



Cisco NX-OS ソフトウェアのオプション

この章では、Cisco NX-OS ソフトウェアのオプション性について説明します。

- [Cisco NX-OS ソフトウェアのオプション \(1 ページ\)](#)
- [モジュラ パッケージの使用 \(3 ページ\)](#)
- [ベース モードまたはフル モードでの NX-OS イメージの起動 \(4 ページ\)](#)
- [RPM に関する情報 \(4 ページ\)](#)
- [DNF コマンドに関する情報 \(20 ページ\)](#)
- [FTP サーバの設定とローカル FTP YUM リポジトリのセットアップ \(35 ページ\)](#)
- [インストール操作ユーザ ロールの作成 \(40 ページ\)](#)
- [Compacting Cisco NX-OS ソフトウェア イメージの圧縮 \(40 ページ\)](#)

Cisco NX-OS ソフトウェアのオプション

Cisco NX-OS リリース 9.2 (1) 以降では、Cisco NX-OS ソフトウェア イメージでモジュラ パッケージの管理をサポートします。これにより、Cisco NX-OS ソフトウェアは、基礎となる NX-OS ソフトウェアを変更することなく、機能を選択的に追加、削除、およびアップグレードする柔軟性を提供します。

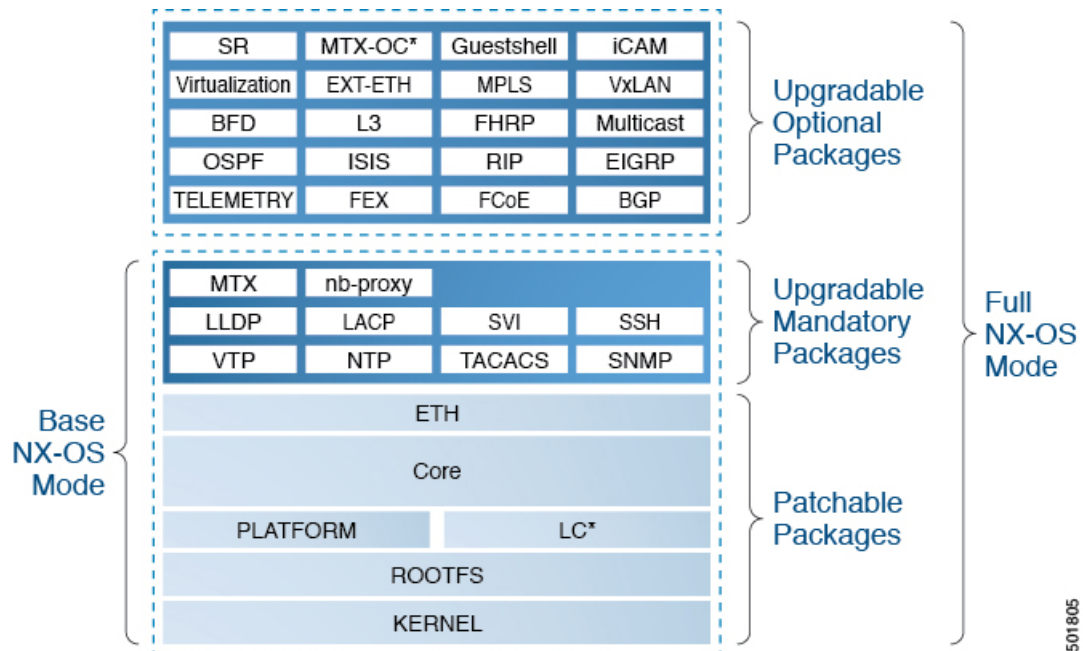
モジュール式の Cisco NX-OS ソフトウェアを使用する利点は次のとおりです。

- 無駄のない NX-OS ソフトウェア
- 機能と修正の非同期的な提供: クイック フィックス (新機能を含む) はリリースとは独立して提供されます。
- 実行時のバイナリとライブラリのフットプリントの削減

Cisco NX-OS ソフトウェアは、次の図に示すように 2 つのモードで NX-OS ソフトウェアをブートするようにプロビジョニングされています。

- ベース NX-OS モード
- フル NX-OS モード

図 1: Cisco NX-OS ソフトウェアのオプション



- ベース NX-OS モードには次が含まれます。
 - アップグレード可能な必須パッケージ
 - パッチ適用可能パッケージ
- フル NX-OS モードには次が含まれます。
 - アップグレード可能なオプション パッケージ
 - アップグレード可能な必須パッケージ
 - パッチ適用可能パッケージ



(注) デフォルトのモードは、フル NX-OS モードです。

ベース NX-OS モードでは、レイヤ 2 およびレイヤ 3 の基本的な機能が提供されます。すべてのダイナミック ルーティング機能 (BGP、OSPF、EIGRP、RIP、ISIS など) やその他のオプション機能 RPM はデフォルトでは使用できません。オプションの機能 RPM は、ベース イメージの上にインストールする必要があります。

フル NX-OS モードでは、ブート時にイーサネット プラグインがプラグイン マネージャによりアクティブ化されるときにすべての機能 RPM がインストールされます。以前のリリースと比較して、ユーザの動作に変更はありません。

モジュラ パッケージの使用

Cisco NX-OS ソフトウェア イメージは、以前から Cisco Linux ディストリビューションを形成するパッケージングで構成されています。各パッケージのサイズが大きいため、特定のパッケージのアップグレードが困難になっています。

このセクションでは、Cisco NX-OS ソフトウェア イメージの新しいパッケージの管理について説明します。Cisco NX-OS リリース 9.2(1) 以降では、BGP、OSPF、VXLAN、MPLS、セグメントルーティングなどの一部の NX-OS 機能はオプションと見なされます。

各モジュラ パッケージには、次の重要な特徴があります。

- **アップグレード機能:** モジュラ パッケージは個別にアップグレード可能です。モジュラ パッケージは、同じリリースのものを使用する必要があります。複数のリリースにまたがるパッケージでのアップグレードの実行はサポートされていません。
- **オプション性:** モジュラ パッケージはオプションです。たとえば、これらのパッケージは実行時に削除またはアンインストールが可能です。モジュラ パッケージの削除はシステムの稼働に影響を与えず、スイッチのその他の機能にも影響を与えません。



(注) モジュラ パッケージでエクスポートされたすべての API は、機能のインストール後にのみ使用する必要があります。

RPM と DNF

RPM (Red Hat Package Manager) は、Linux Standard Base (LSB) 内のパッケージングに使用されるパッケージ管理システムです。RPM コマンド オプションは、次の 3 つのサブグループにまとめられます。

- パッケージのクエリと確認
- パッケージのインストール、アップグレードおよび削除
- その他の機能の実行

rpm は RPM で使用されるメイン コマンドのコマンド名です。一方、**.rpm** は RPM ファイルに使用される拡張子です。

洗練された YUM (Yellowdog Updater, Modified) または DNF は、RPM ベース Linux システム用のオープン ソース コマンドライン ツールです。これにより、ユーザとシステム管理者はシステム上のソフトウェアパッケージのインストール、アップデート、削除、または検索を簡単に行うことができます。DNF により、自動アップデートとパッケージ管理 (依存関係管理を含む) の機能が RPM システムに追加されます。DNF は、システムにインストールされたパッケージを把握するだけでなく、パッケージのコレクションであるリポジトリと連携します。通常、リポジトリにはネットワーク接続を介してアクセスできます。

ベース モードまたはフル モードでの NX-OS イメージの起動

NX-OS イメージは、ベースモードまたはフルモードでブートできるようになりました。フルブートモードでは、以前のリリースのソフトウェアと同様な完全な NX-OS ソフトウェアがインストールされます。これは、デフォルトのブート モードです。ベース ブート モードでは、オプションの RPM はインストールされません。

コマンドライン オプションを使用するには、次の手順を参照してください。

- VSH プロンプトで **install reset nxos base** オプションを使用して、NX-OS イメージを基本ブートモードでインストールします。リロード後にスイッチはベースモードになり、オプションパッケージはインストールされません。
- VSH プロンプトで **install reset nxos full** オプションを使用して、NX-OS イメージをフルモードでインストールします。リロード後にスイッチはフルモードになり、オプションのパッケージが自動的にインストールされます。

詳細については、「機能 RPM の操作に install CLI を使用する」セクションを参照してください。

RPM に関する情報

RPM は、NX-OS の **install** コマンドを使用して、または **DNF** コマンドを使用して新しいソフトウェアバージョンにアップグレードまたはダウングレードできます。アップグレード可能な RPM には、オプションと必須があります。

オプションおよび必須の RPM の詳細については、以降のセクションを参照してください。

RPM の形式

RPM の一般的な形式は、<name>-<version>-<release>.<arch>.rpm です。同じ形式が NX-OS 機能 RPM にも適用されます。

- **name:** パッケージ名（例：BGP）
- **version**（<X.y.x.b> 形式）：<major.minor.patch.build_number>（例：2.0.1.0）
- **release:** RPM 作成元のブランチ（例：9.2.1）
- **arch:** RPM のアーキテクチャ タイプ（例：lib32_n9000）

次の表では、fex-2.0.0.0-9.2.1.lib32_n9000.rpm を例として命名規則の詳細を説明しています。

表 1: RPM 命名規則

RPM 命名規則 例: fex-2.0.0.0-9.2.1.lib32_n9000.rpm	説明
fex	コンポーネントの名前を示しています。
2	RPMに後方互換性がないことを示します。アップグレード中に設定の損失が発生します。
0	後方互換性がある増分 API 変更/CLI 変更/スキーマ変更を示します。既存の機能上の新しい機能が該当します。アップグレード中に失われる設定はありません。
0	機能の変更がないバグ修正を示します。アップグレード中に失われる設定はありません。
0	この番号は、リリースの開発サイクルの間にコンポーネントが変更された回数を追跡します。この値はすべてのリリースイメージで0となります。
9.2.1	RPMのリリース番号またはディストリビューションバージョンを示します。NVR形式に沿っています。機能 RPM は NX-OS リリースにのみ適用可能であるため、このフィールドには存在する NX-OS リリース バージョンのみが指定されます。
lib32_n9000	RPM のアーキテクチャ タイプを示します。

オプション RPM とその関連機能

オプション RPM をインストールすると、ネイティブの NX-OS 動作に影響を与えずに機能を有効化できます。また、オプション RPM は、**install deactivate** コマンドを使用してスイッチから削除できます。

EIGRP などのオプション RPM は、基本ソフトウェアの一部ではありません。これらの RPM は、**dnf** または **install CLI** コマンドを使用して、必要に応じてスイッチに対して追加、アップグレード、削除が可能です。

次のリストは、オプション RPM とその関連機能を示しています。

表 2: オプション RPM とその関連機能

パッケージ名	関連機能
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
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

パッケージ名	関連機能
マルチキャスト	• feature pim • feature pim6 • feature msdp • feature ngmvpn
NIA	N/A
NXSDK	N/A
OSPF	• feature ospf • feature ospfv3
RIP	feature rip
SDAA	N/A
サービス	feature catena
SR	feature mpls segment-routing traffic-engineering
TELEMETRY	feature telemetry
仮想化	該当なし
VM トラッカー	機能 vmtracker
VXLAN	• feature nv overlay • feature fabric forwarding

NX-OS 機能 RPM インストールに関するガイドライン

Cisco NX-OS シリーズ スイッチに存在する次の RPM 管理用 NX-OS システム RPM リポジトリを参照してください。



(注) RPM をシステムのリポジトリに手動でコピーをすることは避けてください。代わりに、install または DNF コマンドを使用してください。

表 3: スイッチに存在する RPM リポジトリ

リポジトリ名	リポジトリのパス	説明
groups-repo	/rpms	バンドルされている NX-OS イメージの一部です。NX-OS イメージの一部としてバンドルされているすべての RPM を保持するために使用されます。このリポジトリに格納されているすべての RPM は、基本 RPM と呼ばれます。
localdb	/bootflash/.rpmstore/patching/localrepo	RPM の保持に使用されます。ユーザが NX-OS 機能 RPM を install add コマンドの一部として追加すると、RPM がこの場所にコピーされ、リロード時に常駐します。ユーザは、リポジトリをクリーンアップする責任があります。 このリポジトリに RPM を追加するには、install add コマンドを使用します。 このリポジトリから RPM を削除するには、install remove コマンドを使用します。 DNF コマンドも、リポジトリに追加するために使用できます。 Cisco Nexus 3000 シリーズ スイッチを除き、リポジトリの最大領域は 200 Mb です（Cisco Nexus 9000 シリーズ スイッチのパッチリポジトリを含む）。 Cisco Nexus 3000 シリーズ スイッチでは、リポジトリの最大サイズは 20 Mb です。
patching	/bootflash/.rpmstore/patching/patchrepo	RPM の保持に使用されます。ユーザが NX-OS パッチ RPM をスイッチに追加すると、パッチ RPM がこのリポジトリにコピーされます。

リポジトリ名	リポジトリのパス	説明
Third_Party	/bootflash/.rpmstore/thirdparty	ユーザがサードパーティ RPM を追加したときに、RPM の保持に使用されます。

groups-repo と **localdb** リポジトリには、システム ブート時またはアクティベーション時にインストールする必要がある NX-OS 機能 RPM が保持されます。DNF コマンドまたは **install** コマンドは、これらの RPM のインストールまたは削除に使用できます。

次のルールが、ブートまたはインストール時の機能 RPM のインストール手順に適用されます。

- 同じ NX-OS リリース番号の RPM のみをインストール用に選択する必要があります。
- ベース RPM は、**localdb** リポジトリに追加できません。

サードパーティ製RPMインストールのガイドライン

現在、シスコによって提供されていないサードパーティ製パッケージは、シスコによって署名されていない場合でも、デバイスにインストールできます。

リリース10.1 (x) 以降、シスコによって署名されていないサードパーティパッケージは、デバイスにインストールできません。ただし、これをバイパスしてソフトウェアをインストールする場合は、サードパーティ製ソフトウェアのインストールを有効にするようにデバイスを設定できます。設定は通常の設定として保持され、コマンドを使用して確認できます。**running-config** この設定に従って、既知のリスクがあるサードパーティ製ソフトウェアをインストールできます。

機能またはサードパーティ RPM の操作に **install CLI** を使用する

機能 RPM の操作に **install CLI** を使用するには、次のリファレンス表を参照してください。

表 4: 機能 **RPM** 操作用の **install CLI** リファレンス

CLI	説明
install reset	この操作は、すべてのパッチ、保持されたコンフィグレーション、アップグレードされたパッケージ、-サードパーティのインストール済みパッケージ、未保存のコンフィグレーションを削除し、デフォルトのパッケージを使用してスイッチの以前のモード（フル/基本）をリロードします。 install reset コマンドも write erase 操作を実行します。次のメッセージがプロンプトに表示されます。 <pre>switch(config)# install reset</pre> WARNING!!This operation will remove all patches, 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]
install reset nxos base	この操作は、すべてのパッチ、アップグレードされたパッケージ、保持された etc コンフィグレーション、-サードパーティのインストール済みパッケージ、スタートアップ設定（write erase）を削除して NX-OS をベース モードでインストールして、デフォルトのパッケージを使用してスイッチをリロードします。
install reset nxos full	この操作は、すべてのパッチ、アップグレードされたパッケージ、保持された etc コンフィグレーション、-サードパーティのインストール済みパッケージ、スタートアップ設定（write erase）を削除して NXOS をフル モードでインストールして、デフォルトのパッケージ（必須およびオプションの RPM による）を使用してスイッチをリロードします。
install add <>	それぞれのリポジトリに RPM ファイルを追加して、リポジトリ（patch/feature/third-party）を更新します。

CLI	説明
install activate <rpm name>	リポジトリに存在する RPM をインストールします。
install commit <rpm name>	パッチ RPM に使用します。リロード時にパッチを保持します。
install deactivate <rpm name>	RPM をアンインストールします。 Cisco NX-OS リリース 10.1（1）以降では、このコマンドを使用して RPM を非アクティブ化すると、RPM の基本バージョンにダウングレードするオプションまたは RPM をアンインストールするオプションが表示されます。必要なオプションを選択すると、操作が続行されます。
install remove <rpm name>	リポジトリから RPM ファイルを削除してリポジトリを更新します。
sh install active	ベース rootfs RPM 以外の、システムにインストールされている RPM のリストを表示します（機能/パッチ/サードパーティ）。
sh install inactive	リポジトリに保持されていてインストールされていない RPM のリストを表示します。
sh install packages	Rootfs RPM を含む、インストールされているすべての RPM をリストします。

CLI	説明
[no] system software allow third-party	Cisco NX-OS リリース 10.1 (1) 以降では、デフォルトでは、サードパーティ製 RPM をデバイスにインストールできません。このコマンドは、この制限をバイパスし、サードパーティ製ソフトウェアのインストールを有効にするようにデバイスを設定します。 次のコマンドは、サードパーティ コンフィギュレーションを適用せずにサードパーティ RPM をアクティブにした場合のエラーメッセージを示しています。 <pre>switch(config)# install activate pbwMonitor-1.0-1.5.0.x86_64.rpm</pre> 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 次のコマンドは、設定適用後のサードパーティ製 RPM インストールのアクティブ化を示しています。 <pre>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</pre> <pre>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</pre> 次のコマンドは、サードパーティ コンフィギュレーションの無効化を示しています。 <pre>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</pre>

(注) ISSUを使用している場合、または以前のバージョンから Cisco NX-OS リリース 10.1.1 リリースにアップグレードしている場合は、アップグレード後最初の30分以内にサードパーティの設定を手動で適用して、サードパーティの RPM をインストールする必要があります。

デジタル署名のサポートにインストール CLI を使用する

次の CLI コマンドを使用して、デジタル署名サポート用の install CLI を実行します。

手順の概要

1. switch#install add bootflash:<keyfile> gpg-key
2. switch#install verify package <package-name>
3. または switch#install verify bootflash:<RPM file>

手順の詳細

手順

	コマンドまたはアクション	目的
Step 1	switch#install add bootflash:<keyfile> gpg-key 例: <pre>install add bootflash:RPM-GPG-KEY-puppetlabs gpg-key [#####] 100% Install operation 304 completed successfully at Thu Jun 19 16:40:28 2018</pre>	Cisco GPG（NU Privacy Guard）キーを使用して Cisco リリース RPM に署名します。公開 GPG キーは、 /etc/pki/rpm-gpg/arm-Nexus9k-rel.gpg に存在します。異なるソースからその他の公開キーを追加するには、このセクションの手順を使用してください。
Step 2	switch#install verify package <package-name>	パッケージを検証します。
Step 3	または switch#install verify bootflash:<RPM file> 例: <pre>switch# install verify bootflash:vxlan-2.0.0.0-9.2.1.lib32_n9000.rpm RSA signed switch#</pre>	RPM ファイルが署名済みか未署名であるかどうかを確認するには、ステップ 2 または 3 を使用します。

インストールされているすべての RPM のクエリ

インストール済みのすべての RPM をクエリするには、次のステップを実行します。

手順の概要

1. show install packages

手順の詳細

手順

	コマンドまたはアクション	目的
Step 1	show install packages 例: <pre>switch# show install packages</pre> <pre>Boot Image: NXOS Image: bootflash:/nxos.9.2.1.bin</pre> ----- <pre>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-udhcp.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#</pre>	インストールされているすべての RPM をクエリします。

1 ステップ手順による RPM のインストール

RPM をインストールする CIL とアップグレードする CLI は同じです。1 ステップ手順を使用して RPM をインストールするには、次の手順を参照してください。

手順

	コマンドまたはアクション	目的
Step 1	install add <rpm>activate 例: <pre>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</pre>	RPM をインストールしてアクティブ化します。

例

```
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
switch(config)#
```

2 ステップ手順による RPM のインストール

RPM をインストールする CIL とアップグレードする CLI は同じです。2 ステップ手順を使用して RPM をインストールするには、以下の手順を参照してください。

手順の概要

1. **install add** <rpm>
2. **install activate** <rpm>

手順の詳細

手順

	コマンドまたはアクション	目的
Step 1	install add <rpm> 例: <pre>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 switch(config)# sh install inactive grep vxlan vxlan-2.0.1.0-9.2.1.lib32_n9000</pre>	RPM をインストールします。
Step 2	install activate <rpm> 例:	RPM をアクティブ化します。

例

```
switch#install activate vxlan

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

switch# show install active | grep vxlan

vxlan-2.0.0.0-9.2.1.lib32_n9000

switch# sh install inactive | grep vxlan

switch#
```

1 ステップの RPM のアップグレード

RPM をインストールする CIL とアップグレードする CLI は同じです。RPM をアップグレードするには、以下の手順を参照してください。

手順

	コマンドまたはアクション	目的
Step 1	install add <rpm>activate upgrade 例: <pre>switch(config)# install add bootflash:bgp-2.0.2.0-9.2.1.lib32_n9000.rpm activate upgrade</pre> <pre>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</pre> <pre>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</pre>	RPM をインストールします。

例

```
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
```

RPM のダウングレード

ダウングレード手順では、特別な CLI 属性を必要とします。1 ステップ手順を使用して RPM をダウングレードするには、次の手順を参照してください。

手順

	コマンドまたはアクション	目的
Step 1	install add <rpm>activate downgrade 例: <pre>switch(config)# install add bootflash:bgp-2.0.1.0-9.2.1.lib32_n9000.rpm activate downgrade</pre> <pre>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</pre> <pre>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</pre>	RPM をダウングレードします。

例

```
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.e15.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)#

```

RPM のインストール

RPM をアップグレードするには、以下の手順を参照してください。

手順の概要

1. install deactivate <rpm>

手順の詳細

手順

	コマンドまたはアクション	目的
Step 1	install deactivate <rpm> 例: <pre> 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 </pre> 例: <pre> switch(config)# install deactivate bgp Base RPM found. Do you want to downgrade to base version(y/n) [n] n </pre> <pre> 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. </pre> <pre> [#####] 100% Install operation 9 completed successfully at Tue Nov 17 05:05:59 2020 </pre>	groups-repo (/rpms) に RPM がある場合は、RPM の基本バージョンにダウングレードします。または、RPM をスイッチから完全にアンインストールします。 基本バージョンにダウングレードするには、yと入力します。RPM を完全にアンインストールするには、コマンドプロンプトで n と入力します。

RPM の削除

RPM を削除するには、次の手順を参照してください。

手順の概要

1. **install remove <rpm>**

手順の詳細

手順

	コマンドまたはアクション	目的
Step 1	install remove <rpm> 例: <pre>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</pre>	リポジトリから、RPM を削除します。

DNF コマンドに関する情報

DNF コマンドの詳細については、以降のセクションを参照してください。



- (注) DNF コマンドは **ctrl + c** をサポートしていません。インストール コマンドは **ctrl + c** をサポートしています。DNF コマンドを **ctrl + c** を使用して中断した場合は、「/isan/bin/patching_utils.py --unlock」を使用して手動でクリーンアップする必要があります。

DNF コマンドを使用したパッケージの操作

DNF コマンドを使用してパッケージを操作するには、以下のセクションを参照してください。



- (注) DNF コマンドは、ボックスの BASH シェルからのみアクセスできます。NXOS VSH ターミナルからはアクセスできません。



- (注) **sudo ユーザ**として、スーパー ユーザ権限にアクセスできることを確認してください。

イメージのベースバージョン RPM を特定する

ls/rpms コマンドを使用して、イメージのベースバージョン RPM を特定します。ベース RPM バージョンは、システム イメージにアーカイブされた、事前インストール済みの RPM です。

```
#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
```

インストール済み RPM のリストをチェックする

dnf list installed コマンドを使用して機能 RPM とサードパーティ RPM をクエリして、特定の RPM を **grep** 検索します。機能 RPM については、次の例を参照してください。

```
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
```

インストール済み RPM の詳細を取得する

```
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
...
```

インストール済み RPM の詳細を取得する

dnf info <rpmname> コマンドは、インストール済み RPM の詳細情報リストを出力します。

dnf info vxlan

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

```
localdb | 1.1 kB 00:00 ...
```

```
patching | 951 B 00:00 ...
```

```
thirdparty | 951 B 00:00 ...
```

```
| 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
```

RPM のインストール

RPM をインストールすると、RPM がダウンロードされ、それぞれのプログラムがスイッチにコピーされます。次の例は、RPM をリモートサーバ（ネットワークで到達可能）からインストールする場合を示しています。

```
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
```

```
localdb | 1.1 kB 00:00 ...
```

```
| 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	Repository	Arch	Version	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!

次の例は、RPM をローカル ブートフラッシュからインストールする場合を示しています。

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

```

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!

```

次の例は、RPM がリポジトリ内で利用可能な場合の RPM のインストールを示しています。

```
dnf install eigrp
```

RPM のアップグレード

次の例は、RPMをリモートサーバ（ネットワークで到達可能）からアップグレードする場合を示しています。

```
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	Repository	Arch	Version 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!

次の例は、RPM をローカル ブートフラッシュからアップグレードする場合を示しています。

```
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

```

```

localdb          | 1.1 kB      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	Repository
Version	Size	
vxlan	lib32_n9000	/vxlan-2.0.1.0-9.2.1.lib32_n9000
2.0.1.0-9.2.1	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!

次の例は、RPM がリポジトリ内で利用可能な場合の RPM のアップグレードを示しています。

```
dnf upgrade eigrp
```

RPM のダウングレード

次の例は、RPM をリモートサーバ（ネットワークで到達可能）からダウングレードする場合を示しています。

```
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	Version	Size	Arch	Repository
Downgrading:				
vxlan	2.0.0.0-9.2.1	1.6 M	lib32_n9000	groups-repo

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!

次の例は、RPM をローカルブートフラッシュからダウングレードする場合を示しています。

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

次の例は、RPM がいずれかのリポジトリ内で利用可能な場合の RPM のダウングレードを示しています。

```
dnf downgrade eigrp
```

RPM の削除

RPM を削除すると、RPM がアンインストールされ、機能のコンフィグレーション CLI がすべて削除されます。RPM を削除するには、**dnf erase <rpm>** コマンドを使用します。

```
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	Repository	Version	Size
Removing:				
vxlan	lib32_n9000	@/vxlan-2.0.1.0-9.2.1.lib32_n9000	2.0.1.0-9.2.1	6.4 M

```
Transaction Summary
```

```
Remove      1 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!
```

DNF のグループのサポート

DNF のグループのサポートは、パッケージ管理の一部です。これにより、管理者によるパッケージの管理が簡素化され、高度な柔軟性が提供されます。

管理者は、パッケージ（RPM）のリストを論理グループにグループ化して、さまざまな操作を実行することができます。DNF では、次のグループ コマンドがサポートされています。

- grouplist
- groupinfo

- groupinstall
- groupremove
- groupupdate

DNF のグループは、大きく分けて、L2、L3、ルーティング、および管理として分類できます。

groupinstall コマンドを使用する

Linux では、複数のパッケージを特定のグループにまとめることができます。dnf でパッケージを個別にインストールするのではなく、特定のグループをインストールして、そのグループに属するすべての関連パッケージをインストールできます。たとえば、使用可能なグループをすべてリストするには **dnf grouplist** コマンドを使用します。

```
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 コマンドを使用する

あるパッケージグループの説明と内容を表示するには、**dnf groupinfo** コマンドを使用します。このコマンドは、グループの機能メンバのリストを出力します。

```
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 コマンドを使用する

このコマンドは、メンバRPMのインストールとアップグレードの両方に使用します。メンバがインストールされていない場合は、使用可能な最も高いバージョンがインストールされます。メンバがすでにインストールされていてより高いバージョンのRPMが使用可能である場合、このコマンドでそのメンバがアップグレードされます。

```
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 コマンドを使用する

既存のインストール済みグループパッケージをすべて更新するには、**dnf groupupdate** コマンドを使用します。

```
bash-4.4# dnf groupupdate 13
```

```

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      :
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

```

grouperase コマンドを使用する

```

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 コマンドを使用する

グループ、またはグループのすべての RPM メンバを削除するには、**dnf grouperase** コマンドを使用します。

```

bash-4.4# dnf grouperase l3
Dependencies resolved.

```

Group	Packages
Marking packages as removed by the group:	
@L3	bfd

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

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#
```

リポジトリを特定する

このコマンドは、スイッチに存在するリポジトリに加え、これらのリポジトリに含まれる RPM の数をリストします。

```
bash-4.3# dnf repolist all
```

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

```

localdb          | 1.1 kB      00:00 ...
patching         | 951 B       00:00 ...
thirdparty       | 951 B       00:00 ...
repo id          | 951 B       00:00 ...
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
```

インストールされている DNF のバージョンを特定する

次に、インストールされている DNF のバージョンを表示する方法を示します。

```
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
```

NX-OS CLI と DNF コマンドとの対応

次の表は、NX-OS CLI とそれに対応する DNF コマンドを示しています。

表 5: パッチ適用コマンドリファレンス

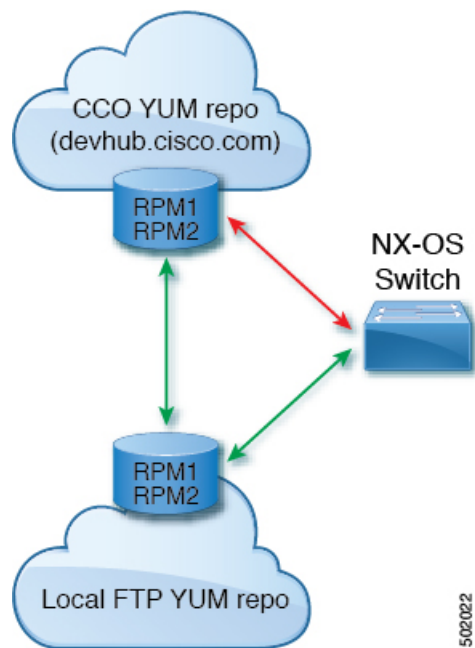
NX-OS CLI コマンド	DNF コマンド
show install inactive	dnf list --patch-only available
show install active	dnf list --patch-only installed
show install committed	dnf list --patch-only committed
show install packages	dnf list --patch-only
show install pkg-info	dnf info --patch-only
show install log	dnf history --show-patch-log ここで log_cmd は次のとおりです。 • opid: ある操作 ID に固有なログ • last: 最新の操作のログを示します。 • reverse: 逆の順序でログを表示します。 • detail: 詳細ログを表示します。 • from: 特定の操作 ID 以降のログを示します。
clear install log	dnf history --clear-patch-log= ここで clear_log_cmd は次のとおりです。 • all: ログを完全にクリアします。 • : この操作 ID 以降のログをクリアします。
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 (注) デフォルトでは、すべてのパッケージがアクティブ化され、コミットされます。

NX-OS CLI コマンド	DNF コマンド
install deactivate	dnf erase --nocommit (注) デフォルトでは、すべてのパッケージが非アクティブ化され、コミットされます。
install commit	dnf install --commit
Install commit	dnf install --commit all

FTP サーバの設定とローカル FTP YUM リポジトリのセットアップ

ローカル FTP YUM リポジトリをセットアップするには、次の図に示すように、初めに FTP サーバを作成して、ローカル FTP YUM リポジトリを作成し、FTP サーバに到達するように Cisco NX-OS スイッチを設定します。

図 2: FTP サーバの設定とローカル FTP YUM リポジトリのセットアップ



502022



(注) Cisco NX-OS リリース 10.1 (1) については、Cisco リポジトリを参照してください。
<https://devhub.cisco.com/artifactory/open-nxos/10.1.1/open-nxos>

Red Hat Enterprise Linux 7 (RHEL7) 仮想マシン上に FTP サーバを作成する

Red Hat Enterprise Linux 7 (RHEL7) 仮想マシン上に FTP サーバを作成するには、次の手順を実行します。

手順の概要

1. `dnf install vsftpd`
2. `systemctl start vsftpd`
3. `systemctl status vsftpd`
4. `firewall-cmd --zone=public --permanent --add-port=21/tcp`
5. `firewall-cmd --zone=public --permanent --add-service=ftp`
6. `firewall-cmd --reload`
7. `wget ftp:// <ip of FTP server> /test.txt`

手順の詳細

手順

	コマンドまたはアクション	目的
Step 1	<code>dnf install vsftpd</code>	Vsftpd (FTP サーバ) をインストールします。
Step 2	<code>systemctl start vsftpd</code>	FTP サーバを開始します。
Step 3	<code>systemctl status vsftpd</code>	サーバのステータスを確認します。
Step 4	<code>firewall-cmd --zone=public --permanent --add-port=21/tcp</code>	外部システムからの FTP サービスへのアクセスを許可し、ポート 21 を開きます。
Step 5	<code>firewall-cmd --zone=public --permanent --add-service=ftp</code>	FTP サービスを追加します。
Step 6	<code>firewall-cmd --reload</code>	サーバをリロードします。
Step 7	<code>wget ftp:// <ip of FTP server> /test.txt</code>	FTP サーバ内のファイル (test.txt など) をホストし、そのファイルの Wget を試みます。 (注) <code>/var/ftp/</code> は、FTP サーバのデフォルト ホームディレクトリです。

ローカル FTP YUM リポジトリを作成する

外部リポジトリ RPM と FTP サーバを同期し、ローカル FTP YUM リポジトリを作成するには、次の手順を実行します。

手順の概要

1. **touch /etc/yum.repos.d/local.repo**
2. **vim /etc/yum.repos.d/local.repo**
3. **cat /etc/yum.repos.d/local.repo**
4. **dnf repolist**
5. **nohup reposync -r <repo-name mentioned in the local.repo> -p <directory path to sync>&**
6. **tail -f nouhup.out**

手順の詳細

手順

	コマンドまたはアクション	目的
Step 1	touch /etc/yum.repos.d/local.repo 例: <pre>bash-4.3#touch /etc/yum.repos.d/local.repo</pre>	/etc/yum.repos.d/ 下にリポジトリ ファイルを作成します。たとえば、 local.repo リポジトリを作成してベース URL を追加します。
Step 2	vim /etc/yum.repos.d/local.repo 例: <pre>bash-4.3#vim /etc/yum.repos.d/local.repo</pre> <pre>[localrepo] name=localrepo baseurl= https://devhub.cisco.com/artifactory/open-nxos/7.0-3-I2-1/x86_64/ enabled=1 gpgcheck=0 ssilverify=0</pre>	リポジトリ ファイルを編集し、localrepo の詳細をコピーします。 (注) ベース URL を必要なリポジトリ URL に変更します。
Step 3	cat /etc/yum.repos.d/local.repo 例: <pre>bash-4.3#cat /etc/yum.repos.d/local.repo</pre> <pre>[localrepo] name=localrepo baseurl= https://devhub.cisco.com/artifactory/open-nxos/7.0-3-I2-1/x86_64/ enabled=1 gpgcheck=0 ssilverify=0</pre>	ローカルリポジトリ データを確認して続行します。
Step 4	dnf repolist 例: <pre>bash-4.3# dnf repolist</pre> <pre>Loaded plugins: fastestmirror, langpacks Loading mirror speeds from cached hostfile * base: mirror.dhakacom.com * extras: mirror.dhakacom.com * updates: mirror.dhakacom.com</pre>	リポジトリの到達可能性を確認します。

FTP サーバに到達するようにスイッチを設定する

	コマンドまたはアクション	目的
	<pre>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</pre>	
Step 5	nohup reposync -r <repo-name mentioned in the local.repo> -p <directory path to sync>& 例: <pre>nohup reposync -r localrepo -p /var/ftp/ &</pre> このコマンドは、/var/ftp/ 内部に名前 local.repo でディレクトリを作成し、すべてのパッケージを devhub.cisco.com からこのディレクトリにダウンロードします。	すべてのパッケージを外部リポジトリから FTP サーバのホーム ディレクトリに同期します。
Step 6	tail -f nouhup.out	同期のステータスを確認します。

FTP サーバに到達するようにスイッチを設定する

FTP サーバに到達するようにスイッチを設定するには、次の手順を実行します。

手順の概要

1. **run bash sudo su**
2. **ip netns exec management ping <ip_address>**
3. **touch /etc/yum/repos.d/ftp.repo**
4. **vim /etc/yum/repos.d/ftp.repo**
5. **cat /etc/yum/repos.d/ftp.repo**
6. **ip netns exec management bash**
7. **dnf repolist**
8. **dnf list available**

手順の詳細

手順

	コマンドまたはアクション	目的
Step 1	run bash sudo su	Sudo ユーザとしてログインします。
Step 2	ip netns exec management ping <ip_address>	スイッチから ping コマンドを使用して、FTP サーバアドレスの到達可能性を確認します。

	コマンドまたはアクション	目的
Step 3	touch /etc/yum/repos.d/ftp.repo 例: <pre>bash-4.3# touch /etc/yum/repos.d/ftp.repo</pre>	/etc/yum/repos.d/の下にリポジトリファイルを作成します。たとえば、 ftp.repo リポジトリを作成します。
Step 4	vim /etc/yum/repos.d/ftp.repo 例: <pre>bash-4.3# vim /etc/yum/repos.d/ftp.repo</pre> <pre>[ftp] name=ftp baseurl= ftp://198.51.100.1/localrepo/ enabled=1 gpgcheck=0 ssilverify=0</pre>	リポジトリファイルを編集し、 ftp リポジトリの詳細をコピーします。 (注) ベース URL を必要な ftp サーバー IP に変更します。
Step 5	cat /etc/yum/repos.d/ftp.repo 例: <pre>bash-4.3# cat /etc/yum/repos.d/ftp.repo</pre> <pre>[ftp] name=ftp baseurl=ftp://198.51.100.1/localrepo/ enabled=1 gpgcheck=0 ssilverify=0</pre>	URL として FTP サーバアドレスを使用して、スイッチ上にリポジトリ ファイルを作成します。
Step 6	ip netns exec management bash	Bash シェル プロンプトを使用します。
Step 7	dnf repolist 例: <pre>bash-4.3# dnf repolist</pre> <pre>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</pre>	新しく作成されたリポジトリの到達可能性を確認します。
Step 8	dnf list available	新しいリポジトリで利用可能なパッケージをリストします。

インストール操作ユーザ ロールの作成

install コマンドは、**admin** ロールのユーザのみが使用できます。**install** コマンドは、RBAC によりユーザが利用できるようになります。該当する RBAC 設定ガイドラインを参照してください。

Compacting Cisco NX-OS ソフトウェア イメージの圧縮

Cisco NX-OS ソフトウェア画像の圧縮は、コピー要求を完了する前に画像ファイルのサイズを縮小します。送信元として SCP、HTTP、または HTTPS を使用し、宛先としてブートフラッシュまたは USB を使用します。次の例では SCP およびブートフラッシュを使用します。

```
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.
```

ファイルをスーパーバイザモジュールにコピーする前に、**compact** キーワードは NX-OS 画像を圧縮します。



- (注) ソフトウェア画像の圧縮は、SCP、HTTP、または HTTPS でのみサポートされています。その他のプロトコルで圧縮しようとする場合、システムは次のエラーを返します。

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



- (注) 圧縮された画像は、LXC 起動モードではサポートされません。



- (注) ソフトウェア画像の圧縮は、Cisco Nexus 9300 シリーズ プラットフォーム スイッチでのみサポートされます。

翻訳について

このドキュメントは、米国シスコ発行ドキュメントの参考和訳です。リンク情報につきましては、日本語版掲載時点で、英語版にアップデートがあり、リンク先のページが移動/変更されている場合がありますことをご了承ください。あくまでも参考和訳となりますので、正式な内容については米国サイトのドキュメントを参照ください。