

Key Concepts

Use this information to understand the key terms, concepts, types of releases relevant to setting up and upgrading Cisco IOS XR software on Cisco 8000 series routers.

This section contains the following topics:

- Key Terms and Concepts, on page 1
- Types of Releases, on page 5
- Files in Cisco Software Download Page, on page 5
- Command Modes, on page 6

Key Terms and Concepts

Applicable Variants for Cisco 8000 Series Routers

The Cisco 8000 series routers run on IOS XR software with XR7 architecture. The procedures for setting up and upgrading the software are applicable for these variants of the series:

- Cisco 8201
- Cisco 8202
- Cisco 8808
- Cisco 8812
- Cisco 8818

Setup

When the router with the pre-installed software is powered ON for the first time, the pre-installed version of the IOS XR software starts functioning automatically. You set up and configure the router for network capabilities.

Upgrade

When a new version of the IOS XR software is available, you may choose to upgrade to that version.

Upgrade Methods

After your upgrade, you apply the changes by using one of two methods:

- Process restart – the system is upgraded while it's in service by restarting only those processes that are affected by the upgrade. This applies to Software Maintenance Upgrades (SMUs).

  However, some SMUs can be applied during system reload as well.

- Drain and reboot – traffic is drained before the upgrade and changes are applied by reloading the router with the updated version.

Packages and Red Hat Package Manager (RPM)

Cisco uses the Red Hat Package Manager (RPM) package management system to package required and optional files for installing and upgrading the IOS XR software.

Base Image and Optional Packages

The .iso is the bare minimum software image that is required to run IOS XR on the router. Additional IOS XR packages are optional and are needed depending on the router configuration and required features. If you wish to add features like multicast, manageability, BNG, you must install the appropriate optional package.

Customizable ISO or Golden ISO (GISO)

Golden ISO (GISO) is a customized ISO image that is built to contain preferable packages to suit diverse installation requirements. GISO can be customized to include a standard base image with the basic functional components, additional RPMs, bug fixes, and configuration files based on your requirement.

Active and Committed Packages

An *active package* is the software version on the router after a version upgrade. If the upgraded version is to be retained after reloads, you need to *commit* the changes. After committing the changes, the active packages would then match the list of committed packges retained till the next applicable install upgrade or downgrade operation, or until the install package-based operations are carried out.

Operation

An operation is the sum of all work carried out to fulfill a user’s request provided through CLIs or RPCs.

The internal work performed to complete an operation is regarded to be at different levels of operation: transactions, atomic software changes, and packaging operations.

Each operation is assigned an Operation ID, which is a function of one of these:

Table 1: Operation IDs

Operation ID	Function
1	Transaction ID
1.1	Transaction ID and the atomic change ID if there is one
1.1.1	Transaction ID, the atomic change ID, and packaging operation ID if there is one

For example, these are operations with their Operation IDs carried out through these commands.

- install package add xr-bgp – **1.1.1** - This is starting the **first** transaction, the **first** atomic operation and the **first** packaging operation.
- install package remove xr-bgp – 1.1.**2** - This is starting a **second** packaging operation, within the first atomic operation in the first transaction.

Transactions

Transactions are the highest level of operation. Starting a transaction marks the start of an overall operation. To maintain the software changes carried out during a transaction, you must commit the transaction. If the system reloads during an install transaction, the running software is reverted to its previous state before the transaction was started. Within a transaction, multiple atomic software changes can be performed.

Atomic Software Changes

All atomic changes occur within a transaction. During an atomic software change, any changes to install IOS XR software are not visible to the system. The changes become visible when the atomic change is applied. Within an atomic operation, multiple packaging operations can be performed.

An atomic operation occurs in its entirety, or does not occur it all. During an upgrade, there is a switchover from the old software to the new in a single step.

Packaging Operations

Packaging operations are actions performed to change the packages that are installed on the system. Every packaging operation is contained within an atomic change. Atomic changes may contain multiple packaging operations. Examples of packaging operations are upgrade, downgrade, replace, add, or remove packages.

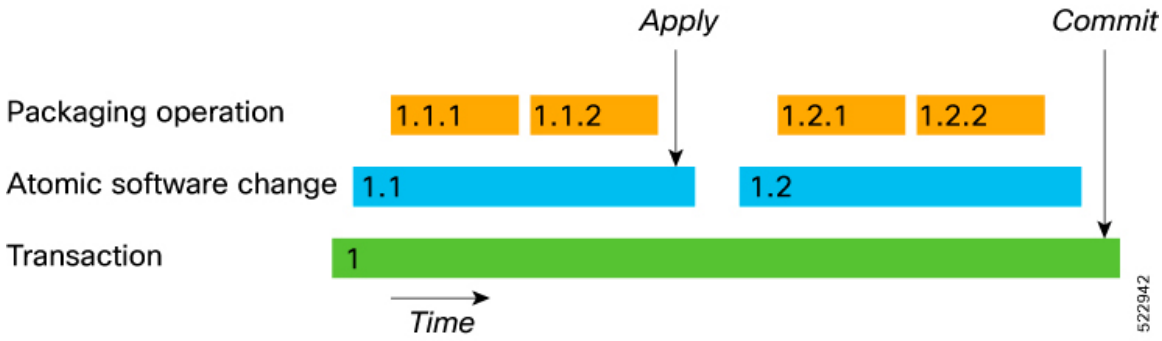
Apply and Commit

One successfully completed operation that modifies the software involves three phases.

- The internal **execution** of the packaging operation that changes the packages that are installed on the system.
- The **Apply** phase that completes an atomic software change and makes the software change visible to the system.
- The **Commit** phase that ends a transaction and ensures that all software changes continue to be present when the router is reloaded.

The following figure shows how the key operations concepts fit together in install and upgrade operations:

Figure 1: Transaction, Atomic Software Change, and Packaging Operation



Example:

- install package add xr-bgp – **1.1.1** This is starting the **first** transaction, the **first** atomic operation and the **first** packaging operation.
- install package remove xr-bgp – 1.1.**2** This is starting a **second** packaging operation, within the first atomic operation in the first transaction.
- install apply - 1.1 This is applying the first atomic operation in the first transaction
- install package add xr-bgp – 1.**2.1**This is starting the **first** packaging operation and the **second** atomic change in the first transaction.
- install package remove xr-bgp – 1.2.**2** This is starting the **second** packaging operation in the second atomic change in the first transaction.
- install apply – 1.2 This is applying the first atomic change in the first transaction.
- install commit - 1This is committing the first transaction
- install package add xr-bgp – **2.1.1** This is starting the **second** transaction, the **first** atomic operation within that transaction and the **first** packaging operation within that atomic operation

Synchronous Action

An asynchronous action allows you to gain access to the prompt and perform another parallel task as the install or upgrade operation continues to its completion.

When installing or upgrading, you can request a synchronous action. Specify the keyword *synchronous* in the install commands, and the prompt is returned only when the request has completed, the Ctrl + C keys are pressed, or a reload occurs.

When the synchronous action is in effect, the user is updated with the status of the request whenever it changes. Pressing Ctrl + C keys during a synchronous action request returns the prompt to the user but does not halt the install or upgrade operation.

Types of Releases

Cisco IOS XR software model has three types of software releases. The software images are available for download at the Cisco Software Download page.

Feature Release

A Feature Release (FR) contains new features and support for new hardware. Feature releases have the X.X.1 (dot one) designation. For example, releases 7.1.1 and 7.5.1 are feature releases. The list of features added to a feature release is provided in the Release notes along with the installation instructions and dependencies.

Maintenance Release

A maintenance release is the primary mechanism to deliver groups of critical bug fixes to the software feature releases.

SMUs Release

A Software Maintenance Unit (SMU) is a fix that is provided until the End of Maintenance (EoM) of the release. The fix is also committed into the next shipping release. SMUs are posted under `Cisco IOS XR Software Maintenance Upgrade` on the Cisco Software Download page. Each SMU is customized for a specific software release.

For more information on release numbering, types of releases, and their timelines, see Software Lifecycle Support Statement - IOS XR.

Files in Cisco Software Download Page

The following table describes the files available for download from the Cisco Software Download page for each variant of the Cisco 8000 series routers:

Table 2: IOS XR Software Installation Files in Cisco Software Download Page

Package File	Example	Description
8000-x64-<*rel. no.*>.iso	8000-x64-7.9.1.iso	Bootable ISO Image of the Operating System required to run IOS XR on a device for basic operations
8000-usb_boot-<*rel. no.*>.zip	8000-x64-usb-7.9.1.zip	USB Boot image of the Operating System
8000-optional-rpms. <*rel. no.*>.tar	8000-optional-rpms.7.9.1.tar	Optional RPMs that provide additional functionality
8000-k9sec-rpms. <*rel. no.*>.tar	8000-k9sec-rpms.7.9.1.tar	Security package that includes software that uses encryption (e.g. SSH) and has export controls for downloads of this software

Package File	Example	Description
8000-<*rel. no.*><*bug-ID*>.tar)	8000-7.9.1.CSCvy99756.tar	Optional or recommended SMUs

Command Modes

The router runs on virtualized Cisco IOS XR software. Therefore, the CLI commands must be executed on virtual machines, namely the XR LXC and the System Admin LXC.

The command modes are applicable to the Cisco Series Routers. This table lists the command modes for the LXCs.

Command Mode	Description
XR EXEC mode (XR LXC execution mode)	Run commands on the XR LXC to display the operational state of the router. Example: `RP/0/RP0/CPU0:router#`
XR Config mode (XR LXC configuration mode)	Perform security, routing, and other XR feature configurations on the XR LXC. Example: `RP/0/RP0/CPU0:router#`**`configure`** `RP/0/RP0/CPU0:router(config)#`