



# サードパーティ製アプリケーション

- サードパーティ製アプリケーションについて (1 ページ)
- 注意事項と制約事項 (1 ページ)
- Python2 および依存パッケージのインストール (2 ページ)
- サードパーティのネイティブ RPM/パッケージのインストール (2 ページ)
- 永続的なサードパーティ RPM (4 ページ)
- VSH からの RPM のインストール (5 ページ)
- サードパーティ製アプリケーション (10 ページ)

## サードパーティ製アプリケーションについて

サードパーティ製アプリケーションの RPM は、<https://devhub.cisco.com/artifactory/open-nxos/9.2.1/> のリポジトリで入手できます。これらのアプリケーションは、Bash シェルで `dnf` コマンドを使用するか、NX-OS CLI を介してネイティブホストにインストールされます。

`dnf install rpm` コマンドを入力すると、Cisco DNF プラグインが実行されます。このプラグインは、RPM を表示されない場所にコピーします。スイッチのリロード時に、システムは RPM を再インストールします。

構成が `/etc` に置かれている場合、Linux プロセス、**incron** は、ディレクトリで作成されたアーティファクトをモニターし、それらを表示されない場所にコピーし、この場所から `/etc` にコピーし直します。

## 注意事項と制約事項

サードパーティ製アプリケーションの RPM には、次の注意事項と制約事項があります。

- Cisco NX-OS リリース 9.2(1) 以降では、エージェントが保存されている Cisco リポジトリにはありません。<https://devhub.cisco.com/artifactory/open-nxos/9.2.1/> のリポジトリでホストされているすべての RPM は、リリースキーで署名されています。
- NX-OS 10.1(1) リリースには、NX-Linux (シスコ独自の Linux ディストリビューション) に基づく新しいオペレーティングシステムと rootfs があるため、WRL5/WRL8 を使用して

構築されたサードパーティ製 RPM は NX-Linux と互換性がない場合がありますそのため、サードパーティ製ソフトウェアが機能しない可能性があります。この場合、以前のリリースで使用されていたアプリケーションの古いバージョンを削除し、NX-Linux と互換性のある新しいソフトウェアに置き換えます。このソフトウェアは、次の場所にあります。

<https://devhub.cisco.com/artifactory/open-nxos/10.1.1/>

- 署名付き RPM のインストールに関するガイドラインと手順については、『Cisco Nexus 9000 Series NX-OS Software Upgrade and Downgrade Guide, Release 9.2(x)』を参照してください。リポジトリの数などです。
- サードパーティ製アプリケーションは、スイッチの起動時に開始されます。サードパーティ製アプリケーションは、その通信インターフェイスが起動する前、またはスイッチと通信ピアまたはサーバー間のルーティングが確立される前に起動される可能性があります。したがって、すべてのサードパーティ製アプリケーションは、通信障害が発生した場合に堅牢になるように作成し、アプリケーションは接続の確立を再試行する必要があります。通信障害が発生してもアプリケーションに復元力がない場合は、「ラッパー」アプリケーションを使用して、目的のアプリケーションを起動する前に通信ピアが到達可能であることを確認するか、必要に応じて目的のアプリケーションを再起動する必要があります。
- Cisco NX-OS リリース 10.2(3)F 以降、Python2 および依存 RPM は NX-OS から削除されます。ただし、Python2 および依存する RPM は、devhub サイトからパッケージグループ packagegroup-nxos-64-python-2-deprecated-rpms としてインストールできます。

## Python2 および依存パッケージのインストール

次に、パッケージのインストールの完全なワークフローを示します。

```
switch# cat /etc/dnf/repos.d/open-nxos.repo
[open-nxos]
name=open-nxos
baseurl=https://devhub.cisco.com/artifactory/open-nxos/10.2.3/
enabled=1
gpgcheck=0
ssilverify=0

dnf info packagegroup-nxos-64-python-2-deprecated-rpms
dnf install packagegroup-nxos-64-python-2-deprecated-rpms
The output of these cmds will be available post KR3F CCO.
```

## サードパーティのネイティブ RPM/パッケージのインストール

パッケージのインストールの完全なワークフローは次のとおりです。

## 手順

エージェントが保存されているシスコのリポジトリを指すように、スイッチのリポジトリを設定します。

```
bash-4.2# cat /etc/dnf/repos.d/open-nxos.repo
[open-nxos]
name=open-nxos

baseurl=https://devhub.cisco.com/artifactory/open-nxos/9.2.1/
baseurl=https://devhub.cisco.com/artifactory/open-nxos/10.1.1/
enabled=1
gpgcheck=0
sslverify=0
```

CLI を使用してデジタル署名をインポートする手順については、『Cisco Nexus 9000 Series NX-OS Software Upgrade and Downgrade Guide, Release 9.2(x)』の「Using Install CLIs for Digital Signature Support」を参照してください。

フルインストールログを使用した yum dnf を使用した RPM のインストールの例。

例：

```
bash-4.2# dnf install splunkforwarder
Loaded plugins: downloadonly, importpubkey, localrpmDB, patchaction, patching, protect-packages
Setting up Install Process
Resolving Dependencies
--> Running transaction check
---> Package splunkforwarder.x86_64 0:6.2.3-264376 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

=====
Package                        Arch                Version             Repository           Size
=====
Installing:
splunkforwarder               x86_64              6.2.3-264376        open-nxos            13 M

Transaction Summary
=====
Install      1 Package

Total size: 13 M
Installed size: 34 M
Is this ok [y/N]: y
Downloading Packages:
Running Transaction Check
Running Transaction Test
Transaction Test Succeeded
Running Transaction
  Installing : splunkforwarder-6.2.3-264376.x86_64
                                                    1/1

complete

Installed:
  splunkforwarder.x86_64 0:6.2.3-264376

Complete!
bash-4.2#
```

パッケージが正常にインストールされたかどうかをスイッチに照会し、そのプロセスまたはサービスが稼働していることを確認する例。

例：

```
bash-4.2# dnf info splunkforwarder
Loaded plugins: downloadonly, importpubkey, localrpmDB, patchaction, patching, protect-packages
Fretta | 951 B 00:00 ...
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 : splunkforwarder
Arch : x86_64
Version : 6.2.3
Release : 264376
Size : 34 M
Repo : installed
From repo : open-nxos
Summary : SplunkForwarder
License : Commercial
Description : The platform for machine data.
```

## 永続的なサードパーティ RPM

次に、永続的なサードパーティ RPM の背後にあるロジックを示します。

- ローカルリポジトリは、永続的なサードパーティ RPM 専用です。**dnf /etc/yum/repos.d/ thirdparty.repo** は **/bootflash/.rpmstore/ thirdparty** を指します。
- コマンドを入力するたびに、RPM のコピーが **//bootflash/.rpmstore/ thirdparty** に保存されます。**dnf install third-party.rpm**
- リブート中に、サードパーティ製リポジトリ内のすべての RPM がスイッチに再インストールされます。
- /etc** 構成ファイルの変更は、**/bootflash/.rpmstore/config/etc** の下に保持され、**/etc** での起動時に再生されます。
- /etc** ディレクトリに作成されたスクリプトは、リロード後も保持されます。たとえば、**/etc/init.d/** の下に作成されたサードパーティのサービススクリプトは、リロード中にアプリケーションを起動します。



(注) iptables のルールは、bash シェルで変更された場合、再起動後は保持されません。

変更した iptables を永続化するには、「」を参照してください。  
[リロード間で Iptable を永続化する](#)

## VSH からの RPM のインストール

### パッケージの追加

NX-OS 機能 RPM は、VSH CLI を使用してインストールすることもできます。

#### 手順の概要

1. **show install package**
2. **install add ?**
3. **install add rpm-packagename**

#### 手順の詳細

##### 手順

|        | コマンドまたはアクション                       | 目的                                                                         |
|--------|------------------------------------|----------------------------------------------------------------------------|
| ステップ 1 | <b>show install package</b>        | すでに存在するパッケージとバージョンを表示します。                                                  |
| ステップ 2 | <b>install add ?</b>               | サポートされている URI を決定します。                                                      |
| ステップ 3 | <b>install add rpm-packagename</b> | The <b>install add</b> コマンドは、ローカルストレージデバイスまたは、ネットワークサーバーへパッケージファイルをコピーします。 |

#### 例

次に、Chef RPM をアクティブにする例を示します：

```
switch# show install package
switch# install add ?
WORD          Package name
bootflash:    Enter package uri
ftp:          Enter package uri
http:         Enter package uri
modflash:     Enter package uri
```

```

scp:      Enter package uri
sftp:     Enter package uri
tftp:     Enter package uri
usb1:     Enter package uri
usb2:     Enter package uri
volatile: Enter package uri
switch# install add
bootflash:chef-12.0.0alpha.2+20150319234423.git.1608.b6eb10f-1.e15.x86_64.rpm
[#####] 100%
Install operation 314 completed successfully at Thu Aug 6 12:58:22 2015

```

### 次のタスク

パッケージをアクティブ化する準備ができたなら、[パッケージのアクティブ化（6 ページ）](#) に移動します。



(注) RPM パッケージの追加とアクティブ化は、次の 1 つのコマンドで実行できます。

```

switch#
install add bootflash:chef-12.0.0alpha.2+20150319234423.git.1608.b6eb10f-1.e15.x86_64.rpm
activate

```

## パッケージのアクティブ化

### 始める前に

RPM は事前に追加しておく必要があります。

### 手順の概要

1. **show install inactive**
2. **install activate rpm-packagename**

### 手順の詳細

#### 手順

|        | コマンドまたはアクション                            | 目的                                    |
|--------|-----------------------------------------|---------------------------------------|
| ステップ 1 | <b>show install inactive</b>            | 追加されていても、アクティブ化されていないパッケージのリストを表示します。 |
| ステップ 2 | <b>install activate rpm-packagename</b> | パッケージをアクティブ化します。                      |

### 例

次に、パッケージをアクティブ化する例を示します：

```

switch# show install inactive
Boot image:
    NXOS Image: bootflash:///yumcli6.bin

Inactive Packages:
    sysinfo-1.0.0-7.0.3.x86_64
Loaded plugins: downloadonly, importpubkey, localrpmDB, patchaction, patching,
               : protect-packages
Available Packages
chef.x86_64          12.0.0alpha.2+20150319234423.git.1608.b6eb10f-1.el5 thirdparty
eigrp.lib32_n9000 1.0.0-r0                                groups-rep
o
sysinfo.x86_64      1.0.0-7.0.3                                patching
switch# install activate chef-12.0-1.el5.x86_64.rpm
[#####] 100%
Install operation completed successfully at Thu Aug  6 12:46:53 2015

```

## パッケージの非アクティブ化

### 手順の概要

1. `install deactivate package-name`

### 手順の詳細

#### 手順

|        | コマンドまたはアクション                                 | 目的                    |
|--------|----------------------------------------------|-----------------------|
| ステップ 1 | <code>install deactivate package-name</code> | RPM パッケージを非アクティブ化します。 |

### 例

次に、Chef RPM パッケージを非アクティブ化する例を示します。

```
switch# install deactivate chef
```

## パッケージの削除

### 始める前に

パッケージを削除する前に非アクティブ化します。非アクティブ化された RPM パッケージのみ削除できます。

### 手順の概要

1. `install remove package-name`

## 手順の詳細

## 手順

|        | コマンドまたはアクション                       | 目的               |
|--------|------------------------------------|------------------|
| ステップ 1 | <b>install remove package-name</b> | RPM パッケージを削除します。 |

## 例

次に、Chef RPM パッケージを削除する例を示します。

```
switch# install remove chef-12.0-1.el5.x86_64.rpm
```

## インストール済みパッケージの表示

## 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                          |
|--------|------------------------------|-----------------------------|
| ステップ 1 | <b>show install packages</b> | インストールされているパッケージのリストを表示します。 |

## 例

次に、インストールされているパッケージのリストを表示する例を示します。

```
switch# show install packages
```

## 詳細ログの表示

## 手順の概要

## 1. show tech-support install

## 手順の詳細

## 手順

|        | コマンドまたはアクション                     | 目的          |
|--------|----------------------------------|-------------|
| ステップ 1 | <b>show tech-support install</b> | 詳細ログを表示します。 |

## 例

次の例は、詳細ログを表示する方法を示しています。

```
switch# show tech-support install
```

## パッケージのアップグレード

## 手順の概要

1. インストール 追加 *package-name* アクティベート アップグレード

## 手順の詳細

## 手順

|        | コマンドまたはアクション                                  | 目的                |
|--------|-----------------------------------------------|-------------------|
| ステップ 1 | インストール 追加 <i>package-name</i> アクティベート アップグレード | パッケージをアップグレードします。 |

## 例

次に、パッケージをアップグレードする例を示します：

```
switch# install add bootflash:bgp-1.0.1-r0.lib32_n9000.rpm activate ?
downgrade Downgrade package
forced      Non-interactive
upgrade     Upgrade package
switch# install add bootflash:bgp-1.0.1-r0.lib32_n9000.rpm activate upgrade
[#####] 100%
Install operation completed successfully at Thu Aug 6 12:46:53 2015
```

## パッケージのダウングレード

### 手順の概要

1. インストール 追加 *package-name* アクティベート ダウングレード

### 手順の詳細

#### 手順

|        | コマンドまたはアクション                                  | 目的                |
|--------|-----------------------------------------------|-------------------|
| ステップ 1 | インストール 追加 <i>package-name</i> アクティベート ダウングレード | パッケージをダウングレードします。 |

#### 例

次の例は、パッケージをダウングレードする方法を示しています。

```
switch# install add bootflash:bgp-1.0.1-r0.lib32_n9000.rpm activate ?
downgrade Downgrade package
forced Non-interactive
upgrade Upgrade package
switch# install add bootflash:bgp-1.0.1-r0.lib32_n9000.rpm activate downgrade
[#####] 100%
Install operation completed successfully at Thu Aug 6 12:46:53 2015
```

## サードパーティ製アプリケーション

### NX-OS

NX-API REST API オブジェクトモデルの仕様の詳細については、<https://developer.cisco.com/docs/nx-api-dme-model-9-3-1-reference/> を参照してください。

## DevOps 構成管理ツール

DevOps 構成管理ツールについては、次のリンクを参照してください。

- Ansible 2.0 リリース (Nexus サポート)、[Ansible リリース インデックス](#)
- Ansible NX-OS サンプル モジュール、[Ansible NX-OS サンプル モジュール](#)
- Puppet、[Puppet Forge Cisco Puppet](#)
- Cisco Puppet モジュール (Git) [Cisco Network Puppet モジュール](#)

- Chef、[Chef Supermarket Cisco クックブック](#)
- Cisco Chef クックブック (Git) [Cisco Network Chef クックブック](#)

## V9K

ESX5.1/5.5、VirtualBox、Fusion、および KVM の場合、仮想 Nexus 9000 スイッチをダウンロードするには、<https://software.cisco.com/portal/pub/download/portal/select.html?&mdfid=286312239&flowid=81422&softwareid=282088129>に移動します。

## 自動化ツールの教育コンテンツ

OpenNX-OS アーキテクチャと自動化に関する無料の書籍については、次を参照してください。  
[http://www.cisco.com/c/dam/en/us/td/docs/switches/datacenter/nexus9000/sw/open\\_nxos/programmability/guide/Programmability\\_Open\\_NX-OS.pdf](http://www.cisco.com/c/dam/en/us/td/docs/switches/datacenter/nexus9000/sw/open_nxos/programmability/guide/Programmability_Open_NX-OS.pdf)

## collectd

collectd は、システムパフォーマンスの統計情報を定期的に収集し、RRD ファイルなどの値を保存する複数の手段を提供するデーモンです。これらの統計情報を使用して、現在のパフォーマンスのボトルネック（パフォーマンス分析など）を見つけたり、将来のシステム負荷を予測したりできます（つまり、キャパシティ プランニング）。

詳細については、<https://collectd.org> を参照してください。

## Ganglia

Ganglia は、クラスタやグリッドなどのハイパフォーマンス コンピューティング システム向けのスケーラブルな分散モニタリングシステムです。これは、クラスタのフェデレーションを対象とした階層設計に基づいています。データ表現のための XML、コンパクトでポータブルなデータ転送のための XDR、データストレージと可視化のための RRDtool など、広く使用されているテクノロジーを活用します。設計されたデータ構造とアルゴリズムを使用して、ノードあたりのオーバーヘッドを低く抑え、同時実行性を高めます。この実装は堅牢であり、広範なオペレーティングシステムとプロセッサアーキテクチャに移植されており、現在、世界中の何千ものクラスタで使用されています。世界中の大学キャンパス間でクラスタをリンクするために使用されており、2000 ノードのクラスタを処理するように拡張できます。

詳細については、<http://ganglia.info> を参照してください。

## Iperf

Iperf は、TCP および UDP の最大帯域幅パフォーマンスを測定するために NLANR/DAST によって開発されました。Iperf を使用すると、さまざまなパラメータと UDP 特性を調整できます。Iperf は、帯域幅、遅延ジッターとデータグラム損失を報告します。

詳細については、<http://sourceforge.net/projects/iperf/>または<http://iperf.sourceforge.net>を参照してください。

## LLDP

リンク層検出プロトコル（LLDP）は、EDPやCDPなどの独自のリンク層プロトコルに代わるように設計された業界標準プロトコルです。LLDPの目的は、隣接するネットワークデバイスにリンク層通知を配信するための、ベンダー間互換性のあるメカニズムを提供することです。

詳細については、「<https://vincentbernat.github.io/lldpd/index.html>」を参照してください。

## Nagios

Nagios は、Nagios Remote Plug-in Executor（NRPE）およびSSHまたはSSLトンネルを介して以下をモニターするオープンソースソフトウェアです。

- ICMP、SNMP、SSH、FTP、HTTP などによるネットワーク サービス
- CPU 負荷、ディスク使用率、システム ログなどのホスト リソース
- サーバー、スイッチ、アプリケーションのアラート サービス
- [サービス（Services）]

詳細については、「<https://www.nagios.org/>」を参照してください。

## OpenSSH

OpenSSHは、盗聴、接続ハイジャック、およびその他の攻撃を排除するために、すべてのトラフィック（パスワードを含む）を暗号化するSSH接続ツールのオープンソースバージョンです。OpenSSHは、セキュアなトンネリング機能と複数の認証方式を提供し、すべてのSSHプロトコルバージョンをサポートします。

詳細については、「<http://www.openssh.com>」を参照してください。

## Quagga

Quaggaは、さまざまなルーティングプロトコルを実装するネットワークルーティングソフトウェアスイートです。Quaggaデーモンは、ネットワークアクセス可能CLI（「vty」という）を使用して構成できます。




---

（注） Quagga BGP のみが検証されています。

---

詳細については、「<http://www.nongnu.org/quagga/>」を参照してください。

## スプラunk

Splunk は、Web ベースのデータ収集、分析、およびモニタリング ツールであり、ユースケースに合わせて検索、可視化、および事前にパッケージ化されたコンテンツを備えています。raw データは、Splunk Universal Forwarder を使用して Splunk サーバーに送信されます。ユニバーサルフォワーダは、リモートソースからの信頼性の高いセキュアなデータ収集を可能にし、そのデータをインデックス作成と統合のために Splunk Enterprise に転送します。数万のリモートシステムに拡張でき、パフォーマンスへの影響を最小限に抑えながらテラバイト単位のデータを収集できます。

詳細については、[http://www.splunk.com/en\\_us/download/universal-forwarder.html](http://www.splunk.com/en_us/download/universal-forwarder.html) を参照してください。

## tccollector

tccollector は、ローカルコレクタからデータを収集し、そのデータをオープン時系列データベース (OpenTSDB) にプッシュするクライアント側プロセスです。

tccollector には次の機能があります。

- データ コレクタを実行し、データを照合します。
- 時系列データベース (TSD) への接続を管理します。
- コレクタに TSD コードを埋め込む必要がなくなります。
- 繰り返される値の重複を排除します。
- ワイヤ プロトコル作業を処理します。

詳細については、[http://opentsdb.net/docs/build/html/user\\_guide/utilities/tcollector.html](http://opentsdb.net/docs/build/html/user_guide/utilities/tcollector.html) を参照してください。

## tcpdump

tcpdump は、ネットワーク インターフェイス上の Boolean 式に一致するパケットの内容に関する説明を出力する CLI アプリケーションです。説明の前にタイムスタンプが表示されます。デフォルトでは、午前 0 時からの時間、分、秒、および小数点以下の秒として出力されます。

tcpdump は、次のフラグを使用して実行できます。

- **-w** : 後で分析するためにパケット データをファイルに保存します。
- **-r** : ネットワーク インターフェイスからパケットを読み取るのではなく、保存されたパケット ファイルから読み取ります。
- **-V** : 保存されたパケット ファイルのリストを読み取ります。

いずれの場合も、tcpdump は式にマッチするパケットだけを処理します。

詳細については、「<http://www.tcpdump.org/manpages/tcpdump.1.html>」を参照してください。

## TShark

TShark は、CLI のネットワーク プロトコル アナライザ です。Tshark を使用すると、ライブ ネットワーク から パケット データ を キャプチャ したり、以前に保存した キャプチャ ファイル から パケット を読み取ったり できます。これらのパケットのデコードされた形式を標準出力に出力するか、パケットをファイルに書き込むことができます。TShark のネイティブ キャプチャ ファイル フォーマットは、**pcap** です。このフォーマットは、**tcpdump** と他のツールに使用されています。TShark は、**cap\_net\_admin** ファイル機能を削除した後、ゲストシェル内で使用できます。

```
setcap  
cap_net_raw=ep /sbin/dumpcap
```



---

(注) このコマンドは、ゲストシェル内で実行する必要があります。

---

詳細については、「<https://www.wireshark.org/docs/man-pages/tshark.html>」を参照してください。

## 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。