Cisco IOS XR Fundamentals

Mobeen Tahir • Mark Ghattas
Dawit Birhanu • Syed Natif Nawaz

ciscopress.com

Cisco IOS XR Fundamentals

Mobeen Tahir, CCIE No. 12643
Mark Ghattas, CCIE No. 19706
Dawit Birhanu, CCIE No. 5602
Syed Natif Nawaz, CCIE No. 8825

Cisco Press

800 East 96th Street

Indianapolis, IN 46240

user-specified path instead of the router's latest running configuration. Because the previous router running configuration is overridden by the user option, the entire existing router running configuration is removed. This includes all the checkpointed configuration files prior to restoration of configuration during reload.

While restoring the router configuration from the running configuration, if the running configuration doesn't exist or is corrupted, it will attempt to restore the router configuration from a saved ASCII copy of the running configuration that is maintained by the system and all previous commit points will be removed. In such a case, the previous configuration will still be applied by the user; however, the user will not be able to roll back the configuration to any previous points. If the saved ASCII running configuration file doesn't exist, the default configuration is applied to the router.

Configuration Rollback

In IOS XR the target configuration built by the user is committed to form the new running config. For every successful commit operation, a unique ID or label is generated. This ID or commit point can be used as a rollback point as seen earlier in Figure 4-3.

Rollback is simply undoing some of the configuration changes that were done by the user. Configuration rollback is an atomic operation that rolls the active configuration back to a previous known state. IOS XR has the capacity to roll back up to 100 commits. If an error is encountered in the rollback operation, active configuration is not changed. The **show rollback points** command can be used to list all the rollbacks, as shown in Example 4-27. Each rollback commit is logged with the user information that made the commit along with a timestamp. You can make the rollback point more user friendly by providing a comment using CLI. Because these are different commit files or refpoints, you can view the configuration changes that went in for every commit using **show configuration rollback changes <***commitid***>**. The command **show configuration commit list** can be used to list all the commit IDs that can be used for rollback with the user and timestamp information of every commit as shown in Example 4-27.

Example 4-27 *Displays the List of All Rollback Points*

```
RP/0/RP0/CPU0:R1#show configuration commit list
SNo. Label/ID    User       Line         Client       Time Stamp
~~~~ ~~~~~~~~    ~~~~       ~~~~         ~~~~~~       ~~~~~~~~~~
1    1000000080  cisco      vty0         Rollback     Mon May  5 03:35:14 2008
2    1000000079  cisco      vty0         CLI          Mon May  5 03:34:44 2008
3    1000000078  cisco      vty0         CLI          Mon May  5 02:48:57 2008
4    1000000077  cisco      vty1         CLI          Thu May  1 23:32:35 2008
5    1000000076  cisco      vty1         CLI          Thu May  1 23:31:01 2008
6    1000000075  cisco      vty1         CLI          Thu May  1 19:22:49 2008
7    1000000074  cisco      vty1         CLI          Thu May  1 19:21:29 2008
8    1000000073  cisco      vty1         CLI          Thu May  1 19:19:56 2008
9    1000000072  cisco      vty1         CLI          Thu May  1 18:46:56 2008
10   1000000071  cisco      vty1         CLI          Thu May  1 17:50:15 2008
11   1000000070  cisco      vty1         CLI          Thu May  1 17:47:42 2008
```

```
12   1000000069  cisco      con0_RP0_C  CLI          Thu May  1 14:22:38 2008
```

show configuration commit list <*number*> **detail** provides the last few commit details, as shown in Example 4-28. It displays the list of the last two commits whose corresponding changes can be rolled back to, along with details of the user and the timestamps. If the commit is through a rollback action, the client of the commit operation will reflect the appropriate agent as seen in Example 4-28.

Example 4-28 *Configuration Commit Details*

```
RP/0/RP0/CPU0:CRS1-3#show configuration commit list 2 detail


   1) CommitId: 1000000083                  Label: NONE
      UserId:   cisco                       Line:  vty0
      Client:   Rollback                    Time:  Tue May  6 05:18:37 2008
      Comment:  NONE

   2) CommitId: 1000000146                  Label: hostname
      UserId:   cisco                       Line:  vty0
      Client:   CLI                         Time:  Tue May 6 05:18:27 2008
      Comment:   change of hostname
RP/0/RP0/CPU0:CRS1-3#
```

show configuration rollback changes last <*number*> can also provide the configuration changes of the last few commits as shown in Example 4-29.

Example 4-29 *Configuration of the Last Three Commit Operations*

```
RP/0/RP0/CPU0:R1#show configuration rollback changes last 3
Building configuration...
hostname CRS1-3
end

RP/0/RP0/CPU0:R1#
```

You can use **show configuration rollback changes** <*commit number*> to display the changes that could happen if you roll back to a specific commit ID, as shown in Example 4-30.

Example 4-30 *Configuration of a Specific Commit Operation Using the Commit ID*

```
RP/0/RP0/CPU0:R1#show configuration rollback changes 1000000080
Building configuration...
hostname CRS1-3
end

RP/0/RP0/CPU0:R1#
```

A commit ID is generated for every successful rollback action. In Example 4-31, the configuration is rolled back to the last commit configuration to undo the hostname change from R1 to CRS1-3.

Example 4-31 *Rollback to the Last Configuration*

```
RP/0/RP0/CPU0:R1#rollback configuration last 1
Loading Rollback Changes.
Loaded Rollback Changes in 1 sec
Committing.
1 items committed in 1 sec (0)items/sec
Updating..

Updated Commit database in 1 sec
Configuration successfully rolled back 1 commits.
RP/0/RP0/CPU0:CRS1-3#
```

You can use **show configuration history** to list the history of all configuration events as seen in Example 4-32. A maximum of 1500 events can be displayed.

Example 4-32 *Displays Configuration Events History*

```
RP/0/RP0/CPU0:CRS1-3#show configuration history
Sno.  Event       Info                              Time Stamp
~~~~  ~~~~~       ~~~~                              ~~~~~~~~~~
1     alarm       inconsistency alarm raised        Mon Apr 14 13:06:59 2008
2     startup     configuration applied             Mon Apr 14 13:08:12 2008
3     OIR config  restore                           Mon Apr 14 13:08:15 2008
4     OIR config  restore                           Mon Apr 14 13:08:55 2008
5     backup      Periodic ASCII backup             Mon Apr 14 14:00:16 2008
158   commit      id 1000000081                     Tue May  6 05:17:56 2008
159   commit      id 1000000082                     Tue May  6 05:18:26 2008
160   commit      id 1000000083                     Tue May  6 05:18:39 2008
```

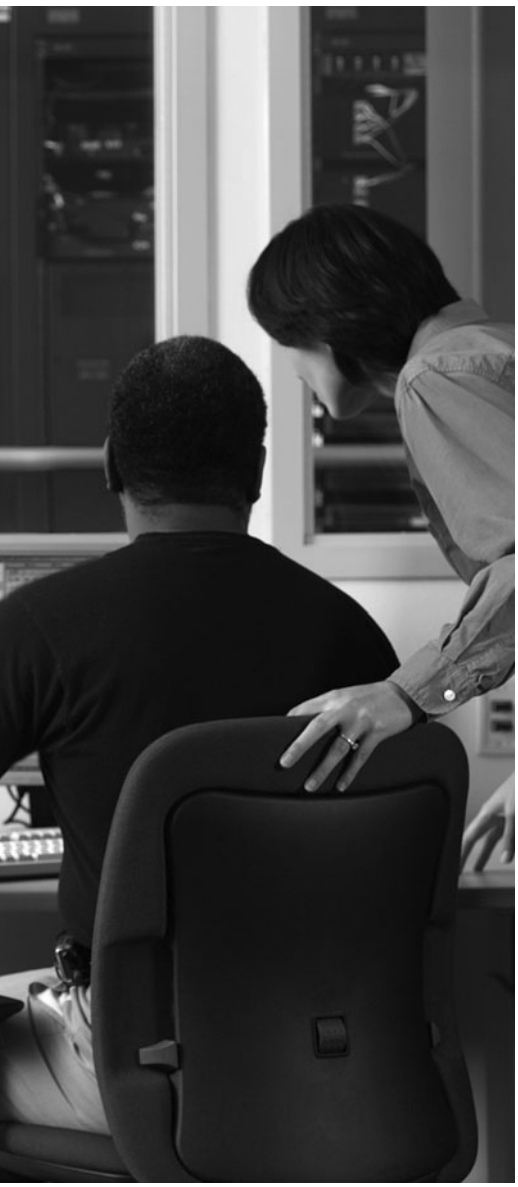
Summary

This chapter provides detailed concepts of Cisco IOS XR configuration management. Also discussed are some of the key components that are required to achieve the restoration of large-scale configurations—namely Configuration Manager, SysDB, CFS, and RDSFS.

Router configuration in IOS XR is based on a two-stage configuration model. The two-stage commit process, which helps to apply the target configuration in bulk, is also discussed in detail.

Restoration of primary persistent configuration, restoration of configuration during boot and node OIR, package activation and deactivation, configuration handling, and session management are implemented using the IOS XR Configuration Management architecture. Some of the salient points of IOS XR configuration, CLIs, and the output of different **show** commands, are explained with examples.

The rollback feature helps the user to undo a previous configuration commit operation. Preconfiguration helps the user program the configuration in advance before installing the corresponding hardware. Last of all, template configuration improves configuration efficiency by creating reusable configurations.



This chapter covers the following topics:

- Using SNMP
- Cisco IOS XR Syslog
- Embedded Event Manager
- Monitoring Processes

Cisco IOS XR finds proven applications in next-generation networks (NGN). Organizations deal with manageability and monitoring aspects as an important part of their day-to-day operations. Monitoring and management functions help reduce the network's operating costs and help improve service response. Therefore, such functions have stringent requirements for accurate measurement of network resources and proactive notification of changes happening on their networks. IOS XR provides features and supports monitoring and troubleshooting commands that integrate well with network management requirements. Monitoring and manageability aspects of the IOS XR operating system have been briefly touched on in the previous chapters. This chapter devotes detailed discussion to these topics.

This chapter outlines the embedded tools and show commands that can be used for monitoring operating system processes and memory. IOS XR also provides real-time monitoring of Layer 2 interfaces and Layer 1 SONET controller counters. This chapter depicts monitoring aspects through examples drawn from a routing platform running IOS XR. Support of syslog and SNMP agent capabilities is an important prerequisite of any OS running on routing and switching devices. IOS XR supports the necessary SNMP management processes that enable it to run as a managed system in compliance with the Internet Network Management Framework. Configuring and monitoring SNMP is also discussed in detail in this chapter.

CHAPTER 5

Cisco IOS XR Monitoring and Operations

This chapter also discusses Embedded Event Management (EEM), where platform or network event stimuli are reported via defined APIs (Application Programming Interfaces). A policy engine then allows a user to trigger responses based on events that might impact a system's health. This chapter devotes a section to demonstrating such policies using a TCL script.

Using SNMP

This section introduces the implementation and configuration of SNMP protocol in IOS XR. As mentioned earlier, SNMP is a protocol used in IP networks for network management. SNMP provides the necessary protocol support to monitor and control network devices, and for network management–related statistics collections. To be managed directly by SNMP, a router must be capable of running an SNMP management process, called an *SNMP agent*. IOS XR has a built-in SNMP agent, whose job is to retrieve and update variables that describe the state of the device or protocols and features enabled in the device. These variables are stored in a data structure called a Management Information Base (MIB).

The Cisco IOS XR software supports the following versions of SNMP:

- Simple Network Management Protocol Version 1 (SNMPv1)
- Simple Network Management Protocol Version 2c (SNMPv2c)
- Simple Network Management Protocol Version 3 (SNMPv3)

There have been several improvements in SNMP as it evolved from v1 to v3; however, the main difference is the improvement in security. The SNMP management stations who have access to the agent MIB are configured by an IP address access control list (ACL) and a community string that serves as a password. In contrast the SNMPv3 ensures security by using Data Encryption Standard (DES) for security. SNMPv3 uses the Hashed Message Authentication Code (HMAC) using the MD5 or SHA-1 algorithm for data integrity. Despite security improvements in SNMP v3, the v2c is still popular and is widespread in SP networks.

Figure 5-1 gives a simplified overview of the exchanges. The figure identifies two devices. One of the devices is a managed device containing the SNMP agent software functionality. The Network Management Station (NMS) contains the applications that helps control or gather the information from the managed device. NMS uses messages called an SNMP