports any mismatched lines.
Use baselines to standardize configurations across similar devices and ensure that all devices are in compliance. Assigning baselines allows you to:

- Know when a config doesn’t match the baseline.
- Compare updated configs against the baseline to determine what changed.
- Quickly roll back to a known good configuration in case of a network outage or unauthorized changes.

NCM uses an algorithm based on line contents to match lines in the config with the selected lines in the baseline.

Complete configurations vs. snippets

A baseline can represent either a complete config or a config snippet:

- If a baseline is the template for a complete config, the baseline must exactly match any config it is compared to (excluding lines that are ignored during the comparison). If the config contains lines that are not in the baseline, the config is flagged as not matching.
- A baseline snippet is the template for a section of the config. The baseline is only compared to that section, and any lines outside that section are automatically ignored. For example, a baseline snippet could define the template for a set of interfaces or access lists. You can assign multiple baseline snippets to a device.

A baseline snippet can be as short as a single line.

Permissions required for baseline management

Baseline management requires the following NCM roles (or higher).

<table>
<thead>
<tr>
<th>Action</th>
<th>Minimum NCM Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access the Baseline Management tab and view existing baselines</td>
<td>WebViewer</td>
</tr>
</tbody>
</table>
Create a baseline

When you create a baseline, you define the baseline contents, identify lines to ignore, and assign it to one or more devices.

Task 1: Define the baseline contents

Specify the contents of the baseline in one of the following ways:

- **Use an existing config:**
  1. Click My Dashboards > Network Configuration > Configuration Management.
  2. On the Config Management tab, expand a node to display the list of associated configs.
  3. Identify the config to use, and click Promote to Baseline.

<table>
<thead>
<tr>
<th>Name</th>
<th>IP Address</th>
<th>Baselines</th>
</tr>
</thead>
<tbody>
<tr>
<td>lab-merakifw-mx64</td>
<td>10.199.10.2</td>
<td>No baseline set</td>
</tr>
<tr>
<td>10/9/2018 02:01:18 am</td>
<td>Startup</td>
<td>Promote to Baseline</td>
</tr>
<tr>
<td>10/9/2018 02:00:48 am</td>
<td>Running</td>
<td>Promote to Baseline</td>
</tr>
</tbody>
</table>

The Promote to Baseline option is visible to users with the NCM role WebUploader or above.

4. On the New Baseline Config page, update the default name and (optionally) add a description.

5. Specify whether this is a **complete config file or a snippet**.

- **Select a file on the server:**
  1. Click My Dashboards > Network Configuration > Configuration Management.
  2. Click the Baseline Management tab.
  3. Click New Baseline.
4. On the New Baseline Config page, enter a name and (optionally) add a description.
5. Specify whether this is a complete config file or a snippet.
6. Click Browse and select the file.

- **Cut and paste the baseline contents:**
  1. Click My Dashboards > Network Configuration > Configuration Management.
  2. Click the Baseline Management tab.
  3. Click New Baseline.
  4. On the New Baseline Config page, enter a name and (optionally) add a description.
  5. Specify whether this is a complete config file or a snippet.
  6. Click Paste, paste the contents, and click Save.

The Configure section of the New Baseline Config page lists each line in the baseline.

<table>
<thead>
<tr>
<th>CONFIGURE</th>
<th>MARK UP BASELINE</th>
</tr>
</thead>
<tbody>
<tr>
<td>version 12.0</td>
<td></td>
</tr>
<tr>
<td>no service pad</td>
<td></td>
</tr>
<tr>
<td>service timestamps debug uptime</td>
<td></td>
</tr>
<tr>
<td>service timestamps log uptime</td>
<td></td>
</tr>
<tr>
<td>no service password-encryption</td>
<td></td>
</tr>
<tr>
<td>!</td>
<td></td>
</tr>
<tr>
<td>hostname uBR7246</td>
<td></td>
</tr>
<tr>
<td>!</td>
<td></td>
</tr>
<tr>
<td>boot system flash slot0:ubr7200-p-mz.120.5.T</td>
<td></td>
</tr>
<tr>
<td>boot system flash</td>
<td></td>
</tr>
</tbody>
</table>

**Task 2: Identify lines to ignore**

To avoid flagging inconsequential changes, identify lines or sections to ignore when the baseline is compared to configs.

On the New Baseline Config page, select one or both of the following options:

- **Apply global comparison criteria**
  
  Use global comparison criteria to exclude individual lines or blocks of text (such as certificate blocks) from comparison.
Choose lines to ignore

Select each line in the configuration to be ignored during a comparison.

NCM ignores the selected lines when the content doesn't match, but NCM reports an issue if an ignored line is missing from the configuration. For example, if you select the hostname line in the baseline, the corresponding line in the config file can have any value, but the line must be present.

To change the number of lines shown per page, click the highlighted line range at the bottom of the dialog and select a new maximum. This change remains in effect while the current baseline is open.

If you selected individual lines to ignore, the Ignored Lines section of the New Baseline Config page lists them.

Task 3: Assign the baseline to one or more devices

1. On the New Baseline Config page, click Assign to Nodes.
2. Select the nodes.

If you promoted an existing config file, the associated node is selected by default.

If you do not have administrator privileges, your account might include limitations that prevent you from seeing all nodes.
3. Under Apply To, select one or more config types (for example, Running and Startup) that this baseline should be compared to.

4. Click Save to apply your selections.

5. Click Save again to save the baseline.

   The baseline is enabled by default, and NCM automatically compares it to configs of the corresponding types downloaded from the selected devices.

Edit a baseline

You can edit a baseline to update the content of the baseline, or to make other changes:

- When you update the content of the baseline, you must **replace** the existing content. Any ignored lines are reset. You must apply global comparison criteria and choose the lines to ignore again.

  If you have a baseline with a large number of ignored lines, consider creating a new baseline and using the existing baseline as a reference to make sure that you select all ignored lines. Then you can delete the original baseline.

- Updating other information (for example, adding or removing ignored lines, changing the name, or assigning the baseline to other devices) does not reset your previous selections.

  You can assign or remove devices when you edit the baseline, or you can use the **Apply/Remove option** to change the associated devices without editing the baseline.
1. Click My Dashboards > Network Configuration > Configuration Management.

2. Click Baseline Management.

3. (Optional) To prevent comparison to the baseline until after the edits are complete, select the baseline and click Disable.

   The baseline is deselected automatically.

4. Select the baseline, and click Edit.

5. Make your changes, and click Save.

6. If you disabled the baseline, select it and click Enable.

   If you assign the baseline to additional devices, NCM automatically compares the updated baseline to the associated configs. If you make other changes, you can manually run a comparison.

**Assign an existing baseline to devices**

1. Click My Dashboards > Network Configuration > Configuration Management.

2. Click the Baseline Management tab.

   To see what nodes a baseline is currently assigned to, click the arrow on the right end of the baseline’s row.

3. Select the baseline.

4. Click Apply/Remove to open the Apply/Remove dialog box.

5. Select the nodes to assign this baseline to. Use filters or the search box to locate nodes.

   If you do not have administrator privileges, your account might include limitations that prevent you from seeing all nodes.
6. Select the config types this baseline should be compared to.

7. Click Save.

If you assigned the baseline to additional devices, NCM automatically compares the baseline to the associated configs.

**Find and review differences between baselines and network configs**

If you have assigned baselines to devices, NCM compares baselines to the associated configs and reports any differences. When NCM runs the comparison, it compares baselines to the most recently downloaded configs of the specified type.

- **When does NCM compare baselines and configs?**
- Manually run a comparison
- Enable or disable comparison to a baseline
- Find devices whose configs do not match the baseline
- Find baselines with mismatched configs
- Review differences between baselines and configs

**When does NCM compare baselines and configs?**

NCM automatically compares baselines and configs when:

- A config is downloaded from a device
- A baseline is assigned to a device
NCM does not automatically compare baselines and configs when:

- **Comparison criteria** are added or updated
- A config is removed from a device
- A baseline’s content or ignored lines are edited

If you make these changes, you can manually run a comparison.

**Manually run a comparison**

1. Click My Dashboards > Network Configuration > Configuration Management.
2. Click the Baseline Management tab.
3. Select one or more baselines.

   - Select the check box above the list to select all baselines on the current page.
   - Updating a large number of baselines can affect performance.

4. Click Update.

   NCM compares the baselines to the most recently downloaded configs from the associated devices.

**Enable or disable comparison to a baseline**

When a baseline is disabled, NCM does not compare configs to it.

1. Click My Dashboards > Network Configuration > Configuration Management.
2. Click the Baseline Management tab, and select one or more baselines.
3. Click Enable or Disable.

   Disabled baseline names are shown in pale text on the Baseline Management tab.
Find devices whose configs do not match the baseline

The Configuration Management tab identifies devices with one or more configs that do not match the baseline. You can click through to review the differences.

1. Click My Dashboards > Network Configuration > Configuration Management.

   The Configuration Management tab lists each device, and the Baselines column displays a status:

<table>
<thead>
<tr>
<th>Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mismatched lines</td>
<td>One or more baselines assigned to this device do not match the latest downloaded config of the corresponding type.</td>
</tr>
<tr>
<td>No issues</td>
<td>All baselines assigned to this device match the latest downloaded configs.</td>
</tr>
<tr>
<td>No configs</td>
<td>No configs have been downloaded from this device.</td>
</tr>
<tr>
<td>No baseline set</td>
<td>No baselines have been assigned to this device.</td>
</tr>
<tr>
<td>Updating</td>
<td>NCM is comparing the configs to the baselines.</td>
</tr>
<tr>
<td>Error</td>
<td>An error occurred during the comparison.</td>
</tr>
</tbody>
</table>

   See the Error details and Suggested action columns for information about the issue. For more information, review the NcmBusinessLayerPlugin.log. The default location is:

   c:\ProgramData\SolarWinds\Logs\Orion\NcmBusinessLayerPlugin.log

2. If a device’s status is Mismatched lines, hover over the status message to see which config does not match the baseline.

In the following example, the device’s startup config matches the baseline but the running config does not match.

3. Click the baseline name to review the differences in the Baseline Diff viewer.
Find baselines with mismatched configs

The Baseline Management tab identifies baselines with one or more mismatched configs. You can click through to review the differences.

1. Click My Dashboards > Network Configuration > Configuration Management.
2. Click the Baseline Management tab.

The Baseline Management tab lists all baselines, with the following status information.

<table>
<thead>
<tr>
<th>Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mismatched lines</td>
<td>This baseline does not match at least one of the corresponding configs on an associated device.</td>
</tr>
<tr>
<td>No issues</td>
<td>This baseline matches the corresponding configs on all associated devices.</td>
</tr>
<tr>
<td>(Empty)</td>
<td>This baseline is not assigned to any devices, or no configs have been downloaded from the associated devices.</td>
</tr>
<tr>
<td>Updating</td>
<td>NCM is comparing this baseline to the associated configs.</td>
</tr>
<tr>
<td>Error</td>
<td>An error occurred during the comparison.</td>
</tr>
<tr>
<td></td>
<td>Click Error to display more information.</td>
</tr>
</tbody>
</table>

3. If a baseline's status is Mismatched lines, click the status message to see which configs on which devices do not match the baseline.

In the following example, a mismatch occurs on one device. The baseline matches the device's startup config but not the running config.
4. Click the Mismatched lines link in the dialog box to review the differences in the Baseline Diff viewer.

Review differences between baselines and configs

Locate mismatches by device or by baseline to view differences in the Baseline Diff viewer:

- Lines in the baseline that are missing from the config are highlighted on the baseline side, with a gray hash on the config side.

- Extra lines in the config are highlighted on the config side.

- Ignored lines have a gray background.

- For baseline snippets, the Baseline Diff viewer initially shows only the config lines that
correspond to the baseline snippet. To view other lines in the config, click the + <number> unchanged line(s) link at the top or bottom of the viewer.

From the top of the viewer, you can:

- Click the baseline name at the top of the viewer to edit the baseline.
- Click the config name to display information about the device on the Node Details page.

**Baseline Diff**

<table>
<thead>
<tr>
<th>Cisco routers baseline</th>
<th>tok-2821-agg.lab.txt</th>
<th>Click to edit the baseline</th>
<th>Click to open the Node Details page</th>
</tr>
</thead>
<tbody>
<tr>
<td>10/15/2018 02:00 AM - Running</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Manually compare two network configuration files**

Use NCM to compare configuration files on two different nodes or to compare different versions of configs on the same node. You can:

- Compare running and startup configs to detect changes that were made to the in-memory config but were not permanently saved to non-volatile memory (NVRAM). These changes will be lost if the device loses power or is rebooted.
- In a multi-vendor environment, compare configs from different vendors for reference purposes.
- Compare different versions of the same config to quickly see what changed.
- Compare the configs on different nodes to ensure they match.

To avoid flagging inconsequential changes (such as the last updated date), specify lines or blocks to exclude from comparisons.

- When you assign a baseline to a device, NCM automatically compares the device’s configs to the baseline and reports any changes.
- Policies that are pushed from a Panorama management server are not part of the main configuration file. Therefore, these elements are not included in the comparison.
Select configs to compare from the Configuration Management page

From the Configuration Management page, you can compare any two configs or any two versions of the same config.

1. Click My Dashboards > Network Configuration > Configuration Management.
2. Select the nodes:
   - To compare configs from two different nodes, select both nodes.
   - To compare two configs from the same node, select a single node.
3. Click Compare Node(s) Configs, and then select Compare node(s) configs from the drop-down menu.
4. Select the configs you want to compare.
5. Review the differences.

Select configs to compare from the Node Details page

From the Node Details page, you can compare different versions of the same config.

1. Click My Dashboards > Network Configuration > Config Summary.
2. In the NCM Node List widget, click a node.
   The Node Details page displays information about that node.
3. ChooseConfigs from the menu on the left.
4. In the Compare Configs widget, select the versions you want to compare.

   The drop-down menus list up to 10 recently downloaded versions of this node's config. If the version you want is not listed, click View All Configs.

5. Click Compare and review the differences.
Review the differences

After you select the configs to compare, the Compare Configs Of Selected Nodes window displays the configs next to each other. To help you quickly identify changes, the page displays only the mismatched lines and the lines immediately surrounding them.

- If the configs match, no lines are displayed.

- Mismatched lines are highlighted in light yellow. The characters that changed are highlighted in darker yellow.

To provide context, NCM displays up to five unchanged lines (by default) before and after each difference. You can change the number of lines in Config Settings.

- A gray hash indicates that lines are missing from one of the configs.
Click the link before or after each change to display additional unchanged lines. Each time you click the link, the next five unchanged lines are shown.

Initially, the comparison shows a maximum of 500 changes. If there are more than 500 changes, a link to display additional changed lines is available. Each time you click the link, the next 100 changed lines are shown.

Exclude config content from comparison in NCM

You can identify lines or blocks of text that you do not want NCM to evaluate when it compares configs. For example, you can exclude:

- Empty lines
- Lines that do not represent substantive changes, such as a Last Updated line
- Expected differences, such as the hostname value
- Blocks of text that do not represent substantive changes, such as the certificate section of the config

Excluding this content speeds up the comparison process and makes reviewing the compared configs easier. It can also prevent NCM from flagging inconsequential or expected differences when it compares a downloaded config to a baseline.
NCM uses regular expressions to identify the content that it should ignore during comparisons. You can create and manage regular expressions on the Comparison Criteria page. NCM uses comparison criteria in the following comparison operations:

<table>
<thead>
<tr>
<th>Comparison operation</th>
<th>When Comparison criteria is used</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manual comparisons</td>
<td>Always</td>
</tr>
<tr>
<td>Config change reports</td>
<td>Always</td>
</tr>
<tr>
<td>Scheduled jobs</td>
<td>Always</td>
</tr>
<tr>
<td>Comparing configs to baselines</td>
<td>If the option to apply comparison criteria is selected in the baseline definition</td>
</tr>
</tbody>
</table>

Access the Comparison Criteria page

From the Comparison Criteria page, you can create, edit, enable, disable, or delete a regular expression.

1. Click Settings > All Settings.
2. Under Product Specific Settings, click NCM Settings.
3. Under Config Settings, click Comparison Criteria.

Create a regular expression

Create a regular expression that identifies the text that NCM should ignore when comparing configs.

See [Regular expression examples](#) for some regular expressions that are commonly used to exclude lines and blocks. Other resources, such as [www.Regular-Expressions.info](http://www-Regular-Expressions.info), are available online to help you learn more about regular expressions.

1. On the Comparison Criteria page, click Add New.
2. Enter a descriptive title.
3. Identify the text to exclude:
To exclude a single line, enter a regular expression that identifies the beginning of the line.

To exclude a block of text:

a. In the RegEx pattern box, enter the regular expression that identifies the beginning of the first line in the block.

b. Select Is block.

c. In the End RegEx pattern box, enter the regular expression that identifies the beginning of the last line in the block.

The search for the pattern that identifies the end of the block is "lazy." That means if the pattern occurs multiple times, NCM finds the first occurrence of the pattern.

4. To activate this regular expression, select Enable NCM to ignore this pattern. Otherwise, leave it disabled for future editing.

5. Click OK.

When you create and enable a new regular expression, it applies to manual config comparisons immediately. However, you must [download the config again](#) before the new regular expression affects config change reports and charts (such as the Overall Configuration Changes Snapshot).

**Edit a regular expression**

1. Select a regular expression on the Comparison Criteria screen, and click Edit.

2. Update any values on the Edit RegEx Pattern dialog.

3. Click OK.

**Enable or disable a regular expression**

Select one or more regular expressions on the Comparison Criteria screen, and click Enable or Disable.

**Delete a regular expression**

Select one or more regular expressions on the Comparison Criteria screen, and click Delete.
Regular expression examples

The following are examples of regular expression patterns you can use to exclude config lines or blocks from comparison.

<table>
<thead>
<tr>
<th>RegEx pattern</th>
<th>End RegEx pattern</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>^!</td>
<td>Last</td>
<td>Ignores the !Last Configuration change line in Cisco configurations.</td>
</tr>
<tr>
<td>^ntp clock-period</td>
<td></td>
<td>Ignores the ntp clock-period line in Cisco configurations.</td>
</tr>
<tr>
<td>^wlccp ap username cisco</td>
<td></td>
<td>Ignores the wlccp line in Cisco access point configurations.</td>
</tr>
<tr>
<td>certificate ca quit</td>
<td></td>
<td>Ignores all lines in a certificate block that starts with certificate ca and ends with quit.</td>
</tr>
</tbody>
</table>
Automate tasks with NCM command scripts and jobs

See the following topics to create and execute command scripts:

- Create an executable command script in NCM
- Execute a script on a node
- Example NCM command script with variables

See the following topics for information about NCM jobs:

- Create and manage NCM jobs
- Create a CRON expression

Create an executable command script in NCM

Several tasks can be automated with command scripts. For example:

- Downloading configuration files
- Uploading configuration files
- Uploading IOS images
- Updating logon banners
- Updating access control lists (ACLs)

With the correct use of variables, a single script can be executed on several different devices, without concern for syntax differences.

Scripts are delivered one line at a time to the target devices.

Create a command script

1. Click My Dashboards > Network Configuration > Configuration Management.
2. Click the Script Management tab.
3. Click Add New Script.
4. Add the script information.
5. Click Save.
6. To execute the script, on the Config Management tab select one or more nodes and follow the steps to Execute a script on a node.
Use variables in scripts

You can use the following types of variables (also called macros) in command scripts:

- Global
- Node
- Transfer
- Command
- Date and time

With Command variables, you can execute the same script on different types of devices. Command variables substitute the right commands based on the device type. For example, the variable `$(EnterConfigMode)` is parsed as `config terminal` when the script runs on Cisco IOS devices, but it is parsed as `configure` when the script run on an HP Procurve Switch.

See an example variable script.

Execute a script on a node

This widget is available to users with the WebUploader, Engineer, or Administrator role.

The script you execute must be saved as a text file that can be uploaded from the client computer. For more information about creating command scripts, see Create an executable command script in NCM.

1. Click My Dashboards > Network Configuration > Configuration Management.

   If you do not have administrator privileges you may not be able to see some nodes. Your account must at least have the WebDownloader role to download configs, and the WebUploader role to execute a script against nodes.

2. Select one or more nodes.

3. Click Execute Script.

4. Select a script to execute or click Load Script from File, and locate it in the file system.

   You can specify a delay in seconds inside a script. A delay is the time NCM waits before sending the next command. The following is a sample script that includes a delay:

   ```
   {Command 1}
   ${Delay:20}
   {Command 2}
   ```

   This feature is useful, for example, when uploading a flash image. Some time is required for the formatting of the flash to complete before then performing the image upload.
5. If you entered or updated the script and you want to save it to a file, select Save script and enter a file name.

6. If you want NCM to put the device into configuration mode before executing the script, select Execute Script in Config Mode.
   
   For example, on Cisco devices when this option is selected, NCM sends the `config t` command to the device before executing the script on the device. On some devices, configuration mode is required to make changes on the device.

7. Click Execute.

View the results

1. Click My Dashboards > Network Configuration > Configuration Management.

2. Click Transfer Status.

3. In the Action column, locate the most recent entry labeled Execute Script.

4. Click Show Script Results in the Status/Details column.

Troubleshoot script execution

If a script fails to execute on a NCM node, review the log for a possible cause. The log is located at C:\ProgramData\SolarWinds\NCM\Logging\NCMBusinessLayerPlugin.log.

Alternatively, execute the script by using Create and manage NCM jobs.

Example NCM command script with variables

The following script contains commands with variables to remove the public read-only community string:

```
${EnterConfigMode}
no snmp-server community public RO
${ExitConfigMode}
${SaveConfig}
${Reboot}
```

Parsed for Cisco IOS devices:

```
config terminal
no snmp-server community public RO
```
Define script variables

Script variables are defined in device command templates. You can create new device templates or edit existing custom templates.

Each device template contains a System OID that is used to uniquely identify a device. A list of command names, and the corresponding commands to be sent to the device when the command name is called, are also included in the templates. These command names are the variables used when creating a script.

Consider the following line taken from the Cisco IOS device command template:

```
<Command Name="EnterConfigMode" Value="config terminal"/>
```

When a script is run on a Cisco IOS device, the variable `${EnterConfigMode}` parses as `config terminal`. New command names can be added and existing command names can be modified within these files.

Reference variables with variables

The script engine also allows you to reference variables with variables. For example, you can define a complex variable in the device template:

```
ShowInt = running | include interface
```

Then define another variable that includes the first:

```
Reveal = show ${ShowInt}
```

When you call `${Reveal}`, it equals `show running | include interface`. 

The `${CRLF}` variable equals a carriage return line feed for all devices.
Create and manage NCM jobs

Network Configuration Manager provides job scheduling to automate the management of network devices and configuration files.

You can schedule numerous operations, including configuration file uploads and downloads, node reboots, and command script execution.

If NCM is configured to connect to nodes with user-level login credentials, when NCM processes a job it connects to the node using the credentials of the user who last edited the job. Otherwise, NCM connects to the node using settings in the device connection profile or the associated global connection profile.

Required NCM roles

Users with the NCM role of Engineer or Administrator can view, create, and manage all jobs in the list.

Users with other NCM roles cannot access jobs.

Create a job

1. Click My Dashboards > Network Configuration > Jobs.
2. Click Create New Job.
3. Name the job, but do not use special characters in the name.
4. Select a Job Type.
5. Specify when the job runs:
   - To run the job once or on a simple schedule, select Basic. Click the tab that identifies how frequently the job runs, and then specify the start time and (if needed) the day(s).
   - To create a more complex schedule, select Advanced and then use the five fields to create a CRON expression.
6. Add a comment if this job relates to a business rule, and click Next.
7. Select the nodes to target with this job, and click Next.
8. Select one or more notification options, and click Next.
<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Save Job Log</td>
<td>Select this option to save the results of the job to the database.</td>
</tr>
<tr>
<td></td>
<td>To view the job log, click the icon in the History column of the Jobs List.</td>
</tr>
<tr>
<td>Save Results to File</td>
<td>To save the results of the job to a file, specify the path and file name:</td>
</tr>
<tr>
<td></td>
<td>- NCM must have read and write privileges to the path.</td>
</tr>
<tr>
<td></td>
<td>- The path must exist. NCM can create the file but not the folders in the path.</td>
</tr>
<tr>
<td></td>
<td>- Supported file extensions include <code>.txt</code>, <code>.csv</code>, <code>.xls</code>, <code>.doc</code>, and <code>.htm</code>.</td>
</tr>
<tr>
<td></td>
<td>- You can use Date and Time or Global variables (also called macros) in the file name. For example, include <code>${DATE}</code> or <code>${DATETIME}</code> to prevent NCM from overwriting the file each time the job runs.</td>
</tr>
<tr>
<td></td>
<td>To save job results to a file on a network share, make sure the following requirements are met:</td>
</tr>
<tr>
<td></td>
<td>- The network share must be accessible by all polling engines.</td>
</tr>
<tr>
<td></td>
<td>- All NCM machines must be in the same domain as the machine where the network share is located.</td>
</tr>
<tr>
<td></td>
<td>- Credentials to give NCM write access to the network share must be specified:</td>
</tr>
<tr>
<td></td>
<td>a. Click Settings &gt; All Settings.</td>
</tr>
<tr>
<td></td>
<td>b. Under Product Specific Settings, click NCM Settings.</td>
</tr>
<tr>
<td></td>
<td>c. Under Advanced, click Advanced Settings.</td>
</tr>
<tr>
<td></td>
<td>d. Under Network Share Settings, select Use Custom Credentials to Give NCM Writer Access.</td>
</tr>
<tr>
<td></td>
<td>e. Enter the credentials, and click Validate Credentials.</td>
</tr>
<tr>
<td></td>
<td>If the validation fails, verify that the NCM-related account has Windows permissions for the network storage.</td>
</tr>
<tr>
<td>Option</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Email Results</td>
<td>If you click Email Results, the default email notification and SMTP server settings are populated. These settings can be overridden in each job.</td>
</tr>
<tr>
<td></td>
<td>You can use <a href="#">Date and Time</a> or <a href="#">Global</a> variables (also called macros) in the Subject.</td>
</tr>
</tbody>
</table>

9. Add details based on the specific job, and click Next.

For Export Configs jobs, the exported file name can contain [Date and Time](#), [Node](#), and [Global](#) variables (also called macros).

10. Review the settings for the job, and click Finish.

## Edit a job

1. Click My Dashboards > Network Configuration > Jobs.
2. Select the job and click Edit.
3. Update the job definition as needed.
4. Review the settings for the job, and click Finish.

## Enable a job

The enable operation applies to jobs that run according to a schedule. You must enable a job in NCM before you can start it. A job that is not enabled will not start.

1. Click My Dashboards > Network Configuration > Jobs.
2. Select a disabled job in the list.
3. Click Enable.

## Disable a job

The disable operation applies to jobs that run according to a schedule. You can delete any job in NCM that you no longer use.

Disable a job to suspend it, but not delete it.

1. Click My Dashboards > Network Configuration > Jobs.
2. Select a scheduled job in the list.
3. Click Disable.
Start a job

Though using a schedule is the most efficient way to manage jobs, you can manually start a job if necessary.

You can start any job that is enabled.

1. Click My Dashboards > Network Configuration > Jobs.
2. Select a job in the list.
3. If the job is disabled, click Enable.
4. Click Start Job.

Stop a job

Though using a schedule is the most efficient way to manage jobs, you can manually stop a job if necessary.

A job currently running shows the status: running.

1. Click My Dashboards > Network Configuration > Jobs.
2. Select a running job in the list.
3. Click Stop Job.

View job logs

View the job log to verify that a job was run as scheduled, or to view the history of the job.

1. Click My Dashboards > Network Configuration > Jobs.
2. Select the job in the list.
3. Sort by the Last Date Run column.
4. Click the page icon in the History column to view the log.

Delete a job

Permanently remove a job, rather than temporarily disable it.

1. Click My Dashboards > Network Configuration > Jobs.
2. Select a disabled job in the list.
3. Click Delete.
# NCM job types

When you [create an NCM job](#), the following job types are available. You can run these jobs either manually or with a schedule.

<table>
<thead>
<tr>
<th>Job type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execute Command Script on Devices</td>
<td>Executes a script on the selected nodes.</td>
</tr>
<tr>
<td>Upload Changes to Devices</td>
<td>Uploads a modified configuration file to the selected NCM nodes.</td>
</tr>
<tr>
<td>Export Configs</td>
<td>Exports the configuration files stored in the Orion database in order to create a backup copy. You can select which configs to export, the name of the file that is created, and the file location.</td>
</tr>
<tr>
<td>Perform Routine Database and Archive Maintenance</td>
<td>Purges old config data, job logs, completed approval requests, and completed transfer requests from the Orion database and server.</td>
</tr>
<tr>
<td>Update Inventory</td>
<td>Updates inventory details for specified NCM nodes.</td>
</tr>
<tr>
<td>Generate a Config Change Report</td>
<td>Compares the most recently downloaded config to one of the following:</td>
</tr>
<tr>
<td></td>
<td>● To a baseline or to the last downloaded config of the selected type.</td>
</tr>
<tr>
<td></td>
<td>If you select Baseline, NCM compares the most recently downloaded config to all baselines assigned to the associated device.</td>
</tr>
<tr>
<td></td>
<td>● To a config file from a specified date or date range.</td>
</tr>
<tr>
<td>Generate a Policy Report</td>
<td>Runs the selected policy reports to locate any policy violations and then caches the results. For more information about viewing information about policy violations, see <a href="#">Find and remediate policy violations</a>.</td>
</tr>
<tr>
<td>Purge Old Configs from Database</td>
<td>Removes old configs from the database as an NCM performance safeguard.</td>
</tr>
<tr>
<td>Download Configs from Devices</td>
<td>Downloads configuration files from the selected NCM devices.</td>
</tr>
<tr>
<td>Reboot Selected Devices</td>
<td>Reboots the selected NCM devices.</td>
</tr>
</tbody>
</table>
Create a CRON expression

When you are scheduling an NCM job, you can use a CRON expression to create an advanced schedule. In Unix computing, the CRON daemon performs all scheduled jobs. Each job is defined in a command line syntax that looks like this:

{Minutes: 0-59} [Hours: 0-23] {Day of Month: 1-31} {Month: 1-12} {Day of Week: 0-6 (Sunday to Saturday)} {Command to Execute}

NCM adopts this model in the resource of the advanced schedule type. The five input fields correspond to the five elements of CRON job syntax. Additionally, NCM supports these CRON operators:

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>Asterisk</td>
<td>The wildcard for the value all</td>
</tr>
<tr>
<td>-</td>
<td>Hyphen</td>
<td>Sets a range: #-#</td>
</tr>
<tr>
<td>,</td>
<td>Comma</td>
<td>Specifies terms out of series</td>
</tr>
</tbody>
</table>

Example

If you want a job to run twice a week every month at midnight on Sunday and Thursday morning, the values for the fields would be:

0 0 * * 0, 4 {Command}

This tells NCM to run the job at 0 minutes and 0 hours, on all valid days of month, in all valid months, on Sunday and Thursday.
Config change templates

Config change templates are used to make complex configuration changes on multiple devices. A single config change template can execute a set of CLI commands on different device types in your network.

See the following topics to learn more about working with config change templates:

- About NCM config change templates
- Parameters for NCM config change templates
- Commands for NCM config change templates
- SWIS entities used in config change templates
- Examples of NCM config change templates
- Create an NCM config change template
- Edit an NCM config change template
- Tag a config change template
- Delete an NCM config change template
- Import and export NCM config change templates
- Execute an NCM config change template

About NCM config change templates

Config change templates can be used to change the configurations on multiple devices across the network. Examples of routine changes you can expedite with config change templates include:

- Changing VLAN membership by device port
- Configuring device interfaces based on description
- Enabling IPSLA for VOIP implementations
- Managing NetFlow collection at the source devices

Config change templates are powerful, configurable scripts that allow for:

- Conditional logic (if/then/else statements)
- Control flow (for each interface, do this)
- String and value comparison
- Data access from the Orion Platform database

💡 If the config changes you need to make do not require advanced logic, you can use scripts to change configurations on multiple devices. Scripts can be executed manually or scheduled.
Before devices can be targeted with a config change template, the devices must have been **inventoried in NCM**. Running an inventory makes device data available to NCM through the SolarWinds Information Service (SWIS) API.

**Creating and using config change templates**

NCM ships a variety of config change templates, which you can execute to make common config changes. You can also create custom config change templates.

Two types of NCM users work with config change templates:

1. The **template creator** defines the config change template. The config change template definition includes:
   - Variables that describe the template's purpose and prompt the template user for values when the config change template is executed. For example, if the config change template changes a password, the definition would include a variable to prompt the user for the new password.
   - A script that generates a series of CLI commands. When the config change template is executed, these CLI commands run on each targeted device.

   The template creator must know the basics of writing a script that uses commands, variables, and logical structures. Creating or editing a config change template requires the NCM role of Administrator or Engineer.

2. The **template user** executes the config change template. At execution, a run-time wizard prompts the user to:
   - Select the devices that the config change template targets.
   - Provide values for the variables defined in the template.

   Based on this input, the config change template creates a series of CLI commands and runs them on each targeted device.

   Executing a config change template requires the NCM role of Administrator, Engineer, or WebUploader.
For more information about creating new config change templates or modifying existing templates, see:

- Parameters for NCM config change templates
- Commands for NCM config change templates
- SWIS entities used in config change templates
- Examples of NCM config change templates
- Create an NCM config change template
- Edit an NCM config change template

To run a config change template, see Execute an NCM config change template.

**Parameters for NCM config change templates**

The parameters of a config change template define and label the variables that hold information about the template.

The script of every config change template includes at least five parameters. Only `PARAMETER_LABEL` and `PARAMETER_DESCRIPTION` can occur more than once in a template. The script includes one pair of `PARAMETER_LABEL` and `PARAMETER_DESCRIPTION` variables for each value that the user must enter when the config change template runs.

**Required Parameters**

**CHANGE_TEMPLATE_DESCRIPTION**

This parameter appears at the top of the script and briefly explains the purpose of the template. It does not have any associated variables, and it is not exposed in the run-time wizard.
This parameter holds the tags that NCM uses to provide grouping options in the Config Change Template resource. It does not have any associated variables and is not exposed in the run-time wizard.

**PLATFORM DESCRIPTION**

This parameter defines the type of NCM device for which the template is designed. It does not have any associated variables and is not exposed in the run-time wizard.

**PARAMETER_LABEL @<variable_name>**

The **PARAMETER_LABEL** parameters define the input data that the template requires. They make the variable available to the user at run-time, and they provide metadata (the variable name) so that the user knows what the variable is holding a place for.

For example, **PARAMETER_LABEL** is used in every template with @ContextNode. The user sets the value by selecting the NCM nodes that are targeted for config change. An instance of the parameter appears in a script as follows:

```
.PARAMETER_LABEL @ContextNode NCM Node
```

In this case, NCM Node is the actual label that appears under the field where the NCM nodes are selected in the template’s run-time wizard.

A config change template may have as many instances of **PARAMETER_LABEL** as needed to support the user input required for the template.

**PARAMETER_DESCRIPTION @<variable_name>**

This parameter holds the explanatory text for an input field. It always appears after the corresponding **PARAMETER_LABEL** and includes the same variable name.

For example, **.PARAMETER_DESCRIPTION @ContextNode NCM Node** corresponds to the input field labeled NCM Node. The value might be something like:

```
The NCM nodes the template will operate on. Target nodes are selected during the first part of the wizard and cannot be changed when defining values of variables.
```

### Optional Parameter

**PARAMETER_DISPLAY_TYPE**

This parameter creates a list of options. The format for using this parameter is as follows. The vertical bar character (|) divides the items in the list.

```
PARAMETER_DISPLAY_TYPE @VariableName

Listbox:1=String1|2=String2|3=String3
```
Commands for NCM config change templates

There are two commands in a config change template:

- **script**: Defines the input type for each variable defined by the template parameters.
- **CLI**: Includes the arguments and logical operations needed to produce a set of CLI commands and execute those commands against each NCM node targeted for a specific config change.

**Script Command**

The `script{}` command declares the input type for every variable defined by the template parameters. The form of the script command is:

```
script script_name (  
data_type @variable  
data_type @variable  
data_type @variable  )
```

The `data_type` can be `int` (integer), `string`, or `swis.entity` where `swis.entity` is a SolarWinds Information Service (SWIS) entity (for example, NCM.Nodes). For descriptions of available entities, see [SWIS entities used in config change templates](#).

**Cisco Example**

```
script ConfigureVLANmembershipCiscoIOS (  	NCM.Nodes @ContextNode,  	NCM.Interfaces[] @TargetPorts,  	NCM.VLANs[] @VlansToRemove,  	NCM.VLANs[] @VlanToAssign  )
```

In this example, the four variables that were introduced in the parameter section of the template with an instance of `PARAMETER_LABEL` are each given a specific SolarWinds Information Service entity data type:

- `@ContextNode` is determined with data from the NCM.Nodes entity in the database.
- `@TargetPorts` is determined with data from the NCM.Interfaces entity.
- `@VlansToRemove` and `@VlanToAssign` are determined with data from the NCM.VLANs entity.

Any variable that references an NCM object that NCM knows through device inventory must take a SolarWinds Information Service entity as its data type. In this case, the four variables work with data that NCM captured and stored in the database through the device inventory process. If you attempt to assign a string instead of a SWIS entity in such cases, NCM fails to correctly parse your script.
CLI command

CLI{} defines a specific CLI command that NCM issues on a target device when the config change template is executed. Its purpose is to create a command line statement that NCM can execute directly on the command line of NCM nodes targeted for the template's config changes.

The config change template creator creates a CLI command by including its arguments wrapped by curly brackets { }. At run time, NCM parses any variables contained within CLI{}. Often a CLI command is as simple as the command you would type directly on the command line of an NCM device.

Cisco Example

To enter config mode on Cisco IOS devices, type configure terminal. In your config change template script, add the command as follows:

```plaintext
CLI
{
  configure terminal
}
```

NCM parses the argument of the CLI{} command by passing through the string itself (configure terminal) as a command to execute against each targeted NCM node at template run time:

Cisco Example with Variables

```plaintext
script ConfigureVLANmembershipCiscoIOS (  
  NCM.Nodes @ContextNode,  
  NCM.Interfaces[] @TargetPorts,  
  NCM.VLANs[] @VlansToRemove,  
  NCM.VLANs[] @VlanToAssign
)  
{
  CLI
  {
    vlan database vlan @vlaniddescription @vlandesc exit
  }
}
```

This example shows a CLI statement with variables to specify VLAN properties while using the vlan database command line editor. For purposes of demonstration, we assume that:

- @vlanid = 1
- @vlandesc = Local-Office

At run time, NCM parses the CLI{} command as:
vlan database vlan 1 description Local-Office exit

If this were all that is included in the Change VLAN Membership on Ports Cisco IOS template, then the config change result would be to set the description of vlan 1 to Local-Office on all NCM nodes selected as targets.

This config change template, however, actually changes the VLANs associated with targeted NCM node ports. For that we need to introduce advanced CLI{} command logic.

Advanced CLI command logic

The scripting framework for change config templates allows you to create CLI{} command arguments that include foreach loops, if/else conditional operations, and functions for manipulating string patterns.

Foreach Loops

A foreach statement iterates through an array of items based on a SWIS entity data type. Foreach statements use the following pattern:

```
foreach (@ItemVariable in @EntityArrayVariable)
```

A primary purpose of a foreach loop is to allow the template user to select multiple NCM objects for config change. The loop instructs NCM to perform the same config change on all items in scope as determined by the SWIS entity in the database and delimited at run time by the template user's selections in the template wizard.

Cisco Example

```
foreach (@portItem in @TargetPorts)
{
    CLI
    {
        interface @portItem.InterfaceDescription
    }
}
```

The foreach statement creates a set that contains two related variables: @portItem and @TargetPorts.

The @TargetPorts variable holds an array of objects with the data type of an SWIS entity called NCM.Interfaces[]. The array will be a set of interfaces on NCM nodes.

The @TargetPorts variable is associated with the PARAMETER_LABELSelect port(s) and the template user selects one or more ports at run time. The template user determines the set of interfaces to fill the array NCM.Interfaces[], and the template will perform VLAN membership config changes on each interface in that array.
@portItem is a dynamic variable that the loop uses during its iterating to hold the value of the current interface from the array represented by @TargetPorts.

The foreach loop format is fixed and NCM expects it to include the dynamic variable.

The user interacts with this template wizard screen:

Click Select Interfaces List to load a tree that displays available interfaces and NCM nodes previously selected in the wizard.

Conditional Statements

Conditional logic in a config change template script uses an if/else pattern to define two branches of possible action, enclosing specific conditions within parentheses. Within each branch of the conditional pattern are CLI{} commands to execute if that branch meets the specific conditions.

Here is the basic structure:

```plaintext
if (condition is true)
CLI
{
  execute commands
}
else
CLI
{
  Execute other commands
}
```

The else section is optional. If you omit it, and the if condition is false, NCM excludes the relevant CLI{} commands from the template output.
Operators

Use any of the following operators to specify a parenthetical condition. Use single quotes around string values.

<table>
<thead>
<tr>
<th>Operator</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>==</code></td>
<td>Is Equal To</td>
</tr>
<tr>
<td><code>&gt;</code></td>
<td>Is Greater Than</td>
</tr>
<tr>
<td><code>&gt;=</code></td>
<td>Is Greater Than or Equal To</td>
</tr>
<tr>
<td><code>&lt;</code></td>
<td>Is Less Than</td>
</tr>
<tr>
<td><code>&lt;=</code></td>
<td>Is Less Than or Equal To</td>
</tr>
<tr>
<td><code>!=</code></td>
<td>Is Not Equal To</td>
</tr>
<tr>
<td>Contains</td>
<td>'string'</td>
</tr>
<tr>
<td>containsExact</td>
<td>'case sensitive string'</td>
</tr>
<tr>
<td>startsWith</td>
<td>'string'</td>
</tr>
<tr>
<td>startsWithExact</td>
<td>'case sensitive string'</td>
</tr>
<tr>
<td>endsWith</td>
<td>'string'</td>
</tr>
<tr>
<td>endsWithExact</td>
<td>'case sensitive string'</td>
</tr>
</tbody>
</table>

Cisco Example

Add conditional logic in the foreach loop to prevent errors that may occur if the user accidentally selects an incorrect interface (for example, the loopback address).

```bash
foreach @portItem in @TargetPorts) {
    if (@PortItem.InterfaceDescription != 'Loop0') {
        CLI {
            interface @portItem.InterfaceDescription
        }
    }
}
```

If the template encounters the loopback interface, it does nothing and passes on to the next interface. This code prevents damage from template user error.
Manipulating Strings

Five functions for manipulating strings are a scripting resource that you can use (for example) to manage ACL config changes for network firewalls, in which a config change template needs to iterate through a predictably variable set of IP addresses.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>Declaration</th>
<th>Variable Definitions</th>
</tr>
</thead>
</table>
| Substring  | Specify a starting point within a string and the length from the starting point that you want to capture for manipulation. | string Substring (string str, int startIndex, int length) | • str is the full string from which the substring comes  
• startIndex marks the position where the substring begins  
• length is the number of characters that the substring includes |
| StrLength  | Return the length of a string.                                   | In StrLength (string str)          | • str is the user-input string whose length is used as the integer value            |
| IndexOf    | Find the number of characters in a string.                      | int IndexOf (string str, string search) | • str is a string to search on  
• search is a user-input string NCM uses to find the numerical value of the string being searched |
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>Declaration</th>
<th>Variable Definitions</th>
</tr>
</thead>
</table>
| SetOctet     | Replace an octet within an IP address | string SetOctet (string ipAddress, int octetPosition, string octet) | - ipAddress is the IP address  
- octetPosition marks the position where the target octet begins  
- octet is the new value of the target octet |
| GetOctet     | Retrieve an octet from a user-specified IP address and octet position. | string GetOctet (string ipAddress, int octetPosition) | - ipAddress is a user-input IP address  
- octetPosition is the user-input value for the place where the function finds the beginning of the octet to get |

Example 1: Manipulating a String

```script
script IPshuffle(string @str, string @search)
{
    int @length = strlength(@str)
    int @startIndex = indexof(@str, @search)
    int @substringLength = @length - @startIndex
    @res = substring(@strA, @startIndex, @substringLength)

    CLI
    {
        @res
    }
}
```
The user enters ABCDEF for the @str variable and CD for the @search variable in the template’s wizard. Based on those values, the script does the following:

1. Uses ABCDEF in the strlen function to give a value of 6 to a variable called @length.
2. Uses CD as the substring of ABCDEF to set a value 2 for the variable called @startIndex.
3. Subtracts 2 (@startIndex) from 6 (@length) to determine the value of @substringLength as 4.
4. Takes the original string ABCDEF and calculates a result (@res) using @startIndex to count in two positions and @substringlength to count four positions from the start index.
5. Outputs CDEF as the result.

Example 2: Changing an Access Control List

This example creates a block of Access Control List (ACL) instructions that predictably vary the value of a specific octet within an IP address. The instructions conform to the pattern 10.10.@id.10, where the value of @id is determined by user input.

The user enters 10.10.10.10 as the value of @ipaddress in the config change template’s run-time wizard. The user enters 1, 22, and 222 for the @indexes variable declared in the script command.

```plaintext
.PARAMETER_LABEL @ipaddress
   IP address
.PARAMETER_DESCRIPTION @ipaddress
   Enter an IP address

.PARAMETER_LABEL @indexes
   Octets
.PARAMETER_DESCRIPTION @indexes
   Enter a pattern of octet replacements. Separate numbers with a comma.
*/

script ACLChanges(string @ipaddress, int[] @indexes)
{
   string @ipnew

   foreach(@id in @indexes)
   {
      @ipnew = setoctet(@ipaddress,3,@id)
      CLI
   }
```
The script uses the `SetOctet` function to determine the value of an `@ipnew` variable. `SetOctet` is defined to take the user-input IP address and create a new IP address by iteratively replacing the third octet with user-input values. For each new IP address, the script produces a command to create outgoing UDP transmission access through port 2055:

- `Allow 10.10.1.10 out Allow 10.10.1.10 UDP 2055 OUT`
- `Allow 10.10.22.10 out Allow 10.10.22.10 UDP 2055 OUT`
- `Allow 10.10.222.10 out Allow 10.10.222.10 UDP 2055 OUT`

**Example 3: Managing an Access Control List for Multiple Routers**

In this example, a config change template generates a block of ACL instructions for a router in a store. We create an ACL block of instructions for this device that varies based on a portion of the device's IP address.

If the store has four routers, 10.1.1.1, 10.1.4.1, 10.1.6.1, and 10.1.10.1, the template script generates an ACL block that appears this way on the selected router (10.1.1.1):

- `Allow 10.1.2.0/24 out Allow 10.1.2.4 UDP 2055 OUT`
- `Allow 10.1.4.0/24 out Allow 10.1.4.4 UDP 2055 OUT`
- `Allow 10.1.6.0/24 out Allow 10.1.6.4 UDP 2055 OUT`
- `Allow 10.1.10.0/24 out Allow 10.1.10.4 UDP 2055 OUT`

Here is the script that produces the output:

```plaintext
script OpenACLs(NCM.Nodes @ContextNode, string[] @IpRouters)
{
    foreach(@ipRouter in @IpRouters)
    {
        # Code to create ACL instructions
    }
}
```
This script does the following:

- Uses a foreach loop to go through a user-input series of router IP addresses.
- Uses the GetOctet function to focus the third octet of the current router IP address.
- Uses the SetOctet function to create a new IP address as a value for @ipnew.
- Creates a CLI { } command that will execute Allow operations for each of the selected routers.

The result is a set of Allow commands that open access in the ACL so that the router 10.1.1.1 can send OUT traffic via UDP on port 2055 to 10.1.4.1, 10.1.6.1, and 10.1.10.1.

Here are the parameters for this config change template. The template user selects the router on which to make ACL changes and inputs the target router IP address through this template:

```
.string @octet = getoctet(@IpRouter,3)
.string @ipnew = setoctet(@ContextNode,3,@octet)
CLI
{
   Allow @ipnew out
   Allow @ipnew UDP 2055 OUT
}
```

Tips

Including special symbols in CLI commands

If you need to include a special symbol, such as a pipe, in a CLI command, you cannot do it directly. For example, the following would break the script:

```
show clock | append disk0:show tech
```

Instead, define a string variable with the symbol as its value, and then place the variable in the command:
Including variables in CLI commands

To keep config change templates as device-independent as possible, you can include command variables defined in device templates. You can also include node variables (also called macros) to reference attributes of each node. To include these, define a string variable in the CLI command with the command or node variable as its value.

The following example uses the node variable `${StorageAddress}` to reference the IP address of the TFTP server. Because the IP address is not hard-coded in the script, the script does not break if the IP address changes.

```plaintext
script BaseChangeTemplate(NCM.Nodes @ContextNode)
{
  string @myTFTPImage='@myTFTPImage' + '/image.bin'
  CLI
  {
    copy tftp://@myTFTPImage flash
    ...
  }
}
```

The resulting script is:

```
  copy tftp://@myTFTPImage/image.bin flash
```

**SWIS entities used in config change templates**

The following tables document the SolarWinds Information Service (SWIS) entities and properties that you can use in NCM config change templates.
### NCM.ArpTables

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>InterfaceIndex</td>
<td>System.Int32</td>
<td>The interface on which this entry’s equivalence is effective. The interface identified by a particular value of this index is the same interface as identified by the same value of RFC 1573’s ifIndex.</td>
</tr>
<tr>
<td>InterfaceID</td>
<td>System.String</td>
<td>A unique GUID ID from ncm.Interfaces table.</td>
</tr>
<tr>
<td>MAC</td>
<td>System.String</td>
<td>The media dependent `physical' address.</td>
</tr>
<tr>
<td>IPAddress</td>
<td>System.String</td>
<td>The IP address corresponding to the media dependent physical address.</td>
</tr>
<tr>
<td>IPSort</td>
<td>System.Double</td>
<td>A list of IP addresses sorted with octet markers (dots) omitted.</td>
</tr>
<tr>
<td>Source</td>
<td>System.String</td>
<td>The type of IP address associated with an ARP operation and media dependent address. Possible Values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Other (1)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Invalid (2)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Dynamic (3)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Static (4)</td>
</tr>
<tr>
<td>RDNSLookup</td>
<td>System.String</td>
<td>Result of DNS lookup on IPAddress.</td>
</tr>
<tr>
<td>NodeID</td>
<td>System.String</td>
<td>A SWIS-generated unique identifier of a network node in the current inventory. (Instances of this property recur in this table according to the number of interfaces for which ARP data is reported.)</td>
</tr>
</tbody>
</table>

Setting this object to the value invalid (2) has the effect of invalidating the corresponding entry in the ipNetToMediaTable. That is, it effectively disassociates the interface identified with said entry from the mapping identified with said entry. It is an implementation specific matter as to whether the agent removes an Invalidated entry from the table. Accordingly, management stations must be prepared to receive tabular information from agents that corresponds to entries not currently in use. Proper interpretation of such entries requires examination of the relevant ipNetToMediaType object.
<table>
<thead>
<tr>
<th>Property Name</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LastDiscovery</td>
<td>System.DateTime</td>
<td>A SWIS-generated date and time marker for when NCM last discovered the device during inventory.</td>
</tr>
<tr>
<td>FirstDiscovery</td>
<td>System.DateTime</td>
<td>A SWIS-generated date and time marker for when NCM first discovered the device during inventory.</td>
</tr>
</tbody>
</table>

**NCM.ARPTables entity relationships**

<table>
<thead>
<tr>
<th>Type</th>
<th>Entity</th>
<th>Joined Data Entity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node</td>
<td>NCM.Nodes</td>
<td>NCM.NodeHostsArpTables (System.Hosting)</td>
</tr>
</tbody>
</table>

**NCM.BridgePorts**

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>InterfaceIndex</td>
<td>System.Int32</td>
<td>The value of the instance of the ifIndex object, defined in MIB-II, for the interface corresponding to this port.</td>
</tr>
<tr>
<td>Port</td>
<td>System.Int32</td>
<td>The port number of the port for which this entry contains bridge management information.</td>
</tr>
<tr>
<td>SpanningTreeEnabled</td>
<td>System.String</td>
<td>The enabled/disabled status of the port. Possible Values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Enabled (1)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Disabled (2)</td>
</tr>
<tr>
<td>SpanningTreeState</td>
<td>System.String</td>
<td>The port’s current state as defined by application of the Spanning Tree Protocol. This state controls what action a port takes on reception of a frame. If the bridge has detected a port that is malfunctioning, it places that port into the broken(6) state. For ports that are disabled (see dot1dStpPortEnable), this object has a value of disabled(1).</td>
</tr>
<tr>
<td>Property Name</td>
<td>Datatype</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>----------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| VlanType      | System.String  | The type of VLAN membership assigned to this port. A port with static VLAN membership is assigned to a single VLAN directly. A port with dynamic membership is assigned a single VLAN based on content of packets received on the port and through VQP queries to VMPS. A port with multiple VLAN membership may be assigned to one or more VLANs directly. A static or dynamic port membership is specified by the value of vmVlan. A multiVlan port membership is specified by the value of vmVlans. Possible Values:  
  - Static(1)  
  - Dynamic(2)  
  - MultiVlan(3)  

<table>
<thead>
<tr>
<th>VLANID</th>
<th>System.Int32</th>
<th>The VLAN id of the VLAN the port is assigned to when vmVlanType is set to static or dynamic. This object is not instantiated if not applicable. The value may be 0 if the port is not assigned to a VLAN.</th>
</tr>
</thead>
</table>
| PortStatus    | System.String  | An indication of the current VLAN status of the port. A status of inactive(1) indicates that a dynamic port does not yet have a VLAN assigned, or a port is assigned to a VLAN that is currently not active. A status of active(2) indicates that the currently assigned VLAN is active. A status of shutdown(3) indicates that the port has been disabled as a result of VQP shutdown response. Possible Values:  
  - inactive(1)  
  - active(2)  
  - shutdown(3)  

| NodeID        | System.String  | A SWIS-generated unique identifier of a network node in the current inventory. (Instances of this property recur in this table according to the number of interfaces for which bridge port data is reported.)                                                                                                                                                                                                                                                                                                                                                   |
### Property Name | Datatype | Description
--- | --- | ---
LastDiscovery | System.DateTime | A SWIS-generated date and time marker for when NCM last discovered the device during inventory.
FirstDiscovery | System.DateTime | A SWIS-generated date and time marker for when NCM first discovered the device during inventory.

### NCM.BridgePorts entity relationships

<table>
<thead>
<tr>
<th>Type</th>
<th>Entity</th>
<th>Joined Data Entity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node</td>
<td>NCM.Nodes</td>
<td>NCM.NodeHostsBridgePorts (System.Hosting)</td>
</tr>
</tbody>
</table>

### NCM.CatalystCards

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CardIndex</td>
<td>System.Int32</td>
<td>A unique value for each module within the chassis.</td>
</tr>
<tr>
<td>Property Name</td>
<td>Datatype</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>----------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>CardType</td>
<td>System.Int32</td>
<td>The type of module.</td>
</tr>
</tbody>
</table>

Possible Values:

(notdefined(0), version1(1), version2(2), version3(3), version4(4), version5(5), version6(6), version7(7), version8(8), version9(9), version10(10), vi2(11), vi4(12), vi30(13), s1b(14), sa2(15), as16(16), new8as(17), lsa(18), fx2(19), fxo2(20), em2(21), fx4(22), fxo4(23), em4(24), s3ab(25), e1vi(26), am12(27), am6(28), ndec(29), newsa2(30), aux(31), console(32), sic-wan(33), sic-1fe(34), sic-1sa(35), sic-3as(36), sic-1e1(37), sic-1t1(38), sic-1bu(39), sic-2bu(40), sic-1bs(41), sic-2bs(42), sic-1am(43), sic-2am(44), sic-1em(45), sic-2em(46), sic-1fxs(47), sic-2fxs(48), sic-1fxo(49), sic-2fxo(50), fcm6(51), sa8(52), t11(53), t12(54), t14(55), t1vi(56), fcm4(57), fcm2(58), rtb21ce3(59), am6(60), ame12(61), wsx5162(62), e11-f(65), e12-f(66), e14-f(67), t11-f(68), t12-f(69), t14-f(70), e11-f-17(71), t11-f-17(72), rtb21ct3(73), atmads1(74), atmads2(75), atm155m(76), ase8(77), ase16(78), sae4(79), sae2(80), wsx5012a(81), wsx5167(82), wsx5239(83), wsx5168(84), wsx5305(85), wsx5550(87), wsf5541(88), atmshdsl1(90), atmshdsl2(91), atmshdsl4(92), atm25m(93), atm3(95), xdsl-fec(96), xdsl-ads1(97), xdsl-gshdsl(98), xdsl-bri(99), xdsl-scc(100), ge1(101), pos155m(102), cpos(103), fe1op(104), sae8(105), atm155mm(106), atm155m-sm(107), atm155msm(108), fe1op-sfx(109), fe1op-mfx(110), cpos-t1(111), ge1-op(112), ge2-op(113), ge2(114), fix-1wan(115), fix-1sae(116), cavium(117), sic-1eth(118), atm1adsl(119), atm2adsl(120), fix-e11(121), fx11(122), e18-75(123), e18-120(124), t18(125), sic-1vfx(126), sic-1vfxo(127), sic-2vfx(128), sic-2vfxo(129), xdsl-fec-new(130), xdsl-sa(131), bk4(132), ima-8e175(133), ima-8e1120(134), ima-4e175(135), ima-4e1120(136), ima-8t1(137), ima-4t1(138), sic-1t1f(139), sic-1e1f(140), fe4(149), atm1shdsl4wire(151), atm1ma4shdsl(152), ls4(153), ls8(154), ls16(155). sic-adls2plus-isdn(156). sic-adls2plus-pots(157), ft3(158), ce32(159), bsv2(160), bsv4(161). rpu(162). erpu(163). ssl(164), nsa(165), wsx6ksup12ge(200), wsx6408gbic(201). wsx6224mmmt(202). wsx6248rj45(203). wsx6248tel(204). wsx6302msm(206), wsf6kmsfc(207), wsx6024flmt(208).
<table>
<thead>
<tr>
<th>Property Name</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>wsx6101oc12mmf(209), wsx6101oc12smf(210), wsx6416gemt(211), wsx61822pa(212), osm2oc12AtmMM(213),</td>
<td>wsx61201oc12mmf(214), osmoc12AtmSl(214), osm4oc12PosMM(216), osm4oc12PosSl(217), osm4oc12PosSL(218), wsx6ksup1a2ge(219), fe18-75(220). fe18-120(221). ft18(222), cf-card(223), bsv2-v2(224), e1vi1-v2(225), e1vi2(226), t1vi1-v2(227), t1vi2(228). osm(229), sd707(230), dm-epri(231), dm-tpri(232), erpu-h(233), ws6kmsfc2(234), wsx6244mmmt(235), wsx6064rj45(236), wsx6ksup22ge(237), wsx6324sm(238), wsx6516gbic(239), osm4geWanGbic(240), osmoc48PosSS(241), osmoc48PosSI(242), osmoc48PosSL(243), wsx6381ids(244), wsc6500sfm(245), osm16oc3PosMM(246), osm16oc3PosSI(247), osm16oc3PosSL(248), osmoc12PosMM(249), osmoc12PosSl(250), osmoc12PosSL(251), wsx6064rj45(252), osm8oc3PosMM(253), osm8oc3PosSl(254), osm8oc3PosSL(255), wsx6548rj45(256), wsx6524mmmt(257), wsx6066slbapc(260), wsx6516getx(261), osmoc48OneDptSS(265), osmoc48OneDptSl(266), osmoc48OneDptSSD(268), osmoc48OneDptSlD(269), osmoc48OneDptSlDual(270), wsx6816gbic(271). osm4choc12T3MM(272), osm4choc12T3Sl(273), osm8choc12T3MM(274), osm8choc12T3Sl(275), osm1choc48T3SS(276), osm2choc48T3SS(277). wsx6500sfm2(278). osm1choc48T3Sl(279), osm2choc48T3Sl(280), wsx6348rj21(281), wsx6546rj21(282), wsSvcCmm(284), wsx650110gex4(285), osmoc48PosSI(286), osmoc48PosMM(289), wsvCldsm(290), wsvCm(291), wsvCfwm(292), wsvCcs(293). wsvCsl(294). osm8choc3DS0SI(295), osm4choc3DS0SI(296), osm1choc12T1Sl(297). wsx4012(300). wsx4148rj(301). wsx4232gb(302). wsx4306gb(303), wsx4418gb(304), wsx44162gbtx(305), wsx4912gb(306), wsx2948gb(307), wsx2948(308), wsx4912(310), wsx4424sxmt(311). wsx4232rjxx(312). wsx4148rj(313), wsx4124fxmt(317), wsx2901(318). wsx4232l3(319), wsx4604gyw(320), wsx44122gbtx(321), wsx2980(322), wsx2980rj(323), wsx2980gb(324). wsx4019(325). wsx4148rj45v(326), wsx4424gb(330), wsx4148fxmt(331), wsx4448gb(332), wsx4448gb(334), wsx4148lxmt(337), wsx4548gb(339). wsx4548gb(340). wsx4248rj45v(341), wsx4302gb(342), wsx4248rj45v(343), wsx4296ggetx(345), wsx4296ggetxvbrj</td>
<td></td>
</tr>
<tr>
<td>Property Name</td>
<td>Datatype</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>----------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>CardName</td>
<td>System.String</td>
<td>A descriptive string used by the network administrator to name the module.</td>
</tr>
<tr>
<td>ModuleModel</td>
<td>System.String</td>
<td>The manufacturer’s model number for the module.</td>
</tr>
<tr>
<td>CardSerial</td>
<td>System.String</td>
<td>The serial number of the module. This MIB object returns the module serial number for any module that uses either a numeric or an alphanumeric serial number.</td>
</tr>
<tr>
<td>HWVersion</td>
<td>System.String</td>
<td>The hardware version of the module.</td>
</tr>
<tr>
<td>FWVersion</td>
<td>System.String</td>
<td>The firmware version of the module.</td>
</tr>
<tr>
<td>SWVersion</td>
<td>System.String</td>
<td>The software version of the module.</td>
</tr>
<tr>
<td>Slot</td>
<td>System.Int32</td>
<td>This value is determined by the chassis slot number where the module is located. Valid entries are 1 to the value of chassisNumSlots.</td>
</tr>
<tr>
<td>Parent</td>
<td>System.Int32</td>
<td>The value of the instance of the entPhysicalIndex object, defined in ENTITY-MIB, for the entity physical index corresponding to this module.</td>
</tr>
<tr>
<td>OperStatus</td>
<td>System.String</td>
<td>The operational status of the module. If the status is not ok, the value of moduleTestResult gives more detailed information about the module’s failure condition(s).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Possible Values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- other(1)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- ok(2)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- minorFault(3)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- majorFault(4)</td>
</tr>
<tr>
<td>SlotsOnCard</td>
<td>System.Int32</td>
<td>The number of ports supported by the module.</td>
</tr>
<tr>
<td>NodeID</td>
<td>System.String</td>
<td>A SWIS-generated unique identifier of a network node in the current inventory. (Instances of this property recur in this table according to the number of interfaces for which card data is reported.)</td>
</tr>
<tr>
<td>LastDiscovery</td>
<td>System.DateTime</td>
<td>A SWIS-generated date and time marker for when NCM last discovered the device during inventory.</td>
</tr>
<tr>
<td>Property Name</td>
<td>Datatype</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>FirstDiscovery</td>
<td>System.DateTime</td>
<td>A SWIS-generated date and time marker for when NCM first discovered the device during inventory.</td>
</tr>
</tbody>
</table>

**NCM.CatalystCards entity relationships**

<table>
<thead>
<tr>
<th>Type</th>
<th>Entity</th>
<th>Joined Data Entity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node</td>
<td>NCM.Nodes</td>
<td>NCM.NodeHostsCatalystCards (System.Hosting)</td>
</tr>
</tbody>
</table>

**NCM.CiscoCards**

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CardIndex</td>
<td>System.Int32</td>
<td>Index into cardTable (not physical chassis slot number).</td>
</tr>
<tr>
<td>CardType</td>
<td>System.Int32</td>
<td>Functional type of this card (integer value).</td>
</tr>
<tr>
<td>CardName</td>
<td>System.String</td>
<td>Functional type of this card (parsed from type name value).</td>
</tr>
<tr>
<td>CardDescr</td>
<td>System.String</td>
<td>Text description of this card.</td>
</tr>
<tr>
<td>CardSerial</td>
<td>System.String</td>
<td>The serial number of this card, or zero if unavailable.</td>
</tr>
<tr>
<td>HWVersion</td>
<td>System.String</td>
<td>Hardware revision level of this card, or an empty string if unavailable.</td>
</tr>
<tr>
<td>SWVersion</td>
<td>System.String</td>
<td>Version of the firmware or microcode installed on this card, or an empty string if unavailable.</td>
</tr>
<tr>
<td>Slot</td>
<td>System.Int32</td>
<td>Number of slots on this card, or 0 if no slots or not applicable, or -1 if not determinable.</td>
</tr>
<tr>
<td>Parent</td>
<td>System.Int32</td>
<td>cardIndex of the parent card that directly contains this card, or 0 if contained by the chassis, or -1 if not applicable or determinable.</td>
</tr>
<tr>
<td>Property Name</td>
<td>Datatype</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>---------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| OperStatus         | System.String       | The operational status of the card. cardOperStatus is up when a card is recognized by the device and is enabled for operation. cardOperStatus is down if the card is not recognized by the device, or if it is not enabled for operation. cardOperStatus is standby if the card is enabled and acting as a standby slave. Possible Values:  
  - not-specified(1)  
  - up (2)  
  - down (3)  
  - standby (4)  
  - standbyMaster (5)  
  - activeMaster (6)  
  - outOfService (7)  
  - masterBooting(8)  
  - activeMasterBooting(9)  
  - standbyMasterBooting(10)  
  - slaveBooting(11) |
| SlotsOnCard        | System.Int32        | Number of slots on this card, or 0 if no slots or not applicable, or -1 if not determinable.                                                                                                               |
| NodeID             | System.String       | A SWIS-generated unique identifier of a network node in the current inventory. (Instances of this property recur in this table according to the number of interfaces for which card data is reported.) |
| LastDiscovery      | System.DateTime     | A SWIS-generated date and time marker for when NCM last discovered the device during inventory.                                                                                                           |
| FirstDiscovery     | System.DateTime     | A SWIS-generated date and time marker for when NCM first discovered the device during inventory.                                                                                                          |

NCM.CiscoCards entity relationships

<table>
<thead>
<tr>
<th>Type</th>
<th>Entity</th>
<th>Joined Data Entity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node</td>
<td>NCM.Nodes</td>
<td>NCM.NodeHostsCiscoCards (System.Hosting)</td>
</tr>
<tr>
<td>Property Name</td>
<td>Datatype</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ifIndex</td>
<td>System.Int32</td>
<td>An indication of the type of address contained in the corresponding instance of cdpCacheAddress (parse just ifIndex from value). For example 1,2,3.</td>
</tr>
<tr>
<td>CDPIndex</td>
<td>System.String</td>
<td>An indication of the type of address contained in the corresponding instance of cdpCacheAddress (full value). For example 1.6, 2.108, 2.3.</td>
</tr>
<tr>
<td>RemoteDevice</td>
<td>System.String</td>
<td>The Device-ID string as reported in the most recent CDP message. A zero-length string indicates no Device-ID field (TLV) was reported in the most recent CDP message.</td>
</tr>
<tr>
<td>RemoteIPAddress</td>
<td>System.String</td>
<td>The (first) network-layer address of the device’s SNMP-agent as reported in the most recent CDP message. For example, if the corresponding instance of cacheAddressType had the value 'ip(1)', then this object would be an IP address.</td>
</tr>
<tr>
<td>RemoteVersion</td>
<td>System.String</td>
<td>The Version string as reported in the most recent CDP message. A zero-length string indicates no Version field (TLV) was reported in the most recent CDP message.</td>
</tr>
<tr>
<td>RemotePort</td>
<td>System.String</td>
<td>The Port-ID string as reported in the most recent CDP message. This is typically the value of the ifName object (for example, 'Ethernet0'). A zero-length string indicates no Port-ID field (TLV) was reported in the most recent CDP message.</td>
</tr>
<tr>
<td>RemoteCapability</td>
<td>System.String</td>
<td>The Device’s Functional Capabilities as reported in the most recent CDP message. For latest set of specific values, see the latest version of the CDP specification. A zero-length string indicates no Capabilities field (TLV) was reported in the most recent CDP message.</td>
</tr>
<tr>
<td>RemotePlatform</td>
<td>System.String</td>
<td>The Device’s Hardware Platform as reported in the most recent CDP message. A zero-length string indicates that no Platform field (TLV) was reported in the most recent CDP message.</td>
</tr>
<tr>
<td>Property Name</td>
<td>Datatype</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>---------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>RemoteDuplex</td>
<td>System.String</td>
<td>The remote device’s interface’s duplex mode, as reported in the most recent CDP message. The value unknown(1) indicates no duplex mode field (TLV) was reported in the most recent CDP message. Possible Values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- unknown(1)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- halfduplex(2)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- fullduplex(3)</td>
</tr>
<tr>
<td>RemoteNativeVLAN</td>
<td>System.Int32</td>
<td>The remote device’s interface's native VLAN, as reported in the most recent CDP message. The value 0 indicates no native VLAN field (TLV) was reported in the most recent CDP message.</td>
</tr>
<tr>
<td>NodeID</td>
<td>System.String</td>
<td>A SWIS-generated unique identifier of a network node in the current inventory. (Instances of this property recur in this table according to the number of remote devices for which remote device data is reported.)</td>
</tr>
<tr>
<td>LastDiscovery</td>
<td>System.DateTime</td>
<td>A SWIS-generated date and time marker for when NCM last discovered the device during inventory.</td>
</tr>
<tr>
<td>FirstDiscovery</td>
<td>System.DateTime</td>
<td>A SWIS-generated date and time marker for when NCM first discovered the device during inventory.</td>
</tr>
</tbody>
</table>

**NCM.CiscoCdp entity relationships**

<table>
<thead>
<tr>
<th>Type</th>
<th>Entity</th>
<th>Joined Data Entity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node</td>
<td>NCM.Nodes</td>
<td>NCM.NodeHostsCiscoCdp (System.Hosting)</td>
</tr>
</tbody>
</table>

**NCM.CiscoChassis**

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>chassisType</td>
<td>System.Int32</td>
<td>Chassis type (integer value). Possible Values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- unknown(1)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- multibus(2)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- agsplus(3)</td>
</tr>
<tr>
<td>Property Name</td>
<td>Datatype</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>chassisName</td>
<td>System.String</td>
<td>Chassis type (parsed string value). Possible Values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• unknown(1)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• multibus(2)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• agsplus(3)</td>
</tr>
<tr>
<td>chassisVersion</td>
<td>System.String</td>
<td>Chassis hardware revision level, or an empty string if unavailable.</td>
</tr>
<tr>
<td>chassisID</td>
<td>System.String</td>
<td>Unique ID string. Defaults to chassis serial number if available, otherwise empty. Can also be set with ‘snmp-server chassis-id’.</td>
</tr>
<tr>
<td>chassisSerialNumberString</td>
<td>System.String</td>
<td>The serial number of the chassis. This MIB object returns the chassis serial number for any chassis that uses either a numeric or an alphanumeric serial number.</td>
</tr>
<tr>
<td>RomSysVersion</td>
<td>System.String</td>
<td>ROM system software version or an empty string if unavailable.</td>
</tr>
<tr>
<td>processorRAM</td>
<td>System.Int32</td>
<td>Bytes of RAM available to CPU.</td>
</tr>
<tr>
<td>nvRAMSize</td>
<td>System.Int32</td>
<td>Bytes of nonvolatile configuration memory.</td>
</tr>
<tr>
<td>nvRAMUsed</td>
<td>System.Int32</td>
<td>Bytes of non-volatile configuration memory in use.</td>
</tr>
<tr>
<td>chassisSlots</td>
<td>System.Int32</td>
<td>Number of slots in this chassis, or -1 of neither applicable nor determinable.</td>
</tr>
<tr>
<td>romID</td>
<td>System.String</td>
<td>This variable contains a printable octet string that contains the System Bootstrap description and version identification.</td>
</tr>
<tr>
<td>whyReload</td>
<td>System.String</td>
<td>This variable contains a printable octet string that contains the reason why the system was last restarted.</td>
</tr>
</tbody>
</table>
### NCM.CiscoChassis entity relationships

<table>
<thead>
<tr>
<th>Type</th>
<th>Entity</th>
<th>Joined Data Entity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node</td>
<td>NCM.Nodes</td>
<td>NCM.NodeHostsCiscoChassis (System.Hosting)</td>
</tr>
</tbody>
</table>

### NCM.CiscoFlash

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FlashSize</td>
<td>System.Int32</td>
<td>Total size of the Flash device. For a removable device, the size is zero if the device has been removed.</td>
</tr>
<tr>
<td>Name</td>
<td>System.String</td>
<td>Flash device name. This name is used to refer to the device within the system. Flash operations get directed to a device based on this name. The system has a concept of a default device. This would be the primary or most used device in case of multiple devices. The system directs an operation to the default device whenever a device name is not specified. The device name is therefore mandatory except when the operation is being done on the default device, or the system supports only a single Flash device. The device name is always available for a removable device, even when the device has been removed.</td>
</tr>
<tr>
<td>Property Name</td>
<td>Datatype</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>----------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>FlashDescription</td>
<td>System.String</td>
<td>Description of a Flash device. The description is meant to explain what the Flash device and its purpose is. Current values are: System flash - for the primary Flash used to store full system images. Boot flash: for the secondary Flash used to store bootstrap images. The ciscoFlashDeviceDescr, CiscoFlashDeviceController (if applicable), and ciscoFlashPhyEntIndex objects are expected to collectively give all information about a Flash device. The device description is always available for a removable device, even when the device has been removed.</td>
</tr>
<tr>
<td>PartitionCount</td>
<td>System.Int32</td>
<td>Flash device partitions actually present. The number of partitions cannot exceed the minimum of ciscoFlashDeviceMaxPartitions and (ciscoFlashDeviceSize / ciscoFlashDeviceMinPartitionSize). This is equal to at least 1, in the case where the partition spans the entire device (actually no partitioning). A partition contains one or more minimum partition units (where a minimum partition unit is defined by ciscoFlashDeviceMinPartitionSize).</td>
</tr>
<tr>
<td>Property Name</td>
<td>Datatype</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>-------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>MinPartitionSize</td>
<td>System.Int32</td>
<td>This object gives the minimum partition size supported for this device. For systems that execute code directly out of Flash, the minimum partition size needs to be the bank size. (Bank size is equal to the size of a chip multiplied by the width of the device. In most cases, the device width is 4 bytes, and so the bank size would be four times the size of a chip). This has to be so because all programming commands affect the operation of an entire chip (in our case, an entire bank because all operations are done on the entire width of the device) even though the actual command may be localized to a small portion of each chip. So when executing code out of Flash, one needs to be able to write and erase some portion of Flash without affecting the code execution. For systems that execute code out of DRAM or ROM, it is possible to partition Flash with a finer granularity (for example, at erase sector boundaries) if the system code supports such granularity. This object lets a management entity know the minimum partition size as defined by the system. If the system does not support partitioning, the value is equal to the device size in ciscoFlashDeviceSize. The maximum number of partitions that could be configured is equal to the minimum of ciscoFlashDeviceMaxPartitions and (ciscoFlashDeviceSize / CiscoFlashDeviceMinPartitionSize).</td>
</tr>
<tr>
<td>Controller</td>
<td>System.String</td>
<td>Flash device controller. The h/w card that actually controls Flash read/write/erase. Relevant for the AGS+ systems where Flash may be controlled by the MC+, STR or the ENVM cards, cards that may not actually contain the Flash chips. For systems that have removable PCMCIA flash cards that are controlled by a PCMCIA controller chip, this object may contain a description of that controller chip. Where irrelevant (Flash is a direct memory mapped device accessed directly by the main processor), this object has an empty (NULL) string.</td>
</tr>
<tr>
<td>Property Name</td>
<td>Datatype</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>----------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| WriteProtectJumper  | System.String  | This object gives the state of a jumper (if present and can be determined) that controls the programming voltage called Vpp to the Flash device. Vpp is required for programming (erasing and writing) Flash. For certain older technology chips it is also required for identifying the chips (which in turn is required to identify which programming algorithms to use; different chips require different algorithms and commands). The purpose of the jumper, on systems where it is available, is to write protect a Flash device. On most of the newer remote access routers, this jumper is unavailable since users are not expected to visit remote sites just to install and remove the jumpers when upgrading software in the Flash device. The unknown(3) value is returned for such systems and can be interpreted to mean that a programming jumper is not present or not required on those systems. On systems where the programming jumper state can be read back through a hardware register, the installed (1) or notInstalled (2) value is returned. This object is expected to be used in conjunction with the ciscoFlashPartitionStatus object whenever that object has the readOnly(1) value. In such a case, this object indicates whether the programming jumper is a possible reason for the readOnly state. Possible Values:  
  - installed(1)  
  - notInstalled(2)  
  - unknown(3)     |
<table>
<thead>
<tr>
<th>Property Name</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MaxPartitions</td>
<td>System.Int32</td>
<td>Max number of partitions supported by the system for this Flash device. The default is 1, which actually means that partitioning is not supported. Note that this value is defined by system limitations, not by the flash device itself (for example, the system may impose a limit of 2 partitions even though the device may be large enough to be partitioned into 4 based on the smallest partition unit supported). On systems that execute code out of Flash, partitioning is a way of creating multiple file systems in the Flash device so that writing into or erasing of one file system can be done while executing code residing in another file system. For systems executing code out of DRAM, partitioning gives a way of sub-dividing a large Flash device for easier management of files.</td>
</tr>
<tr>
<td>Initialized</td>
<td>System.DateTime</td>
<td>System time at which device was initialized. For fixed devices, this is the system time at boot up. For removable devices, it is the time at which the device was inserted, which may be boot up time or a later time (if device was inserted later). If a device (fixed or removable) was repartitioned, it is the time of repartitioning. The purpose of this object is to help a management station determine if a removable device has been changed. The application should retrieve this object prior to any operation and compare with the previously retrieved value. Note that this time is not real time but a running time maintained by the system. This running time starts from zero when the system boots up. For a removable device that has been removed, this value is zero.</td>
</tr>
<tr>
<td>Removable</td>
<td>System.String</td>
<td>Whether Flash device is removable. Generally, only PCMCIA Flash cards are treated as removable. Socketed Flash chips and Flash SIMM modules are not treated as removable. Simply put, only those Flash devices that can be inserted or removed without opening the hardware casing are considered removable. Further, removable Flash devices are expected to have the necessary hardware support: 1) on-line removal and insertion; 2) interrupt generation on removal or insertion.</td>
</tr>
<tr>
<td>Property Name</td>
<td>Datatype</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>-------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>NodeID</td>
<td>System.String</td>
<td>A SWIS-generated unique identifier of a network node in the current inventory. (Instances of this property recur in this table according to the number of Cisco devices for which data is reported.)</td>
</tr>
<tr>
<td>LastDiscovery</td>
<td>System.DateTime</td>
<td>A SWIS-generated date and time marker for when NCM last discovered the device during inventory.</td>
</tr>
<tr>
<td>FirstDiscovery</td>
<td>System.DateTime</td>
<td>A SWIS-generated date and time marker for when NCM first discovered the device during inventory.</td>
</tr>
</tbody>
</table>

**NCM.CiscoFlash entity relationships**

<table>
<thead>
<tr>
<th>Type</th>
<th>Entity</th>
<th>Joined Data Entity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node</td>
<td>NCM.Nodes</td>
<td>NCM.NodeHostsCiscoFlash (SystemHosting)</td>
</tr>
</tbody>
</table>

**NCM.CiscoFlashFiles**

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FlashFileName</td>
<td>System.String</td>
<td>Flash file name as specified by the user copying in the file. The name should not include the colon (:) character as it is a special separator character used to delineate the device name, partition name, and the file name.</td>
</tr>
<tr>
<td>FlashFileSize</td>
<td>System.Int32</td>
<td>Size of the file in bytes. Note that this size does not include the size of the filesystem file header. File size is always non-zero.</td>
</tr>
<tr>
<td>FlashCheckSum</td>
<td>System.String</td>
<td>File checksum stored in the file header. This checksum is computed and stored when the file is written into Flash. It serves to validate the data written into Flash. Whereas the system generates and stores the checksum internally in hexadecimal form, this object provides the checksum in a string form. The checksum is available for all valid and invalid checksum files.</td>
</tr>
<tr>
<td>Property Name</td>
<td>Datatype</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>--------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| FlashFileStatus   | System.String| Status of a file. A file could be explicitly deleted if the file system supports such a user command facility. Alternately, an existing good file would be automatically deleted if another good file with the same name were copied in. Note that deleted files continue to occupy prime Flash real estate. A file is marked as having an invalid checksum if any checksum mismatch was detected while writing or reading the file. Incomplete files (files truncated either because of lack of free space or a network download failure) are also written with a bad checksum and marked as invalid. Possible Values:  
  - deleted(1)  
  - invalidChecksum(2)  
  - valid(3) |
| NodeID            | System.String| A SWIS-generated unique identifier of a network node in the current inventory. (Instances of this property recur in this table according to the number of Cisco devices for which data is reported.) |
| LastDiscovery     | System.DateTime| A SWIS-generated date and time marker for when NCM last discovered the device during inventory.                                               |
| FirstDiscovery    | System.DateTime| A SWIS-generated date and time marker for when NCM first discovered the device during inventory.                                           |

**NCM.CiscoFlashFiles entity relationships**

<table>
<thead>
<tr>
<th>Type</th>
<th>Entity</th>
<th>Joined Data Entity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node</td>
<td>NCM.Nodes</td>
<td>NCM.NodeHostsCiscoFlashFiles (System.Hosting)</td>
</tr>
</tbody>
</table>

**NCM.CiscoImageMIB**

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>System.String</td>
<td>The string of this entry.</td>
</tr>
<tr>
<td>Value</td>
<td>System.String</td>
<td>The string of this entry.</td>
</tr>
</tbody>
</table>
### Property Name | Datatype       | Description                                                                                                                                                                                                 |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>NodeID</td>
<td>System.String</td>
<td>A SWIS-generated unique identifier of a network node in the current inventory. (Instances of this property recur in this table according to the number of Cisco devices for which data is reported.)</td>
</tr>
<tr>
<td>LastDiscovery</td>
<td>System.DateTime</td>
<td>A SWIS-generated date and time marker for when NCM last discovered the device during inventory.</td>
</tr>
<tr>
<td>FirstDiscovery</td>
<td>System.DateTime</td>
<td>A SWIS-generated date and time marker for when NCM first discovered the device during inventory.</td>
</tr>
</tbody>
</table>

### NCM.CiscoImageMIB entity relationships

<table>
<thead>
<tr>
<th>Type</th>
<th>Entity</th>
<th>Joined Data Entity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node</td>
<td>NCM.Nodes</td>
<td>NCM.NodeHostsCiscoImageMIB (System.Hosting)</td>
</tr>
</tbody>
</table>

### NCM.CiscoMemoryPools

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PoolName</td>
<td>System.String</td>
<td>A textual name assigned to the memory pool. This object is suitable for output to a human operator, and may also be used to distinguish among the various pool types, especially among dynamic pools.</td>
</tr>
<tr>
<td>PoolUsed</td>
<td>System.Int32</td>
<td>Indicates the number of bytes from the memory pool that are currently in use by applications on the managed device.</td>
</tr>
<tr>
<td>PoolFree</td>
<td>System.Int32</td>
<td>Indicates the number of bytes from the memory pool that are currently unused on the managed device. Note that the sum of ciscoMemoryPoolUsed and ciscoMemoryPoolFree is the total amount of memory in the pool.</td>
</tr>
<tr>
<td>PoolLargestFree</td>
<td>System.Int32</td>
<td>Indicates the largest number of contiguous bytes from the memory pool that are currently unused on the managed device.</td>
</tr>
<tr>
<td>NodeID</td>
<td>System.String</td>
<td>A SWIS-generated unique identifier of a network node in the current inventory. (Instances of this property recur in this table according to the number of Cisco devices for which data is reported.)</td>
</tr>
</tbody>
</table>

---

page 232
<table>
<thead>
<tr>
<th>Property Name</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LastDiscovery</td>
<td>System.DateTime</td>
<td>A SWIS-generated date and time marker for when NCM last discovered the device during inventory.</td>
</tr>
<tr>
<td>FirstDiscovery</td>
<td>System.DateTime</td>
<td>A SWIS-generated date and time marker for when NCM first discovered the device during inventory.</td>
</tr>
</tbody>
</table>

**NCM.CiscoMemoryPools entity relationships**

<table>
<thead>
<tr>
<th>Type</th>
<th>Entity</th>
<th>Joined Data Entity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node</td>
<td>NCM.Nodes</td>
<td>NCM.NodeHostsCiscoMemoryPools (System.Hosting)</td>
</tr>
</tbody>
</table>

**NCM.EntityLogical**

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>System.String</td>
<td>A textual description of the logical entity. This object should contain a string that identifies the manufacturer’s name for the logical entity, and should be set to a distinct value for each version of the logical entity.</td>
</tr>
<tr>
<td>TDomain</td>
<td>System.String</td>
<td>Indicates the kind of transport service by which the logical entity receives network management traffic. Possible values for this object are presently found in the Transport Mappings for SNMPv2 document (RFC 1906 [RFC1906]).</td>
</tr>
<tr>
<td>Type</td>
<td>System.String</td>
<td>An indication of the type of logical entity. This is typically the OBJECT-IDENTIFIER name of the node in the SMI’s naming hierarchy that represents the major MIB module, or the majority of the MIB modules, supported by the logical entity. For example: a logical entity of a regular host/router &gt; mib-2 a logical entity of a 802.1d bridge &gt; dot1dBridge a logical entity of a 802.3 repeater &gt; snmpDot3RptrMgmt If an appropriate node in the naming hierarchy of SMI cannot be identified, the value ‘mib-2’ should be used.</td>
</tr>
<tr>
<td>Property Name</td>
<td>Datatype</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>----------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Community</td>
<td>System.String</td>
<td>An SNMPv1 or SNMPv2C community-string that can be used to access detailed management information for this logical entity. The agent should allow read access with this community string (to an appropriate subset of all managed objects) and may also return a community string based on the privileges of the request used to read this object. Note that an agent may return a community string with read-only privileges, even if this object is accessed with a read-write community string. However, the agent must take care not to return a community string that allows more privileges than the community string used to access this object.</td>
</tr>
<tr>
<td>TAddress</td>
<td>System.String</td>
<td>The transport service address by which the logical entity receives network management traffic, formatted according to the corresponding value of entLogicalTDomain. For snmpUDPDoman, a TAddress is 6 octets long, the initial 4 octets containing the IP-address in network-byte order and the last 2 containing the UDP port in network-byte order. Consult 'Transport Mappings for Version 2 of the Simple Network Management Protocol' (RFC 1906 [RFC1906]) for further information on snmpUDPDoman.</td>
</tr>
<tr>
<td>NodeID</td>
<td>System.String</td>
<td>A SWIS-generated unique identifier of a network node in the current inventory. (Instances of this property recur in this table according to the number of Cisco devices for which data is reported.)</td>
</tr>
<tr>
<td>LastDiscovery</td>
<td>System.DateTime</td>
<td>A SWIS-generated date and time marker for when NCM last discovered the device during inventory.</td>
</tr>
<tr>
<td>FirstDiscovery</td>
<td>System.DateTime</td>
<td>A SWIS-generated date and time marker for when NCM first discovered the device during inventory.</td>
</tr>
</tbody>
</table>

**NCM.EntityLogical entity relationships**

<table>
<thead>
<tr>
<th>Type</th>
<th>Entity</th>
<th>Joined Data Entity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node</td>
<td>NCM.Nodes</td>
<td>NCM.NodeHostsEntityLogical (System.Hosting)</td>
</tr>
<tr>
<td>Property Name</td>
<td>Datatype</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>EntityName</td>
<td>System.String</td>
<td>The textual name of the physical entity. The value of this object should be the name of the component as assigned by the local device and should be suitable for use in commands entered at the device's 'console'. This might be a text name, such as 'console' or a simple component number (for example, port or module number), such as '1', depending on the physical component naming syntax of the device. If there is no local name, or this object is otherwise not applicable, then this object contains a zero-length string.</td>
</tr>
<tr>
<td>EntityDescription</td>
<td>System.String</td>
<td>A textual description of physical entity. This object should contain a string that identifies the manufacturer's name for the physical entity, and should be set to a distinct value for each version or model of the physical entity.</td>
</tr>
<tr>
<td>EntityType</td>
<td>System.String</td>
<td>An indication of the vendor-specific hardware type of the physical entity. Note that this is different from the definition of MIB-II’s sysObjectID. An agent should set this object to a enterprise-specific registration identifier value indicating the specific equipment type in detail. The associated instance of entPhysicalClass is used to indicate the general type of hardware device. If no vendor-specific registration identifier exists for this physical entity, or the value is unknown by this agent then the value { 0 } is returned.</td>
</tr>
<tr>
<td>Property Name</td>
<td>Datatype</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ContainedIn</td>
<td>System.String</td>
<td>The value of entPhysicalIndex for the physical entity that 'contains' this physical entity. A value of zero indicates this physical entity is not contained in any other physical entity. Note that the set of 'containment' relationships define a strict hierarchy; that is, recursion is not allowed. In the event a physical entity is contained by more than one physical entity (for example, double-wide modules), this object should identify the containing entity with the lowest value of entPhysicalIndex.</td>
</tr>
<tr>
<td>EntityClass</td>
<td>System.String</td>
<td>An indication of the general hardware type of the physical entity. An agent should set this object to the standard enumeration value that most accurately indicates the general class of the physical entity or the primary class if there is more than one. If no appropriate standard registration identifier exists for this physical entity, then the value 'other(1)' is returned. If the value is unknown by this agent, then the value 'unknown(2)' is returned.</td>
</tr>
<tr>
<td>Position</td>
<td>System.Int32</td>
<td>An indication of the relative position of this 'child' component among all its 'sibling' components. Sibling components are defined as entPhysicalEntries that share the same instance values of each of the entPhysicalContainedIn and entPhysicalClass objects.</td>
</tr>
<tr>
<td>HardwareRevision</td>
<td>System.String</td>
<td>The vendor-specific hardware revision string for the physical entity. The preferred value is the hardware revision identifier actually printed on the component itself (if present). Note that if revision information is stored internally in a non-printable (for example, binary) format, then the agent must convert such information to a printable format, in an implementation-specific manner. If no specific hardware revision string is associated with the physical component, or this information is unknown to the agent, then this object contains a zero-length string.</td>
</tr>
<tr>
<td>Property Name</td>
<td>Datatype</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>----------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>FirmwareRevision</td>
<td>System.String</td>
<td>The vendor-specific firmware revision string for the physical entity. Note that if revision information is stored internally in a non-printable (for example, binary) format, then the agent must convert such information to a printable format, in an implementation-specific manner. If no specific firmware programs are associated with the physical component, or this information is unknown to the agent, then this object contains a zero-length string.</td>
</tr>
<tr>
<td>SoftwareRevision</td>
<td>System.String</td>
<td>The vendor-specific software revision string for the physical entity. Note that if revision information is stored internally in a non-printable (for example, binary) format, then the agent must convert such information to a printable format, in an implementation-specific manner. If no specific software programs are associated with the physical component, or this information is unknown to the agent, then this object contains a zero-length string.</td>
</tr>
<tr>
<td>Serial</td>
<td>System.String</td>
<td>The vendor-specific serial number string for the physical entity. The preferred value is the serial number string actually printed on the component itself (if present).</td>
</tr>
<tr>
<td>Manufacturer</td>
<td>System.String</td>
<td>The name of the manufacturer of this physical component. The preferred value is the manufacturer name string actually printed on the component itself (if present).</td>
</tr>
<tr>
<td>Model</td>
<td>System.String</td>
<td>The vendor-specific model name identifier string associated with this physical component. The preferred value is the customer-visible part number, which may be printed on the component itself. If the model name string associated with the physical component is unknown to the agent, then this object contains a zero-length string.</td>
</tr>
<tr>
<td>Alias</td>
<td>System.String</td>
<td>This object is an 'alias' name for the physical entity as specified by a network manager, and provides a non-volatile 'handle' for the physical entity.</td>
</tr>
<tr>
<td>AssetID</td>
<td>System.String</td>
<td>This object is a user-assigned asset tracking identifier for the physical entity as specified by a network manager, and provides non-volatile storage of this information.</td>
</tr>
</tbody>
</table>
**Property Name** | **Datatype** | **Description**
--- | --- | ---
FieldReplaceable | System.String | This object indicates whether or not this physical entity is considered a 'field replaceable unit' by the vendor. If this object contains the value 'true(1)' then this entPhysicalEntry identifies a field replaceable unit. For all entPhysicalEntries that represent components that are permanently contained within a field replaceable unit, the value 'false(2)' should be returned for this object.

**Property Name** | **Datatype** | **Description**
--- | --- | ---
NodeID | System.String | A SWIS-generated unique identifier of a network node in the current inventory. (Instances of this property recur in this table according to the number of Cisco devices for which data is reported.)

**Property Name** | **Datatype** | **Description**
--- | --- | ---
LastDiscovery | System.DateTime | A SWIS-generated date and time marker for when NCM last discovered the device during inventory.

**Property Name** | **Datatype** | **Description**
--- | --- | ---
FirstDiscovery | System.DateTime | A SWIS-generated date and time marker for when NCM first discovered the device during inventory.

**NCM. EntityPhysical entity relationships**

<table>
<thead>
<tr>
<th>Type</th>
<th>Entity</th>
<th>Joined Data Entity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node</td>
<td>NCM.Nodes</td>
<td>NCM.NodeHostsEntityPhysical (System.Hosting)</td>
</tr>
</tbody>
</table>

**NCM. Interfaces**

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>InterfaceID</td>
<td>System.String</td>
<td>[Swis]</td>
</tr>
<tr>
<td>InterfaceIndex</td>
<td>System.Int32</td>
<td>A unique value, greater than zero, for each interface. SolarWinds recommends assigning values contiguously starting from 1. The value for each interface sub-layer must remain constant at least from one reinitialization of the entity's network management system to the next reinitialization.</td>
</tr>
<tr>
<td>InterfaceDescription</td>
<td>System.String</td>
<td>A textual string containing information about the interface. This string should include the name of the manufacturer, the product name, and the version of the interface hardware or software.</td>
</tr>
<tr>
<td>Property Name</td>
<td>Datatype</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>----------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>VLANID</td>
<td>System.Int32</td>
<td>The set of the device's member ports that belong to the VLAN. Each octet within the value of this object specifies a set of eight ports, with the first octet specifying ports 1 through 8, the second octet specifying ports 9 through 16, etc. Within each octet, the most significant bit represents the lowest numbered port, and the least significant bit represents the highest numbered port. Thus, each port of the VLAN is represented by a single bit within the value of this object. If that bit has a value of '1' then that port is included in the set of ports. The port is not included if its bit has a value of '0'. A port number is the value of dot1dBasePort for the port in the BRIDGE-MIB (RFC 1493).</td>
</tr>
<tr>
<td>PortStatus</td>
<td>System.String</td>
<td>An indication of the current VLAN status of the port. A status of inactive(1) indicates that a dynamic port does not yet have a VLAN assigned, or a port is assigned to a VLAN that is currently not active. A status of active(2) indicates that the currently assigned VLAN is active. A status of shutdown(3) indicates that the port has been disabled as a result of VQP shutdown response. Possible Values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- inactive(1)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- active(2)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- shutdown(3)</td>
</tr>
<tr>
<td>VLANType</td>
<td>System.Int32</td>
<td>The type of this VLAN</td>
</tr>
<tr>
<td>Property Name</td>
<td>Datatype</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>---------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>InterfaceName</td>
<td>System.String</td>
<td>The textual name of the interface. The value of this object should be the name of the interface as assigned by the local device and should be suitable for use in commands entered at the device's 'console'. This might be a text name, such as 'le0' or a simple port number, such as '1', depending on the interface naming syntax of the device. If several entries in the ifTable together represent a single interface as named by the device, then each entry has the same value of ifName. If there is no local name, or this object is otherwise not applicable, then this object contains a 0-length string.</td>
</tr>
<tr>
<td>InterfaceAlias</td>
<td>System.String</td>
<td>This object is an 'alias' name for the interface as specified by a network manager, and provides a non-volatile 'handle' for the interface.</td>
</tr>
<tr>
<td>InterfaceType</td>
<td>System.Int32</td>
<td>The type of interface. Additional values for ifType are assigned by the Internet Assigned Numbers Authority (IANA), through updating the syntax of the IANAifType textual convention.</td>
</tr>
<tr>
<td>InterfaceTypeName</td>
<td>System.String</td>
<td>The type of interface. Additional values for ifType are assigned by the Internet Assigned Numbers Authority (IANA), through updating the syntax of the IANAifType textual convention.</td>
</tr>
<tr>
<td>InterfaceTypeDescription</td>
<td>System.String</td>
<td>The type of interface. Additional values for ifType are assigned by the Internet Assigned Numbers Authority (IANA), through updating the syntax of the IANAifType textual convention.</td>
</tr>
<tr>
<td>Property Name</td>
<td>Datatype</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>InterfaceSpeed</td>
<td>System.Single</td>
<td>An estimate of the interface’s current bandwidth in bits per second. For interfaces that do not vary in bandwidth or for those where no accurate estimation can be made, this object should contain the nominal bandwidth. If the bandwidth of the interface is greater than the maximum value reportable by this object then this object should report its maximum value (4, 294, 967, 295) and ifHighSpeed must be used to report the interface’s speed. For a sub-layer that has no concept of bandwidth, this object should be zero.</td>
</tr>
<tr>
<td>MACAddress</td>
<td>System.String</td>
<td>The interface’s address at its protocol sublayer. The interface’s media specific MIB must define the bit and byte ordering and format of the value contained by this object. For interfaces that do not have such an address (for example, a serial line), this object should contain an octet string of zero length.</td>
</tr>
<tr>
<td>AdminStatus</td>
<td>System.String</td>
<td>The desired state of the interface. The testing(3) state indicates that no operational packets can be passed. When a managed system initializes, all interfaces start with ifAdminStatus in the down(2) state. As a result of either explicit management action or per configuration information retained by the managed system, ifAdminStatus is then changed to either the up (1) or testing (3) states (or remains in the down (2) state).</td>
</tr>
<tr>
<td>OperStatus</td>
<td>System.String</td>
<td>The current operational state of the interface. The testing (3) state indicates that no operational packets can be passed. If ifAdminStatus is down (2) then ifOperStatus should be down (2). If ifAdminStatus is changed to up (1) then ifOperStatus should change to up (1) if the interface is ready to transmit and receive network traffic; it should change to dormant(5) if the interface is waiting for external actions (such as a serial line waiting for an incoming connection); it should remain in the down (2) state if and only if there is a fault that prevents it from going to the up (1) state.</td>
</tr>
<tr>
<td>Property Name</td>
<td>Datatype</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>----------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>InterfaceMTU</td>
<td>System.Int32</td>
<td>The size of the largest packet that can be sent or received on the interface, specified in octets. For interfaces that are used for transmitting network datagrams, this is the size of the largest network datagram that can be sent on the interface.</td>
</tr>
<tr>
<td>LastChange</td>
<td>System.DateTime</td>
<td>The value of sysUpTime at the time the interface entered its current operational state. If the current state was entered prior to the last re-initialization of the local network management subsystem, then this object contains a zero value.</td>
</tr>
<tr>
<td>PhysicalInterface</td>
<td>System.Char</td>
<td>This object has the value 'true(1)' if the interface sublayer has a physical connector and the value 'false(2)' otherwise.</td>
</tr>
<tr>
<td>Promiscuous</td>
<td>System.Char</td>
<td>This object has a value of false(2) if this interface only accepts packets/frames that are addressed to this station. This object has a value of true(1) when the station accepts all packets/frames transmitted on the media. The value true(1) is only legal on certain types of media. If legal, setting this object to a value of true(1) may require the interface to be reset before becoming effective. The value of ifPromiscuousMode does not affect the reception of broadcast and multicast packets/frames by the interface.</td>
</tr>
<tr>
<td>NodeID</td>
<td>System.String</td>
<td>A SWIS-generated unique identifier of a network node in the current inventory. (Instances of this property recur in this table according to the number of Cisco devices for which data is reported.)</td>
</tr>
<tr>
<td>LastDiscovery</td>
<td>System.DateTime</td>
<td>A SWIS-generated date and time marker for when NCM last discovered the device during inventory.</td>
</tr>
<tr>
<td>FirstDiscovery</td>
<td>System.DateTime</td>
<td>A SWIS-generated date and time marker for when NCM first discovered the device during inventory.</td>
</tr>
</tbody>
</table>
NCM.Interfaces entity relationships

<table>
<thead>
<tr>
<th>Type</th>
<th>Entity</th>
<th>Joined Data Entity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node</td>
<td>NCM.Nodes</td>
<td>NCM.NodeHostsInterfaces (System.Hosting)</td>
</tr>
<tr>
<td>IpAddresses</td>
<td>NCM.IpAddresses</td>
<td>NCM.InterfaceHostsIpAddresses (System.Hosting)</td>
</tr>
</tbody>
</table>

NCM.IpAddresses

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>InterfaceIndex</td>
<td>System.Int32</td>
<td>The index value that uniquely identifies the interface to which this entry is applicable. The interface identified by a particular value of this index is the same interface as identified by the same value of RFC 1573's ifIndex.</td>
</tr>
<tr>
<td>IPAddress</td>
<td>System.String</td>
<td>The IP address to which this entry's addressing information pertains.</td>
</tr>
<tr>
<td>IPAddrIPSort</td>
<td>System.Double</td>
<td>Store IP address in double representation.</td>
</tr>
<tr>
<td>SubnetMask</td>
<td>System.String</td>
<td>The subnet mask associated with the IP address of this entry. The value of the mask is an IP address with all the network bits set to 1 and all the hosts bits set to 0.</td>
</tr>
<tr>
<td>InterfaceID</td>
<td>System.String</td>
<td>InterfaceId from interfaces table.</td>
</tr>
<tr>
<td>NodeID</td>
<td>System.String</td>
<td>A SWIS-generated unique identifier of a network node in the current inventory. (Instances of this property recur in this table according to the number of Cisco devices for which data is reported.)</td>
</tr>
<tr>
<td>LastDiscovery</td>
<td>System.DateTime</td>
<td>A SWIS-generated date and time marker for when NCM last discovered the device during inventory.</td>
</tr>
<tr>
<td>FirstDiscovery</td>
<td>System.DateTime</td>
<td>A SWIS-generated date and time marker for when NCM first discovered the device during inventory.</td>
</tr>
</tbody>
</table>

NCM.IpAddresses entity relationships

<table>
<thead>
<tr>
<th>Type</th>
<th>Entity</th>
<th>Joined Data Entity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interfaces</td>
<td>NCM.Interfaces</td>
<td>NCM.InterfaceHostsIpAddresses (System.Hosting)</td>
</tr>
</tbody>
</table>
## NCM.MacForwarding

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port</td>
<td>System.Int32</td>
<td>Either the value '0', or the port number of the port on which a frame having a source address equal to the value of the corresponding instance of dot1dTpFdbAddress has been seen. A value of '0' indicates that the port number has not been learned but that the bridge does have some forwarding/filtering information about this address (for example, in the dot1dStaticTable). Implementers are encouraged to assign the port value to this object whenever it is learned even for addresses for which the corresponding value of dot1dTpFdbStatus is not learned(3).</td>
</tr>
<tr>
<td>MAC</td>
<td>System.String</td>
<td>A unicast MAC address for which the bridge has forwarding and/or filtering information.</td>
</tr>
<tr>
<td>Source</td>
<td>System.String</td>
<td>The status of this entry. The meanings of the values are: other (1) : none of the following. This would include the case where some other MIB object (not the corresponding instance of dot1dTpFdbPort, nor an entry in the dot1dStaticTable) is being used to determine if and how frames addressed to the value of the corresponding instance of dot1dTpFdbAddress are being forwarded.</td>
</tr>
<tr>
<td>NodeID</td>
<td>System.String</td>
<td>A SWIS-generated unique identifier of a network node in the current inventory. (Instances of this property recur in this table according to the number of Cisco devices for which data is reported.)</td>
</tr>
<tr>
<td>LastDiscovery</td>
<td>System.DateTime</td>
<td>A SWIS-generated date and time marker for when NCM last discovered the device during inventory.</td>
</tr>
<tr>
<td>FirstDiscovery</td>
<td>System.DateTime</td>
<td>A SWIS-generated date and time marker for when NCM first discovered the device during inventory.</td>
</tr>
</tbody>
</table>

### NCM.MacForwarding entity relationships

<table>
<thead>
<tr>
<th>Type</th>
<th>Entity</th>
<th>Joined Data Entity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node</td>
<td>NCM.Nodes</td>
<td>NCM.NodeHostsMacForwarding (System.Hosting)</td>
</tr>
</tbody>
</table>
## NCM.Nodes

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AgentIP</td>
<td>System.String</td>
<td>IP address of the device entered manually by the user.</td>
</tr>
<tr>
<td>Status</td>
<td>System.Byte</td>
<td>NCM-only specific status of the device:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Unknown = 0 (not polled yet)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Up = 1 (based on ICMP pool)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Down = 2 (based on ICMP pool)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Warning = 3 (based on ICMP pool)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- MonitoringDisabled = 10 (NCM node monitoring is disabled by user)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- UnManaged = 9 (device is unmanaged in NCM)</td>
</tr>
<tr>
<td>Community</td>
<td>System.String</td>
<td>SNMP community string entered by the user.</td>
</tr>
<tr>
<td>ReverseDNS</td>
<td>System.String</td>
<td>DNS name of the device.</td>
</tr>
<tr>
<td>SysName</td>
<td>System.String</td>
<td>An administratively-assigned name for this managed node.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>By convention, this is the node's fully-qualified domain name.</td>
</tr>
<tr>
<td>SysDescr</td>
<td>System.String</td>
<td>A textual description of the entity. This value should include the full name</td>
</tr>
<tr>
<td></td>
<td></td>
<td>and version identification of the system's hardware type, software</td>
</tr>
<tr>
<td></td>
<td></td>
<td>operating-system, and networking software. It is mandatory that this</td>
</tr>
<tr>
<td></td>
<td></td>
<td>contains only printable ASCII characters.</td>
</tr>
<tr>
<td>SysContact</td>
<td>System.String</td>
<td>The textual identification of the contact person for this managed node.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>together with information on how to contact this person.</td>
</tr>
<tr>
<td>SysLocation</td>
<td>System.String</td>
<td>The physical location of this node (for example, 'telephone closet, 3rd</td>
</tr>
<tr>
<td></td>
<td></td>
<td>floor').</td>
</tr>
<tr>
<td>SystemOID</td>
<td>System.String</td>
<td>The vendor's authoritative identification of the network management</td>
</tr>
<tr>
<td></td>
<td></td>
<td>subsystem contained in the entity. This value is allocated within the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SMI enterprises subtree (1.3.6.1.4.1) and provides an easy and</td>
</tr>
<tr>
<td></td>
<td></td>
<td>unambiguous means for determining 'what kind of box' is being managed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For example, if vendor 'Flintstones, Inc.' was assigned the subtree 1.3.6.1.4.1.4242,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>it could assign the identifier 1.3.6.1.4.1.4242.1.1 to its 'Fred Router'.</td>
</tr>
<tr>
<td>Vendor</td>
<td>System.String</td>
<td>Vendor of device- determined based on SystemOID.</td>
</tr>
<tr>
<td>Property Name</td>
<td>Datatype</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>----------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>VendorIcon</td>
<td>System.String</td>
<td>Vendor icon of device - determined based on SystemOID.</td>
</tr>
<tr>
<td>MachineType</td>
<td>System.String</td>
<td>Machine Type - determined based on SystemOID.</td>
</tr>
<tr>
<td>LastBoot</td>
<td>System.DateTime</td>
<td>The time (in hundredths of a second) since the network management portion of the system was last re-initialized.</td>
</tr>
<tr>
<td>OSImage</td>
<td>System.String</td>
<td>Determined based on SysDescr.</td>
</tr>
<tr>
<td>OSVersion</td>
<td>System.String</td>
<td>Determined based on SysDescr.</td>
</tr>
<tr>
<td>SNMPLevel</td>
<td>System.Byte</td>
<td>SNMP version selected by the user (1,2 or 3).</td>
</tr>
<tr>
<td>SNMPContext</td>
<td>System.String</td>
<td>SNMPv3 credentials entered by the user.</td>
</tr>
<tr>
<td>SNMPUsername</td>
<td>System.String</td>
<td>SNMPv3 credentials entered by the user.</td>
</tr>
<tr>
<td>SNMPAuthType</td>
<td>System.String</td>
<td>SNMPv3 credentials entered by the user.</td>
</tr>
<tr>
<td>SNMPAuthPass</td>
<td>System.String</td>
<td>SNMPv3 credentials entered by the user.</td>
</tr>
<tr>
<td>SNMPEncryptType</td>
<td>System.String</td>
<td>SNMPv3 credentials entered by the user.</td>
</tr>
<tr>
<td>SNMPEncryptPass</td>
<td>System.String</td>
<td>SNMPv3 credentials entered by the user.</td>
</tr>
<tr>
<td>SNMPStatus</td>
<td>System.String</td>
<td>The status of SNMP connection to the device (OK, No SNMP support, SNMP error description if any).</td>
</tr>
<tr>
<td>NodeID</td>
<td>System.String</td>
<td>A SWIS-generated unique identifier of a network node in the current inventory. (Instances of this property recur in this table according to the number of devices for which data is reported.)</td>
</tr>
<tr>
<td>LastDiscovery</td>
<td>System.DateTime</td>
<td>A SWIS-generated date and time marker for when NCM last discovered the device during inventory.</td>
</tr>
<tr>
<td>FirstDiscovery</td>
<td>System.DateTime</td>
<td>A SWIS-generated date and time marker for when NCM first discovered the device during inventory.</td>
</tr>
</tbody>
</table>

**NCM.Nodes entity relationships**

<table>
<thead>
<tr>
<th>Type</th>
<th>Entity</th>
<th>Joined Data Entity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interfaces</td>
<td>NCM.Interfaces</td>
<td>NCM.NodeHostsInterfaces (System.Hosting)</td>
</tr>
<tr>
<td>MacForwarding</td>
<td>NCM.MacForwarding</td>
<td>NCM.NodeHostsMacForwarding (System.Hosting)</td>
</tr>
<tr>
<td>VLANs</td>
<td>NCM.VLANs</td>
<td>NCM.NodeHostsVLANs (System.Hosting)</td>
</tr>
<tr>
<td>Type</td>
<td>Entity</td>
<td>Joined Data Entity</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-------------------------</td>
<td>--------------------------------------------------------</td>
</tr>
<tr>
<td>BridgePorts</td>
<td>NCM.BridgePorts</td>
<td>NCM.NodeHostsBridgePorts (System.Hosting)</td>
</tr>
<tr>
<td>ArpTables</td>
<td>NCM.ArpTables</td>
<td>NCM.NodeHostsArpTables (System.Hosting)</td>
</tr>
<tr>
<td>CiscoCards</td>
<td>NCM.CiscoCards</td>
<td>NCM.NodeHostsCiscoCards (System.Hosting)</td>
</tr>
<tr>
<td>CiscoCdp</td>
<td>NCM.CiscoCdp</td>
<td>NCM.NodeHostsCiscoCdp (System.Hosting)</td>
</tr>
<tr>
<td>CiscoChassis</td>
<td>NCM.CiscoChassis</td>
<td>NCM.NodeHostsCiscoChassis (System.Hosting)</td>
</tr>
<tr>
<td>CiscoFlash</td>
<td>NCM.CiscoFlash</td>
<td>NCM.NodeHostsCiscoFlash (System.Hosting)</td>
</tr>
<tr>
<td>CiscoFlashFiles</td>
<td>NCM.CiscoFlashFiles</td>
<td>NCM.NodeHostsCiscoFlashFiles (System.Hosting)</td>
</tr>
<tr>
<td>CiscoImageMIB</td>
<td>NCM.CiscoImageMIB</td>
<td>NCM.NodeHostsCiscoImageMIB (System.Hosting)</td>
</tr>
<tr>
<td>CiscoMemoryPools</td>
<td>NCM.CiscoMemoryPools</td>
<td>NCM.NodeHostsCiscoMemoryPools (System.Hosting)</td>
</tr>
<tr>
<td>EntityLogical</td>
<td>NCM.EntityLogical</td>
<td>NCM.NodeHostsEntityLogical (System.Hosting)</td>
</tr>
<tr>
<td>EntityPhysical</td>
<td>NCM.EntityPhysical</td>
<td>NCM.NodeHostsEntityPhysical (System.Hosting)</td>
</tr>
<tr>
<td>PortsTcp</td>
<td>NCM.PortsTcp</td>
<td>NCM.NodeHostsPortsTcp (SystemHosting)</td>
</tr>
<tr>
<td>PortsUdp</td>
<td>NCM.PortsUdp</td>
<td>NCM.NodeHostsPortsUdp (System.Hosting)</td>
</tr>
<tr>
<td>RouteTable</td>
<td>NCM.RouteTable</td>
<td>NCM.NodeHostsRouteTable (System.Hosting)</td>
</tr>
</tbody>
</table>

**NCM.PortsTcp**

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NodeID</td>
<td>System.String</td>
<td>The unique identifier of a network node subject to configuration actions.</td>
</tr>
<tr>
<td>LastDiscovery</td>
<td>System.DateTime</td>
<td>Date and time NCM last discovered the device during inventory.</td>
</tr>
<tr>
<td>FirstDiscovery</td>
<td>System.DateTime</td>
<td>Date and time NCM first discovered the device during inventory.</td>
</tr>
<tr>
<td>TCPLocalAddress</td>
<td>System.String</td>
<td>The local IP address for this TCP connection. In the case of a connection in the listen state that is willing to accept connections for any IP interface associated with the node, the value 0.0.0.0 is used.</td>
</tr>
<tr>
<td>TCPLocalPort</td>
<td>System.Int32</td>
<td>The local port number for this TCP connection.</td>
</tr>
</tbody>
</table>

**ADMINISTRATOR GUIDE: NETWORK CONFIGURATION MANAGER**

page 247
<table>
<thead>
<tr>
<th>Property Name</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCPLocalPortName</td>
<td>System.String</td>
<td>Port description based on TCPLocalPort value.</td>
</tr>
<tr>
<td>TCPRemoteAddress</td>
<td>System.String</td>
<td>The remote IP address for this TCP connection.</td>
</tr>
<tr>
<td>TCPRemotePort</td>
<td>System.Int32</td>
<td>The remote port number for this TCP connection.</td>
</tr>
<tr>
<td>TCPState</td>
<td>System.String</td>
<td>The state of this TCP connection.</td>
</tr>
</tbody>
</table>

The only value that may be set by a management station is deleteTCB(12). Accordingly, it is appropriate for an agent to return a 'badValue' response if a management station attempts to set this object to any other value.

If a management station sets this object to the value deleteTCB(12), then this has the effect of deleting the TCB (as defined in RFC 793) of the corresponding connection on the managed node, resulting in immediate termination of the connection.

As an implementation-specific option, a RST segment may be sent from the managed node to the other TCP endpoint (note however that RST segments are not sent reliably).

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCPRemotePortName</td>
<td>System.String</td>
<td>Port description based on TCPRemotePort value.</td>
</tr>
</tbody>
</table>

NCM.PortsTcp entity relationships

<table>
<thead>
<tr>
<th>Type</th>
<th>Entity</th>
<th>Joined Data Entity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node</td>
<td>NCM.Notes</td>
<td>NCM.NodeHostsPortsTcp (System.Hosting)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NodeID</td>
<td>System.String</td>
<td>The unique identifier of a network node subject to configuration actions.</td>
</tr>
<tr>
<td>LastDiscovery</td>
<td>System.DateTime</td>
<td>Date and time NCM last discovered the device during inventory.</td>
</tr>
<tr>
<td>FirstDiscovery</td>
<td>System.DateTime</td>
<td>Date and time NCM first discovered the device during inventory.</td>
</tr>
</tbody>
</table>
### NCM.PortsUdp entity relationships

<table>
<thead>
<tr>
<th>Type</th>
<th>Entity</th>
<th>Joined Data Entity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node</td>
<td>NCM.Nodes</td>
<td>NCM.NodeHostsPortsUdp (System.Hosting)</td>
</tr>
</tbody>
</table>

### NCM.RouteTable

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NodeID</td>
<td>System.String</td>
<td>The unique identifier of a network node subject to configuration actions.</td>
</tr>
<tr>
<td>LastDiscovery</td>
<td>System.DateTime</td>
<td>Date and time NCM last discovered the device during inventory.</td>
</tr>
<tr>
<td>FirstDiscovery</td>
<td>System.DateTime</td>
<td>Date and time NCM first discovered the device during inventory.</td>
</tr>
<tr>
<td>InterfaceIndex</td>
<td>System.Int32</td>
<td>The index value that uniquely identifies the local interface through which the next hop of this route should be reached. The interface identified by a particular value of this index is the same interface as identified by the same value of ifIndex.</td>
</tr>
<tr>
<td>Property Name</td>
<td>Datatype</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>----------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| InterfaceID   | System.String  | Indicate the mask to be logical-ANDed with the destination address before being compared to the value in the ipRouteDest field. For those systems that do not support arbitrary subnet masks, an agent constructs the value of the ipRouteMask by determining whether the value of the correspondent ipRouteDest field belong to a class-A, B, or C network, and then using one of:  
  - mask network  
  - 255.0.0.0 class-A  
  - 255.255.0.0 class-B  
  - 255.255.255.0 class-C  

  If the value of the ipRouteDest is 0.0.0.0 (a default route), then the mask value is also 0.0.0.0. Note that all IP routing subsystems implicitly use this mechanism. |
| Destination   | System.String  | The type of route. Note that the values direct(3) and indirect(4) refer to the notion of direct and indirect routing in the IP architecture. Setting this object to the value invalid(2) has the effect of invalidating the corresponding entry in the ipRouteTable object. That is, it effectively disassociates the destination identified with said entry from the route identified with said entry. It is an implementation-specific matter as to whether the agent removes an invalidated entry from the table.  

  Accordingly, management stations must be prepared to receive tabular information from agents that corresponds to entries not currently in use. Proper interpretation of such entries requires examination of the relevant ipRouteType object.  

  Possible Values:  
  - other(1)  
  - invalid(2)  
  - direct(3)  
  - indirect(4) |

```
<table>
<thead>
<tr>
<th>Property Name</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
</table>
| Mask          | System.String  | The routing mechanism through which this route was learned. Inclusion of values for gateway routing protocols is not intended to imply that hosts should support those protocols. Possible Values:  
  - other(1)  
  - local(2)  
  - netmgmt(3)  
  - icmp(4)  
  - egp(5)  
  - ggp(6)  
  - hello(7)  
  - rip(8)  
  - is-is(9)  
  - es-is(10)  
  - ciscoIgrp(11)  
  - bbnSpfIgp(12)  
  - ospf(13)  
  - bgp(14)  |
<p>| NextHop       | System.String  | The number of seconds since this route was last updated or otherwise determined to be correct. Note that no semantics of 'too old' can be implied except through knowledge of the routing protocol by which the route was learned. |
| RouteType     | System.String  | The IP address of the next hop of this route. (In the case of a route bound to an interface that is realized via a broadcast media, the value of this field is the agent's IP address on that interface.) |
| RouteProtocol | System.String  | The primary routing metric for this route. The semantics of this metric are determined by the routing-protocol specified in the route’s ipRouteProto value. If this metric is not used, its value should be set to -1. |
| RouteAge      | System.Int32   | An alternate routing metric for this route. The semantics of this metric are determined by the routing-protocol specified in the route’s ipRouteProto value. If this metric is not used, its value should be set to -1. |</p>
<table>
<thead>
<tr>
<th>Property Name</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NextHopAS</td>
<td>System.Int32</td>
<td>An alternate routing metric for this route. The semantics of this metric are determined by the routing-protocol specified in the route’s ipRouteProto value. If this metric is not used, its value should be set to -1.</td>
</tr>
<tr>
<td>Metric1</td>
<td>System.Int32</td>
<td>An alternate routing metric for this route. The semantics of this metric are determined by the routing-protocol specified in the route’s ipRouteProto value. If this metric is not used, its value should be set to -1.</td>
</tr>
<tr>
<td>Metric2</td>
<td>System.Int32</td>
<td>An alternate routing metric for this route. The semantics of this metric are determined by the routing-protocol specified in the route’s ipRouteProto value. If this metric is not used, its value should be set to -1.</td>
</tr>
<tr>
<td>Metric3</td>
<td>System.Int32</td>
<td>The index value that uniquely identifies the local interface through which the next hop of this route should be reached. The interface identified by a particular value of this index is the same interface as identified by the same value of ifIndex.</td>
</tr>
</tbody>
</table>
| Metric4       | System.Int32  | Indicate the mask to be logical-ANDed with the destination address before being compared to the value in the ipRouteDest field. For those systems that do not support arbitrary subnet masks, an agent constructs the value of the ipRouteMask by determining whether the value of the correspondent ipRouteDest field belong to a class-A, B, or C network, and then using one of:  
  - mask network  
  - 255.0.0.0 class-A  
  - 255.255.0.0 class-B  
  - 255.255.255.0 class-C  
  If the value of the ipRouteDest is 0.0.0.0 (a default route), then the mask value is also 0.0.0.0. Note that all IP routing subsystems implicitly use this mechanism. |
The type of route. Note that the values direct(3) and indirect(4) refer to the notion of direct and indirect routing in the IP architecture.

Setting this object to the value invalid(2) has the effect of invalidating the corresponding entry in the ipRouteTable object. That is, it effectively disassociates the destination identified with said entry from the route identified with said entry. It is an implementation-specific matter as to whether the agent removes an invalidated entry from the table. Accordingly, management stations must be prepared to receive tabular information from agents that corresponds to entries not currently in use. Proper interpretation of such entries requires examination of the relevant ipRouteType object.

Possible Values:
- other(1)
- invalid(2)
- direct(3)
- indirect(4)
<table>
<thead>
<tr>
<th>Property Name</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VLANID</td>
<td>System.Int32</td>
<td>The set of the device's member ports that belong to the VLAN. Each octet within the value of this object specifies a set of eight ports, with the first octet specifying ports 1 through 8, the second octet specifying ports 9 through 16, etc. Within each octet, the most significant bit represents the lowest numbered port, and the least significant bit represents the highest numbered port. Thus, each port of the VLAN is represented by a single bit within the value of this object. If that bit has a value of '1' then that port is included in the set of ports; the port is not included if its bit has a value of '0'. A port number is the value of dot1dBasePort for the port in the BRIDGE-MIB (RFC 1493).</td>
</tr>
<tr>
<td>VLANName</td>
<td>System.String</td>
<td>The name of this VLAN. This name is used as the ELAN-name for an ATM LAN-Emulation segment of this VLAN.</td>
</tr>
<tr>
<td>VLANMTU</td>
<td>System.Int32</td>
<td>The MTU size on this VLAN, defined as the size of largest MAC-layer (information field portion of the) data frame that can be transmitted on the VLAN.</td>
</tr>
<tr>
<td>VLANType</td>
<td>System.Int32</td>
<td>The type of this VLAN</td>
</tr>
</tbody>
</table>
| VLANState     | System.Int32   | The state of this VLAN. Possible Values:  
- operational(1)  
- suspended(2)  
- mtuTooBigForDevice(3)  
- mtuTooBigForTrunk(4) |

**NCM.VLANs entity relationships**

<table>
<thead>
<tr>
<th>Type</th>
<th>Entity</th>
<th>Joined Data Entity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node</td>
<td>NCM.Nodes</td>
<td>NCM.NodeHostsVLANs (System.Hosting)</td>
</tr>
</tbody>
</table>
Examples of NCM config change templates

A config change template consists of two parts: parameters and commands. The following sections describe the parameters and commands in two example templates:

- Change VLAN Membership on Ports Cisco IOS example
- Enable NetFlow on Cisco ASA example

Change VLAN Membership on Ports Cisco IOS example

A config change template named Change VLAN Membership on Ports Cisco IOS is installed with NCM. Its purpose is to change VLAN membership on Cisco IOS device ports.

The following sections explain the specific components of a config change template by demonstrating how to use the Change VLAN Membership on Ports Cisco IOS template to make VLAN membership config changes on hypothetical Cisco device interfaces.

This section assumes that you know how to make VLAN membership changes to device interfaces from the Cisco IOS command line. This section also assumes that you are familiar with using variables, data arrays, foreach loops, if/else conditional statements, and logical operators in creating system administration scripts.

The following sections show the reference template broken up into parameter, command, and output sections.

Parameters

These are the parameters for the Change VLAN Membership on Ports Cisco IOS template. Notice that the parameters already have values associated with them, which are either a string or a variable.

```bash
/*
.CHANGE_TEMPLATE_DESCRIPTION
   This change template configures VLAN membership on Cisco IOS devices. The template was verified on Cisco 2950 Catalyst Switch running IOS software version 12.1(12c).

.CHANGE_TEMPLATE_TAGS
   Cisco, IOS, VLAN Membership

.PLATFORM_DESCRIPTION
   Cisco IOS

.PARAMETER_LABEL @ContextNode
   NCM Node

.PARAMETER_DESCRIPTION @ContextNode
*/
```
The node the template will operate on. All templates require this by default. The target node is selected during the first part of the wizard so it will not be available for selection when defining values of variables.

**.PARAMETER_LABEL @TargetPorts**
Select Port(s)

**.PARAMETER_DESCRIPTION @TargetPorts**
Select the port(s) for which you would like to change VLAN membership.

**.PARAMETER_LABEL @VlansToRemove**
VLAN(s) to remove

**.PARAMETER_DESCRIPTION @VlansToRemove**
Select the VLAN(s) to remove. Selecting VLANs irrelevant to interfaces results in no actions taken for those interfaces.

**.PARAMETER_LABEL @VlanToAssign**
VLAN to assign

**.PARAMETER_DESCRIPTION @VlanToAssign**
Select the VLAN you would like to assign.

Commands

There is one instance of the script command and multiple instances of the **CLI( )** command, and all variables have declarations.

```plaintext
script ConfigureVLANmembershipCiscoIOS (  
  NCM.Nodes @ContextNode,
  NCM.Interfaces[] @TargetPorts,
  NCM.VLANs[] @VlansToRemove,
  NCM.VLANs @VlanToAssign )
{
  // Enter configuration mode
  CLI
  { configure terminal }  

  // Loop through selected ports
```
foreach (@portItem in @TargetPorts)
{
    CLI
    { interface @portItem.InterfaceDescription }
}

// Loop through list of vlans to remove
foreach (@vlanRemove in @VlansToRemove)
{
    CLI
    { no switchport access vlan @vlanRemove.VLANID }
}

CLI
{ switchport access vlan @VlanToAssign.VLANID }
CLI
{ exit }
}

// Exit configuration mode
CLI
{ exit }
}

Output Commands

These are the commands that NCM executes after logging on to the NCM device(s) selected as the target for this config change template. We are changing VLAN membership on one interface of two different Cisco switches.

bgp-2651-03

    configure terminal
    interface FastEthernet0/0
    no switchport access vlan 1004
    switchport access vlan 1002
    exit
    exit

cur-3725

    configure terminal
    interface FastEthernet0/1
    no switchport access vlan 1004
Enable NetFlow on Cisco ASA example

A config change template named Enable NetFlow on CiscoASA is installed with NCM. It configures a Cisco ASA for NetFlow export.

Here are the commands that this template executes on the command line of the targeted devices selected in the template's run-time setup wizard. For this example, we are including values as if the user entered them in the wizard interface.

```plaintext
switchport access vlan 1002
exit
exit
```

```plaintext
configure terminal
flow-export destination inside 10.10.18.157 2055
flow-export template timeout-rate 1
flow-export delay flow-create 60
logging flow-export syslogs disable
access-list netflow-export extended permit ip any
class-map netflow-export-class
match access-list netflow-export
policy-map netflow-policy
class netflow-export-class
flow-export event-type all destination 10.10.18.157
service-policy netflow-policy global
flow-export enable
exit
end
```

You could execute this set of CLI commands on your target device and the result would be config changes in the status of NetFlow data processing by the device.

The config change template that produces this output of CLI commands is:

```plaintext
/*
.CHANGE_TEMPLATE_DESCRIPTION
   This change template configures your Cisco ASA for NetFlow export. This was verified on an ASA 5505 running ASA software version 8.2(1)12.
.CHANGE_TEMPLATE_TAGS
   Cisco, ASA, NetFlow
*/
```
The node the template will operate on. All templates require this by default. The target node is selected during the first part of the wizard so it will not be available for selection when defining values of variables.

Enter the IP address of the server running the NetFlow traffic analysis solution (for example: SolarWinds NetFlow Traffic Analyzer).

Enter the NetFlow export port. The default for SolarWinds NTA is 2055.

```bash
script EnableNetflowOnCiscoASA ( 
    NCM.Nodes @ContextNode, 
    string @NetFlowCollectorIPAddress, 
    int @NetFlowExportPort 
) 
{
    // Enter configuration mode and generate NetFlow commands
    CLI
    {
        configure terminal
        flow-export destination inside @NetFlowCollectorIPAddress @NetFlowExportPort
        flow-export template timeout-rate 1
        flow-export delay flow-create 60
        logging flow-export-syslogs disable
    }
}
```
access-list netflow-export extended permit ip any any
class-map netflow-export-class
match access-list netflow-export
policy-map netflow-policy
class netflow-export-class
flow-export event-type all destination @NetFlowCollectorIPAddress
service-policy netflow-policy global
flow-export enable
exit
}
}

Parameters

The parameters defined at the beginning of this script create an interface in which the user types the IP address and port of the NetFlow receiver.

.PARAMETER_LABEL @NetFlowCollectorIPAddress
    NetFlow Collector IP Address
.PARAMETER_DESCRIPTION @NetFlowCollectorIPAddress
    Enter the IP address of the server running the NetFlow traffic analysis solution (e.g. SolarWinds NetFlow Traffic Analyzer--NTA).

.PARAMETER_LABEL @NetFlowExportPort
    NetFlow Export Port
.PARAMETER_DESCRIPTION @NetFlowExportPort
    Enter the NetFlow export port (default for SolarWinds NTA is 2055).

The first line defines the parameter or variable name (in this case, @NetFlowCollectorIPAddress) for which the user enters a value in the wizard interface text box at run time. The second line defines the label (in this case, NetFlow Collector IP Address) that appears in the wizard interface to prompt the user to enter the IP address. The third and fourth lines define the description that appears below the wizard interface text box.

The parameters for NetFlow Export Port (in lines 5-12) function exactly the same way as the first four. The parameter variables, labels, and descriptions guide the config change template’s run-time execution by receiving specific user input.

Command Declarations (script)

The script declarations include all the variables for which the template prompts the user to provide input. In this case, three variables and their data types are declared:
EnableNetflowOnCiscoASA (  
NCM.Nodes @ContextNode,  
string @NetFlowCollectorIPAddress,  
int @NetFlowExportPort  )
{

NCM.Nodes is applied to the @ContextNode variable. NCM.Nodes refers to the Nodes entity in the SolarWinds Information Service (SWIS). In the interface wizard, the user enters a string value for the NetFlow Collector IP Address and an integer value for the NetFlow Export Port on the device.

For a complete list of entities and properties, see SWIS entities used in config change templates.

CLI Commands

The majority of config change template code outputs original CLI commands with only a few parsed variables. Any time a variable is referenced, a value is used in its place. For example, since the user typed 10.10.18.157 as the IP address and 2055 as the collector port number, NetFlowCollectorIPAddress is replaced with 10.10.18.157 and @NetFlowExportPort is replaced with 2055 when the script runs.

flow-export destination inside @NetFlowCollectorIPAddress @NetFlowExportPort

The previous line of code generates the following output:

flow-export destination inside 10.10.18.157 2055

Create an NCM config change template

NCM ships a variety of config change templates. If none of these meet your needs, you can create a custom config change template.

You must have the NCM role of Administrator or Engineer to create or edit a config change template. You must also understand the basics of writing a script that uses commands, variables, and logical structures. For more information, see examples of config change templates.

If the config changes you need to make do not require advanced logic, you can use scripts to change configurations on multiple devices.
1. Click My Dashboards > Network Configuration > Config Change Templates.

2. Do one of the following:
   - To start with a template definition that includes required variables and examples of other template elements, select Base-Template and click Advanced Modify.
   - To start with a template definition that is similar to the one you want to create, select the existing template and click Advanced Modify.
   - To start with a blank template definition, click Create New Config Change Template.

3. Name the template, enter a description, and add tags.

4. Define the parameters and commands for the config change template.

5. Click Validate to check syntax.
   If validation fails, NCM displays an error that guides you to make changes.

6. Save the template:
   - If you created this template by modifying an existing template definition, click Save As Copy.
   - If you created this template from a blank template definition, click Submit to save the template, or Execute to save and run it.
     If you choose to execute the template, NCM validates the syntax of the template. If validation succeeds, NCM saves a copy of the template and loads the relevant interface for user input.

**Edit an NCM config change template**

Complete the following steps to modify the properties or the logic of an existing config change template.

1. Click My Dashboards > Network Configuration > Config Change Templates.

2. Select a template in the list.

3. Click Advanced Modify.

4. Change the name, edit the description, and add tags, if necessary.
5. Modify the parameters or commands as needed.
6. Click Validate to check syntax.
7. Click Submit.

Tag a config change template

Assigning tags to config change templates enables you to filter search results when working with the templates in NCM. You can select from a set of predefined tags or add your own.

1. Click My Dashboards > Network Configuration > Config Change Templates.
2. Select a template in the list.
3. Click Tags.
4. Select one of the following options, and then add or remove tags:
   - Add Existing
   - Add New
   - Remove
5. Click Submit.

Delete an NCM config change template

You must have the NCM role of Administrator or Engineer to delete a config change template.

1. Click My Dashboards > Network Configuration > Config Change Templates.
2. Select a template in the list.
3. Click Delete.
4. Click Yes at the confirmation prompt.

Import and export NCM config change templates

If you have multiple Orion servers, you can share config change templates between them. You can also share config change templates on THWACK.

Share config change templates between Orion servers

You can export a config change template to a file, and then import it to another Orion server.
Export a config change template as a file

1. Click My Dashboards > Network Configuration > Config Change Templates.
2. Select a template in the list.
3. Click Export as File. Verify that a pop-up blocker does not prevent the file from being downloaded.
4. Download the file to a local folder.
5. Click Return to Config Change Templates.

The config change template is exported to the Downloads directory. The file name is the template name with a `.ncm-template` file extension.

Import a config change template

1. Click My Dashboards > Network Configuration > Config Change Templates.
2. Click Import.
3. Click Choose File, locate the file on your computer, and click Open to select it.

```
> Config change template files have a `.ncm-template` file extension.
```

The selected file name is displayed next to the Choose File button.

4. Click Submit.

The imported template opens in the config change template editor.

5. Change the name, edit the description, and add tags, if necessary.
6. Modify the logic of your Config Change Template.
7. Click Validate to check syntax.
8. Click Submit.

The imported template is listed with other config change templates.

Share config change upgrade templates with the THWACK community

You can share your config change templates with other NCM users by exporting them to THWACK, and you can import config change templates created by other NCM users.
Export a config change template to THWACK

1. Click My Dashboards > Network Configuration > Config Change Templates.
2. Select a template in list.
3. Click Export to THWACK.
4. If prompted, enter your user name and password, and click Log In.

Import a config change template from THWACK

1. Click My Dashboards > Network Configuration > Config Change Templates.
2. Click the Shared Config Change Templates on THWACK tab.
3. Locate the template you need:
   - Use the Search box to search for a string in the template name.
   - Click a tag in the left pane to filter the list.
   - Click the template name to display its description.
4. Select a template and click Import.
5. If prompted, enter your THWACK user name and password, and click Log In.
   The template is copied to your Orion server.

Execute an NCM config change template

When you execute a config change template, you select the nodes that template runs against and enter other values in a run-time wizard.

Before you can execute a config change template against a device, the device must have been inventoried in NCM.

You must have the NCM role of Administrator, Engineer, or WebUploader to execute a config change template.

1. Click My Dashboards > Network Configuration > Config Change Templates.
2. Select a template in list.
3. Click Define Variables & Run.
4. Select the target nodes.
5. Click Next.
6. Enter values in the input fields.

   Input fields are different for each config change template, and they depend on the purpose of the template. For example, in a template that enables NetFlow data exporting for a set of Cisco devices, you might be asked to enter the IP address of the relevant NetFlow collector and the port on which your target device exports flow data.

7. Click Next.

8. If you want to see the CLI commands for a device, expand the device list and click Show Commands in New Window.

9. If you want to save your changes to the startup config, select Write config to NVRAM after execute.

10. Click Execute.
Approval system for device configuration changes

See the following topics to configure and use an approval system for device configuration changes:

- Set up config change approval in NCM
- Turn on the NCM configuration change approval system
- View, approve, and decline network config change requests
- Manage NCM user accounts and assign NCM roles

Set up config change approval in NCM

Network Configuration Manager enables you to define a semi-automated approval process for making configuration changes on network devices. NCM uses roles to determine which Orion accounts are able to perform the tasks of changing device configurations (WebUploader), approving those changes (Administrator), and changing the roles of Orion accounts (Engineer, Administrator). NCM uses email to relay config change approval requests. As part of setting up the config change approval system, you must provide SMTP information.

The setup wizard guides you through the process of specifying an email server for notifications, addresses of change approvers, and accounts of team members who manage device configurations.

1. Click Settings > All Settings.
2. Under Product Specific Settings, click NCM Settings.
4. Select an approval mode, and click Submit.
5. Enter SMTP settings, and click Submit.
6. Enter Admin Email settings.
   - If you are using one-level approval, the addresses in the To field receive notifications of pending device config changes.
   - The address in the From field is shown as the sending address for config change approval requests.
   - The Subject field appears in the subject line of config change approval requests.
   - Add an additional message regarding to inform approvers of the standard action to take.
7. Click Submit.
8. If you selected one-level approval, click Manage Users to adjust the NCM role for accounts that will be submitting upload requests, and follow the steps in Manage NCM user accounts and assign NCM roles. Otherwise, click Finish.

9. If you selected a form of two-level approval, select a user and enter a valid address for Approval Level 1 and Approval Level 2, and click Finish.

   For each user in the list that you want to have a WebUploader role but neither the Engineer or Administrator role, select the user name, and click Edit.

10. When you are finished managing users and adding level 1 and level 2 addresses, click Submit.

   When you enable the change approval system, NCM prevents the system from executing device uploads until an NCM administrator has approved the scheduled job.

**Turn on the NCM configuration change approval system**

After setting up the configuration change approval system in NCM, you need to enable the system to send approval notifications and process acknowledgments. The approval system can be disabled at any time.

**Enable config change approval**

When you enable the change approval system, NCM prevents the system from executing device uploads until an NCM administrator has approved the scheduled job.

> The default setting is to allow any NCM user with the WebUploader role or higher to make and push a device configuration change with no approval necessary. Enabling the Change Approval System without changing the default setting is the same as not enabling the system at all. Review the default settings and user roles and make adjustments as necessary before enabling the approval system.

1. Click Settings > All Settings.
2. Under Product Specific Settings, click NCM Settings.
3. Under Config Management Change Approval, click Enable Approval System.
4. Click Enable.

**Disable config change approval**

When you disable the system, NCM processes device configuration changes normally, either as scheduled or immediately, depending on the actions of the relevant team member with the WebUploader account privileges.
1. Click Settings > All Settings.
2. Under Product Specific Settings, click NCM Settings.
3. Under Config Management Change Approval, click Disable Approval System.

**View, approve, and decline network config change requests**

1. Click Settings > All Settings.
2. Under Product Specific Settings, click NCM Settings.
3. Under Config Management Change Approval, click Pending Approval.
4. Select a request in the list, and click Approve or Decline.
Manage Cisco ACLs and Palo Alto policies in NCM

Use NCM to manage access control list (ACL) rules for Cisco ASA firewalls and Cisco Nexus devices. You can:

- View and compare Cisco ASA ACL rules
  You can compare ACLs (or different versions of the same ACL), and view and manage the rules within an ACL. NCM automatically detects redundant and shadowed rules.
- Learn about overlapping (shadowed and redundant) ACL rules
  Eliminating shadowed and redundant rules reduces the size of the rule set, making it easier to manage, and helps you ensure that the rules achieve the intended results.
- View information about the policies defined for Palo Alto devices that run OS 7.1 and later.

Manage ACLs on Cisco ASA and Nexus devices

Use NCM to help you manage the access control lists (ACLs) for your Cisco ASA and Cisco Nexus devices. Find rules that are not being applied as intended, and identify unnecessary or redundant rules that can be removed. Streamlining ACLs makes them easier to manage and saves CPU and memory on your devices.

- Display the ACLs on a device.
- Compare ACLs on the same device or different devices.
- Display the rules in an ACL.
- Identify rules that have not been applied.
- Investigate overlapping rules in an ACL.
- Display information about objects or objects groups in a rule.

Display the access control lists on a device

Before you can complete other management tasks, you must display a list of the ACLs on a device.

2. Double-click the name of a Cisco ASA or Cisco Nexus device.
   The Node Details page opens.
3. From the menu on the left, choose Access Lists.
The Access Lists page lists the ACLs configured for that device. If an ACL has changed, click the arrow to display a list of previous versions.

A warning icon ▲ indicates that the ACL contains **overlapping rules**. You can display the ACL rules to find out which rules overlap.

### Compare ACLs

Use NCM to quickly locate the differences between ACLs or ACL versions. For example, you can compare two versions of the same ACL to determine what changed, or to verify that changes were implemented correctly. You can compare ACLs on different nodes to verify that the same rules are being applied on both devices.

#### Compare ACLs on the same device

1. Display the list of ACLs on a device.
2. Select the ACLs or ACL versions to compare.

   To compare the current version to a previous version, expand the node to list previous versions.

3. At the top of the page, click Compare ACL.

   The rules from both ACLs are displayed beside each other. The line number of rules with differences are highlighted.

#### Compare ACLs on different devices

To compare ACLs on different devices, first select two ACLs on the same device, and then change one of the ACLs being compared.
1. **Display the list of ACLs** on a device.

2. Select the ACL that you want to compare, and any other ACL on that node. (You will change the second selection later.)

3. At the top of the page, click Compare ACL.

   The rules from both ACLs are displayed beside each other. The line number of rules with differences are highlighted.

4. Change the second ACL to an ACL on a different device:
   
   a. Near the top of the page, click Change ACLs Compared.
   
   b. Select the node, interface, name, and version of the ACL you want to compare.
   
   c. Click Change.

**Display ACL rules**

When you display ACL rules, also known as Access Control Entries (ACEs), SolarWinds NCM identifies **overlapping rules**, which might require additional investigation.

1. **Display the list of ACLs** on a device.

2. Click an ACL name.

   The rules (or ACEs) are listed on the Rules of This Access List page. The right column shows the number of hits, and a warning icon `⚠` indicates that the rule **overlaps another rule**.

   Use the search and filter options to find a specific rule quickly. Or use filters to display all rules that meet certain criteria.

   Click Edit Filter Properties to change the options available for filtering.

**Identify rules that have not been applied**

1. **Display ACL rules**.

   On the right side of each line, the ACL browser displays the hit count for the rule. The hit count indicates the number of times the rule has been applied.

   By default, rules are sorted by line number.

2. To sort the list by hit count, click the down-arrow on the sort menu and choose Hit Count.
Rules that have never been applied (0 hits) are at the top of the list.

Investigate overlapping rules

1. Display ACL rules.
2. Point to the warning icon ❗ that identifies a rule that overlaps another rule. A message describes the issue.
3. Click Show the details. A dialog box displays the preceding rules that shadow the rule or make it redundant.

Display information about objects or object groups

1. Display ACL rules.
2. If an object or object group is listed as the source or destination of a rule, click the name of an object or object group. Information about the current version of the object or object group is shown on the right.
3. To view information about a previous version of the object or object group, select a version from the drop-down menu.
4. To compare another version to the currently selected version, click Compare Diff and select the version.
Overlapping ACL rules

Cisco ASA and Nexus devices evaluate rules in order, from top to bottom. Overlapping rules occur when some or all of the traffic that would have been processed by one rule has already been processed by a previous rule. When you view ACL rules for a Cisco ASA or Nexus device, SolarWinds NCM displays a warning icon to identify overlapping rules.

Finding and eliminating overlapping rules reduces the size of the rule set, making it easier to manage, and also helps you ensure that the rules achieve the intended results.

NCM detects four types of overlapping rules on Cisco ASA and Nexus devices:

- **Fully shadowed**
- **Partially shadowed**
- **Fully redundant**
- **Partially redundant**

> When detecting overlapping rules, NCM supports both contiguous and discontiguous masks.

### Fully shadowed rules

A fully shadowed rule is detected when:

- The criteria for one rule matches all of the traffic covered by a second rule.
- The two rules apply different actions.

The second rule is fully shadowed by the first. The rules conflict, but the shadowed rule is never applied to any traffic because it comes later in the access list. For example:

```plaintext
access-list acl1 permit ip 1.0.0.0 255.0.0.0 any
access-list acl1 deny ip 1.229.0.0 255.255.0.0 any
```

### Partially shadowed rules

A partially shadowed rule is detected when:

- The criteria for one rule matches some of the traffic covered by a second rule.
- The two rules apply different actions.
The second rule is partially shadowed by the first. It is applied to only some of the intended traffic. For example:

```
access-list acl102 permit ip 10.0.0.0 255.255.255.0 any
access-list acl102 deny ip any any
```

In some cases, a partially shadowed rule might be intentional. For example, you might want to permit traffic from specific IP addresses, but deny all others.

### Fully redundant rules

A fully redundant rule is detected when:

- The criteria for one rule matches all of the traffic covered by a second rule.
- The two rules apply the same action.

The second rule is fully redundant because of the first. It is never applied to any traffic. For example:

```
access-list acl3 permit ip 1.0.0.0 255.0.0.0 any
access-list acl3 permit ip 1.229.0.0 255.255.0.0 any
```

### Partially redundant rules

A partially redundant rule is detected when:

- The criteria for one rule matches some of the traffic covered by a second rule.
- The two rules apply the same action.

The second rule is partially redundant because of the first. It is applied to only some of the intended traffic. For example:

```
access-list acl4 deny ip 192.168.169.0 255.255.255.0 any
access-list acl4 deny tcp any any range echo acl
```

### View Palo Alto policies

Use NCM 8.0 and later to view information about the policies defined for Palo Alto devices that run OS 7.1 and later.
What policies can I view in NCM?

NCM displays information about the policies defined in the default virtual system (vsys) of the Palo Alto device. If you have defined policies in a non-default vsys, they are not available to NCM.

NCM displays information about Security Policies on Palo Alto devices. NCM does **not** display information about the following types of policies:

- NAT
- QoS
- Policy Based Forwarding
- Decryption
- Tunnel Inspection
- Application Override
- Authentication
- DoS Protection

View policies

1. Click My Dashboards > Network Configuration > Config Summary.

2. In the NCM Node List, click a Palo Alto device.

   The Node Details page displays information about the selected device.

3. In the menu on the left, choose Policies.

   The Policies view displays a summary of each policy, including the name, source zones, destination zones, and origin.

   - Policies pushed from a Panorama management server have an origin of either 'Pre Policy (Panorama)' or 'Post Policy (Panorama)'.
   - Policies that are not pushed from a Panorama management server have an origin of 'Local'.

   ! Information is available on the Policies view only **after** you have downloaded configurations from the Palo Alto device.

4. Use filters or search to locate the policy that you’re interested in, and click the policy name.

   The Policy Details page displays information about the policy definition, as well as other information to help you evaluate and manage your policies. For example:
The Policy Changes widget displays the time and date of that affect this policy. This includes changes that were made directly to the policy, and changes to tags, applications, services, and other objects that are referenced by the policy.

For changes made directly to the policy, click the View diff link to open the Compare Configs window and see what changed.

The Other Firewalls Using this Policy widget lists the devices that would be directly affected by changes to this policy.

Other firewalls are listed only if their configurations have been downloaded to NCM. NCM uses the policy name to determine if other firewalls are using the policy.
If you have SolarWinds NTA with netflow enabled, the Top XX Conversations on Policy widget shows application traffic conversations that are affected by the selected policy. When a policy changes, use this widget to see how the change affects network traffic.
NCM reports

The Orion Platform database accumulates much information that can be presented in a variety of formats. See the following topics to learn more about NCM reports:

- Run an NCM config change report once
- Schedule a recurring network config change report
- Access predefined NCM reports
- Predefined NCM reports

For more information about working with reports in the Orion Platform, see Manage and view reports in the Orion Platform.

Run an NCM config change report once

1. Click My Dashboards > Network Configuration > Jobs.
2. Click Create New Job.
3. Name the job, and elect Generate a Config Change Report from Job Type.
4. Select Basic as the Schedule Type.
5. Select Once, and then enter a day and time that is at least 15 minutes from the current NCM server time.
6. Add a comment if this job relates to a business rule, and click Next.
7. Select the nodes to target with this job, and click Next.
8. Select an email notification option, and click Next. If you click Email Results, the default email notification and SMTP server settings are populated. These settings can be overridden in each job.
9. Select the type of config change report to generate, and click Next.
10. Select Only Send Devices That Had Changes if desired, and click Next.
11. Review the settings for the job, and click Finish.

To create a config change report to run recurrently, see Schedule a recurring network config change report.
If NCM is configured to connect to nodes with user-level login credentials, when NCM processes a job it connects to the node using the credentials of the user who last edited the job. Otherwise, NCM connects to the node using settings in the device connection profile or the associated global connection profile.

### Schedule a recurring network config change report

1. Click My Dashboards > Network Configuration > Jobs.
2. Click Create New Job.
3. Name the job, and select Generate a Config Change Report from Job Type.
4. Specify when the job runs:
   - To run the job once or on a simple schedule, select Basic. Click the tab that identifies how frequently the job runs, and then specify the start time and (if needed) the day(s).
   - To create a more complex schedule, select Advanced and then use the five fields to create a CRON expression.
5. Add a comment if this job relates to a business rule, and click Next.
6. Select the nodes to target with this job, and click Next.
7. Select an email notification option, and click Next. If you click Email Results, the default email notification and SMTP server settings are populated. These settings can be overridden in each job.
8. Select the type of config change report to generate, and click Next.
9. Review the settings for the job, and click Finish.

### Access predefined NCM reports

To access predefined NCM reports, complete the following steps.

> For more information about working with reports in the Orion Platform, see Manage and view reports in the Orion Platform.

1. Choose Reports > All Reports to open the All Reports page for the Orion Platform.
2. To locate NCM reports, group by Report Category, and then scroll down to the report category names that begin with "NCM" (for example, NCM Audit).
3. Click the category name to display the list of reports in that category.

4. Click the report title to open the report.

**Predefined NCM reports**

The following predefined reports are included with Network Configuration Manager.

To access these reports, see [Access predefined NCM reports](#).

For more information about working with reports in the Orion Platform, see [Manage and view reports in the Orion Platform](#).
<table>
<thead>
<tr>
<th>Category</th>
<th>Reports</th>
</tr>
</thead>
<tbody>
<tr>
<td>NCM Audit</td>
<td>These reports display the results of audit checks.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Config Transfer Audit</strong>: Displays information on config transfers.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Real Time Change Detection Audit</strong>: Displays change notification messages on nodes within the network.</td>
</tr>
<tr>
<td></td>
<td>- <strong>User Activity Tracking Report</strong>: Displays relevant information on node activity within the network.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Vulnerability State Change Audit</strong>: Displays information about vulnerability state changes.</td>
</tr>
<tr>
<td>NCM Brocade Inventory</td>
<td>These reports display information about the physical entities and agent config modules running on the chassis of Brocade devices.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Brocade Agent Config Module</strong>: Displays information on Brocade agent config modules running in chassis.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Brocade Chassis Serial Number</strong>: Displays the serial number for a Brocade chassis.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Brocade Chassis Unit Serial Number</strong>: Displays the serial numbers for units within a Brocade chassis.</td>
</tr>
<tr>
<td>Category</td>
<td>Reports</td>
</tr>
<tr>
<td>------------------</td>
<td>--------------------------------------------------------------------------</td>
</tr>
<tr>
<td>NCM Cisco Inventory</td>
<td>These reports display information about the physical properties of the system.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Cisco 3750 Stack - Physical Entity</strong>: Displays information about physical entities within each device.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Cisco Bootload Images</strong>: Displays information related to bootload images within each device.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Cisco Card Data</strong>: Displays information about Cisco card data within each device.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Cisco Catalyst Cards</strong>: Displays the card in each Catalyst switch.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Cisco Chassis IDs</strong>: Displays the chassis ID and serial number for Cisco devices.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Cisco Discovery Protocol</strong>: Displays the CDP discovered from each device.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Cisco Fan Tray Status</strong>: Lists the operational status information for a device’s fans or fan trays.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Cisco Flash File System</strong>: Displays the list of files in the flash memory of Cisco devices.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Cisco Flash Memory</strong>: Displays the amount of flash memory in Cisco devices.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Cisco FRU Power Status</strong>: Lists the power-related administrative status and operational status of the manageable components in the system.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Cisco FRU Power Supply Groups in the system</strong>: Displays the redundancy mode and the operational status of the power supply groups in the system.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Cisco IOS Image Details</strong>: Displays details of IOS in each Cisco device.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Cisco Memory Pools</strong>: Displays memory pools in Cisco devices.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Cisco VLANs</strong>: Displays the VLANs to which devices belong.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Memory in Cisco Devices</strong>: Displays memory pools in Cisco devices.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Old Cisco Cards</strong>: Displays the cards in Cisco devices.</td>
</tr>
<tr>
<td></td>
<td>- <strong>ROM Bootstrap for Cisco Devices</strong>: Displays the ROM bootstrap version for selected devices.</td>
</tr>
<tr>
<td>Category</td>
<td>Reports</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>NCM F5 Inventory</td>
<td>These reports provide information about the physical entities of the discovered F5 systems and LTM and GTM object activity details.</td>
</tr>
<tr>
<td></td>
<td>1. <strong>Capacity Planning</strong>: Displays capacity details such as CPU usage, memory, and throughput relevant to the F5 devices in your network.</td>
</tr>
<tr>
<td></td>
<td>2. <strong>Device Inventory</strong>: Displays information about the physical entities, software, and state of each F5 device discovered by NCM in your network.</td>
</tr>
<tr>
<td></td>
<td>3. <strong>GTM Object Activity</strong>: Displays details about Global Traffic Manager object activity that indicates usage rates for the relevant devices.</td>
</tr>
<tr>
<td></td>
<td>4. <strong>LTM Node Inventory</strong>: Displays status and configuration information about the Local Traffic Manager node.</td>
</tr>
<tr>
<td></td>
<td>5. <strong>LTM Object Activity</strong>: Displays activity details that indicate usage rates for Local Traffic Manager objects in your NCM-managed network.</td>
</tr>
<tr>
<td>NCM Inventory</td>
<td>These reports provide information about physical entities and network services configured on NCM discovered devices.</td>
</tr>
<tr>
<td></td>
<td>1. <strong>ARP Tables</strong>: Displays ARP tables from each device.</td>
</tr>
<tr>
<td></td>
<td>2. <strong>Interfaces</strong>: Displays details about each interface.</td>
</tr>
<tr>
<td></td>
<td>3. <strong>IP Addresses on Each Interface</strong>: Displays the IP addresses assigned to each device.</td>
</tr>
<tr>
<td></td>
<td>4. <strong>Juniper Physical Entities</strong>: Displays information about Juniper physical entities within each device.</td>
</tr>
<tr>
<td></td>
<td>5. <strong>Last Status Change for Each Interface</strong>: Displays the time the status last changed on each interface.</td>
</tr>
<tr>
<td></td>
<td>6. <strong>Logical Entities</strong>: Displays the logical entities within each device.</td>
</tr>
<tr>
<td></td>
<td>7. <strong>Physical Entities (Serial Number)</strong>: Displays information about physical entities within each device.</td>
</tr>
<tr>
<td></td>
<td>8. <strong>Physical Entities (Serial Number) v2</strong>: Displays information about physical entities within each device.</td>
</tr>
<tr>
<td></td>
<td>9. <strong>Switch Ports</strong>: Displays spanning tree and VLAN information on each switch port.</td>
</tr>
<tr>
<td></td>
<td>10. <strong>TCP Services</strong>: Displays the TCP services on each device.</td>
</tr>
<tr>
<td></td>
<td>11. <strong>UDP Services</strong>: Displays the UDP services on each device.</td>
</tr>
<tr>
<td>Category</td>
<td>Reports</td>
</tr>
<tr>
<td>------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>NCM Node Details</td>
<td>These reports provide details such as status, availability, response times, events, and alerts for each node under NCM management.</td>
</tr>
<tr>
<td></td>
<td>- <strong>ACLs with overlapping rules</strong>: Shows a summary of ACLs with overlapping rules for all Cisco and Nexus devices.</td>
</tr>
<tr>
<td></td>
<td><strong>All Nodes</strong>: Displays all nodes managed by NCM.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Backup Status of Running Config</strong>: Displays the timestamp for when the current Running config was backed-up or an indication that it has never been backed up.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Backup Status of Startup Config</strong>: Displays the timestamp for when the current Startup config was backed-up or an indication that it has never been backed up.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Baseline vs. Config Conflicts</strong>: Lists configs that differ from the associated baselines.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Connection Profiles</strong>: Displays the Connection Profile used by each device.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Current IOS Image and Version</strong>: Displays the current IOS image and version.</td>
</tr>
<tr>
<td></td>
<td>- <strong>End of Support Devices</strong>: Displays devices that are at the end of their support.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Last Boot Time for Each Device</strong>: Displays the timestamp for when the device was last boot.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Last Inventory of Each Device</strong>: Displays the timestamp for when the device was last inventoried.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Overall Configuration Changes Snapshot</strong>: Displays all changes to the Running, Baseline, and Startup configs.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Overall Devices Backed Up vs. Not Backed Up</strong>: Displays whether nodes have been backed up.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Overall Devices Inventoried vs. Not Inventoried</strong>: Displays whether nodes have been inventoried.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Overall Running vs. Startup Config Conflicts</strong>: Displays where the Running config differs from the Startup config.</td>
</tr>
<tr>
<td></td>
<td>- <strong>System Information of Each Device</strong>: Displays the System Description, Location and Contact Information discovered from each device.</td>
</tr>
<tr>
<td>Category</td>
<td>Reports</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>NCM Polling Status</td>
<td>These reports provide information about polling and rediscovery cycles.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Devices that do not respond to SNMP</strong>: Displays a list of devices that are not responding to SNMP. Use this report to validate the SNMP Community string on each device.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Down Nodes</strong>: Displays a list of currently down nodes as recorded in the database.</td>
</tr>
<tr>
<td>NCM Route Tables Inventory</td>
<td>- <strong>Route Tables</strong>: Displays route tables for relevant devices.</td>
</tr>
<tr>
<td>NCM Security</td>
<td>These reports provide information about logins and security settings.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Community Strings for Each Node</strong>: Displays the SNMP community string used by each device.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Login Failure Report</strong>: Displays the devices that cannot be logged into and the error information.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Login Information for Each Device</strong>: Displays the user name, enable level, and community strings for each device.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Login Status</strong>: Displays any error messages relating to login information for each device.</td>
</tr>
<tr>
<td>NCM Windows Server Inventory</td>
<td>These reports provide information about the software and services installed on Windows nodes.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Installed Services</strong>: Displays installed services for Windows nodes.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Installed Software</strong>: Displays installed software for Windows nodes.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Windows Accounts</strong>: Displays Windows accounts for relevant nodes.</td>
</tr>
</tbody>
</table>
Alerts in NCM

Orion Platform alerting capabilities can be used to alert you to potential issues with configurations. The following topics provide information about NCM alerts and alert actions:

- **Types of NCM alert actions**
- **Apply the default NCM alert**
- **Add scripted commands to the default NCM alert**

**Types of NCM alert actions**

In executing one of its alert-related actions, NCM requires a role with sufficient permissions and cannot use device access credentials to authorize its action.

As a security enhancement related to executing NCM actions, NCM account passwords are not stored in the database. As part of configuring NCM, the installation software removes passwords from the database as part of the Configuration Wizard session.

Sequence the actions with an awareness that some NCM actions require others to complete first. For example, NCM cannot execute a notification email action before it downloads the config from an NCM-managed device.

You can use three types of NCM actions in processing an Orion Platform alert:

- Backup Running Config
- Execute Config Script
- Show Last Config Changes

**Backup Running Config**

NCM downloads the latest configuration from the context node. It is the same as running Node Details > Configs > Download Config. Unlike a normal execution of this action, however, the results of this download are written to an alerts table in the Orion database and this data is used when an alert is processed.

**Execute Config Script**

NCM executes the command(s) that you entered in the Command Script to Execute field. For example, if you enter `show version`, and include it as a Trigger Action on an alert, NCM runs the `show` command as part of alert processing and includes the results with the alert notification.
Show Last Config Changes

NCM performs a SQL query to find the most recent changes and compares those changes either to the baseline config or the next-to-last downloaded config, depending on how you set up your alert action.

When the alert is triggered, the results of the NCM action are stored in the Orion database (in ${Notes}) and used as part of runtime processing of an alert. You can also view this information as part of the Alert Details on any relevant alert reported through the Orion Web Console at Alerts & Activity > Alerts.

If an alert is triggered for a node without relevant config history, NCM cannot contribute any data and the Orion alert is processed without it. So selecting this action only makes sense if you already have a history of device configurations.

Apply the default NCM alert

When you install NCM, the software automatically creates a predefined alert called Alert Me and Trigger an NCM Action in the Orion Alert Manager.

By default, if changes in the device configuration trigger it, this predefined alert does these three things in order:

1. Backs up the running config on the alerting device.
2. Determines the last config changes made on the device.
3. Sends an email regarding the alert to a relevant administrator that includes the results of both NCM actions.

Apply the default NCM alert

1. Click Alerts & Activity > Alerts.
2. Click Manage Alerts.
3. Select Object Type from the Group By list, and select Node.
4. Select Alert Me and Trigger an NCM Action.
5. If you want to edit the configuration, click Edit Alert.
6. If you want to make a copy of the alert before editing it, click Duplicate & Edit.
7. Edit the Alert Properties, and set Enable On/Off to On.
8. On Trigger Condition, define the conditions in which the software launches the alert. The default conditions are any node in the NCM environment that responds in 200 ms or more.
9. On Reset Condition, define the conditions in which the software resets the alert. The default condition is that the triggered nodes respond in 100 ms or less.

10. On Time of Day, define the days and times during which the software actively evaluates the database for trigger conditions. The default range is 24/7.

11. On Trigger Actions, create actions to execute when the software triggers the alert, and enter your NCM credentials.

12. On Reset Conditions, define actions to execute when the software resets the alert. Default actions are to back up the config running on the alerting device, determine the last config changes, and send an email to a contact. For other available NCM actions, see Types of NCM alert actions.

13. On Reset Actions, create actions to execute when the resets the alert, and click Next. By default, resetting an alert sends an email notification.

14. On Summary, review and edit settings. We recommend enabling Alert Integration.

15. Click Submit.

**Add scripted commands to the default NCM alert**

You can modify the default NCM advanced alert to execute specific command scripts at the time the alert is triggered. The following example adds a simple `show version` command as a scripted action.

1. Click Alerts & Activity > Alerts.

2. Click Manage Alerts.

3. Select Object Type from the Group By list, and select Node.

4. Select Alert Me and Trigger an NCM Action.

5. Click Duplicate & Edit.


7. Select Execute an NCM Action, and click Configure Action.

8. Select Execute Config Script, and enter a command in Command Script to Execute. For example:

   ```
   show version
   ```

   When executed, this command runs on the context node, receives detailed software and hardware information, and includes it in the `$\{Notes\}` macro of an Orion database alerts table.

9. Click Add Action.
10. Drag the new action into the desired position in the list.

   By default, the software positions a new action at the end of the action list. In this case, it makes sense to position this action third, after NCM backs up the running config and determines the last config changes.

11. Click Summary.

12. On Summary, review and edit settings. We recommend enabling Alert Integration.

13. Click Submit.
Servers for NCM file transfers and email notifications

Use the following topics to configure servers for file transfers and email notifications regarding real-time change detection:

- Specify the SolarWinds TFTP server settings for NCM config transfers
- Use the SolarWinds SFTP & SCP server for NCM config transfers
- Use a third-party SCP server for NCM config change transfers
- Configure NCM email notifications

Specify the SolarWinds TFTP server settings for NCM config transfers

When you install NCM, the SolarWinds TFTP server is automatically installed on the Orion server and on any additional polling engines (APEs). If TFTP (Trivial File Transfer Protocol) is used to transfer configuration files to and from your network devices, NCM uses the SolarWinds TFTP server.

> A device's **connection profile** specifies what protocol NCM uses to transfer configs to and from a device. For more information about the protocols NCM can use, see [Protocols NCM uses to transfer configurations](#).

Verify the TFTP server status and configure settings

Open the SolarWinds TFTP server application to verify that the service is running, view recent log entries, and configure settings to meet your organization's needs.

1. From the `<OrionInstallationDir>\TFTP Server folder, open the TFTPServer.exe application. The default location is:

   C:\Program Files (x86)\SolarWinds\TFTP Server\TFTPServer.exe

   The main window lists log entries for any recent activity.
2. Click File > Configure.

3. If the TFTP Server Service Status is Stopped, click Start.

   - You can also control TFTP as a service from Administrative Tools and set the service to start automatically.

4. Select Add TFTP Server to Windows System Tray to provide access to the right-click context menu from the system tray.

   - The TFTP Server Root Directory displays the default file transfer directory, which was created during installation. NCM uses this directory for temporary file storage during a transfer. In most cases, there is no need to change the directory that NCM uses. However, if you do need to change it, you must change the TFTP settings in NCM.

5. Click the Server Bindings tab, and then set the IP address from which the server will manage transfers.

6. Click the Security tab, and set any relevant restrictions on transfers.

7. Click OK.

View or change the TFTP server and storage directory locations in NCM

When you install NCM, the TFTP server and config transfer directory are set up automatically, and the TFTP Server Settings page reflects these default values. In most cases, you do not need to change the default values.
1. Click Settings > All Settings.

2. Under Product Specific Settings, click NCM Settings.


   The TFTP Server Settings page shows the default settings for the SolarWinds TFTP server installed on your Orion server and any additional polling engines (APEs).

   ![TFTP Server Settings](image)

4. If you want to manually enter each IP address, select Allow Me to Specify the IP Address of the TFTP Server. Otherwise, NCM automatically resolves each IP address via DNS.

5. Verify the settings for the Orion server and each APE:
   - Under TFTP Server, most environments use the default value.
     - When you install NCM, the SolarWinds TFTP server is automatically installed on the Orion server and on any APEs. The TFTP server on each APE is used to transfer configs to and from devices controlled by that APE.
     - To change the default, select Allow Me to Specify the IP Address of the TFTP Server, and then enter the IP address where the SolarWinds TFTP server you want to use is installed.
   - Under Specify a config transfer directory, specify the location where configurations are temporarily stored during transfer.
     - When you install NCM, the default config transfer directory is created automatically. If you change the default location, click Validate to verify that NCM can access the new
To specify a shared directory, enter the network path and make sure that NCM has access to the directory.

6. Click Submit to save the settings.

**Use the SolarWinds SFTP & SCP server for NCM config transfers**

When you install NCM, the SolarWinds SFTP & SCP server is automatically installed on the Orion server and on any additional polling engines (APEs). If NCM uses SCP (Secure Copy Protocol) to transfer configuration files, you can use the SolarWinds SFTP & SCP server or you can use a third-party SCP server.

A device’s **connection profile** specifies what protocol NCM uses to transfer configs to or from a device. For more information about the protocols NCM can use, see Protocols NCM uses to transfer configurations.

**Verify the SCP server status and configure settings**

The SFTP & SCP Server runs as a service. You can open the SFTP & SCP Server application to verify that the service is running, view recent messages, and, if necessary, configure settings to meet your organization’s needs. In many cases, organizations can use the default settings.

An SFTP & SCP Server is not an FTP server. You cannot connect to an SFTP & SCP server with an FTP client.

1. On the Orion server, start the SolarWinds SFTP & SCP Server. To do so:
   a. Expand the Windows System Tray.
   b. Right-click the SolarWinds SFTP/SCP Server icon.
   c. Select Open SolarWinds SFTP/SCP Server.

If the icon is not in the System Tray, open the `<OrionInstallionDir>\SFTP & SCP Server` folder, and then run the `ScpServerTray.exe` application. The default location is:

`C:\Program Files (x86)\SolarWinds\SFTP & SCP Server\ScpServerTray.exe`

The main window lists the most recent messages.
2. If the service is stopped, open the Service status drop-down menu and select Start.

   You can also open Windows Administrative Tools and set the service to start automatically.

3. Click File > Configure to open the SFTP/SCP Server Settings dialog box.

   The Root Directory box displays the default root directory, which was created during installation. In most cases, there is no need to change the root directory. However, if you do need to change it, you must also change the SCP settings in NCM.

4. Select the protocols you want the server to support from the Allowed Protocols list.

5. Select options you want to enable in the Permitted File Transfer Operations section.

6. Click the TCP/IP Settings tab, and then specify the port number you want to use in the TCP Port field.

7. If you want to specify the IP address configuration, select Use Custom IP Address Binding, and select the IP address you want to use.

8. If you want to enable user authentication on the server, complete the following:
   a. Click the Users tab.
   b. Click New User.
   c. Type the user name and password, and click Apply Changes.
   d. If you want to remove a user, select the user, and click Remove.
9. On the Startup & System Tray tab:
   a. If you want the SFTP & SCP Server application window to open automatically, select
      Automatically run this application when I log on to Windows.
      The SFTP &SCP Server service can still run in the background even if the application
      window is closed.
   b. Select what you want the application to do when you close the application window.

10. Click OK.

View or change the SCP server IP address and storage directory in NCM

When you install NCM, the SolarWinds SCP server is automatically installed on the Orion server and
on any APEs. The SCP server on each APE is used to transfer configs to and from devices controlled
by that APE. The default config transfer directory is set up automatically on each server.

The SCP Server Settings page displays these default values. You can modify these values if needed.

1. Click Settings > All Settings.
2. Under Product Specific Settings, click NCM Settings.

   The SCP Server Settings page shows the default settings for the SolarWinds SFTP & SCP server
   installed on your Orion server and any additional polling engines (APEs).
4. To manually enter each IP address, select Allow Me to Specify the IP Address of the SCP Server. Then enter the IP address for the Orion server and each APE.

   By default, NCM attempts to automatically resolve each IP address via DNS. You might need to enter the IP address, for example, if a device has multiple network interfaces and the IP address cannot be automatically detected.

5. If you want to change the default values, enter the Username and Password that NCM will use to access the SCP server.

   NCM cannot use public key authentication to access the SolarWinds SCP server.

6. Under Specify a config transfer directory, specify the location where configurations are temporarily stored during transfer.

   When you install NCM, the default config transfer directory is created automatically. If you change the default location, click Validate to verify that NCM can access the new location.

   To specify a shared directory, enter the network path and make sure that NCM has access to the directory.

7. Click Submit to save the settings.
Use a third-party SCP server for NCM config change transfers

If NCM uses SCP (Secure Copy Protocol) to transfer configuration files, you can use the SolarWinds SFTP & SCP server or you can use a third-party SCP server.

A device's connection profile specifies what protocol NCM uses to transfer configs to or from a device. For more information about the protocols NCM can use, see Protocols NCM uses to transfer configurations.

Complete the following steps to specify the third-party SCP server’s location and login credentials. The user must have receive and transmit permissions configured in the third-party SCP server.

1. Click Settings > All Settings.
2. Under Product Specific Settings, click NCM Settings.
   
   The SCP Server Settings page shows the default settings for the SPC servers used by your Orion server and any additional polling engines (APes).

4. Select Use Third-Party SCP Server.
5. Select Allow Me to Specify the IP Address of the SCP Server to prevent NCM from resolving the SCP server based on its own host and disabling other entries.

6. Enter the user name and password that NCM should use to connect to the SCP server.

7. Specify a config transfer directory, and click Validate to verify that the SCP Root Directory is set to be the same as the third-party server.

8. Click Submit to save the settings.

**Configure NCM email notifications**

SMTP server credentials used to send notifications for config change approvals, real time change detection, and running jobs

1. Click Settings > All Settings.

2. Under Product Specific Settings, click NCM Settings.

3. Under Manage Notifications, click SMTP Server.

4. Enter the fully qualified domain name (FQDN) of the server in Email Server Address.

5. Enter the Port Number on which the server handles messages.

6. Select an Authentication type.

7. If you selected Password as your Authentication type, enter a user name and password that the server accepts.

8. Click Submit.

**Email notification defaults used in new scheduled jobs**

1. Click Settings > All Settings.

2. Under Product Specific Settings, click NCM Settings.

3. Under Manage Notifications, click Email Notification.

4. Enter in Sender Name the name you want to appear in a job completion notification.

5. Enter a Reply Address if desired.

6. Enter the Subject to be used to send a job completion notification.

7. Enter recipients in the To field.

8. Click Submit.
EnergyWise devices

This topic explores how you can use Network Configuration Manager in conjunction with Network Performance Monitor to enable and manage your Cisco EnergyWise devices.

What is EnergyWise?

EnergyWise is Cisco’s response to the call to cut energy costs, address environmental concerns, and adhere to government directives around green technologies. By purchasing EnergyWise capable devices and enabling their energy-saving features, you can retain business critical systems in a fully powered state while allowing less critical power over ethernet (PoE) devices to power down or drop into standby during off hours.

EnergyWise gives you the ability to control your energy cost. NCM gives you the ability to remotely apply recurrence policies and schedule power usage, helping you use less power. And, SolarWinds NPM allows you to monitor your energy use and power levels. SolarWinds perfectly partners with Cisco and the EnergyWise technologies to help you save more and monitor your savings.

Manage and enable EnergyWise nodes

Cisco devices that support the EnergyWise technology can be enabled and their EnergyWise settings managed through the NCM integration with NPM.

Before completing the following procedure, EnergyWise nodes must be managed in both NCM and SolarWinds NPM. You must discover and add nodes to the Orion Platform, and then manage the nodes with NCM.

1. Click Settings > All Settings.
2. Under Node & Group Management, click Manage Nodes.
3. Select the Cisco node for which you want EnergyWise enabled, and click More Actions > Manage EnergyWise.
4. Click Enable EnergyWise on these nodes.
5. Specify the values on the Manage EnergyWise Node page.
6. Click Execute Config Actions.
Manage Power over Ethernet ports

Power over Ethernet (PoE) devices are connected to your devices on an interface and are managed at the interface level. Before completing the following procedure, you must have added your EnergyWise capable nodes to both NCM and NPM.

1. Click Settings > All Settings.
2. Under Node & Group Management, click Manage Nodes.
3. Expand the Cisco node containing the interface you want to configure.
4. Select the interface you want to enable EnergyWise, and click More Actions > Manage EnergyWise.
5. Click Enable EnergyWise on these nodes.
6. Specify the values on the Manage EnergyWise Interface page.
7. Click Execute Config Actions.
Common NCM tasks

The following topics present example scenarios to help demonstrate how you can use Network Configuration Manager in different network environments.

- Change the community string on multiple nodes
- Block a MAC address on a wireless access point
- Change an interface description
- Block all private addresses with an access control list
- Customize the login banner of a device

Change the community string on multiple nodes

This procedure replaces the public read-only community string with a new read-only community string on several network nodes at the same time.

1. Click My Dashboards > Network Configuration > Jobs.
2. Click Create New Job.
3. Name the job, and select Execute Command Script on Devices from Job Type.
4. Specify when the job runs:
   - To run the job once or on a simple schedule, select Basic. Click the tab that identifies how frequently the job runs, and then specify the start time and (if needed) the day(s).
   - To create a more complex schedule, select Advanced and then use the five fields to create a CRON expression.
5. Add a comment if this job relates to a business rule, and click Next.
6. Select the nodes to target with this job, and click Next.
7. Select an email notification option, and click Next. If you click Email Results, the default email notification and SMTP server settings are populated. These settings can be overridden in each job.
8. Add the script to execute.
   - If you want to load a saved script, click Load Saved Script, and select a script.
   - If you want to load an existing script from disk, click Load Script from File, browse to the file, and click Open.
   - If you want to create a new script, enter your script in the text box. For example:
config
t
no snmp-server community public RO
snmp-server community 123@dm1n RO
exit
wr mem

Where 123@dm1n is the new community string.

9. If you want to save a script, click Save Script, specify a location, and click Save.

10. Select Filter Results that match a pattern if you want to see in the script output only those lines that match a specific regular expression pattern.

11. Select Show Commands in Output to view what NCM sent to the targeted devices.

12. Click Next.

13. Review the settings for the job, and click Finish.

Verify success by comparing configs

1. Click My Dashboards > Network Configuration > Configuration Management.

2. Select one or more nodes, and click Compare Node(s) Configs.

3. Select a config from each list.

4. Click Compare Selected. Changes to are highlighted in red, green, and yellow.

Block a MAC address on a wireless access point

If you discover a device utilizing unauthorized access through your wireless network, you can block the MAC address to prevent future access. The following procedure uses an access control list on a wireless access point to block a specific MAC address.

Back up the running configuration

1. Click My Dashboards > Network Configuration > Configuration Management.

2. Select one or more nodes.

3. Click Download > Running.
Block a MAC address on a wireless access point

1. Click My Dashboards > Network Configuration > Jobs.
2. Click Create New Job.
3. Name the job, and select Execute Command Script on Devices from Job Type.
4. Specify when the job runs:
   - To run the job once or on a simple schedule, select Basic. Click the tab that identifies how frequently the job runs, and then specify the start time and (if needed) the day(s).
   - To create a more complex schedule, select Advanced and then use the five fields to create a CRON expression.
5. Add a comment if this job relates to a business rule, and click Next.
6. Select the nodes to target with this job, and click Next.
7. Select an email notification option, and click Next. If you click Email Results, the default email notification and SMTP server settings are populated. These settings can be overridden in each job.
8. Add the script to execute.
   - If you want to load a saved script, click Load Saved Script, and select a script.
   - If you want to load an existing script from disk, click Load Script from File, browse to the file, and click Open.
   - If you want to create a new script, enter your script in the text box. For example:

     ```bash
     ${EnterConfigMode}
     access-list 724 deny 000e.0ca1.a2b4 0000.0000.0000
     exit
     wr mem
     ```

     Where 724 is the name of the ACL, and 000E.0CA1.A2B4 is the MAC address to block. ${EnterConfigMode} is a variable that is equivalent to config terminal on Cisco devices.
9. If you want to save a script, click Save Script, specify a location, and click Save.
10. Select Filter Results that match a pattern if you want to see in the script output only those lines that match a specific regular expression pattern.
11. Select Show Commands in Output to view what NCM sent to the targeted devices.
12. Click Next.
13. Review the settings for the job, and click Finish.
Verify success by comparing configs

1. Click My Dashboards > Network Configuration > Configuration Management.
2. Select one or more nodes, and click Compare Node(s) Configs.
3. Select a config from each list.
4. Click Compare Selected. Changes to are highlighted in red, green, and yellow.

Change an interface description

Updating interface descriptions with Network Configuration Manager saves time because you do not have to remember IP addresses or login credentials for the device you are updating.

Back up the running configuration

1. Click My Dashboards > Network Configuration > Configuration Management.
2. Select one or more nodes.
3. Click Download > Running.

Update the interface descriptions

1. Click My Dashboards > Network Configuration > Jobs.
2. Click Create New Job.
3. Name the job, and select Execute Command Script on Devices from Job Type.
4. Specify when the job runs:
   - To run the job once or on a simple schedule, select Basic. Click the tab that identifies how frequently the job runs, and then specify the start time and (if needed) the day(s).
   - To create a more complex schedule, select Advanced and then use the five fields to create a CRON expression.
5. Add a comment if this job relates to a business rule, and click Next.
6. Select the nodes to target with this job, and click Next.
7. Select an email notification option, and click Next. If you click Email Results, the default email notification and SMTP server settings are populated. These settings can be overridden in each job.
8. Add the script to execute.
   - If you want to load a saved script, click Load Saved Script, and select a script.
   - If you want to load an existing script from disk, click Load Script from File, browse to the file, and click Open.
   - If you want to create a new script, enter your script in the text box. For example:

     ```
     config t
     interface Ethernet0
     no description
     description Link to Upstairs Lab
     exit
     exit
     wr mem
     ```
     
     Where `Link to Upstairs Lab` is the new description.

9. If you want to save a script, click Save Script to File, specify a location, and click Save.

10. Select Filter Results that match a pattern if you want to see in the script output only those lines that match a specific regular expression pattern.

11. Select Show Commands in Output to view what NCM sent to the targeted devices.

12. Click Next.

13. Review the settings for the job, and click Finish.

**Verify success by checking the interface description**

1. Click My Dashboards > Network Configuration > Config Summary.

2. Click the updated node in the list.

3. Click the Configs tab.

4. Under Config List, select the running config, and click Edit Config.

5. Locate the interface definitions.

6. Verify that the new description has been applied to the interface you modified.
Block all private addresses with an access control list

Routers connected the Internet are normally configured to discard any traffic using private IP addresses. This isolation gives your private network a basic form of security as it is not usually possible for the outside world to establish a connection directly one of your network devices using these addresses. The following procedure updates the access control list to block all private IP addresses on several devices at the same time.

Back up the running configuration

1. Click My Dashboards > Network Configuration > Configuration Management.
2. Select one or more nodes.
3. Click Download > Running.

Update the access control list for a group of nodes

1. Click My Dashboards > Network Configuration > Jobs.
2. Click Create New Job.
3. Name the job, and select Execute Command Script on Devices from Job Type.
4. Specify when the job runs:
   - To run the job once or on a simple schedule, select Basic. Click the tab that identifies how frequently the job runs, and then specify the start time and (if needed) the day(s).
   - To create a more complex schedule, select Advanced and then use the five fields to create a CRON expression.
5. Add a comment if this job relates to a business rule, and click Next.
6. Select the nodes to target with this job, and click Next.
7. Select an email notification option, and click Next. If you click Email Results, the default email notification and SMTP server settings are populated. These settings can be overridden in each job.
8. Add the script to execute.
   - If you want to load a saved script, click Load Saved Script, and select a script.
   - If you want to load an existing script from disk, click Load Script from File, browse to the file, and click Open.
If you want to create a new script, enter your script in the text box. For example:

```plaintext
${EnterConfigMode}
access-list 102 deny ip 10.0.0.0 0.255.255.255 any log
access-list 102 deny ip 172.16.1.0 0.15.255.255 any log
access-list 102 deny ip 192.168.0.0 0.0.255.255 any log
exit
write memory
```

*Where 102 is the name of the ACL. ${EnterConfigMode} is a variable that is equivalent to `config terminal` on Cisco devices.*

9. If you want to save a script, click Save Script, specify a location, and click Save.

10. Select Filter Results that match a pattern if you want to see in the script output only those lines that match a specific regular expression pattern.

11. Select Show Commands in Output to view what NCM sent to the targeted devices.

12. Click Next.

13. Review the settings for the job, and click Finish.

**Verify success by comparing configs**

1. Click My Dashboards > Network Configuration > Configuration Management.

2. Select one or more nodes, and click Compare Node(s) Configs.

3. Select a config from each list.

4. Click Compare Selected. Changes to are highlighted in red, green, and yellow.

**Customize the login banner of a device**

You can easily change the login banner for a router, switch, or firewall using Network Configuration Manager. This customization can be rolled out to a single or multiple devices.

**Modify the login banner**

1. Click My Dashboards > Network Configuration > Jobs.

2. Click Create New Job.

3. Name the job, and select Execute Command Script on Devices from Job Type.

4. Specify when the job runs:
To run the job once or on a simple schedule, select Basic. Click the tab that identifies how frequently the job runs, and then specify the start time and (if needed) the day(s).

To create a more complex schedule, select Advanced and then use the five fields to create a CRON expression.

5. Add a comment if this job relates to a business rule, and click Next.

6. Select the nodes to target with this job, and click Next.

7. Select an email notification option, and click Next. If you click Email Results, the default email notification and SMTP server settings are populated. These settings can be overridden in each job.

8. Add the script to execute.

   - If you want to load a saved script, click Load Saved Script, and select a script.
   - If you want to load an existing script from disk, click Load Script from File, browse to the file, and click Open.
   - If you want to create a new script, enter your script in the text box. For example:
     ```
     config t
     no banner login
     banner login ^Unauthorized use of these systems is punishable by law^ exit
     wr mem
     ```
     Where Unauthorized use of these systems is punishable by law is the new banner.

9. If you want to save a script, click Save Script, specify a location, and click Save.

10. Select Filter Results that match a pattern if you want to see in the script output only those lines that match a specific regular expression pattern.

11. Select Show Commands in Output to view what NCM sent to the targeted devices.

12. Click Next.

13. Review the settings for the job, and click Finish.

**Verify success by comparing configs**

1. Click My Dashboards > Network Configuration > Configuration Management.

2. Select one or more nodes, and click Compare Node(s) Configs.

3. Select a config from each list.

4. Click Compare Selected. Changes to are highlighted in red, green, and yellow.
NCM macros (variables)

NCM provides the following types of macros (also called variables):

- Global macros
- Node macros
- Transfer macros
- Command macros
- Date and time macros
- Keystroke macros
- Prompt detection and delay macros

All NCM macros are enclosed in `{ }`. For example, the macro for system name is `{SysName}`. Macros are not case-sensitive.

Global macros

The values for global macros are specified by [global device defaults](#). The value for a global macro is the same for all nodes.

ℹ️ You can also create custom global macros.

<table>
<thead>
<tr>
<th>Macro</th>
<th>Inserts the value of this Global Device Defaults field</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>{GlobalUserName}</code></td>
<td>Username</td>
</tr>
<tr>
<td><code>{GlobalPassword}</code></td>
<td>Password</td>
</tr>
<tr>
<td><code>{GlobalEnableLevel}</code></td>
<td>Enable Level</td>
</tr>
<tr>
<td><code>{GlobalEnablePassword}</code></td>
<td>Enable Password</td>
</tr>
<tr>
<td><code>{GlobalExecProtocol}</code></td>
<td>Execute Commands and Scripts</td>
</tr>
<tr>
<td><code>{GlobalConfigRequestProtocol}</code></td>
<td>Config Request</td>
</tr>
<tr>
<td><code>{GlobalConfigTransferProtocol}</code></td>
<td>Transfer Configs</td>
</tr>
<tr>
<td><code>{GlobalTelnetPort}</code></td>
<td>Telnet Port</td>
</tr>
<tr>
<td><code>{GlobalSSHPort}</code></td>
<td>SSH Port</td>
</tr>
</tbody>
</table>
## Node macros

Node macros point to a column in the NCM_Nodes table. The value of a node macro is unique for each node. The following table lists the NCM_Nodes table column that supplies the value for each macro. Table column names are shown in square brackets.

<table>
<thead>
<tr>
<th>Macro</th>
<th>Inserts this value</th>
<th>From this [columnName]</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>${AgentIP}</code></td>
<td>The IP address of the node.</td>
<td>• If [ManagedProtocol] is 0, then [AgentIP]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If [ManagedProtocol] is <strong>not</strong> 0, then [AgentIPv6]</td>
</tr>
<tr>
<td><code>${Caption}</code></td>
<td>The name used to identify the node in the Orion Platform.</td>
<td>[NodeCaption]</td>
</tr>
<tr>
<td></td>
<td>During discovery, this value is populated with the host name (if available) or the DNS domain name. You can change it by editing the node property Name. This value is <strong>not</strong> updated when the node is rediscovered.</td>
<td></td>
</tr>
<tr>
<td><code>${SystemName}</code></td>
<td>The host name of the node, if it can be polled during discovery. This value is updated when the node is rediscovered.</td>
<td>• If [SysName] is empty, same as <code>${AGENTIP}</code></td>
</tr>
<tr>
<td><code>${SysName}</code></td>
<td></td>
<td>• If [SysName] is <strong>not</strong> empty then [SysName]</td>
</tr>
<tr>
<td><code>${SystemLocation}</code></td>
<td>The node location, which is obtained from the device via SNMP polling.</td>
<td>[SysLocation]</td>
</tr>
<tr>
<td><code>${SysLocation}</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>${SystemContact}</code></td>
<td>The node contact, which is obtained from the device via SNMP polling.</td>
<td>[SysContact]</td>
</tr>
<tr>
<td><code>${SysContact}</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>${Description}</code></td>
<td>The node description.</td>
<td>[SysDescr]</td>
</tr>
<tr>
<td><code>${SysDescr}</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>${SystemDescription}</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>${SysDescription}</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>${SCPStorageAddress}</code></td>
<td>The IP address or host name of the NCM SCP server.</td>
<td>• If [Settings.SCPAutoRetrieveIP], then [EngineIP]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If not, then [Settings.SCPServerAddress]</td>
</tr>
</tbody>
</table>
Transfer macros

Transfer macros are used when NCM uploads or downloads configuration files.

<table>
<thead>
<tr>
<th>Macro</th>
<th>Inserts this value</th>
<th>From this [columnName]</th>
</tr>
</thead>
<tbody>
<tr>
<td>${ConfigType}</td>
<td>The type of configuration (for example, running or startup).</td>
<td></td>
</tr>
<tr>
<td>${StorageFileName}</td>
<td>The name generated by NCM for the downloaded configuration file.</td>
<td></td>
</tr>
<tr>
<td>${ConfigText}</td>
<td>In a direct upload operation, the config to be uploaded to the device.</td>
<td></td>
</tr>
</tbody>
</table>

Command macros

Device template commands are used to execute commands on devices. Device templates define the Name and Value for each device command. For example, EnterConfigMode is a command name, and the value is the command required to enter config mode for a specific device type. For a description of all device template commands, see NCM device template commands.

When a command is defined in a device template, you can include the command as a macro within a script. To do so, format the command Name as a macro (for example, ${EnterConfigMode}).
Device command templates for each device type can define different values for commands with the same name. For example, the value of `EnterConfigMode` is `config terminal` for Cisco IOS devices, but it is `configure` for HP Procurve switches. When you use a command macro in a script instead of the actual command, you can use the same script for multiple device types.

## Date and time macros

Date and time macros are replaced with current date or time.

<table>
<thead>
<tr>
<th>Macro</th>
<th>Inserts this value</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>${DateTime}</td>
<td>Local date and time in short date and local time format (the Short Date and Short Time format defined in the Windows Control Panel)</td>
<td>10-29-2019 2-55 PM</td>
</tr>
<tr>
<td>${Date}</td>
<td>Date in short date format</td>
<td>10-29-2019</td>
</tr>
<tr>
<td>${ShortDate}</td>
<td>Date in short date format</td>
<td>29-Oct-19</td>
</tr>
<tr>
<td>${MediumDate}</td>
<td>Date in medium date format</td>
<td>Tuesday, October 29, 2019</td>
</tr>
<tr>
<td>${Time}</td>
<td>Time in short time format</td>
<td>14-55</td>
</tr>
<tr>
<td>${ShortTime}</td>
<td>Time in short time format</td>
<td>02-55 PM</td>
</tr>
<tr>
<td>${LongTime}</td>
<td>Time in long time format</td>
<td>2-55-48 PM</td>
</tr>
<tr>
<td>${DOW}</td>
<td>Day of the week, spelled out</td>
<td>Tuesday</td>
</tr>
<tr>
<td>${AbreviatedDOW}</td>
<td>Abbreviated day of the week</td>
<td>Tue</td>
</tr>
<tr>
<td>${Day}</td>
<td>One- or two-digit day of the month</td>
<td>5 29</td>
</tr>
<tr>
<td>${AbreviatedDay}</td>
<td>Abbreviated day of the week</td>
<td></td>
</tr>
<tr>
<td>Macro</td>
<td>Inserts this value</td>
<td>Examples</td>
</tr>
<tr>
<td>----------</td>
<td>--------------------------------------------------------</td>
<td>----------</td>
</tr>
<tr>
<td>${DD}</td>
<td>Two-digit day of the month, with a leading zero if needed</td>
<td>05 29</td>
</tr>
<tr>
<td>${MMMM}</td>
<td>Name of the month, spelled out</td>
<td>October</td>
</tr>
<tr>
<td>$MonthName$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$Month Name$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>${MMM}</td>
<td>Abbreviated name of the month</td>
<td>Oct</td>
</tr>
<tr>
<td>$AbreviatedMonth$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$Abreviated Month$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>${Month}</td>
<td>One- or two-digit month number</td>
<td>5</td>
</tr>
<tr>
<td>$M$</td>
<td>One- or two-digit month number</td>
<td>12</td>
</tr>
<tr>
<td>${MM}</td>
<td>Two-digit month number, with a leading zero if needed</td>
<td>05</td>
</tr>
<tr>
<td>${DayOfYear}</td>
<td>Number of the day of the year</td>
<td>302</td>
</tr>
<tr>
<td>$DayOfYear$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$Day Of Year$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>${Year2}</td>
<td>Two-digit year</td>
<td>20</td>
</tr>
<tr>
<td>$Year2$</td>
<td>Two-digit year</td>
<td></td>
</tr>
<tr>
<td>$Year$</td>
<td>Two-digit year</td>
<td></td>
</tr>
<tr>
<td>$YY$</td>
<td>Two-digit year</td>
<td></td>
</tr>
<tr>
<td>${Year4}</td>
<td>Four-digit year</td>
<td>2020</td>
</tr>
<tr>
<td>$Year4$</td>
<td>Four-digit year</td>
<td></td>
</tr>
<tr>
<td>$Year$</td>
<td>Four-digit year</td>
<td></td>
</tr>
<tr>
<td>$YYYY$</td>
<td>Four-digit year</td>
<td></td>
</tr>
<tr>
<td>${H}</td>
<td>One- or two-digit hour</td>
<td>02</td>
</tr>
<tr>
<td>$Hour$</td>
<td>One- or two-digit hour</td>
<td>11</td>
</tr>
<tr>
<td>${HH}</td>
<td>Two-digit hour, with a leading zero if needed</td>
<td>02</td>
</tr>
<tr>
<td>$HH$</td>
<td>Two-digit hour, with a leading zero if needed</td>
<td>11</td>
</tr>
<tr>
<td>${NN}</td>
<td>Two-digit minute, with a leading zero if needed</td>
<td>05</td>
</tr>
<tr>
<td>$Minute$</td>
<td></td>
<td>55</td>
</tr>
<tr>
<td>${N}</td>
<td>One- or two-digit minute</td>
<td>5</td>
</tr>
<tr>
<td>$Minute$</td>
<td></td>
<td>55</td>
</tr>
<tr>
<td>${SS}</td>
<td>Two-digit second, with a leading zero if needed</td>
<td>05</td>
</tr>
<tr>
<td>$Second$</td>
<td></td>
<td>30</td>
</tr>
<tr>
<td>${S}</td>
<td>One- or two-digit second</td>
<td>5</td>
</tr>
<tr>
<td>$Second$</td>
<td></td>
<td>30</td>
</tr>
</tbody>
</table>
Macro | Inserts this value | Examples
--- | --- | ---
${AMPM}$ | AM or PM | PM
${AM/PM}$ |  |  

**Keystroke macros**

Keystroke macros send keystrokes to menu-based or CLI devices. For example, `${DownArrow}` simulates sending a Down Arrow keystroke to the device.

<table>
<thead>
<tr>
<th>Macro</th>
<th>Sends these characters</th>
</tr>
</thead>
<tbody>
<tr>
<td>${CTRL+@}</td>
<td>CTRL+@</td>
</tr>
<tr>
<td>${CTRL+A}</td>
<td>CTRL + A</td>
</tr>
<tr>
<td>${CTRL+B}</td>
<td>CTRL+B</td>
</tr>
<tr>
<td>${CTRL+C}</td>
<td>CTRL+C</td>
</tr>
<tr>
<td>${CTRL+D}</td>
<td>CTRL+D</td>
</tr>
<tr>
<td>${CTRL+E}</td>
<td>CTRL+E</td>
</tr>
<tr>
<td>${CTRL+F}</td>
<td>CTRL+F</td>
</tr>
<tr>
<td>${CTRL+G}</td>
<td>CTRL+G</td>
</tr>
<tr>
<td>${CTRL+H}</td>
<td>CTRL+H</td>
</tr>
<tr>
<td>${CTRL+I}</td>
<td>CTRL+I</td>
</tr>
<tr>
<td>${CTRL+J}</td>
<td>CTRL+J</td>
</tr>
<tr>
<td>${CTRL+K}</td>
<td>CTRL+K</td>
</tr>
<tr>
<td>${CTRL+L}</td>
<td>CTRL+L</td>
</tr>
<tr>
<td>${CTRL+M}</td>
<td>CTRL+M</td>
</tr>
<tr>
<td>${CTRL+N}</td>
<td>CTRL+N</td>
</tr>
<tr>
<td>${CTRL+O}</td>
<td>CTRL+O</td>
</tr>
<tr>
<td>${CTRL+P}</td>
<td>CTRL+P</td>
</tr>
<tr>
<td>${CTRL+Q}</td>
<td>CTRL+Q</td>
</tr>
<tr>
<td>${CTRL+R}</td>
<td>CTRL+R</td>
</tr>
<tr>
<td>Macro</td>
<td>Sends these characters</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>${CTRL+S}</td>
<td>CTRL+S</td>
</tr>
<tr>
<td>${CTRL+T}</td>
<td>CTRL+T</td>
</tr>
<tr>
<td>${CTRL+U}</td>
<td>CTRL+U</td>
</tr>
<tr>
<td>${CTRL+V}</td>
<td>CTRL+V</td>
</tr>
<tr>
<td>${CTRL+W}</td>
<td>CTRL+W</td>
</tr>
<tr>
<td>${CTRL+X}</td>
<td>CTRL+X</td>
</tr>
<tr>
<td>${CTRL+Y}</td>
<td>CTRL+Y</td>
</tr>
<tr>
<td>${CTRL+Z}</td>
<td>CTRL+Z</td>
</tr>
<tr>
<td>${CTRL+[]}</td>
<td>CTRL+{} (Equivalent to pressing the Esc key)</td>
</tr>
<tr>
<td>${CTRL+}</td>
<td>CTRL+\</td>
</tr>
<tr>
<td>${CTRL+]}</td>
<td>CTRL+]}</td>
</tr>
<tr>
<td>${CTRL+CTRL}</td>
<td>CTRL+CTRL</td>
</tr>
<tr>
<td>${CTRL+} }</td>
<td>CTRL+</td>
</tr>
<tr>
<td>${UpArrow}</td>
<td>Up Arrow</td>
</tr>
<tr>
<td>${DownArrow}</td>
<td>Down Arrow</td>
</tr>
<tr>
<td>${RightArrow}</td>
<td>Right Arrow</td>
</tr>
<tr>
<td>${LeftArrow}</td>
<td>Left Arrow</td>
</tr>
<tr>
<td>${CR}</td>
<td>Carriage return</td>
</tr>
<tr>
<td>${CRLF}</td>
<td>Carriage return and line feed combination: \n</td>
</tr>
</tbody>
</table>

**Prompt detection and delay macros**

The following macros can be included in scripts.
<table>
<thead>
<tr>
<th>Macro</th>
<th>Description</th>
</tr>
</thead>
</table>
| `${DisablePromptDetection}` | By default, NCM waits for the device to send a command mode prompt before it sends the next command. If a device does not send a command mode prompt immediately after each command, you can include the following macros in a script to turn off prompt detection for a block of commands in order to improve the performance of the script.  

`${DisablePromptDetection}`  
Command1  
Command2  
...  
`${EnablePromptDetection}`  

The `${DisablePromptDetection}` macro temporarily disables prompt detection. NCM stops waiting for the command mode prompt before it sends the next command. The `${EnablePromptDetection}` macro turns prompt detection back on.  

⚠️ Use these macros to surround a specific group of commands. Always use `${DisablePromptDetection}` to turn prompt detection back on at the end of the group of commands. |
| `${Delay:nn}`            | In a command script, use the Delay macro to specify the number of seconds that NCM should wait before it sends the next command. Use this macro when the first command will take some time to complete. The following example waits 20 seconds before it sends Command2:  

Command1  
`${Delay:20}`  
Command2  

| `${NewPromptRegEx:regEx}` | If a device’s command mode prompt changes frequently and the UseVirtualPromptForCommands in the device template is causing false prompt detection, you can use the NewPromptRegEx macro to identify the new prompt with the executed command. For example, after the sys command is executed, a device’s command prompt changes from `<Prompt>` to `[Prompt]`. The following line tells NCM to execute the sys command and then identify any line that contains `[]` as the command mode prompt:  

sys `${NewPromptRegEx:[]}` |
Where can macros be used?

You can use macros in the following operations.

<table>
<thead>
<tr>
<th>NCM functionality</th>
<th>Macros you can use</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Connection profiles</strong> that enable NCM to connect to a node</td>
<td>• Global</td>
</tr>
<tr>
<td><strong>Command scripts</strong></td>
<td>• Global</td>
</tr>
<tr>
<td></td>
<td>• Node</td>
</tr>
<tr>
<td></td>
<td>• Transfer</td>
</tr>
<tr>
<td></td>
<td>• Command</td>
</tr>
<tr>
<td></td>
<td>• Date and time</td>
</tr>
<tr>
<td></td>
<td>• Prompt detection and delay macros</td>
</tr>
<tr>
<td><strong>NCM jobs</strong></td>
<td>• Global</td>
</tr>
<tr>
<td></td>
<td>• Node</td>
</tr>
<tr>
<td></td>
<td>• Date and time</td>
</tr>
<tr>
<td><strong>Template for naming archived config files</strong></td>
<td>• Global</td>
</tr>
<tr>
<td></td>
<td>• Node</td>
</tr>
<tr>
<td></td>
<td>• Date and time</td>
</tr>
<tr>
<td></td>
<td>• ${ConfigType}</td>
</tr>
<tr>
<td><strong>Device templates</strong></td>
<td>• Node</td>
</tr>
<tr>
<td></td>
<td>• Transfer</td>
</tr>
<tr>
<td></td>
<td>• Date and time</td>
</tr>
<tr>
<td></td>
<td>• Keystroke</td>
</tr>
<tr>
<td><strong>Real-time change detection</strong> email notification options</td>
<td>• Global</td>
</tr>
<tr>
<td></td>
<td>• Node</td>
</tr>
<tr>
<td></td>
<td>• Date and time</td>
</tr>
<tr>
<td></td>
<td>• Keystroke</td>
</tr>
<tr>
<td><strong>Firmware upgrade templates</strong></td>
<td>• Macros documented in Manage NCM firmware upgrade templates</td>
</tr>
<tr>
<td></td>
<td>• Command macros</td>
</tr>
<tr>
<td><strong>Alerts</strong></td>
<td>• Orion Platform variables</td>
</tr>
<tr>
<td><strong>The SWIS verb</strong> NodeInvoker.ParseMacros</td>
<td>• Global</td>
</tr>
<tr>
<td></td>
<td>• Node</td>
</tr>
<tr>
<td></td>
<td>• Date and time</td>
</tr>
</tbody>
</table>
In policy rules, NCM automatically adds certain policy rule macros, such as ${ConfigBlockStartLine}. Policy rule macros are for internal use and should **not** be manually added to policy rules.

### Custom macros (variables) in NCM

A custom macro is a variable with a global value. You can use custom macros in scripts or jobs, and they can also be used for a property that applies across all managed nodes.

#### Access custom macros

1. Click Settings > All Settings.
2. Under Product Specific Settings, click NCM Settings.
3. Under Advanced, click Manage Macros.

#### Create a custom macro

1. Click Add New.
2. Enter a name and value for the new macro.
   
   Example: You want to define a macro to track devices that were provisioned in a certain location on a certain date. Enter a name to identify the macro, and then enter a value to identify the location and date:
   
   - Macro Name: ProvisionedAustin1
   - Macro Value: Austin 6/10/2019

3. Click Submit.

#### Edit a custom macro

You can change the value of the macro, but not the name.

1. Select the macro.
2. Click Edit.
3. Modify the value of the macro.
4. Click Submit.
Delete a custom macro

1. Select the macro.
2. Click Delete.