Optionality in NX-OS Software

Optionality in NX-OS software

Optionality is a feature in NX-OS software that

- uses modular packages for selective feature upgrades
- supports both base and full modes, and
- enables independent upgrade or removal of optional RPMs without service disruption.

NX-OS software image supports modular package management. NX-OS software now provides flexibility to add, remove, and upgrade the features selectively without changing the base NX-OS software.

Using modular NX-OS software provides several advantages:

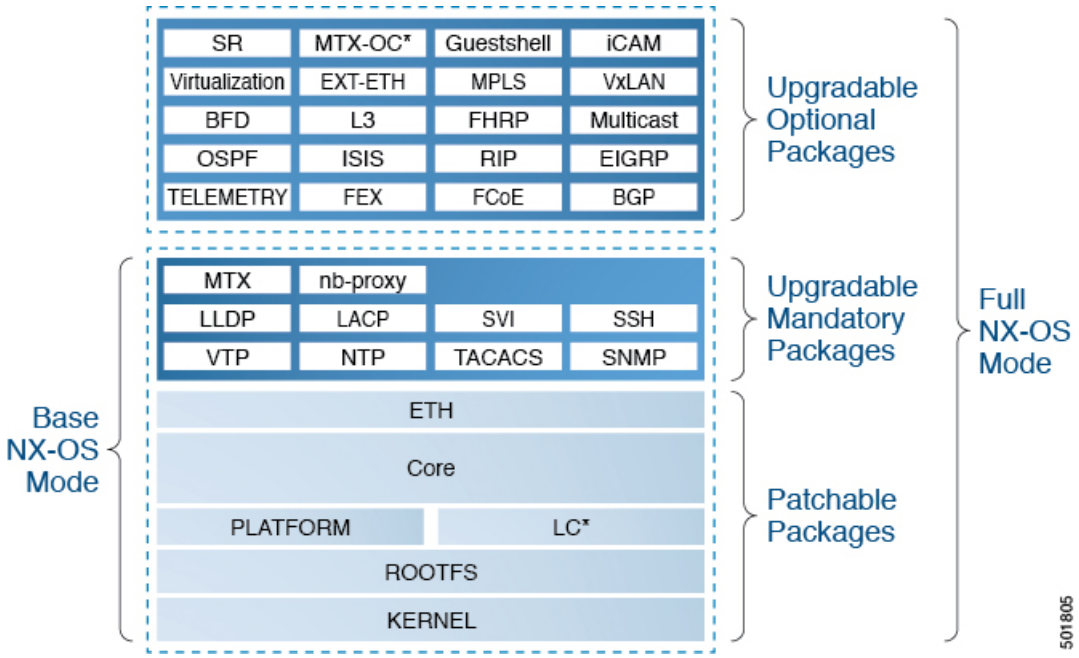
- Leaner NX-OS software
- Asynchronous delivery of the features and the fixes—provide quick fixes that are independent of the releases, including new features
- Reduced footprint of binaries and libraries at run time

Modes

NX-OS software is provisioned to boot the NX-OS software in two modes as described in the illustration:

- Base NX-OS mode
- Full NX-OS mode

Figure 1: Optionality in NX-OS software



- Base NX-OS mode contains:
  - Upgradable mandatory packages
  - Patchable packages
- Full NX-OS mode contains:
  - Upgradable optional packages
  - Upgradable mandatory packages
  - Patchable packages

Note The default mode is full NX-OS mode.

In base NX-OS mode, basic Layer 2 and Layer 3 features are available. All dynamic routing features (for example, BGP, OSPF, EIGRP, RIP, and ISIS) and other optional feature RPMs are not available by default. You have to install the optional feature RPMs on top of the base image.

In full NX-OS mode, all feature RPMs are installed during boot time when Ethernet plugin is activated by the plug-in manager. There is no change in the user behavior as compared to the previous releases.

Modular packages

A modular package is a software package that

- enables independent upgrades within the same release
- allows runtime removal without impacting system startup or other functions, and
- requires feature APIs to be used only after package installation.

The NX-OS software image is traditionally constructed with the packaging that forms a Linux distribution. It makes upgrading certain packages difficult as each package is large in size. This section describes a new package management for the NX-OS software image. Beginning with NX-OS Release 9.2(1), some NXOS features are considered as optional, for example, BGP, OSPF, VXLAN, MPLS, Segment Routing.

Each modular package has the following important characteristics:

- Upgrade functionality—The modular packages can be independently upgraded. The modular packages should be used from the same release as performing upgrades on these packages across multiple releases is not supported.
- Optionality—The modular packages are optional, for example, these packages can be removed or uninstalled at run time. The removal of the modular packages does not affect bringing-up the system and it does not affect any other functionality of the switches.

Note All APIs exported by the modular package should be used only after the installation of the feature.

RPM and DNF

RPM (Red Hat Package Manager) is the package management system used for packaging in the Linux Standard Base (LSB). The RPM command options are grouped into three subgroups for:

- Querying and verifying packages
- Installing, upgrading, and removing packages
- Performing miscellaneous functions

rpm is the command name for the main command that is used with RPM, whereas .rpm is the extension that is used for the RPM files.

Dandified YUM (Yellowdog Updater, Modified) or DNF is an open source command-line tool for RPM based Linux systems. It allows users and system administrators to easily install, update, remove, or search software packages on the systems. DNF adds the automatic updates and the package management, including dependency management, to the RPM systems. In addition to understanding the installed packages on a system, DNF works with the repositories that are collections of the packages and they are typically accessible over a network connection.

NX-OS image boot modes

You can boot the NX-OS image in base or full mode. The full boot mode installs the complete NX-OS software which is similar to the software of the previous releases. This is the default boot mode. The base boot mode has no optional RPMs installed.

To use the command line option, perform one of these steps depending on the mode.

- Use the **install reset nxos base** option to install the NX-OS image in the base boot mode using the VSH prompt. After reload, the switch is in the base mode with no optional packages installed.
- Use the **install reset nxos full** option to install the NX-OS image in the full boot mode using the VSH prompt. After reload, the switch is in the full mode with the optional packages automatically installed.

For more information, see *Using install commands for feature RPM operation* section.

Red Hat Package Managers

You can upgrade or downgrade Red Hat Package Manager (RPM) to a new software version using NX-OS install commands or DNF commands. An upgradable RPM can be optional or mandatory.

See the following sections for more information about optional and mandatory RPMs.

Format of the RPM

This section describes the general format and naming convention of RPM files for NX-OS features.

The general format of a RPM is <name>-<version>-<release>.<arch>.rpm. The same format is followed for NX-OS feature RPMs.

- Name: package name, for example, BGP
- Version in <x.y.x.b> format: <major.minor.patch.build_number>, for example, 2.0.1.0
- Release: The branch from which the RPM is created, for example, 9.2.1
- Arch: The architecture type of the RPM, for example, lib32_n9000

This table provides more information on the naming convention, for example, fex-2.0.0.0-9.2.1.lib32_n9000.rpm.

Table 1: RPM naming convention

RPM Naming Convention **Example: fex-2.0.0.0-9.2.1.lib32_n9000.rpm**	**Description**
fex	Indicates the name of the component.
2	Indicates that the RPM is not backward compatible. Configuration loss takes place during an upgrade.
0	Indicates the incremental API changes/command changes/Schema changes with backward compatibility. It is applicable to the new features on top of the existing capabilities. No configuration is lost during an upgrade.
0	Indicates a bug fix without any functionality change. No configuration is lost during an upgrade.

RPM Naming Convention Example: fex-2.0.0.0-9.2.1.lib32_n9000.rpm	Description
0	This number tracks how many times the component has changed during the development cycle of a release. This value will be 0 for all the release images.
9.2.1	Indicates the release number or the distribution version for the RPM. It aligns to the NVR format. Since the feature RPM is only applicable to a NXOS release, this field has NXOS release version number present.
lib32_n9000	Indicates the architecture type of the RPM.

Optional RPMs and their associated features

The optional RPMs are the RPMs that can be installed to enable the features without affecting the native NX-OS behavior or they can be removed using the **install deactivate** command from the switch.

Optional RPMs, for example, EIGRP are not a part of the base software. They can be added, upgraded, and removed as required using either **dnf** or **install** commands from the switch.

This table contains the list of the optional RPMs and their associated features.

Table 2: List of optional RPMs and their associated features

Package Name	Associated Features
BGP	feature bgp
BFD	feature bfd
Container-tracker	feature container-tracker
EIGRP	feature eigrp
Ext-Eth	• feature openflow • feature evb • feature imp • feature netflow • feature sla_sender • feature sla_responder • feature sla twamp-server • feature sflow
EXT_ETH_LOWMEM	• feature evb • feature netflow

Package Name	Associated Features
FCoE	• feature-set fcoe • feature-set fcoe-npv
FEX	feature-set fex
FHRP	• feature hsrp • feature vrrpv3
HW TELEMETRY	feature hw telemetry
iCAM	feature icam
ISIS	feature isis
MPLS	• feature mpls segment-routing • feature mpls evpn
Multicast	• feature pim • feature pim6 • feature msdp • feature ngmvpn
NIA	NA
NXSDK	NA
OSPF	• feature ospf • feature ospfv3
RIP	feature rip
SDAA	NA
Services	feature catena
SR	feature mpls segment-routing traffic-engineering
TELEMETRY	feature telemetry
Virtualization	NA
VM Tracker	feature vmtracker
VXLAN	• feature nv overlay • feature fabric forwarding

Guidelines for NX-OS feature RPM installation

The NX-OS system RPM repositories are present in NX-OS Series switches for RPM management.

Note Avoid manually copying the RPMs to system repositories. Instead use the install or DNF commands.

Table 3: RPM repositories that are present in the switches

Repository Name	Repository Path	Description
groups-repo	/rpms	Part of the bundled NX-OS image. It is used to keep all the RPMs that are bundled as part of the NX-OS image. All RPMs based in this repository are known as base RPMs.
localdb	/bootflash/.rpmstore/patching/localrepo	Used for RPM persistency. When a user adds a NX-OS feature RPM as part of **install add** command, the RPM is copied to this location and it is persisted during the reloads. User has the responsibility to clean the repository. To add a RPM to this repository, use **install add** command. To remove a RPM from this repository, use **install remove** command. DNF commands can be used to populate the repository too. The maximum space for the repository is 200Mb along with the patching repository for Nexus 9000 Series switches except Nexus 3000 Series switches. For Nexus 3000 Series switches, the maximum space for the repository is 20 Mb only.
patching	/bootflash/.rpmstore/patching/patchrepo	Used for RPM persistency. When a user adds a NX-OS patch RPM to the switch, the patch RPM is copied to this repository.
thirdparty	/bootflash/.rpmstore/thirdparty	Used for RPM persistency when a user adds a third party RPM.

The `groups-repo` and `localdb` repositories hold the NX-OS feature RPMs that should be installed during the system boot or during activation. DNF commands or **install** command can be used for the installation or the removal of these RPMs.

The listed rules are applied to the feature RPM installation procedure during boot or install time:

- Only RPMs with the same NX-OS release number should be selected for the installation.
- Base RPMs cannot be added to the `localdb` repository.

Guidelines for third-party RPM installation

In releases prior to 10.1(x), you can install any third-party package on the device, even if it is not provided or signed by Cisco.

Starting with release 10.1(x) any third-party package that is not signed by Cisco is not allowed to be installed on the device. However, if you wish to bypass this and install the software, you can configure the device to enable the third-party software installation. The configuration persists as a normal configuration and can be verified by using the **running-config** command. Following this configuration, you can install any third-party software with the known risks.

Install command options for feature and third-party RPMs

Use the reference table for using install commands for the feature RPM operations.

Table 4: Reference for install commands for the feature RPM operations

Command	Description

install reset	This operation removes all the patches, persisted configurations, upgraded packages, third-party installed packages, unsaved configurations, and reloads the switch's previous mode (Full/Base) with the default packages.
	The **install reset** command also performs write erase operation. The following message is displayed at the prompt:
	`switch(config)# install reset`
	`WARNING!!This operation will remove all pactches, upgraded packages, persisted etc configs, third party packages installed, startup configuration(write erase) and reload the switch with default packages.`
	`Do you want to proceed with reset operation? (y/n)? [n]`

Command	Description
install reset nxos base	This operation installs NX-OS in base mode by removing all patches, upgraded packages, persisted etc configurations, third-party packages installed, startup configuration (write erase), and reloads the switch with the default packages.
install reset nxos full	This operation installs NX-OS with full mode by removing all patches, upgraded packages, persisted etc configs, third-party packages installed, startup configuration (write erase), and reloads the switch with the default packages (with mandatory and optional RPMs).
install add <>	Adds an RPM file to the respective repository and updates the repository (**patch/feature/third-party**).
install activate <*rpm name*>	Installs an RPM that is present in the repository.
install commit <*rpm name*>	Used for the patch RPMs. Makes the patch persist during the reload.
install deactivate <*rpm name*>	Uninstalls an RPM. Beginning with NX-OS Release 10.1(1), when you use this command to deactivate RPMs, the options to either downgrade to the base version of RPM or to uninstall RPM appear. You can select the option that you desire and the operation will proceed.
install remove <*rpm name*>	Removes an RPM file from the repository and updates the repository.
sh install active	Displays the list of the installed RPMs in the system apart from base rootfs RPMs. (features/patch/third-party).
sh install inactive	Displays the list of the RPMs that are present in the repository but they are not installed.
sh install packages	Lists all the RPMs that are installed including rootfs RPMs.

Command	Description

[no] system software allow third-party	Beginning with NX-OS Release 10.1(1) the third-party RPM installations are not allowed to be installed on the device by default. This command bypasses this restriction and configures the device to enable the third-party software installation.
	The following command shows the error message when you activate third-party RPM without applying the third-party configuration:
	`switch(config)# install activate pbwMonitor-1.0-1.5.0.x86_64.rpm`
	`Install operation 193 failed because package is not signed by Cisco.Enable TPS installation using 'system software allow third-party' CLI at Tue Nov 17 04:23:10 2020`
	The following command shows activating third-party RPM installations after applying the configuration:
	`switch(config)# system software allow third-party`
	`switch(config)# 2020 Nov 17 04:25:41 switch %$ VDC-1 %$ %USER-2-SYSTEM_MSG: <<%PATCH-INSTALLER-2-TPS_FEATURE_ENABLED>> User has enabled TPS installation - patch_installer`
	`switch(config)# install activate pbwMonitor-1.0-1.5.0.x86_64.rpm`
	`[####################] 100%`
	`Install operation 194 completed successfully at Tue Nov 17 04:25:58 2020`
	The following command shows disabling the third-party configuration:
	`switch(config)# no system software allow third-party`
	`switch(config)# 2020 Nov 17 04:27:17 switch %$ VDC-1 %$ %USER-2-SYSTEM_MSG: <<%PATCH-INSTALLER-2-TPS_FEATURE_DISABLED>> User has disabled TPS installation - patch_installer`

Note If you are using ISSU or upgrading to NX-OS Release 10.1.1 release from an earlier version, you must manually apply the third-party configuration within the first 30 minutes after the upgrade to ensure the third-party RPMs get installed.

Use install commands for digital signature support

Use the install commands for digital signature support.

Procedure

Step 1 Import and add a GPG (GNU Privacy Guard) key from a file located in the bootflash using the **install add bootflash**:<keyfile> **gpg-key** command.

Example:

```
switch# install add bootflash:RPM-GPG-KEY-puppetlabs gpg-key
[####################] 100%
Install operation 304 completed successfully at Thu Jun 19 16:40:28 2018
```

Release RPMs are signed with GPG (GNU Privacy Guard) key. The public GPG key is present at **/etc/pki/rpm-gpg/arm-Nexus9k-rel.gpg**. To add other public keys from different sources, use the steps in this section.

Step 2 Use one of the two steps to verify whether the RPM file is a signed or non-signed file.

a) Verify that the package is a signed file using the **install verify package** <package-name> command.

b) Verify that the RPM file is a signed file using the **install verify bootflash**:<RPM file> command.

Example:

```
switch# install verify bootflash:vxlan-2.0.0.0-9.2.1.lib32_n9000.rpm

RSA signed
switch#
```

Query all installed RPMs

Perform this step to query all the installed RPMs

Procedure

Query all the installed RPMs using the **show install packages** command.

Example:

```
switch# show install packages

Boot Image:
NXOS Image: bootflash:/nxos.9.2.1.bin

--------------------------------------------------
Installed Packages
attr.x86_64 2.4.47-r0.0 installed Unsigned
aufs-util.x86_64 3.14+git0+b59a2167a1-r0.0 installed Unsigned
base-files.n9000 3.0.14-r89.0 installed Unsigned
base-passwd.lib32_x86 3.5.29-r0.1.0 installed Unsigned
bash.lib32_x86 4.3.30-r0.0 installed Unsigned
bfd.lib32_n9000 2.0.0.0-9.2.1 installed Signed
bgp.lib32_n9000 2.0.0.0-9.2.1 installed Signed
binutils.x86_64 2.25.1-r0.0 installed Unsigned
bridge-utils.x86_64 1.5-r0.0 installed Unsigned
busybox.x86_64 1.23.2-r0.0 installed Unsigned
busybox-udhcpc.x86_64 1.23.2-r0.0 installed Unsigned
```

```
bzip2.x86_64 1.0.6-r5.0 installed Unsigned
ca-certificates.all 20150426-r0.0 installed Unsigned
cgroup-lite.x86_64 1.1-r0.0 installed Unsigned
chkconfig.x86_64 1.3.58-r7.0 installed Unsigned
container-tracker.lib32_n9000 2.0.0.0-9.2.1 installed Signed
containerd-docker.x86_64 0.2.3+gitaa8187dbd3b7ad67d8e5e3a15115d3eef43a7ed1-r0.0
installed Unsigned
core.lib32_n9000 2.0.0.0-9.2.1 installed Signed
coreutils.lib32_x86 8.24-r0.0 installed Unsigned
cpio.x86_64 2.12-r0.0 installed Unsigned
cracklib.lib32_x86 2.9.5-r0.0 installed Unsigned
cracklib.x86_64 2.9.5-r0.0 installed Unsigned
createrepo.x86_64 0.4.11-r9.0 installed Unsigned
cronie.x86_64 1.5.0-r0.0 installed Unsigned
curl.lib32_x86 7.60.0-r0.0 installed Unsigned
db.x86_64 6.0.30-r0.0 installed Unsigned
dbus-1.lib32_x86 1.8.20-r0.0 installed Unsigned
dhcp-client.x86_64 4.3.2-r0.0 installed Unsigned
dhcp-server.x86_64 4.3.2-r0.0 installed Unsigned
switch#
```

Install RPMs using the one-step procedure

The commands for both install and upgrade RPMs are the same. Use this one-step procedure to install the RPMs.

Procedure

Step 1 Install and activate the RPM using the **install add** <*rpm*> **activate** command.

Example:

```
switch# install add bootflash:chef.rpm activate
Adding the patch (/chef.rpm)
[####################] 100%
Install operation 868 completed successfully at Tue May  8 11:20:10 2018

Activating the patch (/chef.rpm)
[####################] 100%
Install operation 869 completed successfully at Tue May  8 11:20:20 2018
```

Step 2 Verify the output of the **show install active** command.

Example:

```
switch# show install active
Boot Image:
        NXOS Image: bootflash:/nxos.9.2.1.bin

Active Packages:
bgp-2.0.1.0-9.2.1.lib32_n9000
chef-12.0.0alpha.2+20150319234423.git.1608.b6eb10f-1.el5.x86_64

Active Base Packages:
        lacp-2.0.0.0-9.2.1.lib32_n9000
        lldp-2.0.0.0-9.2.1.lib32_n9000
        mtx-device-2.0.0.0-9.2.1.lib32_n9000
        mtx-grpc-agent-2.0.0.0-9.2.1.lib32_n9000
```

```
mtx-infra-2.0.0.0-9.2.1.lib32_n9000
mtx-netconf-agent-2.0.0.0-9.2.1.lib32_n9000
mtx-restconf-agent-2.0.0.0-9.2.1.lib32_n9000
mtx-telemetry-2.0.0.0-9.2.1.lib32_n9000
ntp-2.0.0.0-9.2.1.lib32_n9000
nxos-ssh-2.0.0.0-9.2.1.lib32_n9000
snmp-2.0.0.0-9.2.1.lib32_n9000
svi-2.0.0.0-9.2.1.lib32_n9000
tacacs-2.0.0.0-9.2.1.lib32_n9000
vtp-2.0.0.0-9.2.1.lib32_n9000
```

Install RPMs using the two-step procedure

The commands for both install and upgrade RPMs are the same. Use this two-step procedure to install the RPMs.

Procedure

Step 1 Install the RPM using the **install add** <*rpm*> command.

Example:

```
switch# install add bootflash:vxlan-2.0.1.0-9.2.1.lib32_n9000.rpm

[####################] 100%
Install operation 892 completed successfully at Thu Jun  7 13:56:38 2018
```

Step 2 Verify using the **show install inactive** command.

Example:

```
switch(config)# show install inactive | grep vxlan
vxlan-2.0.1.0-9.2.1.lib32_n9000
```

Step 3 Activate the RPM using the **install activate** <*rpm*> command.

Example:

```
switch# install activate vxlan

[####################] 100%
Install operation 891 completed successfully at Thu Jun  7 13:53:07 2018
```

Step 4 Verify using the **show install active** command.

Example:

```
switch# show install active | grep vxlan
vxlan-2.0.0.0-9.2.1.lib32_n9000
switch# show install inactive | grep vxlan
switch#
```

Upgrade the RPMs

The commands for both install and upgrade RPMs are the same. Perform this procedure to upgrade the RPMs:

Procedure

Step 1 Install the RPM using the **install add** <*rpm*>**activate upgrade** command.

Example:

```
switch(config)# install add bootflash:bgp-2.0.2.0-9.2.1.lib32_n9000.rpm activate upgrade

Adding the patch (/bgp-2.0.2.0-9.2.1.lib32_n9000.rpm)
[####################] 100%
Install operation 870 completed successfully at Tue May 8 11:22:30 2018

Activating the patch (/bgp-2.0.2.0-9.2.1.lib32_n9000.rpm)
[####################] 100%
Install operation 871 completed successfully at Tue May 8 11:22:40 2018
```

Step 2 Verify the output using the **show install active** command.

Example:

```
switch(config)# show install active

Boot Image:
NXOS Image: bootflash:/nxos.9.2.1.bin

Active Packages:
bgp-2.0.2.0-9.2.1.lib32_n9000
chef-12.0.0alpha.2+20150319234423.git.1608.b6eb10f-1.el5.x86_64

Active Base Packages:
lacp-2.0.0.0-9.2.1.lib32_n9000
lldp-2.0.0.0-9.2.1.lib32_n9000
mtx-device-2.0.0.0-9.2.1.lib32_n9000
mtx-grpc-agent-2.0.0.0-9.2.1.lib32_n9000
mtx-infra-2.0.0.0-9.2.1.lib32_n9000
mtx-netconf-agent-2.0.0.0-9.2.1.lib32_n9000
mtx-restconf-agent-2.0.0.0-9.2.1.lib32_n9000
mtx-telemetry-2.0.0.0-9.2.1.lib32_n9000
ntp-2.0.0.0-9.2.1.lib32_n9000
nxos-ssh-2.0.0.0-9.2.1.lib32_n9000
snmp-2.0.0.0-9.2.1.lib32_n9000
svi-2.0.0.0-9.2.1.lib32_n9000
tacacs-2.0.0.0-9.2.1.lib32_n9000
vtp-2.0.0.0-9.2.1.lib32_n9000
```

Downgrade RPMs

The downgrade procedure needs a special command attribute. Downgrade the RPMs using the one-step procedure.

Procedure

Step 1 Downgrade the RPM using the **install add** <*rpm*>**activate downgrade** command.

Example:

```
switch(config)# install add bootflash:bgp-2.0.1.0-9.2.1.lib32_n9000.rpm activate downgrade


Adding the patch (/bgp-2.0.1.0-9.2.1.lib32_n9000.rpm)
[####################] 100%
Install operation 872 completed successfully at Tue May 8 11:24:43 2018

Activating the patch (/bgp-2.0.1.0-9.2.1.lib32_n9000.rpm)
[####################] 100%
Install operation 873 completed successfully at Tue May 8 11:24:52 2018
```

Step 2 Verify the output using the **show install active** command.

Example:

```
switch(config)# show install active
Boot Image:
 NXOS Image: bootflash:/nxos.9.2.1.bin

Active Packages:
 bgp-2.0.1.0-9.2.1.lib32_n9000
 chef-12.0.0alpha.2+20150319234423.git.1608.b6eb10f-1.el5.x86_64

Active Base Packages:
 lacp-2.0.0.0-9.2.1.lib32_n9000
 lldp-2.0.0.0-9.2.1.lib32_n9000
 mtx-device-2.0.0.0-9.2.1.lib32_n9000
 mtx-grpc-agent-2.0.0.0-9.2.1.lib32_n9000
 mtx-infra-2.0.0.0-9.2.1.lib32_n9000
 mtx-netconf-agent-2.0.0.0-9.2.1.lib32_n9000
 mtx-restconf-agent-2.0.0.0-9.2.1.lib32_n9000
 mtx-telemetry-2.0.0.0-9.2.1.lib32_n9000
 ntp-2.0.0.0-9.2.1.lib32_n9000
 nxos-ssh-2.0.0.0-9.2.1.lib32_n9000
 snmp-2.0.0.0-9.2.1.lib32_n9000
 svi-2.0.0.0-9.2.1.lib32_n9000
 tacacs-2.0.0.0-9.2.1.lib32_n9000
 vtp-2.0.0.0-9.2.1.lib32_n9000
switch(config)#
```

Uninstall the RPMs

Perform this procedure to uninstall the RPMs.

Procedure

Downgrade to the base version of RPM, if one exists in the groups-repo (/rpms), or uninstall the RPM completely from the switch using the **install deactivate** <*rpm*> command.

- To downgrade to the base version, enter **y**.
- To completely uninstall the RPM, enter **n** in the command prompt.

Example:

```
switch(config)# install deactivate bgp
Base RPM found. Do you want to downgrade to base version(y/n) [n] y
Downgrading to the base version
[####################] 100%
Install operation 190 completed successfully at Tue Nov 17 04:10:40 2020
```

Example:

```
switch(config)# install deactivate bgp
Base RPM found. Do you want to downgrade to base version(y/n) [n] n

================================================================================
 WARNING!!
 This operation will remove 'bgp-3.0.0.0-9.4.1.lib32_n9000' related configuration from
running-configuration
 on successful completion. Update startup-configuration accordingly.
================================================================================
[####################] 100%
Install operation 9 completed successfully at Tue Nov 17 05:05:59 2020
```

Remove the RPMs

Perform this procedure to remove the RPMs.

Procedure

Remove the RPM from the repository using the **install remove** <*rpm*> command.

Example:

```
switch(config)# show install inactive | grep vxlan

vxlan-2.0.0.0-9.2.1.lib32_n9000
switch(config)# install remove vxlan

Proceed with removing vxlan? (y/n)?  [n] y
[####################] 100%
Install operation 890 Removal of base rpm package is not permitted  at Thu Jun  7 13:52:15 2018
```

Dandified YUM commands

In Nexus 9000 switches, DNF (Dandified YUM) is the package manager used to manage modular software components (RPMs) within the NX-OS environment. Beginning with NX-OS Release 10.1(x), DNF replaced YUM as the primary tool for the Optionality feature, allowing you to install, upgrade, or remove specific

features (such as Layer 3, BGP, or OSPF) without performing a full system binary upgrade or reloading the switch.

Note DNF commands do not support CTRL+C. Install commands do support CTRL+C. If DNF commands are aborted using CTRL+C, manual cleanup must be performed using **/isan/bin/patching_utils.py --unlock**.

Package operations with DNF commands

This section describes how to perform package operations using the DNF commands.

This section describes how to perform package operations using the DNF commands:

Note

- DNF commands are accessed only from the BASH shell on the box and they are not allowed from the NX-OS VSH terminal.
- Make sure that as a sudo user, you have access to the super user privileges.

Find the base version RPM of the image

The base RPM version is the pre-installed RPM that is archived in the system image.

Use the **ls /rpms** command to find the base version RPM of the image.

```
switch# ls /rpms

bfd-2.0.0.0-9.2.1.lib32_n9000.rpm
ins_tor_sdk_t2-1.0.0.0-9.2.0.77.lib32_n9000.rpm
mtx-netconf-agent-2.0.0.0-9.2.1.lib32_n9000.rpm   snmp-2.0.0.0-9.2.1.lib32_n9000.rpm
bgp-2.0.0.0-9.2.1.lib32_n9000.rpm
ins_tor_sdk_t3-1.0.0.0-9.2.0.77.lib32_n9000.rpm
mtx-restconf-agent-2.0.0.0-9.2.1.lib32_n9000.rpm  sr-2.0.0.0-9.2.1.lib32_n9000.rpm
container-tracker-2.0.0.0-9.2.1.lib32_n9000.rpm  isis-2.0.0.0-9.2.1.lib32_n9000.rpm
         mtx-telemetry-2.0.0.0-9.2.1.lib32_n9000.rpm      svi-2.0.0.0-9.2.1.lib32_n9000.rpm
eigrp-2.0.0.0-9.2.1.lib32_n9000.rpm              lacp-2.0.0.0-9.2.1.lib32_n9000.rpm
          nbproxy-2.0.0.0-9.2.1.lib32_n9000.rpm
tacacs-2.0.0.0-9.2.1.lib32_n9000.rpm
ext-eth-2.0.0.0-9.2.1.lib32_n9000.rpm            lldp-2.0.0.0-9.2.1.lib32_n9000.rpm
          ntp-2.0.0.0-9.2.1.lib32_n9000.rpm
telemetry-2.3.4.0-9.2.1.lib32_n9000.rpm
fcoe-2.0.0.0-9.2.1.lib32_n9000.rpm               mcast-2.0.0.0-9.2.1.lib32_n9000.rpm
          nxos-ssh-2.0.0.0-9.2.1.lib32_n9000.rpm
virtualization-2.0.0.0-9.2.1.lib32_n9000.rpm
fex-2.0.0.0-9.2.1.lib32_n9000.rpm                mpls-2.0.0.0-9.2.1.lib32_n9000.rpm
         ospf-2.0.0.0-9.2.1.lib32_n9000.rpm              vtp-2.0.0.0-9.2.1.lib32_n9000.rpm
fhrp-2.0.0.0-9.2.1.lib32_n9000.rpm               mtx-device-2.0.0.0-9.2.1.lib32_n9000.rpm
          repodata
vxlan-2.0.0.0-9.2.1.lib32_n9000.rpm
guestshell-2.0.0.0-9.2.1.lib32_n9000.rpm        mtx-grpc-agent-2.0.0.0-9.2.1.lib32_n9000.rpm
       rip-2.0.0.0-9.2.1.lib32_n9000.rpm
icam-2.0.0.0-9.2.1.lib32_n9000.rpm               mtx-infra-2.0.0.0-9.2.1.lib32_n9000.rpm
          services-2.0.0.0-9.2.1.lib32_n9000.rpm
```

Check the list of the installed RPMs

Use the **dnf list installed** command to query the feature and third party RPMs and grep a specific RPM.

Here is an example for feature RPMs.

```
bash-4.2# dnf list installed | grep lib32_n9000

bfd.lib32_n9000                          2.0.0.0-9.2.1           @groups-repo
core.lib32_n9000                         2.0.0.0-9.2.1           installed
eth.lib32_n9000                          2.0.0.0-9.2.1           installed
guestshell.lib32_n9000                   2.0.0.0-9.2.1           @groups-repo
lacp.lib32_n9000                         2.0.0.0-9.2.1           installed
linecard2.lib32_n9000                    2.0.0.0-9.2.1           installed
lldp.lib32_n9000                         2.0.0.0-9.2.1           installed
mcast.lib32_n9000                        2.0.0.0-9.2.1           @groups-repo
mtx-device.lib32_n9000                   2.0.0.0-9.2.1           installed
mtx-grpc-agent.lib32_n9000               2.0.0.0-9.2.1           installed
mtx-infra.lib32_n9000                    2.0.0.0-9.2.1           installed
mtx-netconf-agent.lib32_n9000            2.0.0.0-9.2.1           installed
mtx-restconf-agent.lib32_n9000           2.0.0.0-9.2.1           installed
mtx-telemetry.lib32_n9000                2.0.0.0-9.2.1           installed
nbproxy.lib32_n9000                      2.0.0.0-9.2.1           installed
ntp.lib32_n9000                          2.0.0.0-9.2.1           installed
nxos-ssh.lib32_n9000                     2.0.0.0-9.2.1           installed
ospf.lib32_n9000                         2.0.0.0-9.2.1           @groups-repo
platform.lib32_n9000                     2.0.0.0-9.2.1           installed
snmp.lib32_n9000                         2.0.0.0-9.2.1           installed
svi.lib32_n9000                          2.0.0.0-9.2.1           installed
tacacs.lib32_n9000                       2.0.0.0-9.2.1           installed
tor.lib32_n9000                          2.0.0.0-9.2.0.77        installed
virtualization.lib32_n9000               2.0.1.0-9.2.1           @localdb
vtp.lib32_n9000                          2.0.0.0-9.2.1           installed
vxlan.lib32_n9000                        2.0.0.0-9.2.1           @groups-repo
...
```

Get details of the installed RPMs

The **dnf info** <*rpmname*> command lists out the detailed information of the installed RPM.

```
dnf info vxlan

Loaded plugins: downloadonly, importpubkey, localrpmDB, patchaction, patching,
protect-packages
groups-repo

                     | 1.1 kB     00:00 ...
localdb

                     |  951 B     00:00 ...
patching

                     |  951 B     00:00 ...
thirdparty

                     |  951 B     00:00 ...
Installed Packages
Name        : vxlan
Arch        : lib32_n9000
Version     : 2.0.0.0
Release     : 9.2.1
Size        : 6.4 M
Repo        : installed
```

```
From repo   : groups-repo
Summary     : Cisco NXOS VxLAN
URL         : http://cisco.com/
License     : Proprietary
Description : Provides VxLAN support
```

Install RPMs

Installing the RPMs downloads the RPMs and copies the respective program to the switches. This is an example for installing the RPMs from a remote server (that is reachable in the network).

```
bash-4.3# dnf install
http://10.0.0.2/modularity/rpms/vxlan-2.0.1.0-9.2.1.lib32_n9000.rpm

Loaded plugins: downloadonly, importpubkey, localrpmDB, patchaction, patching,
protect-packages
groups-repo
                                                     | 1.1 kB     00:00 ...
localdb
                                                     |  951 B     00:00 ...
localdb/primary
                                                     |  886 B     00:00 ...
localdb
                                                                    1/1
patching
                                                     |  951 B     00:00 ...
thirdparty
                                                     |  951 B     00:00 ...
Setting up Install Process
vxlan-2.0.1.0-9.2.1.lib32_n9000.rpm
                                                 | 1.6 MB     00:00
Examining /var/tmp/yum-root-RaANgb/vxlan-2.0.1.0-9.2.1.lib32_n9000.rpm:
vxlan-2.0.1.0-9.2.1.lib32_n9000
Marking /var/tmp/yum-root-RaANgb/vxlan-2.0.1.0-9.2.1.lib32_n9000.rpm to be installed
Resolving Dependencies
--> Running transaction check
---> Package vxlan.lib32_n9000 0:2.0.1.0-9.2.1 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================

 Package                   Arch                          Version
           Repository                                   Size
================================================================================
Installing:
vxlan                     lib32_n9000                   2.0.1.0-9.2.1
     /vxlan-2.0.1.0-9.2.1.lib32_n9000                   6.4 M
Transaction Summary
================================================================================
Install       1 Package

Total size: 6.4 M
Installed size: 6.4 M
Is this ok [y/N]: y
Downloading Packages:
Running Transaction Check
Running Transaction Test
Transaction Test Succeeded
Running Transaction
  Installing : vxlan-2.0.1.0-9.2.1.lib32_n9000
                                                                    1/1
```

```
starting pre-install package version mgmt for vxlan
pre-install for vxlan complete
starting post-install package version mgmt for vxlan
post-install for vxlan complete

Installed:
  vxlan.lib32_n9000 0:2.0.1.0-9.2.1


Complete!
```

This is an example for installing the RPMs from local bootflash.

sudo dnf install /bootflash/vxlan-2.0.1.0-9.2.1.lib32_n9000.rpm

```
Loaded plugins: downloadonly, importpubkey, localrpmDB, patchaction, patching,
protect-packages
groups-repo

                      | 1.1 kB     00:00 ...
localdb

                      |  951 B     00:00 ...
patching

                      |  951 B     00:00 ...
thirdparty

                      |  951 B     00:00 ...
Setting up Install Process
Examining /bootflash/vxlan-2.0.1.0-9.2.1.lib32_n9000.rpm: vxlan-2.0.1.0-9.2.1.lib32_n9000
Marking /bootflash/vxlan-2.0.1.0-9.2.1.lib32_n9000.rpm as an update to
vxlan-2.0.0.0-9.2.1.lib32_n9000
Resolving Dependencies
--> Running transaction check
---> Package vxlan.lib32_n9000 0:2.0.0.0-9.2.1 will be updated
---> Package vxlan.lib32_n9000 0:2.0.1.0-9.2.1 will be an update
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================

 Package                                  Arch
Version                                                  Repository
                                      Size
================================================================================
Updating:
 vxlan                                    lib32_n9000
2.0.1.0-9.2.1                                    /vxlan-2.0.1.0-9.2.1.lib32_n9000
                            6.4 M

Transaction Summary
================================================================================
Upgrade       1 Package

Total size: 6.4 M
Is this ok [y/N]: y
Downloading Packages:
Running Transaction Check
Running Transaction Test
Transaction Test Succeeded
Running Transaction
  Updating   : vxlan-2.0.1.0-9.2.1.lib32_n9000
```

```
                                    1/2
starting pre-install package version mgmt for vxlan
pre-install for vxlan complete
starting post-install package version mgmt for vxlan
post-install for vxlan complete
  Cleanup    : vxlan-2.0.0.0-9.2.1.lib32_n9000

                                    2/2

Updated:
  vxlan.lib32_n9000 0:2.0.1.0-9.2.1


Complete!
```

This is an example for installing the RPM if it is available in a repository.

dnf install eigrp

Upgrade RPMs

This topic provides examples for upgrading RPMs from a remote server (that is reachable in the network).

```
bash-4.3# dnf upgrade
http://10.0.0.2/modularity/rpms/vxlan-2.0.1.0-9.2.1.lib32_n9000.rpm

Loaded plugins: downloadonly, importpubkey, localrpmDB, patchaction, patching,
protect-packages
groups-repo
                                                      | 1.1 kB     00:00 ...
localdb
                                                      |  951 B     00:00 ...
patching
                                                      |  951 B     00:00 ...
thirdparty
                                                      |  951 B     00:00 ...
Setting up Upgrade Process
vxlan-2.0.1.0-9.2.1.lib32_n9000.rpm
                                                 | 1.6 MB     00:00
Examining /var/tmp/yum-root-RaANgb/vxlan-2.0.1.0-9.2.1.lib32_n9000.rpm:
vxlan-2.0.1.0-9.2.1.lib32_n9000
Marking /var/tmp/yum-root-RaANgb/vxlan-2.0.1.0-9.2.1.lib32_n9000.rpm as an update to
vxlan-2.0.0.0-9.2.1.lib32_n9000
Resolving Dependencies
--> Running transaction check
---> Package vxlan.lib32_n9000 0:2.0.0.0-9.2.1 will be updated
---> Package vxlan.lib32_n9000 0:2.0.1.0-9.2.1 will be an update
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================

 Package                  Arch                          Version
          Repository                                   Size
================================================================================
Updating:
vxlan                    lib32_n9000                   2.0.1.0-9.2.1
     /vxlan-2.0.1.0-9.2.1.lib32_n9000                  6.4 M
Transaction Summary
================================================================================
Upgrade       1 Package
```

```
Total size: 6.4 M
Is this ok [y/N]: y
Downloading Packages:
Running Transaction Check
Running Transaction Test
Transaction Test Succeeded
Running Transaction
** Found 1 pre-existing rpmdb problem(s), 'yum check' output follows:
busybox-1.23.2-r0.0.x86_64 has missing requires of busybox-syslog
  Updating   : vxlan-2.0.1.0-9.2.1.lib32_n9000
                                                                        1/2
starting pre-install package version mgmt for vxlan
pre-install for vxlan complete
starting post-install package version mgmt for vxlan
post-install for vxlan complete
  Cleanup    : vxlan-2.0.0.0-9.2.1.lib32_n9000
                                                                        2/2

Updated:
  vxlan.lib32_n9000 0:2.0.1.0-9.2.1


Complete!
```

This is an example for upgrading the RPMs from local bootflash.

sudo dnf upgrade /bootflash/vxlan-2.0.1.0-9.2.1.lib32_n9000.rpm

```
Loaded plugins: downloadonly, importpubkey, localrpmDB, patchaction, patching,
protect-packages
groups-repo

                     | 1.1 kB     00:00 ...
localdb

                     |  951 B     00:00 ...
patching

                     |  951 B     00:00 ...
thirdparty

                     |  951 B     00:00 ...
Setting up Upgrade Process
Examining /bootflash/vxlan-2.0.1.0-9.2.1.lib32_n9000.rpm: vxlan-2.0.1.0-9.2.1.lib32_n9000
Marking /bootflash/vxlan-2.0.1.0-9.2.1.lib32_n9000.rpm as an update to
vxlan-2.0.0.0-9.2.1.lib32_n9000
Resolving Dependencies
--> Running transaction check
---> Package vxlan.lib32_n9000 0:2.0.0.0-9.2.1 will be updated
---> Package vxlan.lib32_n9000 0:2.0.1.0-9.2.1 will be an update
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================

 Package                                Arch
Version                                         Repository
                                  Size
================================================================================
Updating:
vxlan                                  lib32_n9000
2.0.1.0-9.2.1                                    /vxlan-2.0.1.0-9.2.1.lib32_n9000
                            6.4 M
```

```
Transaction Summary
================================================================================
Upgrade       1 Package

Total size: 6.4 M
Is this ok [y/N]: y
Downloading Packages:
Running Transaction Check
Running Transaction Test
Transaction Test Succeeded
Running Transaction
  Updating   : vxlan-2.0.1.0-9.2.1.lib32_n9000

                                     1/2
starting pre-install package version mgmt for vxlan
pre-install for vxlan complete
starting post-install package version mgmt for vxlan
post-install for vxlan complete
  Cleanup    : vxlan-2.0.0.0-9.2.1.lib32_n9000

                                     2/2

Updated:
  vxlan.lib32_n9000 0:2.0.1.0-9.2.1


Complete!
```

This is an example for upgrading the RPMs if it is available in any repository.

```
dnf upgrade eigrp
```

Downgrade RPMs

This topic provides example for downgrading the RPMs from a remote server (that is reachable in the network).

```
sudo dnf
 downgrade vxlan-2.0.0.0-9.2.1.lib32_n9000

Loaded plugins: downloadonly, importpubkey, localrpmDB, patchaction, patching,
protect-packages
Setting up Downgrade Process
groups-repo

                      | 1.1 kB     00:00 ...
localdb

                      |  951 B     00:00 ...
localdb/primary

                      | 1.3 kB     00:00 ...
localdb

                                        2/2
patching

                      |  951 B     00:00 ...
thirdparty

                      |  951 B     00:00 ...
Resolving Dependencies
--> Running transaction check
```

```
---> Package vxlan.lib32_n9000 0:2.0.0.0-9.2.1 will be a downgrade
---> Package vxlan.lib32_n9000 0:2.0.1.0-9.2.1 will be erased
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package                                    Arch
            Version                                                 Repository
                                  Size
================================================================================
Downgrading:
vxlan                                      lib32_n9000
           2.0.0.0-9.2.1                                            groups-repo
                                  1.6 M
Transaction Summary
================================================================================
Downgrade     1 Package

Total download size: 1.6 M
Is this ok [y/N]: y
Downloading Packages:
Running Transaction Check
Running Transaction Test
Transaction Test Succeeded
Running Transaction
  Installing : vxlan-2.0.0.0-9.2.1.lib32_n9000

                                    1/2
starting pre-install package version mgmt for vxlan
pre-install for vxlan complete
starting post-install package version mgmt for vxlan
post-install for vxlan complete
  Cleanup    : vxlan-2.0.1.0-9.2.1.lib32_n9000

                                    2/2

Removed:
  vxlan.lib32_n9000 0:2.0.1.0-9.2.1



Installed:
  vxlan.lib32_n9000 0:2.0.0.0-9.2.1



Complete!
```

This is an example for downgrading the RPMs from local bootflash.

dnf downgrade /bootflash/eigrp-2.0.0-9.2.1.lib32_n9000.rpm

This is an example for downgrading the RPMs if it is available in any repository.

dnf downgrade eigrp

Delete RPMs

Deleting the RPMs de-installs the RPMs and removes any configuration commands of the feature. Use the **dnf erase** <*rpm*> command to delete the RPMs.

```
bash-4.2# sudo dnf erase vxlan

Loaded plugins: downloadonly, importpubkey, localrpmDB, patchaction, patching,
protect-packages
Setting up Remove Process
Resolving Dependencies
--> Running transaction check
---> Package vxlan.lib32_n9000 0:2.0.1.0-9.2.1 will be erased
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
Package   Arch   Version  RepositorySize
================================================================================
Removing:
vxlan     lib32_n9000    2.0.1.0-9.2.1    @/vxlan-2.0.1.0-9.2.1.lib32_n9000  6.4 M
Transaction Summary
================================================================================
Remove1 Package

Installed size: 6.4 M
Is this ok [y/N]: y
Downloading Packages:
Running Transaction Check
Running Transaction Test
Transaction Test Succeeded
Running Transaction
Erasing    : vxlan-2.0.1.0-9.2.1.lib32_n9000 1/1
starting pre-remove package version mgmt for vxlan
pre-remove for vxlan complete

Removed:
vxlan.lib32_n9000 0:2.0.1.0-9.2.1

Complete!
```

Support for DNF groups

The DNF group is a feature that

- simplifies the management of the packages for the administrators
- provides greater flexibility, and
- allows administrators to manage collections of packages as logical groups.

The support for DNF groups is part of the package management.

The administrators can group a list of packages (RPMs) into a logical group and they can perform various operations. The group commands that DNFsupports include

- **grouplist**
- **groupinfo**
- **groupinstall**
- **groupremove**, and
- **groupupdate**.

The DNF groups can be broadly classified as Layer 2, Layer 3, routing, and management.

Grouplist command for listing available package groups

In Linux, number of packages are bundled to particular group. Instead of installing individual packages with dnf, you can install particular group that will install all the related packages that belongs to the group. For example to list all the available groups, use the **dnf grouplist** command:

List all available package groups in Linux using the **dnfgrouplist** command.

```
bash-4.4# dnf grouplist
Last metadata expiration check: 0:00:00 ago on Fri 08 Mar 2024 12:26:33 PM UTC.
                  ] ---  B/s |   0  B     --:-- ETA
Available Groups:
   management
   routing
   L2
   L3
bash-4.4#
```

Groupmembers command for displaying package group contents

Display the description and the contents of a package group using the **dnf groupinfo** command.

The command lists out the feature members of the group.

```
bash-4.4# dnf groupinfo l2

Last metadata expiration check: 0:00:00 ago on Fri 08 Mar 2024 12:27:44 PM UTC.
                  ] ---  B/s |   0  B     --:-- ETA

Group: L2
 Mandatory Packages:
   lacp
   lldp
   svi
   vtp
bash-4.4#
```

Groupinstall command for install and upgrade of member RPM

This command is for both install and upgrade of the members RPM. If the member is not installed, it installs the highest version available. If the member is already installed and higher RPM is available, it upgrades that member.

```
bash-4.4# dnf groupinstall l3
Last metadata expiration check: 0:00:00 ago on Fri 08 Mar 2024 12:38:05 PM UTC.
                  ] ---  B/s |   0  B     --:-- ETA
Not a redundant system. Nothing todo
Dependencies resolved.
================================================================================

 Group                                          Packages

================================================================================
Marking packages as installed by the group:
 @L3                                            bfd

Is this ok [y/N]: y
Complete!
Install operation 10 completed successfully at Fri Mar  8 12:38:08 2024.

[####################] 100%
```

Groupupdate command for updating package groups

Update any existing installed group packages using the **dnf groupupdate** command.

```
bash-4.4# dnf groupupdate l3

Last metadata expiration check: 0:00:00 ago on Wed 13 Mar 2024 12:30:11 PM UTC.
                                     ] ---  B/s |   0  B     --:-- ETA
Dependencies resolved.

================================================================================$

 Group                                                  Packages
================================================================================$
Marking packages as installed by the group:
 @L3                                                    bfd
================================================================================$

 Package                      Arch                                    Version
                            Repository                               Size
================================================================================$
Installing group packages:
 bfd                          lib32_64_n9000                          2.0.0.0-10.4.3
                            groups-repo                            562 k

Transaction Summary
================================================================================$
Install  1 Package

Total size: 562 k
Installed size: 2.3 M
Is this ok [y/N]: y
Downloading Packages:
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :
                                                                        1/1
  Running scriptlet: bfd-2.0.0.0-10.4.3.lib32_64_n9000
                                                                        1/1
starting pre-install package version mgmt for bfd
pre-install for bfd complete
  Installing       : bfd-2.0.0.0-10.4.3.lib32_64_n9000
                                                                        1/1
  Running scriptlet: bfd-2.0.0.0-10.4.3.lib32_64_n9000
                                                                        1/1
starting post-install package version mgmt for bfd
post-install for bfd complete
  Verifying        : bfd-2.0.0.0-10.4.3.lib32_64_n9000
                                                                        1/1

Installed:
  bfd.lib32_64_n9000 2.0.0.0-10.4.3

Complete!
Install operation 14 completed successfully at Wed Mar 13 12:30:23 2024.

[####################] 100%
bash-4.4#
```

Grouperase command for deleting groups

Delete the groups or all the RPM members of the group using the **dnf grouperase** command.

```
bash-4.4# dnf grouperase l3
Dependencies resolved.
================================================================================

 Group                                              Packages

================================================================================
Marking packages as removed by the group:
 @L3                                                bfd

================================================================================

 Package                    Arch                                Version
               Repository                      Size
================================================================================
Removing:
 bfd                        lib32_64_n9000                      2.0.0.0-10.4.3
               @System                         2.3 M

Transaction Summary
================================================================================
Remove  1 Package

Freed space: 2.3 M
Is this ok [y/N]: y
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :
                                                   1/1
  Running scriptlet: bfd-2.0.0.0-10.4.3.lib32_64_n9000
starting pre-remove package version mgmt for bfd
pre-remove for bfd complete
  Erasing          : bfd-2.0.0.0-10.4.3.lib32_64_n9000
                                                   1/1
  Running scriptlet: bfd-2.0.0.0-10.4.3.lib32_64_n9000
                                                   1/1
starting post-remove package version mgmt for bfd
post-remove for bfd complete

    Verifying        : bfd-2.0.0.0-10.4.3.lib32_64_n9000
                                                     1/1

Removed:
  bfd.lib32_64_n9000 2.0.0.0-10.4.3
                                                          Complete!
Install operation 11 completed successfully at Fri Mar  8 12:38:41 2024.

[####################] 100%
bash-4.4#
```

Find repositories

The **dnf repolist all** command lists the repositories that the switch has along with the number of RPMs it has to those repositories.

```
bash-4.3# dnf repolist all

Loaded plugins: downloadonly, importpubkey, localrpmDB, patchaction, patching,
protect-packages
groups-repo

                     | 1.1 kB     00:00 ...
localdb

                     |  951 B     00:00 ...
patching

                     |  951 B     00:00 ...
thirdparty

                     |  951 B     00:00 ...
repo id
          repo name
                                  status
groups-repo
          Groups-RPM Database
                                  enabled: 37
localdb
          Local RPM Database
                                  enabled:  6
patching
          Patch-RPM Database
                                  enabled:  0
thirdparty
          Thirdparty RPM Database
                                  enabled:  0
open-nxos
          open-nxos
                                  disabled
repolist: 43
```

Installed DNF version

To view the installed version of DNF use the **dnf --version** command.

```
dnf --version

3.4.3
Installed: rpm-5.4.14-r0.0.x86_64 at 2018-06-02 13:04
Built    : Wind River <info@windriver.com> at 2018-04-27 08:36
Committed: Wind River <info@windriver.com> at 2018-04-27

Installed: yum-3.4.3-r9.0.x86_64 at 2018-06-02 13:05
Built    : Wind River <info@windriver.com> at 2018-04-27 08:36
Committed: Wind River <info@windriver.com> at 2018-04-27
```

Mapping of NX-OS commands to the DNF commands

See the table for mapping the NX-OS commands to the DNF commands:

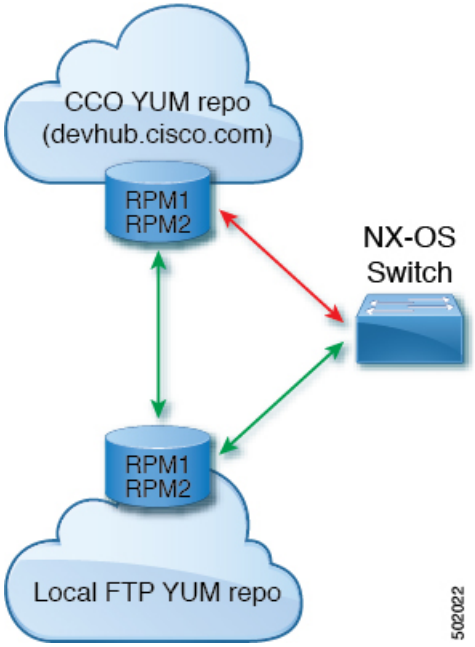
Table 5: Patching command reference

NX-OS Commands	DNF Commands
show install inactive	**dnf list --patch-only available**
show install active	**dnf list --patch-only installed**
show install committed	**dnf list --patch-only committed**
show install packages	**dnf list --patch-only**
show install pkg-info	**dnf info --patch-only**
show install log	**dnf history --show-patch-log** where log_cmd: • opid= - Log that is specific to an operation ID. • last - Shows the latest operation log. • reverse – Shows the log in reverse order. • detail – Show detailed log. • from= - Shows logging from a specific operation ID.
clear install log	**dnf history --clear-patch-log=** where clear_log_cmd: • all - Clears the complete log. • - Clears the logs above this operation ID.
install add	**dnf install --add bootflash:/**
install remove	**dnf install --remove**
install remove inactive	**dnf install --remove all**
install activate	**dnf install --no-persist --nocommit** **Note** By default, all packages are activated and committed.
install deactivate	**dnf erase --nocommit** **Note** By default, all packages are deactivated and committed.
install commit	**dnf install --commit**
Install commit	**dnf install --commit all**

Configure an FTP server and set up a local FTP YUM repository

For setting up a local FTP YUM repository, you have to first create an FTP server, create a local FTP YUM repository, and configure the NX-OS switch to reach the FTP server as outlined in this illustration.

Figure 2: Configure an FTP server and set up a local FTP YUM repository



Note For NX-OS Release 10.1(1), visit https://devhub.cisco.com/artifactory/open-nxos/10.1.1/ for **open-nxos** repository.

Create an FTP server on Red Hat Enterprise Linux 7 (RHEL7) Virtual Machine

Complete the following steps to create an FTP server on Red Hat Enterprise Linux 7 (RHEL7) Virtual Machine (VM):

Procedure

Step 1 Install vsftpd, an FTP server, using the **dnfinstall vsftpd** command.

Step 2 Start the FTP server using the **systemctl start vsftpd** command.

Step 3 Check the status of the FTP server using the **systemctl status vsftpd** command.

Step 4 Provide access to the FTP services from the external systems and open port 21 using the **firewall-cmd --zone=public --permanent --add-port=21/tcp** command.

Step 5 Add the FTP service using the **firewall-cmd --zone=public --permanent --add-service=ftp** command.

Step 6 Reload the server using the **firewall-cmd --reload** command.

Step 7 Host a file in the FTP server (for example, test.txt) and attempt Wget of that file using the**wget ftp://** < *ip of FTP server* > / *test.txt* command.

Note
The **/var/ftp/** directory is the default home directory of the FTP server.

Create a local FTP YUM repository

Complete the steps provided in this procedure to synchronize the external repository RPMs to the FTP server and create a local FTP YUM repository.

Procedure

Step 1 Create a repository file using the **touch/etc/yum.repos.d/local.repo** command.

Create a repository file under **/etc/yum.repos.d/** , for example, creates **local.repo** repository and adds the base URL.

Example:

```
bash-4.3# touch /etc/yum.repos.d/local.repo
```

Step 2 Edit the repository file and copy the localrepo details.using the **vim /etc/yum.repos.d/local.repo** command.

Note
Modify the base URL to the required repository URL.

Example:

```
bash-4.3# vim /etc/yum.repos.d/local.repo

[localrepo]
name=localrepo
baseurl=
https://devhub.cisco.com/artifactory/open-nxos/7.0-3-I2-1/x86_64/
enabled=1
gpgcheck=0
sslverify=0
```

Step 3 Verify the local repository data to proceed further using the **cat /etc/yum.repos.d/local.repo** command.

Example:

```
bash-4.3# cat /etc/yum.repos.d/local.repo

[localrepo]
name=localrepo
baseurl=
https://devhub.cisco.com/artifactory/open-nxos/7.0-3-I2-1/x86_64/
enabled=1
gpgcheck=0
sslverify=0
```

Step 4 Check the reachability of the repository using the **dnf repolist** command.

Example:

```
bash-4.3# dnf repolist
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
* base: mirror.dhakacom.com
* extras: mirror.dhakacom.com
* updates: mirror.dhakacom.com
repo id repo name status
base/7/x86_64 CentOS-7 - Base 9,911
extras/7/x86_64 CentOS-7 - Extras 313
localrepo localrepo 687
updates/7/x86_64 CentOS-7 - Updates 711
repolist: 11,622
```

Step 5 Synchronize all the packages from the external repository to the FTP server home directory using the **nohup reposync -r** < *repo-name mentioned in the local.repo* > **-p** < *directory path to sync* > **&** command.

Example:

```
nohup reposync -r localrepo -p /var/ftp/ &
```

This command creates a directory with the name **local.repo** inside **/var/ftp/** and downloads all the packages from **devhub.cisco.com** to the directory.

Step 6 Check the status of the synchronization using the **tail -f nouhup.out** command.

a) **ls /var/ftp/localrepo**
b) **cd /var/ftp/localrepo/ && createrepo .**

Configure a switch to reach an FTP server

Complete the following steps to configure a switch to reach an FTP server:

Procedure

Step 1 Log in as a sudo user using the **run bash sudo su** command.

Step 2 Verify that the FTP server can be reached using the **ip netns exec management ping** < *ip_address* > command.

This command checks the reachability of the FTP server address from the switch using the **ping** command.

Step 3 Create a repository file using the **touch/etc/yum/repos.d/ftp.repo** command.

This command creates a repository file under **/etc/yum/repos.d/**, for example, it creates the **ftp.repo** repository.

Example:

```
bash-4.3# touch /etc/yum/repos.d/ftp.repo
```

Step 4 Edit the repository file and copy the ftp repo details using the **vim /etc/yum/repos.d/ftp.repo** command.

Note
Modify the base URL to the required ftp server IP.

Example:

```
bash-4.3# vim /etc/yum/repos.d/ftp.repo

[ftp]
name=ftp
baseurl=
ftp://198.51.100.1/localrepo/
enabled=1
gpgcheck=0
sslverify=0
```

Step 5 Create a repository file on the switch with the FTP server address as the URL using the **cat /etc/yum/repos.d/ftp.repo** command.

Example:

```
bash-4.3# cat /etc/yum/repos.d/ftp.repo
[ftp]
name=ftp
baseurl=ftp://198.51.100.1/localrepo/
enabled=1
gpgcheck=0
sslverify=0
```

Step 6 To use the Bash shell prompt, run the **ip netns exec management bash** command.

Step 7 Check the reachability of the newly created repository using the **dnfrepolist** command.

Example:

```
bash-4.3# dnf repolist
Loaded plugins: downloadonly, importpubkey, localrpmDB, patchaction, patching,
: protect-packages
groups-repo | 1.1 kB 00:00 ...
localdb | 951 B 00:00 ...
patching | 951 B 00:00 ...
thirdparty | 951 B 00:00 ...
thirdparty/primary | 758 B 00:00 ...
thirdparty 1/1
repo id repo name status
groups-repo Groups-RPM Database 37
localdb Local RPM Database 0
patching Patch-RPM Database 0
thirdparty Thirdparty RPM Database 1
ftp ftp 686
repolist: 724
```

Step 8 List the available packages in the new repository using the **dnflist available** command.

Create user roles for install operation

The **install** command is only available to the users of admin role. The **install** command can be available to a user by RBAC. For more information about RBAC configuration guidelines, see Guidelines and Limitations for User Accounts and RBAC.

Compacting Cisco NX-OS Software Images

Note This feature is deprecated from Cisco NX-OS Release

Cisco NX-OS software image compaction reduces the size of the image file before completing a copy request. Use SCP, HTTP, or HTTPS as the source and bootflash or USB as the destination. The following example uses SCP and bootflash:

```
switch# copy scp://user@scpserver.cisco.com//download/nxos64.10.1.1.bin
bootflash:nxos64.10.1.1.bin compact vrf management use-kstack

user1@10.65.42.196's password:
nxos64.10.1.1.bin 100% 1501MB 8.4MB/s 02:58
Copy complete, now saving to disk (please wait)...
Copy complete.
```

The **compact** keyword compacts the NX-OS image before copying the file to the supervisor module.

Note Software image compaction is only supported on SCP, HTTP, or HTTPS. If you attempt compaction with any other protocol, the system returns the following error:

```
Compact option is allowed only with source as scp/http/https and destination
as bootflash or usb
```

Note Compacted images are not supported with LXC boot mode.

Note Software image compaction is only supported on Cisco Nexus 9300-series platform switches.