



Cisco IOS XRv 9000 ルータ設置およびコンフィギュレーションガイド

最終更新：2024年11月13日

シスコシステムズ合同会社

〒107-6227 東京都港区赤坂9-7-1 ミッドタウン・タワー

<http://www.cisco.com/jp>

お問い合わせ先：シスコ コンタクトセンター

0120-092-255（フリーコール、携帯・PHS含む）

電話受付時間：平日 10:00～12:00、13:00～17:00

<http://www.cisco.com/jp/go/contactcenter/>

【注意】 シスコ製品をご使用になる前に、安全上の注意（www.cisco.com/jp/go/safety_warning/）をご確認ください。本書は、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。また、契約等の記述については、弊社販売パートナー、または、弊社担当者にご確認ください。

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB's public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED "AS IS" WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

All printed copies and duplicate soft copies of this document are considered uncontrolled. See the current online version for the latest version.

Cisco has more than 200 offices worldwide. Addresses and phone numbers are listed on the Cisco website at www.cisco.com/go/offices.

Cisco and the Cisco logo are trademarks or registered trademarks of Cisco and/or its affiliates in the U.S. and other countries. To view a list of Cisco trademarks, go to this URL: <https://www.cisco.com/c/en/us/about/legal/trademarks.html>. Third-party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1721R)

© 2017–2023 Cisco Systems, Inc. All rights reserved.



はじめに

- [マニュアルの変更履歴](#) (iii ページ)
- [通信、サービス、およびその他の情報](#) (iv ページ)
- [新機能および変更機能に関する情報](#) (v ページ)

マニュアルの変更履歴

この表に、初版後、このマニュアルに加えられた技術的な変更の履歴を示します。

日付	まとめ
2022 年 1 月	Cisco IOS XR Release 7.3.3 の機能に合わせてドキュメントを更新し再発行しました。
2021 年 11 月	Cisco IOS XR Release 7.5.1 の機能に合わせてドキュメントを更新し再発行しました。
2021 年 10 月	Cisco IOS XR Release 7.3.2 の機能に合わせてドキュメントを更新し再発行しました。
2021 年 2 月	Cisco IOS XR Release 7.3.1 の機能に合わせてドキュメントを更新し再発行しました。
2019 年 8 月	Cisco IOS XR Release 7.0.1 の機能に合わせてドキュメントを更新し再発行しました。
2019 年 4 月	Cisco IOS XR リリース 6.6.2 の機能に合わせてドキュメントを更新し再発行しました。
2018 年 12 月	Cisco IOS XR Release 6.6.1 の機能に合わせてドキュメントを更新し再発行しました。
2018 年 7 月	Cisco IOS XR Release 6.4.2 および 6.5.1 の機能に合わせてドキュメントを更新し再発行しました。

日付	まとめ
2018年3月	Cisco IOS XR Release 6.3.2 および 6.4.1 の機能に合わせてドキュメントを更新し再発行しました。
2017年9月	Cisco IOS XR Release 6.3.1 の機能に合わせてドキュメントを更新し再発行しました。
2017年7月	Cisco IOS XR Release 6.2.2 の機能に合わせてドキュメントを更新し再発行しました。
2017年3月	Cisco IOS XR Release 6.2.1 の機能に合わせてドキュメントを更新し再発行しました。
2016年11月	Cisco IOS XR Release 6.1.2 の機能に合わせてドキュメントを更新し再発行しました。
2016年5月	Cisco IOS XR Release 6.0.1 の機能に合わせてドキュメントを更新し再発行しました。
2015年12月	Cisco IOS XR Release 6.0.0 の機能に合わせてドキュメントを更新し再発行しました。
2015年8月	このマニュアルの初版

通信、サービス、およびその他の情報

- シスコからタイムリーな関連情報を受け取るには、[Cisco Profile Manager \[英語\]](#) でサインアップしてください。
- 重要な技術によりビジネスに必要な影響を与えるには、[Cisco Services \[英語\]](#) にアクセスしてください。
- サービスリクエストを送信するには、[Cisco Support \[英語\]](#) にアクセスしてください。
- 安全で検証済みのエンタープライズクラスのアプリケーション、製品、ソリューション、およびサービスを探して参照するには、[Cisco Marketplace \[英語\]](#) にアクセスしてください。
- 一般的なネットワーク、トレーニング、認定関連の出版物を入手するには、[Cisco Press \[英語\]](#) にアクセスしてください。
- 特定の製品または製品ファミリの保証情報を探すには、[Cisco Warranty Finder \[英語\]](#) にアクセスしてください。

Cisco バグ検索ツール

[Cisco バグ検索ツール](#) (BST) は、シスコ製品とソフトウェアの障害と脆弱性の包括的なリストを管理する Cisco バグ追跡システムへのゲートウェイとして機能する、Web ベースのツールです。BST は、製品とソフトウェアに関する詳細な障害情報を提供します。

新機能および変更機能に関する情報

表 1: リリース 7.3.1 の新機能および変更された機能の表

機能	導入されたリリース
ゴールデン ISO (GISO) の導入	リリース 7.3.1

リリース 7.0.1 の新機能および変更された機能の表

機能	導入されたリリース
セグメントルーティング	7.0.1
L3VPN の Carrier Supporting Carrier サポート	7.0.1
L3VPN の Inter-AS サポート	7.0.1
MPLS トラフィック エンジニアリング	7.0.1

リリース 6.6.2 の新機能および変更された機能の表

機能	導入されたリリース
M5 サーバベースのアプライアンス	6.6.2
bgp bestpath igp-metric ignore コマンドのサポート	6.6.2

リリース 6.6.1 の新機能および変更された機能の表

機能	導入されたリリース
vBNG での ACL ベースの転送	6.6.1
vBNG での不明瞭な VLAN	6.6.1
vBNG で PBR を使用した HTTP リダイレクト	6.6.1
vBNG での PPPoE LAC サポート	6.6.1

機能	導入されたリリース
IPv6 を介したセグメント ルーティング	6.6.1

リリース 6.4.2 および 6.5.1 の新機能および変更された機能の表

機能	導入されたリリース
新機能の導入なし	なし

リリース 6.4.1 の新機能および変更された機能の表

機能	導入されたリリース
Cisco IOS XRv 9000 アプライアンスの機能拡張	6.4.1
L2VPN 仮想プライベート ワイヤサービス (VPWS)	6.4.1
vBNG PPPoE	6.4.1
マルチキャスト機能	6.4.1

リリース 6.3.2 の新機能および変更された機能の表

機能	導入されたリリース
新機能の導入なし	なし

リリース 6.3.1 の新機能および変更された機能の表

機能	導入されたリリース
AWS での Cisco IOS XRv 9000 ルータの展開	6.3.1
ブロードバンド ネットワーク ゲートウェイ (BNG)	6.3.1
IPv4 を介した総称ルーティング カプセル化	6.3.1
仮想ローカル モビリティ アンカー (vLMA)	6.3.1
Docker および LXC コンテナでの VRF のサポート	6.3.1

リリース 6.2.2 の新機能および変更された機能の表

機能	導入されたリリース
新機能の導入なし	なし

リリース 6.2.1 の新機能および変更された機能の表

機能	導入されたリリース
BGP Persistenc	6.2.1
ホットスタンバイルータプロトコル (HSRP)	6.2.1
NSH プロキシモード	6.2.1
RT Constriant	6.2.1
仮想ルータ冗長プロトコル (VRRP)	6.2.1

リリース 6.1.2 の新機能および変更された機能の表

機能	導入されたリリース
アプリケーション ホスティング	6.1.2
アプライアンス	6.1.2
論理バンドル上の BFD	6.1.2
リンク バンドルまたは Link Aggregation Group (LAG)	6.1.2
マルチソケット データプレーン	6.1.2
ネットワーク サービス ヘッダー (NSH)	6.1.2

リリース 6.0.1 の新機能および変更された機能の表

機能	導入されたリリース
BGP の最適なルートリフレクタ (BGP-ORR)	6.0.1
双方向アクティブ測定プロトコル (TWAMP)	6.0.1
階層型 QoS	6.0.1
マルチ ソケット スケール アウト	6.0.1



第 1 章

インストールの準備

この章では、Cisco IOS XRv 9000 ルータ インストールの前提条件についての情報を提供します。

- [はじめに \(1 ページ\)](#)
- [ソフトウェアの設定と管理 \(3 ページ\)](#)
- [VM のプロビジョニング \(4 ページ\)](#)
- [ルータ インターフェイス \(4 ページ\)](#)
- [ルータ ネットワーク インターフェイスの VM ネットワーク インターフェイス カードへのマッピング \(5 ページ\)](#)
- [Cisco IOS XRv 9000 インストール ファイル \(7 ページ\)](#)
- [Cisco IOS XRv 9000 ルータのインストール \(9 ページ\)](#)

はじめに

Cisco IOS XRv 9000 ルータは、64 ビット IOS XR ソフトウェアを実行する x86 サーバハードウェアの仮想マシン (VM) インスタンスに展開されるクラウドベースのルータです。Cisco IOS XRv 9000 ルータは、仮想化されたフォーム ファクタで従来型のプロバイダー エッジ サービスを提供するとともに、仮想ルート リフレクタの機能も提供します。Cisco IOS XRv 9000 ルータは、Cisco IOS XR ソフトウェアに基づいているため、その他の IOS XR プラットフォームで使用可能な広範なルーティング機能を継承および共有しています。

Cisco IOS XRv 9000 ルータで使用可能な IOS XR 機能については、Cisco IOS XRv 9000 ルータの最新リリース ノートにある「サポートされる Cisco IOS XR テクノロジー」に関する項を参照してください。

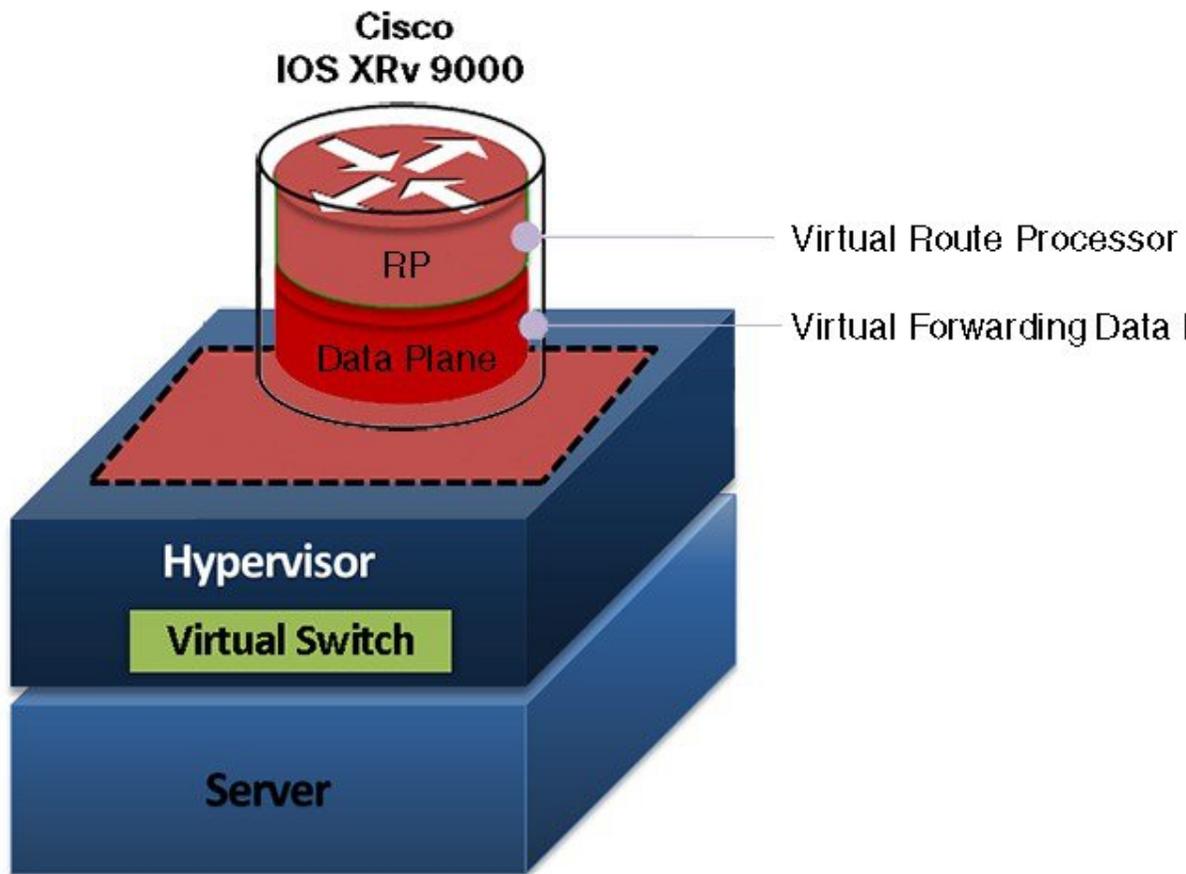
表 2: 最新のリリース ノートの参照先

リリース	リリース ノートの参照先
7.3.1	Cisco IOS XRv 9000 ルータ リリース ノート、IOS XR リリース 7.3.1

リリース	リリース ノートの参照先
6.4.2	Cisco IOS XRv 9000 ルータ リリース ノート、IOS XR リリース 6.4.2
6.4.1	Cisco IOS XRv 9000 ルータ リリース ノート、IOS XR リリース 6.4.1
6.3.2	Cisco IOS XRv 9000 ルータ リリース ノート、IOS XR リリース 6.3.2
6.3.1	Cisco IOS XRv 9000 ルータ リリース ノート、IOS XR リリース 6.3.1
6.2.3	Cisco IOS XRv 9000 ルータ リリース ノート、IOS XR リリース 6.2.3
6.2.25	Cisco IOS XRv 9000 ルータ リリース ノート、IOS XR リリース 6.2.25
6.2.2	Cisco IOS XRv 9000 ルータ リリース ノート、IOS XR リリース 6.2.2
6.2.1	Cisco IOS XRv 9000 ルータ リリース ノート、リリース 6.2.1
6.1.4	Cisco IOS XRv 9000 ルータ リリース ノート、リリース 6.1.4 (注) これは長期メンテナンス リリースです。
6.1.2	Cisco IOS XRv 9000 ルータ リリース ノート、リリース 6.1.2
6.0.2	Cisco IOS XRv 9000 ルータ リリース ノート、リリース 6.0.2
6.0.1	Cisco IOS XRv 9000 ルータ リリース ノート、IOS XR リリース 6.0.1
6.0.0	Cisco IOS XRv 9000 ルータ リリース ノート、リリース 6.0.0
5.4.0	Cisco IOS XRv 9000 ルータ リリース ノート、リリース 5.4.0

Cisco IOS XRv 9000 ルータ仮想 IOS XE ソフトウェアが VM 上に展開される場合、Cisco IOS XR ソフトウェアは、あたかも従来の Cisco IOS XR ハードウェア プラットフォーム上に展開されているかのように機能します。Cisco IOS XRv 9000 ルータは、ルートプロセッサ、ラインカード、および仮想化された転送機能を 1 つの中央集中型転送インスタンスにまとめます。Cisco IOS XRv 9000 ルータには、すべての機能を備えた高速仮想 x86 データ プレーンがあります。

図 1: Cisco IOS XRv 9000 ルータの仮想フォーム ファクタ



この図は、Cisco IOS XRv 9000 ルータの基本的な仮想フォーム ファクタを示しています。Cisco IOS XRv 9000 ルータは、ハイパーバイザ上の VM として展開されます。Cisco IOS XRv 9000 ルータは、さまざまな接続モデルを提供します。このモデルには、仮想スイッチ (vSwitch) に接続された仮想化インターフェイスや、パフォーマンスを最大化するために VM に直接入力される 10 G インターフェイスの物理パススルーが含まれます。

VM の要件と Cisco IOS XRv 9000 ルータの制限についての詳細は、[仮想マシンの要件 \(155 ページ\)](#) の項を参照してください。

Cisco IOS XRv 9000 ルータは、VMware または KVM のハイパーバイザ上に展開されます。サポートされるハイパーバイザの詳細は、[ハイパーバイザのサポート \(156 ページ\)](#) の項を参照してください。

ソフトウェアの設定と管理

ソフトウェアの設定と Cisco IOS XRv 9000 ルータの管理には、次の方法があります。

- VM にシリアルポートをプロビジョニングしてシリアルポートに接続し、Cisco IOS XR CLI コマンドにアクセスします。
- VM コンソールまたは仮想シリアルポートのコンソールを使用して、Cisco IOS XR CLI コマンドにアクセスします。



(注) 基盤となるハイパーバイザが VM とシリアルポートの関連付けをサポートしている場合にのみ、シリアルポートを使用して Cisco IOS XRv 9000 ルータ VM を管理できます。たとえば、Citrix XenServer 環境はシリアルポートの関連付けをサポートしていません。詳細については、ハイパーバイザのドキュメンテーションを参照してください。

- リモート SSH/Telnet を使用して、Cisco IOS XR CLI コマンドにアクセスします。
ルータ コンソールへのアクセス方法の詳細は、次を参照してください。 [仮想シリアルポート経由の Cisco IOS XRv 9000 ルータへのアクセス \(53 ページ\)](#)

システム アップグレードの詳細については、『System Setup and Software Installation Guide for Cisco NCS 6000 Series Routers』を参照してください。

VM のプロビジョニング

Cisco ハードウェア ルータは、通常、Cisco IOS XR ソフトウェアをプレインストールして出荷されます。Cisco IOS XRv 9000 ルータはハードウェア ベースではないため、Cisco.com から Cisco IOS XR ソフトウェアをダウンロードして仮想マシン (VM) に直接インストールする必要があります。ただし、初期インストールプロセスの一環として、Cisco IOS XRv 9000 イメージをインストールして起動できるように、まず VM の属性 (メモリやハードディスクなど) をプロビジョニングする必要があります。

また、Cisco IOS XRv 9000 アプライアンスは、出荷時にベアメタル UCS サーバーハードウェアに事前にインストールされた Cisco IOS XRv 9000 ルータソフトウェアです。Cisco IOS XRv 9000 アプライアンスには、該当するすべてのライセンスが含まれています。アプライアンス パッケージにより、ハードウェアとソフトウェアの所有権についての運用上の心配をせずに、ネットワーク ルーティング機能を仮想化できます。

ルータ インターフェイス

Cisco IOS XRv 9000 ルータ インターフェイスはハードウェア ベースのシスコ ルータと同じ機能を実行します。Cisco IOS XRv 9000 インターフェイスの命名規則は次のとおりです。

- インターフェイスは、GigabitEthernet インターフェイスとして論理的に命名されます。これらのインターフェイスは、VMXNET3 や E1000 などの仮想化されたインターフェイス

か、PCI パススルー経由で VM に渡される物理 1 ギガビット インターフェイスのどちらかです。

- PCI パススルー経由で VM に渡される物理 10 G インターフェイスの場合、インターフェイスは、TenGigabitEthernet インターフェイスとして論理的に命名されます。
- インターフェイスの番号は 0 から始まり、次のように単調に増加します。

```
interface GigabitEthernet 0/0/0/0
interface GigabitEthernet 0/0/0/1
interface GigabitEthernet 0/0/0/2
```

10 ギガビット インターフェイスの場合は次のとおりです。

```
interface TenGigabitEthernet 0/0/0/0
interface TenGigabitEthernet 0/0/0/1
```

ルータ ネットワーク インターフェイスの VM ネットワーク インターフェイス カードへのマッピング

Cisco IOS XRv 9000 ルータは、論理仮想ネットワーク インターフェイス カード (vNIC) または VM に割り当てられた物理 NIC を、管理イーサネット インターフェイス、GigabitEthernet インターフェイス、または TenGigE インターフェイスにマッピングします。

Cisco IOS XRv 9000 のルータが起動されるたびに、最初の NIC は仮想ルータの管理イーサネット インターフェイスとして使用され、2 番目および 3 番目の NIC は仮想ルータによって内部的に使用されます。この 3 つの NIC は、E1000 vNIC である必要があります。その他の NIC は、ライン インターフェイスとしてデータ プレーンにマッピングされます。

ライン インターフェイスの命名規則は、<GigabitEthernet | TenGigE> 0/0/0/<port number> です。

ポート番号は、NIC のタイプに割り当てられます。ポート番号の割り当てのルールを次に示します。

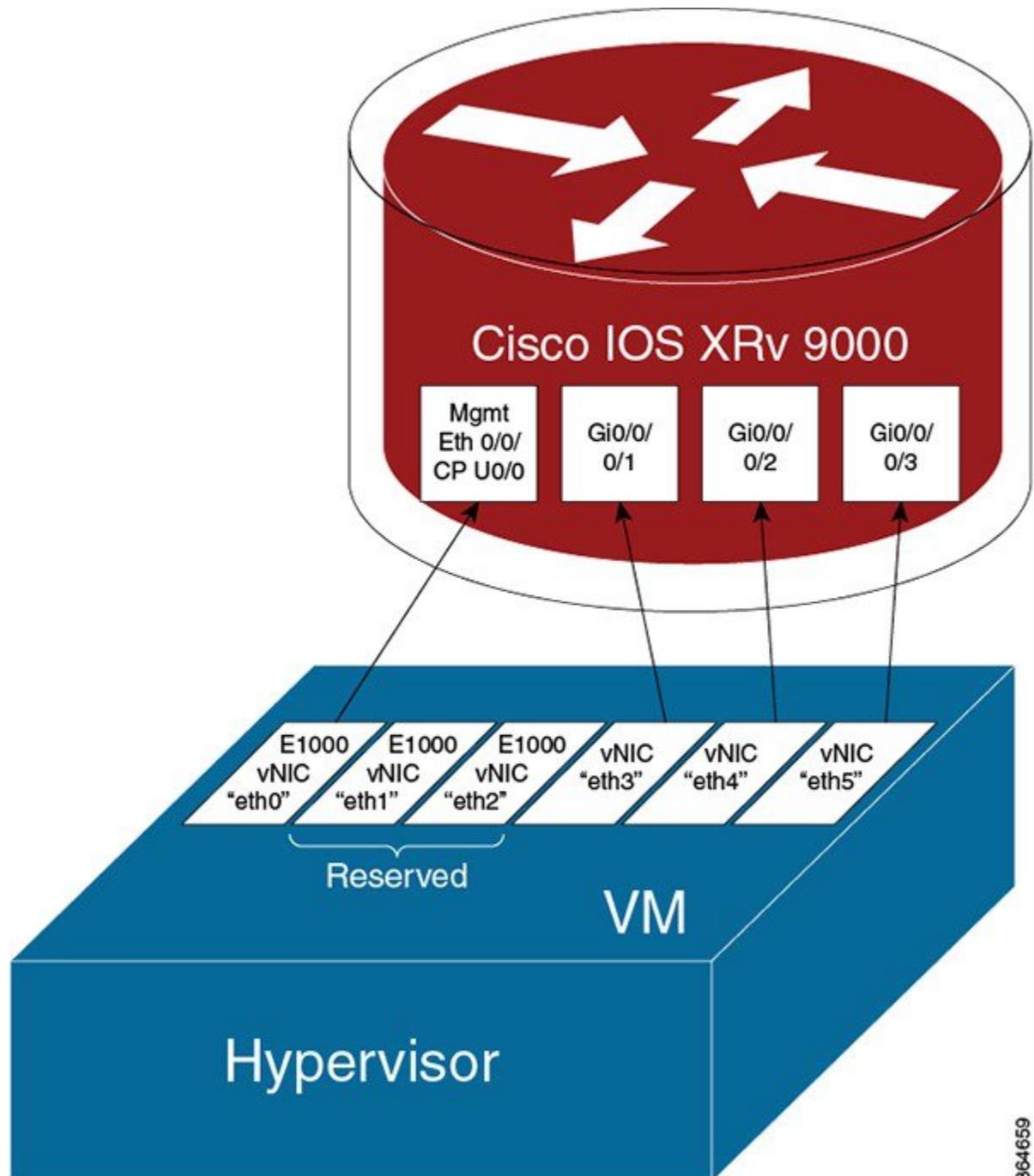
- 低速のインターフェイスの前に、高速のインターフェイスが割り当てられます。たとえば、10GE インターフェイスは 1GE の前に割り当てられます。
- 同じ速度の中では、高い PCI アドレスの前に、低いアドレスのものが割り当てられます。たとえば、04:00.0 は 04.00.1 の前に割り当てられます。

TenGigE NIC が 1 つ、物理 GigabitEthernet が 1 つ、およびデータ プレーンにマッピングされた仮想 NIC が 1 つあると仮定した場合、ライン インターフェイスの名前は次のようになります。

- TenGigE NIC は TenGigE 0/0/0/0 という名前になります。
- 物理 GigabitEthernet NIC は GigabitEthernet 0/0/0/1 という名前になります。
- 仮想 NIC は GigabitEthernet 0/0/0/2 という名前になります。

この図では、Cisco IOS XRv 9000 のルータにマッピングされた 6 つの NIC（仮想および物理）の例が示されています。この図は、VMware および KVM ハイパーバイザに当てはまります。

図 2: NIC と Cisco IOS XRv 9000 ルータとのマッピング



XRv 9000 アプライアンスのインターフェイスマッピングについては、[アプライアンスの物理的接続の概要](#)（124 ページ）を参照してください

Cisco IOS XRv 9000 インストール ファイル

Cisco IOS XRv 9000 ルータのソフトウェア イメージ パッケージには、次に示すファイル タイプが含まれています。これらのタイプは、サポート対象のハイパーバイザに Cisco IOS XRv 9000 ルータをインストールする際に使用します。

- **.iso** : VM にソフトウェア イメージをインストールするために使用されます。これは、サポートされているハイパーバイザ環境で VM を作成するために使用できます。
- **.ova** : VM での Open Virtualization Appliance (OVA) テンプレート (TAR 形式) の展開に使用されます。OVA イメージは、VMware ESXi ハイパーバイザに Cisco IOS XRv 9000 ルータを展開する場合に推奨されています。OVA ファイルには、Cisco IOS XRv 9000 ルータソフトウェアがインストールされた仮想マシンのディスク イメージ (VMDK) が含まれています。



(注) OVA ファイル内の VMDK ディスクは、ストリームに最適化された形式であり、直接起動することはできません。ストリームに最適化された VMDK ディスクを使用するには、そのディスクを標準の読み取り/書き込みディスク形式に変換する必要があります。OVA の導入時に、このディスクは ESXi によって透過的に標準の読み取り/書き込み VMDK 形式に変換されます。ストリームに最適化された VMDK は、標準のディスク ツール (**qemu-img** など) を使用して、標準の読み取り/書き込み VMDK ディスクまたは QCOW2 ディスクに変換することもできます。

- **qcow2** : KVM OpenStack 環境でソフトウェア イメージを起動するために使用します。qcow2 ディスク イメージには、インストール済みの Cisco IOS XRv 9000 ルータ ソフトウェアが含まれています。
- **virsh.xml** : Virsh を使用して KVM 環境で Cisco IOS XRv 9000 ルータを起動するために使用できるサンプルの XML です。
- **CML2** : CML2.1 (Cisco Modeling Lab) でラボを作成し、CML で XRv 9000 ルータを起動できます。

インストール イメージのバリエーション

上記のイメージタイプは **default** または **vrr** (仮想ルートリフレクタ) のバリエーションに含まれます。ファイル名に **vrr** が含まれていないイメージは、**default** のバリエーションです。

- **デフォルト** : 仮想プロバイダーエッジ (vPE) ルータとして動作するように調整されています。vPE は、コントロールプレーンの vRR に対してデータプレーンの CPU が多くなります。VPE は、ESXi などの他のハイパーバイザのデフォルトプロファイルです。VPE は高速ルーティングに使用します。

- `vrr` : 仮想ルートリフレクタ (vRR) として動作するように調整されています。vRR は、ベアメタルアプライアンスのデフォルトプロファイルです。

default タイプのインストールイメージは次のとおりです。

- `xrv9k-fullk9-x.iso`
- `xrv9k-fullk9-x.ova`
- `xrv9k-fullk9-x.qcow2`
- `xrv9k-fullk9-x.virsh.xml`

vrr タイプのインストールイメージは次のとおりです。

- `xrv9k-fullk9-x.vrr.iso`
- `xrv9k-fullk9-x.vrr.ova`
- `xrv9k-fullk9-x.vrr.qcow2`
- `xrv9k-fullk9-x.vrr.virsh.xml`

`xrv9k-li-x.pkg` : 合法的傍受のためのオプションパッケージ。

Cisco IOS XR ソフトウェア リリース 7.2.1 以降、`xrv9k-full(k9).iso` ビルドは廃止されました。Cisco IOS XR ソフトウェア リリース 7.2.1 以降、ポストビルドの一部として GISO イメージを構築しており、これらのイメージも `xrv9k-fullk9.iso` イメージに名前が変更されています。Cisco IOS XR ソフトウェア リリース 7.8.1 XRv 9000 プラットフォームは LJAM ビルドに移行し、LJAM の移行により、自動 GISO ポストビルドスクリプトは廃止されました。そのため、`xrv9k-mini.iso` と必要な RPM を含む GISO ツールを使用して GISO を構築する必要があります。Cisco.com の `xrv9k-full` および `xrv9k-fullk9.iso` は使用しないでください。

`xrv9k-mini.iso` と必要な RPM を含む GISO ツールを使用して GISO イメージを構築し、アップグレード/ダウングレードのインストールを試行します。

GISO ツールリンク : [GitHub](#) から `gisobuild.py` をダウンロード

制限事項は [ゴールドエン ISO ワークフロー \(33 ページ\)](#) を参照してください。

コンソールイメージのバリエーション

シリアルポートを介してログインしたくない場合は、VGA コンソールを提供する VGA イメージを使用できます。ファイル名に `vga` が含まれるインストールイメージは、コンソールイメージのバリエーションです。ルータのインストールに VGA イメージが使用されると、XR コンソールは VGA コンソールにマッピングされます。それ以外の場合は、XR コンソールが最初のシリアルポートにマッピングされます。インストール手順は、VGA イメージの有無にかかわらず同じです。

コンソールイメージのバリエーションは次のとおりです。

- `xrv9k-fullk9_vga-x.iso`
- `xrv9k-fullk9_vga-x.ova`

- xrv9k-fullk9_vga-x.qcow2
- xrv9k-fullk9_vga-x.vrr.iso
- xrv9k-fullk9_vga-x.vrr.ova
- xrv9k-fullk9_vga-x.vrr.qcow2
- xrv9k-fullk9-x_vga.virsh.xml
- xrv9k-fullk9-x.vrr_vga.virish.xml

Cisco IOS XRv 9000 ルータのインストール

Cisco IOS XRv 9000 ルータは、VMware か KVM ハイパーバイザのどちらかにインストールされます。Cisco IOS XRv 9000 ルータのインストールの詳細については、次のセクションを参照してください。

- Cisco IOS XRv 9000 ルータを VMware ESXi 環境にインストールする
- Cisco IOS XRv 9000 ルータを KVM 環境にインストールする

XRv9000 アプライアンスソフトウェアのインストールについては、[アプライアンスの設定 \(127 ページ\)](#) を参照してください



第 2 章

Cisco IOS XRv 9000 ルータを VMware ESXi 環境にインストールする

VMware ESXi ハイパーバイザに Cisco IOS XRv 9000 ルータをインストールするには、以下のファイルタイプが必要です。

- [.iso](#) : VM にイメージをインストールするために使用されます。これは、サポートされているすべてのハイパーバイザ環境で VM を作成するためにも使用できます。
- [.ova](#) : VM での OVA テンプレート (TAR 形式) の展開に使用されます。OVA イメージは、VMware ESXi ハイパーバイザに Cisco IOS XRv 9000 ルータを展開する場合に推奨されています。
- [VMware ESXi のインストール要件 \(11 ページ\)](#)
- [Cisco IOS XRv 9000 ルータを VM にインストールする \(12 ページ\)](#)
- [Cisco IOS XRv 9000 ルータを OVA テンプレートを使用して VM にインストールする \(13 ページ\)](#)
- [Cisco IOS XRv 9000 ルータを ISO テンプレートを使用して VM にインストールする \(14 ページ\)](#)
- [VMware ESXi 構成でのパフォーマンスの向上 \(18 ページ\)](#)

VMware ESXi のインストール要件

Cisco IOS XRv 9000 ルータをインストールする前に、必要なホストとクライアントソフトウェアを含めた VMware 環境のセットアップが必要になります。たとえば、VMware ESXi 環境に Cisco IOS XRv 9000 ルータをインストールする場合は、最初に vSphere クライアントをインストールする必要があります。

VMware ESXi のインストール要件については、Cisco IOS XRv 9000 ルータの最新のリリースノートを参照してください。

リリースノートのリンクについては、[表 2: 最新のリリースノートの参照先 \(1 ページ\)](#) を参照してください。



- (注) Cisco IOS XRv 9000 ルータの展開に OVA ファイルを使用すると、6 つの vNIC が自動的に作成されます。これらの vNIC は、Cisco IOS XRv 9000 ルータの起動後に手動で VM に追加できません。

VMware ESXi のサポートに関する情報とサポートされる VMware の機能および操作に関する情報については、次の各項を参照してください。

- [VMware ESXi のサポート情報](#)
- [サポートされている VMware 機能と操作](#)

Cisco IOS XRv 9000 ルータを VM にインストールする

VMware ESXi は、OVA および ISO ファイルを使用した Cisco IOS XRv 9000 ルータの VM へのインストールをサポートしています。



- (注) Citrix XenServer、KVM および Microsoft Hyper-V の実装は、.ova ファイルを使用した VM の展開をサポートしていません。VM のインストールには、.iso ファイルを使用する必要があります。

OVA ファイルを使用したインストール

OVA ファイルパッケージには、Cisco IOS XR リリースに基づいたデフォルトの VM 設定が入っている OVF ファイルが含まれています。default.ova パッケージは、Cisco IOS XRv 9000 ルータを仮想プロバイダー エッジ（高速仮想ルータ）として展開する場合に使用します。また、vrr.ova パッケージは、Cisco IOS XRv 9000 ルータを仮想ルートリフレクタ（高スケールルートリフレクタ）として展開する場合に使用します。

Cisco IOS XRv 9000 ルータを OVA ファイルを使用してインストールする方法の詳細は、[Cisco IOS XRv 9000 ルータを OVA テンプレートを使用して VM にインストールする](#)（13 ページ）を参照してください。

ISO ファイルを使用したインストール

Cisco IOS XRv 9000 ルータを ISO ファイルを使用してインストールする方法の詳細は、「[Cisco IOS XRv 9000 ルータを ISO テンプレートを使用して VM にインストールする](#)」を参照してください。

Cisco IOS XRv 9000 ルータを OVA テンプレートを使用して VM にインストールする

以下に示す手順は、Cisco IOS XRv 9000 ルータを展開する際の一般的なガイドラインです。ただし、実行する必要がある正確な手順は、VMware 環境と設定の特性に応じて異なる場合があります。

メモリ、CPU、NICなどのVM設定に変更を加えた場合、変更内容を有効にするためにはCisco IOS XRv 9000 ルータを再起動する必要があります。

始める前に

次のことを確認してください。

- vSphere Client がマシンにインストールされていること。
- ネットワーク経由で VM シリアルポートを接続するために適切なファイアウォールオプションを設定してあること。

手順

- ステップ 1** vSphere クライアントで、[File] > [Deploy OVF Template] を選択します。
- ステップ 2** Cisco IOS XRv 9000 ルータの .ova ファイルの保存先の場所を選択し、[Next] をクリックします。
- ステップ 3** OVF テンプレートの詳細を確認して、[Next] をクリックします。
- ステップ 4** VM の名前を指定し、インベントリの場所を選択して、[Next] をクリックします。
- ステップ 5** (Cisco IOS XR リリース 5.4 以降) ハードウェア導入設定をドロップダウンから選択し、[Next] をクリックします。
- ステップ 6** VM ファイルのデータストアを選択して、[Next] をクリックします。
- ステップ 7** 仮想ディスクの保存形式を選択して、[Next] をクリックします。

選択可能ディスク形式は以下のとおりです。

- Thick Provision Lazy Zeroed
- Thick Provision Eager Zeroed
- Thin Provisioned

(注)

Thick Provision Eager Zeroed オプションはインストールにかかる時間が長くなりますが、優れたパフォーマンスが得られます。また、Thick Provisioned は、ディスク上の物理的なスペースの消費量が多くなります。

ステップ 8 [Network Mapping] で、ドロップダウンリストを使用して、宛先ネットワーク上に1つ以上の仮想ネットワーク インターフェイス カード (vNIC) を割り当てます。

インターフェイス マッピングの詳細については、[ルータ ネットワーク インターフェイスの VM ネットワーク インターフェイス カードへのマッピング \(5 ページ\)](#) を参照してください。

ステップ 9 [Virtual Machine Properties] ウィンドウが利用可能なら、VM のプロパティを設定します。これはリリースに応じて異なります。

ステップ 10 VM の電源を自動的にオンにするには、[Power on after deployment] を選択します。

ステップ 11 [Finish] をクリックして OVA を展開します。

OVA により .iso ファイルが展開され、[Power on after deployment] 設定が選択されている場合は VM の電源が自動的にオンになります。VM の電源がオンになると、Cisco IOS XRv 9000 ルータによりインストールと起動プロセスが開始されます。ブートストラップ設定ファイルがOVAに含まれている場合は、ルータ設定が自動的に有効化されます。ブートストラップ設定ファイルについては、次を参照してください。
[CVAC : ブートストラップ構成のサポート \(78 ページ\)](#)

次のタスク

Cisco IOS XRv 9000 ルータのコンソールにアクセスします。詳細については、[コンソールマッピング \(51 ページ\)](#) のセクションを参照してください。

Cisco IOS XRv 9000 ルータを ISO テンプレートを使用して VM にインストールする

以下に示す手順は、VMware vSphere を使用して Cisco IOS XRv 9000 ルータを展開する際の一般的なガイドラインです。ただし、実行する必要がある正確な手順は、VMware 環境と設定の特性に応じて異なる場合があります。この手順のステップは、VMware ESXi 5.5 バージョンに基づいています。

始める前に

次のことを確認してください。

- vSphere Client がマシンにインストールされていること。
- ネットワーク経由で VM シリアルポートを接続するために適切なファイアウォール オプションを設定してあること。

手順

ステップ 1 Cisco IOS XRv 9000 ルータ ISO ファイルをダウンロードし、そのファイルを VM データストアにコピーします。

(注)

ISO ファイルの `_vga` バージョンを使用して XR コンソールを VGA コンソールにマッピングします。そうしない場合、XR コンソールは最初のシリアルポートにマッピングされることになります。[ルータネットワーク インターフェイスの VM ネットワーク インターフェイス カードへのマッピング \(5 ページ\)](#) を参照してください。

ステップ 2 vSphere クライアントで、[Create a New Virtual Machine] を選択します。

ステップ 3 [Configuration] の下で [Create a Custom configuration] を選択し、[Next] をクリックします。

ステップ 4 VM の名前を指定して、[Next] をクリックします。

ステップ 5 [Storage] の下で、VM ファイルのデータストアを選択して、[Next] をクリックします。

ステップ 6 [Virtual Machine version 8] を選択し、[Next] をクリックします。

(注)

Cisco IOS XRv 9000 ルータは、VMware ESXi Server の 5.0 より前のバージョンとは互換性がありません。

ステップ 7 ドロップダウンから [Linux] と [Other 2.6 Linux (64-bit) setting] を選択し、[Next] をクリックします。

ステップ 8 [CPU] の下で、次の設定を選択します。

- 仮想ソケット (仮想 CPU) の数
- ソケットあたりのコア数

ソケットごとのコアの数は、選択されている仮想ソケットの数に関係なく、常に 1 に設定する必要があります。たとえば、4 個の vCPU から構成される Cisco IOS XRv 9000 ルータでは、4 個のソケットとソケットごとに 1 個のコアとして設定する必要があります。

使用しているリリースでサポートされている仮想 CPU の数と対応する RAM 割り当て要件の詳細については、[VMware ESXi のインストール要件 \(11 ページ\)](#) の項を参照してください。

[Next] をクリックします。

ステップ 9 VM のメモリ サイズを設定します。[Next] をクリックします。

(注)

サポートされるメモリ サイズは 16GB です。

ステップ 10 [Network] で、少なくとも 4 個の仮想ネットワーク インターフェイス カード (vNIC) を割り当てます。

1. ドロップダウンから vNIC の数を選択します。

(注)

VMware ESXi 5.5 インターフェイスでは、最初の VM 作成中に 4 個の vNIC の作成が可能です。VM が作成され、Cisco IOS XRv 9000 が最初に起動された後、vNIC をさらに追加できます。

2. vNIC を追加します。

- 各 vNIC に異なるネットワークを選択します。2 番目と 3 番目の NIC は予約済みであることに注意してください。
- ドロップダウンメニューからアダプタ タイプを選択します。最初の 3 つの NIC に E1000 アダプタを選択します。後で、物理インターフェイス（パススルー）または VMXNET3 を選択することができます。使用しているリリースでサポートされるアダプタ タイプの詳細については、[VMware ESXi のインストール要件（11 ページ）](#) の項を参照してください。

（注）

リリース 6.0 以降では、VMXNET3 NIC がサポートされています。

3. 電源オン時に接続する vNIC をすべて選択し、[Next] をクリックします。

（注）

Cisco IOS XRv 9000 の実行中に、vSphere を使用して vNIC を VM に追加することができます。既存の VM に vNICs を追加する方法の詳細は、vSphere のマニュアルを参照してください。

ステップ 11 [SCSI Controller] の下で [LSI Logic Parallel] を選択し、[Next] をクリックします。

ステップ 12 [Select a Disk] の下で、[Create a new virtual disk] をクリックします。

ステップ 13 [Create a Disk] の下で、次の値を選択します。

- Capacity: Disk Size

使用しているリリースに必要な仮想ハードディスクのサイズの詳細については、[VMware ESXi のインストール要件（11 ページ）](#) の項を参照してください。

- Disk Provisioning

次のいずれかを選択します。

- Thick Provision Lazy Zeroed
- Thick Provision Eager Zeroed
- Thin Provisioned

（注）

Thick Provision Eager Zeroed オプションはインストールにかかる時間が長くなりますが、優れたパフォーマンスが得られます。また、Thick Provisioned も、ディスク上の物理的なスペースの消費量が多くなります。

- Location

仮想マシンの保存先。

[Next] をクリックします。

ステップ 14 [Advanced Options] の下で、仮想デバイス ノード用に [IDE (0:0)] を選択します。

ステップ 15 [Ready to Complete] 画面で、[Edit the virtual machine settings before completion] を選択します。

ステップ 16 [Continue] チェックボックスをクリックします。

ステップ 17 [New CD/DVD Drive] をクリックし、以下のことを実行します。

1. VM の起動元のデバイス タイプを選択します。

Cisco IOS XRv 9000 .iso ファイルから起動するため、[Datastore ISO file] オプションを選択します。ステップ 1 で設定したデータ ストアの .iso ファイルの場所を参照します。

2. [Device Status] フィールドで、[Connect at power on] チェックボックスにチェック マークを付けます。
3. VM を起動するホスト上の [Virtual Device Node CD/DVD] ドライブを選択します。

ステップ 18 シリアルポート（コンソールポート）を追加するには、[Hardware] タブで [Add] をクリックします。

（注）

シリアルポートの設定は、非 vga（デフォルト）イメージでは必須です。

シリアル コンソール アクセスの設定の詳細については、次を参照してください。 [VMware ESXi のシリアル コンソール アクセスの設定（53 ページ）](#)

ステップ 19 [Serial Port] を選択し、[Next] をクリックします。

ステップ 20 [Connect via Network] を選択し、[Next] をクリックします。

ステップ 21 [Server] を選択し、ホストの telnet アドレスと、1000 より高い未使用ポートを追加します。[Next] をクリックします。

ステップ 22 [Ready to Complete] 画面で、[Finish] をクリックします。

ステップ 23 ステップ 17 からステップ 22 までを繰り返して、3 つのシリアルポートを追加します。3 つのポートは、XR 補助ポート、admin ポート、admin 補助ポートです。

ステップ 24 [Resources] タブで [CPU setting] をクリックし、[Resource Allocation] の設定を [Unlimited] に設定します。

ステップ 25 [OK] をクリックします。

ステップ 26 [Finish] をクリックします。

これで、VM は Cisco IOS XRv 9000 ルータ用に設定され、起動する準備が整いました。VM の電源がオンになると、Cisco IOS XRv 9000 が起動されます。

次のタスク

VM コンソールの代わりに ESXi ホストのシリアルポートから Cisco IOS XRv 9000 ルータにアクセスして設定する場合は、VM の電源をオンにしてルータを起動する前に、この設定を使用するよう VM をプロビジョニングします。詳細については、[コンソール マッピング（51 ページ）](#) を参照してください。



-
- (注) デフォルトでは、XR コンソールは VM の最初のシリアルポートにマッピングされます。ただし、Cisco IOS XRv 9000 ルータの導入に VGA イメージを使用する場合、XR コンソールは VGA コンソールにマッピングされます。VGA コンソールには、vSphere Client の [Console] タブでアクセスします。ESXi では、VGA コンソールは、インストールが完了すると自動的に開くコンソールです。VGA コンソールに接続するための特定の手順はありません。
-

VMware ESXi 構成でのパフォーマンスの向上

次の操作を実行すると、VMware ESXi 構成でパフォーマンスを向上できます。

- VMware ESXi の電源管理を無効にします。

VMware ESXi の電源管理を無効にするには、[High Performance] 設定を選択します。詳細については、VMware のドキュメンテーションを参照してください。

VMware ESXi でのパフォーマンスの向上には、専用のシステム リソースが必要になるという短所もあります。



第 3 章

Cisco IOS XRv 9000 ルータを KVM 環境にインストールする

KVM ハイパーバイザに Cisco IOS XRv 9000 ルータをインストールするには、以下のファイルタイプが必要です。

- [.qcow2](#) : KVM OpenStack 環境でソフトウェア イメージを起動するために使用されます。
qcow2 ディスク イメージには、プレインストールされている Cisco IOS XRv 9000 ルータのインスタンスがあります。
- [.iso](#) と [.qcow2](#) : Virsh アプリケーションを使用して Cisco IOS XRv 9000 ルータ VM を手動で作成するために使用されます。また、Virsh コマンドを使用して KVM 環境で Cisco IOS XRv 9000 ルータを起動するために使用されるサンプル XML 設定が含まれる [virsh.xml](#) ファイルも必要です。
- [KVM のインストール要件 \(19 ページ\)](#)
- [KVM コマンドラインを使用した Cisco IOS XRv 9000 ルータのインストール \(20 ページ\)](#)
- [Virsh を使用した KVM での Cisco IOS XRv 9000 ルータのインストール \(23 ページ\)](#)
- [Cisco IOS XRv 9000 ルータの KVM インスタンスを OpenStack に作成する \(24 ページ\)](#)
- [KVM 構成でのパフォーマンスの向上 \(28 ページ\)](#)

KVM のインストール要件

Cisco IOS XRv 9000 ルータは、Kernel Virtual Machine (KVM) を使用する Ubuntu および Red Hat のディストリビューションでサポートされます。KVM に Cisco IOS XRv 9000 ルータをインストールするには、[.iso](#) ファイルまたは [.qcow2](#) イメージのどちらかを使用した VM の作成とインストールが必要になります。VM の起動には、KVM コマンドライン、Virsh または OpenStack を使用します。

KVM のインストール要件については、Cisco IOS XRv 9000 ルータの最新のリリース ノートを参照してください。

リリース ノートのリンクについては、[表 2: 最新のリリース ノートの参照先 \(1 ページ\)](#) を参照してください。



- (注)
- リリース 6.0 移行では、45 GB の最小 VM ハードディスク サイズがサポートされていません。
 - `du` コマンドを使用すると、仮想ディスク イメージによって使用される実際のディスク容量を確認できます。
 - PCIe パススルー インタフェースを VM で機能させるには、`grub` 設定でオプションの `intel_iommu=on` コマンドを設定する必要があります。
 - 組み込みデータプレーンを機能させるには、CPU フラグを VM に渡す必要があります。また、このフラグに `sse4_2` フラグが含まれている必要があります。

OpenStack での KVM サポートの詳細については、「[OpenStack での KVM のサポート](#)」の項を参照してください。

KVM コマンドラインを使用した Cisco IOS XRv 9000 ルータのインストール

この手順は、Cisco IOS XRv 9000 ルータ用の VM を手動で作成するための一般的なガイドラインです。実行する必要がある正確な手順は、KVM の環境と設定の特性に応じて異なる場合があります。詳細については、Red Hat Linux または Ubuntu のドキュメンテーションを参照してください。

手順

この手順では、3つのトラフィック インターフェイスと3つの必須インターフェイス（1つはXR管理用、2つは予約済み）のある ISO ブート設定を作成する方法を説明します。

1. `/usr/bin/kvm \`
KVM を起動します。
2. `-smbios type=1,manufacturer="cisco",product="Cisco IOS XRv 9000",uuid=97fc351b-431d-4cf2-9c01-43c283faf2a3 \`
VM インスタンスのユニバーサル固有 ID (UUID) を設定します。UUID は、VM インスタンスを特定するライセンスの一部として使用されます。
3. `-cpu host \`
ホスト CPU フラグをゲストに渡します。
4. `-drive file=/home/username/bnbMay13/workdir-username/disk1.raw,if=virtio,media=disk,index=1 \`
`-drive file=/home/username/bnbMay13/workdir-username/xrv9k-fullk9-x.iso.baked,media=cdrom,index=2`

\

ハードディスクを空にし、ISO を起動します。

5. `-m 16384 \`
メモリを作成します。
6. `-smp cores=4,threads=1,sockets=1 \`
4 つの仮想 CPU と 1 つのソケットを作成します。
7. `-enable-kvm \`
ハードウェア アクセラレーションを有効化します。
8. `-display none \`
コンソールアクセスのためにシリアルポートを使用する場合、VGA コンソールをエミュレートします。このステップは推奨されています。
9. `-rtc base=utc \`
リアルタイムクロック (RTC) を設定します。

10.


```
-netdev tap,id=host1,ifname=usernameLx1,script=no,downscript=no \
-netdev tap,id=host2,ifname=usernameLx2,script=no,downscript=no \
-netdev tap,id=host3,ifname=usernameLx3,script=no,downscript=no \
-device virtio-net-pci,romfile=,netdev=host1,id=host1,mac=52:46:84:57:A0:DA \
-device virtio-net-pci,romfile=,netdev=host2,id=host2,mac=52:46:C4:F4:36:0F \
-device virtio-net-pci,romfile=,netdev=host3,id=host3,mac=52:46:A5:C0:D0:C5 \
```

3 つの NIC を作成します。最初のは XR 管理インターフェイスにマッピングされ、2 番目と 3 番目は予約済みです。

11.


```
-netdev tap,id=data1,ifname=usernameXr1,script=no,downscript=no \
-netdev tap,id=data2,ifname=usernameXr2,script=no,downscript=no \
-netdev tap,id=data3,ifname=usernameXr3,script=no,downscript=no \
-device e1000,romfile=,netdev=data1,id=data1,mac=52:46:87:18:62:DF \
-device e1000,romfile=,netdev=data2,id=data2,mac=52:46:32:02:90:6F \
-device e1000,romfile=,netdev=data3,id=data3,mac=52:46:34:93:52:1F \
```

4 番目から 11 番目の NIC は、トラフィック ポートにマッピングされます。最低 1 つのトラフィック インターフェイスが推奨されています。

12.


```
-serial telnet:0.0.0.0:10621,nowait,server \
-serial telnet:0.0.0.0:14713,nowait,server \
-serial telnet:0.0.0.0:18090,nowait,server \
-serial telnet:0.0.0.0:17181,nowait,server \
```

コンソールにアクセスするための 4 つのシリアルポートを作成します。最初のシリアルポートは XR コンソールにマッピングされます。詳細については、「コンソール マッピング」の項を参照してください。最低 1 つのシリアルポートが推奨されています。

シリアル コンソールアクセスの設定の詳細については、[QEMU を使用した KVM のシリアル コンソールアクセスの設定 \(55 ページ\)](#) の項を参照してください。

13. -boot once=d &

ISO を 1 回のみ起動します。

例：

```
/usr/bin/kvm \
  -smbios type=1,manufacturer="cisco",product="Cisco IOS XRv
9000",uuid=97fc351b-431d-4cf2-9c01-43c283faf2a3 \
  -cpu host \
  -drive file=/home/username/bnbMay13/workdir-username/disk1.raw,if=virtio,media=disk,index=1
\
  -drive file=/home/username/bnbMay13/workdir-username/xrv9k-fullk9-x.iso.baked,media=cdrom,index=2
\
  -m 16384 \
  -smp cores=4,threads=1,sockets=1 \
  -enable-kvm \
  -daemonize \
  -display none \
  -rtc base=utc \
  -name IOS-XRv-9000:username \
  -runas username \
  -netdev tap,id=host1,ifname=usernameLx1,script=no,downscript=no \
  -netdev tap,id=host2,ifname=usernameLx2,script=no,downscript=no \
  -netdev tap,id=host3,ifname=usernameLx3,script=no,downscript=no \
  -device virtio-net-pci,romfile=,netdev=host1,id=host1,mac=52:46:84:57:A0:DA \
  -device virtio-net-pci,romfile=,netdev=host2,id=host2,mac=52:46:C4:F4:36:0F \
  -device virtio-net-pci,romfile=,netdev=host3,id=host3,mac=52:46:A5:C0:D0:C5 \
  -netdev tap,id=data1,ifname=usernameXr1,script=no,downscript=no \
  -netdev tap,id=data2,ifname=usernameXr2,script=no,downscript=no \
  -netdev tap,id=data3,ifname=usernameXr3,script=no,downscript=no \
  -device e1000,romfile=,netdev=data1,id=data1,mac=52:46:87:18:62:DF \
  -device e1000,romfile=,netdev=data2,id=data2,mac=52:46:32:02:90:6F \
  -device e1000,romfile=,netdev=data3,id=data3,mac=52:46:34:93:52:1F \
  -monitor telnet:0.0.0.0:11063,server,nowait \
  -serial telnet:0.0.0.0:10621,nowait,server \
  -serial telnet:0.0.0.0:14713,nowait,server \
  -serial telnet:0.0.0.0:18090,nowait,server \
  -serial telnet:0.0.0.0:17181,nowait,server \
  -boot once=d &
```

(注)

上記の例で `disk1.raw` というラベルのディスクは、`qemu-img` で作成できます。`qemu-img` は、仮想ハードディスクの形式を変換するユーティリティです。上記の例では、生ディスク形式ではなく、`qcow2` ディスク形式を使用することができます。

プレインストールされている `qcow2` イメージも使用できます。その場合、`cdrom` パラメータは削除されま

Virsh を使用した KVM での Cisco IOS XRv 9000 ルータのインストール

この手順は、Cisco IOS XRv 9000 用の VM を手動で作成するための一般的なガイドラインです。実行する必要がある正確な手順は、KVM の環境と設定の特性に応じて異なることがあります。詳細については、Red Hat Linux、Ubuntu、および Virsh のドキュメンテーションを参照してください。

始める前に

VM メモリのバレーニングはサポートされていないため、Virsh ファイルの中で memory と currentmemory の単位値（下記参照）は同じでなければなりません。

```
<memory unit='MB'>XXX</memory>
<currentMemory unit='MB'>XXX</currentMemory>
```

手順

ステップ 1 Cisco.com から .iso または .qcow2 イメージ、およびサンプル Virsh XML ファイルをダウンロードします。

ステップ 2 XML ファイルを編集して .iso または .qcow2 イメージを指すようにし、インターフェイス ソースを編集して目的の接続を指定するようにします。

```
<!-- CDROM HDB Disk -->
  <disk type='file' device='cdrom'>
    <driver name='qemu' type='raw' />
    <source file='/home/<username>/xrv9k-fullk9-x.iso' />
    <target dev='hdc' bus='ide' />
    <alias name='ide-cdrom' />
  </disk>
```

ステップ 3 qcow2 イメージを使用する場合は、CDROM セクションをコメントアウトします。

ステップ 4 iso イメージを使用する場合は、次のようにします。

1. 空の qcow2 ディスクを作成します。

```
qemu-img create -f qcow2 xrv9000.qcow2 45G
```

(注)

サポートされる最小の VM ハードディスク サイズは、リリース 6.0 の場合 45 GB、リリース 5.4 の場合は 55 GB です。

2. XML の HDA Disk セクションを編集して、作成した空の qcow2 ディスクを指すようにします。

```
<!-- HDA Disk -->
  <disk type='file' device='disk'>
    <driver name='qemu' type='qcow2' />
    <source file='/home/<username>/xrv9000.qcow2' />
```

```
<target dev='vda' bus='virtio' />
<alias name='virtio-disk0' />
</disk>
```

ステップ 5 Virsh ドメインを作成します。

```
virsh create xrv9k-fullk9-x.virsh.xml
```

ステップ 6 ステップ 5 で作成された Virsh ドメインを検証します。

```
virsh list xrv9k-fullk9-x.virsh.xml
Id      Name                                State
-----
149     IOS-XRv-9000_username_virsh       running
```

ステップ 7 以下のような Virsh コマンドを使用して、VM を管理します。

- **reboot**

```
virsh reboot IOS-XRv-9000_USER1_virsh
```

- **shutdown**

```
virsh shutdown IOS-XRv-9000_USER1_virsh
```

- **destroy**

```
virsh destroy IOS-XRv-9000_USER1_virsh
```

シリアル コンソール アクセスの設定の詳細については、[Virsh を使用した KVM のシリアル コンソール アクセスの設定 \(54 ページ\)](#) の項を参照してください。

詳細については、[Virsh コマンド リファレンス](#)のドキュメンテーションを参照してください。

Cisco IOS XRv 9000 ルータの KVM インスタンスを OpenStack に作成する

この手順では、Cisco IOS XRv 9000 ルータを RHEL バージョン 7 以降と OpenStack バージョン 5 以降で使用方法について説明します。この手順には OpenStack コマンドライン インターフェイスを使用します。この説明は、読者が OpenStack のコマンドと手順に精通していることを前提としています。詳細については、OpenStack のドキュメンテーションを参照してください。

手順を完了すると、3つの Neutron ネットワークに接続された3つのデータプレーンインターフェイスのある単一の Cisco IOS XRv 9000 ルータが動作することになります。



- (注) トランク インターフェイスが必要な場合は、Neutron ML2 コア プラグインを Nexus 1kv 機能プラグインと併用してください。次の手順では、インターフェイスのアクセス（非タギング）タイプのみ作成します。

始める前に

次のものがが必要です。

- Neutron ML2 コア プラグインを備えた OpenStack 5、6 または 7。機能プラグインとしての Open vSwitch、および Tenants ネットワーク タイプとしての VLAN。
- Cisco IOS XRv 9000 ルータ ISO イメージ（VGA タイプ）。

手順

ステップ 1 イメージの準備

OpenStack に Cisco IOS XRv 9000 ルータを導入するには、次のものがが必要です。

- Cisco IOS XRv 9000 ISO イメージ（VGA タイプ）
- 45 GB の空白 Cinder ボリューム（仮想ディスク）

(注)

サポートされる最小の VM ハードディスク サイズは、リリース 6.0 の場合 45 GB、リリース 5.4 の場合は 55 GB です。

ISO イメージを起動すると、Cisco XRv 9000 ルータが 2 番目のディスク（45 GB ボリュームの空白ディスク）にインストールされます。後で、作成された Cinder ボリュームからルータを起動できます。必要に応じて、ISO イメージの代わりに、プレインストールされていた Cisco XRv 9000 qcow2 ディスクをダウンロードすることもできます。

1. Cisco IOS XRv 9000 ISO イメージ（VGA タイプ）は、OpenStack の glance にインポートする必要があります。イメージを glance にインポートするには、次のコマンドラインを使用します。

```
glance image-create --name xrv9k-fullk9_vga-x --disk-format iso --container bare \
--file xrv9k-fullk9_vga-x.iso
```

2. OpenStack glance のイメージを確認します。

```
glance image-list
```

ID	Name	Disk Format
Container Format	Size	Status
71b44355-32a8-45e7-abfd-f52593f2dc1a	csr1000v-universalk9.03.15.00.S.155-2.S	qcow2
bare	1335427072	active

Cisco IOS XRv 9000 ルータの KVM インスタンスを OpenStack に作成する

```

| e779245a-9491-4bee-bc85-a73e9394b981 | xrv9k-fullk9_vga-x | iso
| bare | 775370752 | active |
| 3b3ade31-fae6-4354-9902-fb77452a65ab | xrvr-initial-config | iso
| bare | 358400 | active |

```

ステップ 2 Cinder ボリュームの準備

次のコマンドラインを使用して、Cinder ボリュームを作成します。

1. Cisco IOS XRv 9000 ルータのディスク イメージの Cinder ボリュームを作成し、ブート可能にします。

```

cinder create --display-name xrv9k-disk 45
cinder set-bootable volume-id True

```

cinder list コマンドを実行すると、ルータの ISO イメージのボリューム ID が表示されます。

2. Cinder ボリュームを確認します。

```

cinder list

```

Bootable	ID	Attached to	Status	Display Name	Size	Volume Type
true	cbef86dd-9819-4daa-81b2-4b905b287974	5262e1fe-37f5-4535-bf76-aab26cb86366	Available	xrv9k-disk	45	None

名前が **xrv9k-disk** で、**Status-Available** と **Bootable-True** に設定された Cinder ボリュームが表示されるはずです。

ステップ 3 Nova Flavor の設定

この仮想ハードウェア テンプレートは、OpenStack では**フレーバー**と呼ばれます。**nova flavor-create** コマンドを使用すると、新しいフレーバーを作成できます。**nova** のフレーバーは、RAM サイズ、ディスク、コア数に関する情報を、VM の新しいインスタンスが開始されるときに **Nova** のスケジューラ プロセスに渡すために使用されます。

Cisco IOS XRv 9000 ルータには、16 GB の RAM、45 GB のハードディスク、および 4 個の vCPU が必要です。

```

nova flavor-create xrv9k-flavor auto 16384 45 4

```

xrv9k-flavor はフレーバーの名前です。

ステップ 4 ネットワークの設定

Cisco IOS XRv 9000 ルータには、最低 4 つのネットワーク インターフェイスが必要です。最初の NIC または vNIC は XR 管理インターフェイスにマッピングされます。2 番目と 3 番目の NIC は予約済みで、残りの NIC はトラフィック インターフェイスにマッピングされます。インターフェイスマッピングの詳細については、「[ルータ ネットワーク インターフェイスの VM ネットワーク インターフェイス カードへのマッピング](#)」の項を参照してください。

ネットワーク設定のサンプルでは、6つの neutron ネットワークと6つの neutron サブネットワークを設定し、VM インスタンスへのパラメータとして6つのネットワーク ID を渡すことができます。これは、3つのトラフィック インターフェイスに相当します。 **neutron net-create <neutron-network-name>** を使用して、Neutron にネットワークを作成します。

次に例を示します。

```
neutron net-create management-xr
neutron net-create management-other
neutron net-create management-host
neutron net-create datalink-1
neutron net-create datalink-2
neutron net-create datalink-3
```

上記の例で、最初の3つのコマンドはコントロールプレーン ネットワークを作成し、最後の3つのコマンドはデータ プレーン ネットワークを作成しています。

次に、対応するネットワーク名を使用して、Neutron にサブネットワークを作成します。 **neutron subnet-create <neutron-network-name> <IP-subnet> --name <neutron-network-name>** コマンドを使用します。一貫性を確保するために、neutron とサブネットワークに同じ名前を指定します。

次に例を示します。

```
neutron subnet-create management-xr 10.50.70.0/26 --name management-xr
neutron subnet-create management-other 10.50.70.64/26 --name management-other
neutron subnet-create management-host 10.50.70.128/26 --name management-host
neutron subnet-create datalink-1 10.57.11.0/24 --name datalink-1
neutron subnet-create datalink-2 10.57.12.0/24 --name datalink-2
neutron subnet-create datalink-3 10.57.13.0/24 --name datalink-3
```

ステップ 5 neutron ルータに management-xr サブネットと management-host サブネットを指定

次のコマンドを使用して、neutron ルータに management-xr サブネットと management-host サブネットを指定します。

```
neutron router-interface-add <Neutron router name> <subnet id of management-xr>
neutron router-interface-add <Neutron router name> <subnet id of management-host>
```

(注)

- DHCP の問題を回避するために、Cisco IOS XRv 9000 ルータを起動する前にこの手順を実行する必要があります。
- **neutron router-list** コマンドを実行すると、Neutron ルータ名の一覧が表示されます。
- **neutron subnet-list** コマンドを実行すると、Neutron サブネットの一覧が表示されます。

ステップ 6 Cisco IOS XRv 9000 ルータの起動

次のコマンドラインを実行して、ルータを起動します。

```
nova boot
--flavor xrv9k-flavor \
--nic net-id={Control plane network ID of management-xr} \
--nic net-id={Control plane network ID of management-other} \
```

```

--nic net-id={Control plane network ID of management-host} \
--nic net-id={Data plane network ID of datalink-1} \
--nic net-id={Data plane network ID of datalink-2} \
--nic net-id={Data plane network ID of datalink-3} \
--block-device id={glance ID of Cisco IOS XRv 9000 router's CD volume available in Step 1},\
--source=image,dest=volume,bus=ide,device=/dev/hdc,size=1,type=cdrom,bootindex=1 \
--block-device source=volume,id={cinder ID of Cisco IOS XRv 9000 Router's disk volume available in Step 1}, dest=volume,size=50,bootindex=0 xrv9k-1
nova list (your instance should be in Active state)
nova get-vnc-console xrv9k-1 novnc

```

nova get-vnc-console コマンドは、Cisco IOS XRv 9000 ルータ コンソールへのアクセスに使用される URL を返します。

(注)

config-drive を実行すると、VM を初期設定で起動できます。これにより、VM の電源をオンにしたときに、**config-drive** に設定されているサービスが実行されます。この Cisco IOS XRv 9000 ルータは VM 上で動作しているため、ルータはプレーンテキストファイルを使用した有効な XR 設定を受け入れます。CLI からコマンドが入力された場合と同様、起動プロセス中にこのテキストファイルはコマンドパーサーによって解析されます。

config-drive のサポートが必要な場合は、**nova boot** コマンドに次の行を追加することにより、XR の初期設定を格納したプレーンテキストファイルを渡すことができます。

```
config-drive true user-data <path>/iosxr_config.txt file /iosxr_config.txt=<path>/iosxr_config.txt
```

シスコの仮想アプライアンス設定 (CVAC) の詳細については、「[CVAC : ブートストラップ構成のサポート](#)」の項を参照してください。

KVM 構成でのパフォーマンスの向上

KVM 環境内の Cisco IOS XRv 9000 ルータは、KVM ホストの設定を変更することでパフォーマンスの向上が可能です。これらの設定は、ルータの Cisco IOS XR の構成時の設定とは無関係です。このオプションは、Red Hat Enterprise Linux 7.0 KVM で使用できます。

CPU ピニングを有効にすると、KVM 構成でのパフォーマンスを向上できます。



(注) Cisco IOS XR ソフトウェア リリース 5.4 は、KVM の VirtIO インターフェイスで 1518 バイトを超えるジャンボ パケットをサポートしていません。1518 バイトを超えるパケットは破棄されます。

CPU ピニングの有効化

KVM 環境で Cisco IOS XRv 9000 ルータのパフォーマンスを向上させるため、KVM CPU アフィニティ オプションを使用して特定のプロセッサに VM を割り当てることができます。このオプションを使用する場合は、KVM ホストで CPU ピニングを構成します。

CPU ピンニングを構成するには、次の手順を実行します。

1. KVM ホスト環境で、ピンニングに使用できる vCPU の数を調べるために、ホストのトポロジを確認します。

```
virsh nodeinfo
```

2. 使用可能な vCPU の数を確認します。

```
virsh capabilities
```

3. vCPU をプロセッサ コアのセットにピンニングします。

```
virsh vcpupin <vm-name> <vcpu-number> <host-core-number>
```

virsh vcpupin コマンドは、Cisco IOS XRv 9000 ルータの vCPU ごとに実行する必要があります。次の例は、ホストに 8 個のコアがあり Cisco IOS XRv 9000 ルータが 4 個の vCPU で構成されている場合に必要になる KVM コマンドを示しています。

```
virsh vcpupin xrv9000 0 2
virsh vcpupin xrv9000 1 3
virsh vcpupin xrv9000 2 4
virsh vcpupin xrv9000 3 5
```

ホストのコア番号は、0 ~ 7 のどの番号でもかまいません。詳細については、KVM のドキュメンテーションを参照してください。



-
- (注) CPU ピンニングを構成する場合は、ホストサーバーの CPU トポロジを慎重に検討してください。複数のコアで構成された Cisco IOS XRv 9000 ルータを使用している場合は、複数のソケットにまたがる CPU ピンニングを構成しないでください。
-

KVM 構成でのパフォーマンスの向上には、専用のシステム リソースが必要になるという短所もあります。



第 4 章

ゴールデンISOを使用したインストールのカスタマイズ

表 3: 機能の履歴 (表)

機能名	リリース情報	説明
ゴールデン ISO (GISO)	リリース 7.3.1	ゴールデン ISO は、カスタマイズ可能な .iso イメージファイルです。このファイルには、iPXE またはシステムアップグレードを使用してインストールできる、導入準備が完了した最小限のソフトウェアイメージ、構成ファイル、およびパッケージが含まれています。GISO ファイルは、要件に基づいてオプションパッケージと SMU を含めるように作成できます。この機能は、Cisco IOS XRv 9000 ルータでサポートされるようになりました。

ゴールデン ISO (GISO) は、ユーザーがインストール要件に合わせて作成できるカスタマイズ ISO です。ユーザーはインストール可能なイメージをカスタマイズして、基本的な機能コンポーネントを備えた標準的な基本イメージを含め、要件に基づいて RPM、SMU、および設定ファイルを追加することができます。

インストールが簡単であること、およびシステムをシームレスにインストールまたはアップグレードするためにかかる時間は、クラウド規模のネットワークで重要な役割を果たします。時間がかかる複雑なインストールプロセスは、ネットワークの復元力と拡張性に影響します。GISO によってインストールプロセスが簡素化され、インストールワークフローが自動化され、RPM および SMU の依存関係が自動的に管理されます。

GISO は、[github](#) の場所 ([Github](#) の場所) で利用可能なスクリプト `gisobuild.py` を使用して作成します。作成スクリプトと GISO の作成手順の詳細については、[ゴールデン ISO の構築 \(33 ページ\)](#) を参照してください。

システムを GISO を使用して起動すると、GISO 内の追加の SMU および RPM が自動的にインストールされ、ルータは GISO の XR 設定で事前に設定されます。GISO のダウンロードおよびインストールの詳細については、[ゴールデン ISO のインストール \(39 ページ\)](#) を参照してください。

GISO の機能は次の場合に使用できます。

- IOS XR 32 ビットから IOS XR 64 ビットへの移行
- ルータの初期展開
- ソフトウェア ディザスタ リカバリ
- 1 つの基本バージョンから別のバージョンへのシステム アップグレード
- 追加 SMU を使用した同じ基本バージョンからのシステム アップグレード
- 更新プログラムのインストールと依存パッケージの識別および更新
- [制限事項 \(32 ページ\)](#)
- [ゴールデン ISO ワークフロー \(33 ページ\)](#)
- [ゴールデン ISO の構築 \(33 ページ\)](#)
- [ゴールデン ISO のインストール \(39 ページ\)](#)
- [ゴールデン ISO への置換のインストール \(40 ページ\)](#)

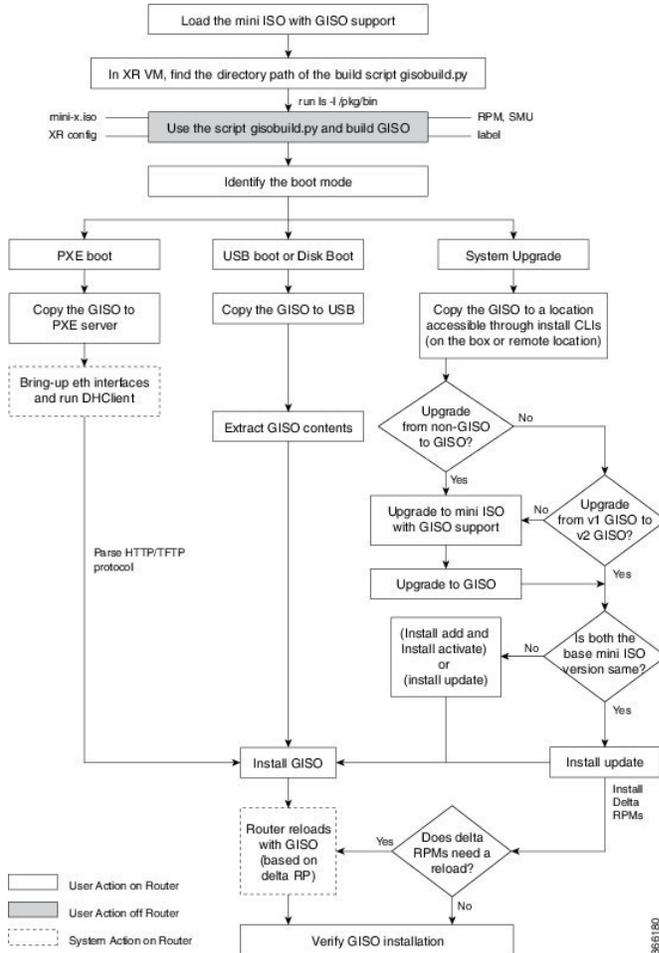
制限事項

次に、カスタム ISO に関する既知の問題および制限事項を示します。

- 非同期パッケージの GISO (ISO とは異なるリリースのパッケージ) の作成と起動はサポートされていません。
- GISO 作成スクリプト `gisobuild.py` は XR 設定の確認をサポートしていません。
- GISO ビルドの名前を変更し、その名前を変更した GISO ビルドのインストールはサポートされていません。
- Cisco IOS XRv 9000 ルータは、カスタム設定や `ZTP.ini` などのオプションファイルを使用した GISO の作成をサポートしていません。
- Cisco IOS XRv 9000 ルータは、XRv9K vRR プロファイルの GISO 作成をサポートしていません。
- 自動 GISO ポストビルドスクリプトは廃止されました。

ゴールデン ISO ワークフロー

次の図は、ゴールデン ISO を構築してインストールするためのワークフローを示しています。



ゴールデン ISO の構築

カスタマイズした ISO は、[Github](#) の場所で利用可能なシスコゴールデン ISO (GISO) 作成スクリプト `gisobuild.py` を使用して構築します。

GISO ツールリンク : [SMU のゴールデン ISO の作成](#)

GISO 作成スクリプトは、自動依存関係管理をサポートし、次の機能を提供します。

- パッケージリポジトリ内に存在するすべてのパッケージの RPM データベースを構築します。
- `xrv9k-mini-x.iso` バージョンと一致しない Cisco RPM をスキップおよび削除します。

- mini-x.iso 内にすでに存在するサードパーティ製の基本パッケージの SMU ではないサードパーティの RPM をスキップおよび削除します。
- 同じリリースで異なるバージョンの基本 RPM が複数ある場合、エラーを表示し、作成プロセスを終了します。
- すべての RPM の互換性チェックと依存関係チェックを実行します。たとえば、子 RPM ncs5500-mpls-te-rsvp は親 RPM ncs5500-mpls に依存します。子 RPM のみが含まれる場合、ゴールデン ISO の作成は失敗します。

GISO を作成するには、スクリプトに次の入力パラメータを指定します。

- 基本 mini-x.iso (必須)
- XR コンフィギュレーションファイル (任意)
- ホスト、XR、およびシステム管理用の 1 つまたは複数のシスコ固有の SMU (必須)
- ホスト、XR、およびシステム管理用の 1 つまたは複数のサードパーティ SMU (必須)
- ゴールデン ISO のラベル (任意)



(注) ゴールデン ISO はミニ ISO からのみ作成できます。full または fullk9 バンドル ISO はサポートされていません。

GISO を作成する場合は、次の命名規則を使用します。

GISO ビルド	書式	例
k9sec RPM を使用しない GISO	<code><platform-name>-golden-x.iso-<version>.<label></code> <code><platform-name>-golden-x-<version>.iso.<label></code>	<プラットフォーム名>-golden-x64.iso-<バージョン>.v1 <プラットフォーム名>-golden-x64-<バージョン>.iso.v1
k9sec RPM を使用した GISO	<code><platform-name>-goldenk9-x.iso-<version>.<label></code> <code><platform-name>-goldenk9-x-<version>.iso.<label></code>	<プラットフォーム名>-goldenk9-x64.iso-<バージョン>.v1 <プラットフォーム名>-goldenk9-x64-<バージョン>.iso.v1



(注) k9sec RPM を GISO に適切に追加するには、**chmod** コマンドを使用してファイルの権限を 644 に変更します。

```
chmod 644 [k9 sec rpm]
```

GISO を作成するには、次の手順を実行します。

始める前に

- 非 GISO から GISO バージョンにアップグレードするには、最初に GISO サポートを使用してミニ ISO にアップグレードする必要があります。NCS 5500 シリーズ ルータの場合は、リリース 6.2.2 以降にアップグレードします。
- GISO が構築されているシステムは、次の要件を満たしている必要があります。
 - システムには Python バージョン 2.7 以降が必要です。
 - システムには、最低 3 ～ 4 GB の空きディスク領域が必要です。
 - システムに Linux ユーティリティ `mount`、`rm`、`cp`、`umount`、`zcat`、`chroot`、`mkisofs` があることを確認します。これらのユーティリティはスクリプトによって使用されます。これらすべての Linux コマンドを実行する権限があることを確認します。
 - システムのカーネルバージョンは、Cisco ISO のカーネルバージョンより後の 3.16 以降である必要があります。
 - Linux カーネルでサポートされている `libyaml rpm` が、ツールで `yaml` を正常に実行できることを確認します。
 - ユーザーは `rpm` リポジトリのセキュリティ `rpm (k9sec-rpm)` に対する適切な権限を持っている必要があります。それ以外の場合は、ゴールデン ISO の作成でセキュリティ `rpm` が無視されます。
- `gisobuild` スクリプトが実行されるシステムには、`root` クレデンシャルを使用する必要があります。

手順

ステップ 1 スクリプト `gisobuild.py` を Github の場所から、GISO を構築するオフラインシステムまたは外部サーバにコピーします。このシステムが上記の「始める前に」セクションに記載された前提条件を満たしていることを確認します。

ステップ 2 スクリプト `gisobuild.py` を実行し、ルータからゴールデン ISO を作成するためのパラメータを指定します。すべての RPM と SMU が同じディレクトリ内に存在することを確認します。ゴールデン ISO の作成に使用できる RPM と SMU の数は 128 です。

(注)

`-i` オプションは必須で、`-r` と `-c` のいずれかまたはその両方を指定する必要があります。

```
[directory-path]$ gisobuild.py [-h] [-i <mini-x.iso>] [-r <rpm repository>]
[-c <config-file>] [-l <giso label>] [-m] [-v]
```

ゴールデン ISO ラベルが指定されていないため、デフォルトは 0 です。次に、スクリプトの出力例を示します。

(注)

NCS5500 ルータには、x86_64 とアームの 2 種類のカードがあります。システム管理は両方のタイプのカードで実行されますが、XR は x86_64 カードでのみ実行されます。

```
[directory-path]$ gisobuild.py -i ncs5500-mini-x.iso -r . -c config-file -l vl

System requirements check [PASS]
Platform: ncs5500 Version: 6.2.1.14I
XR-Config file (<directory>/config-file) will be encapsulated in GISO.
Scanning repository [<directory>/ncs5500-giso]...
Building RPM Database...
Total 54 RPM(s) present in the repository path provided in CLI

Skipping following older version of host os rpm(s) from repository:
ncs5500-sysadmin-hostos-6.2.1.16-r62114I.CSCho88888.admin.x86_64.rpm
ncs5500-sysadmin-hostos-6.2.1.16-r62114I.CSCho88888.host.x86_64.rpm
Skipping following older version of spirit-boot rpm(s) from repository:
ncs5500-spirit-boot-1.0.0.2-r62114I.CSCsb88888.x86_64.rpm
ncs5500-spirit-boot-1.0.0.1-r62114I.CSCsb77777.x86_64.rpm

Following XR x86_64 rpm(s) will be used for building GISO:
ncs5500-iosxr-infra-4.0.0.2-r62114I.CSCxr11111.x86_64.rpm
ncs5500-m2m-1.0.0.0-r62114I.x86_64.rpm
ncs5500-iosxr-infra-4.0.0.4-r62114I.CSCxr11111.x86_64.rpm
ncs5500-iosxr-infra-4.0.0.3-r62114I.CSCxr44444.x86_64.rpm
ncs5500-k9sec-3.1.0.0-r62114I.x86_64.rpm
ncs5500-iosxr-fwding-4.0.0.2-r62114I.CSCxr22222.x86_64.rpm
ncs5500-spirit-boot-1.0.0.3-r62114I.CSCsb77777.x86_64.rpm
ncs5500-k9sec-3.1.0.1-r62114I.CSCxr33333.x86_64.rpm
ncs5500-iosxr-fwding-4.0.0.1-r62114I.CSCxr22222.x86_64.rpm
ncs5500-mgbl-3.0.0.0-r62114I.x86_64.rpm
ncs5500-parser-2.0.0.0-r62114I.x86_64.rpm
ncs5500-iosxr-mgbl-3.0.0.0-r62114I.x86_64.rpm
ncs5500-k9sec-3.1.0.2-r62114I.CSCxr33333.x86_64.rpm
ncs5500-iosxr-infra-4.0.0.1-r62114I.CSCxr44444.x86_64.rpm
kernel-image-3.14.23-wr7.0.0.2-standard-3.14-r0.1.xr.x86_64.rpm
openssh-scp-6.6p1-r0.0.xr.x86_64.rpm
kernel-3.14.23-wr7.0.0.2-standard-3.14-r0.1.xr.x86_64.rpm
openssh-scp-6.6p1.pl-r0.0.CSctp11111.xr.x86_64.rpm
cisco-klm-mifpga-0.1.pl-r0.0.CSctp11111.xr.x86_64.rpm
kernel-image-3.14.23-wr7.0.0.2-standard-3.14.pl-r0.1.CSctp11111.xr.x86_64.rpm
cisco-klm-mifpga-0.1-r0.0.xr.x86_64.rpm
kernel-3.14.23-wr7.0.0.2-standard-3.14.pl-r0.1.CSctp11111.xr.x86_64.rpm
kernel-modules-3.14-r0.1.xr.x86_64.rpm
kernel-modules-3.14.pl-r0.1.CSctp11111.xr.x86_64.rpm

Skipping following rpms from repository since they are already present in base ISO:
ncs5500-mgbl-3.0.0.0-r62114I.x86_64.rpm
ncs5500-m2m-1.0.0.0-r62114I.x86_64.rpm
ncs5500-parser-2.0.0.0-r62114I.x86_64.rpm

Following XR rpm(s) will be used for building GISO:
kernel-modules-3.14-r0.1.xr.x86_64.rpm
kernel-3.14.23-wr7.0.0.2-standard-3.14.pl-r0.1.CSctp11111.xr.x86_64.rpm
ncs5500-iosxr-fwding-4.0.0.2-r62114I.CSCxr22222.x86_64.rpm
ncs5500-iosxr-infra-4.0.0.3-r62114I.CSCxr44444.x86_64.rpm
ncs5500-iosxr-infra-4.0.0.2-r62114I.CSCxr11111.x86_64.rpm
kernel-image-3.14.23-wr7.0.0.2-standard-3.14-r0.1.xr.x86_64.rpm
cisco-klm-mifpga-0.1-r0.0.xr.x86_64.rpm
cisco-klm-mifpga-0.1.pl-r0.0.CSctp11111.xr.x86_64.rpm
ncs5500-iosxr-infra-4.0.0.4-r62114I.CSCxr11111.x86_64.rpm
ncs5500-iosxr-fwding-4.0.0.1-r62114I.CSCxr22222.x86_64.rpm
openssh-scp-6.6p1-r0.0.xr.x86_64.rpm
```

```
ncs5500-k9sec-3.1.0.2-r62114I.CSCXr33333.x86_64.rpm
ncs5500-k9sec-3.1.0.0-r62114I.x86_64.rpm
kernel-modules-3.14.pl-r0.1.CSCTp11111.xr.x86_64.rpm
ncs5500-k9sec-3.1.0.1-r62114I.CSCXr33333.x86_64.rpm
openssh-scp-6.6p1.pl-r0.0.CSCTp11111.xr.x86_64.rpm
ncs5500-spirit-boot-1.0.0.3-r62114I.CSCsb77777.x86_64.rpm
ncs5500-iosxr-mgbl-3.0.0.0-r62114I.x86_64.rpm
kernel-image-3.14.23-wr7.0.0.2-standard-3.14.pl-r0.1.CSCTp11111.xr.x86_64.rpm
ncs5500-iosxr-infra-4.0.0.1-r62114I.CSCXr44444.x86_64.rpm
kernel-3.14.23-wr7.0.0.2-standard-3.14-r0.1.xr.x86_64.rpm
```

...RPM compatibility check [PASS]

Following CALVADOS x86_64 rpm(s) will be used for building GISO:

```
ncs5500-sysadmin-shared-6.2.1.18-r62114I.CSCcv22222.x86_64.rpm
ncs5500-sysadmin-hostos-6.2.1.17-r62114I.CSCho77777.admin.x86_64.rpm
ncs5500-sysadmin-system-6.2.1.18-r62114I.CSCcv11111.x86_64.rpm
ncs5500-sysadmin-shared-6.2.1.17-r62114I.CSCcv33333.x86_64.rpm
ncs5500-sysadmin-system-6.2.1.17-r62114I.CSCcv44444.x86_64.rpm
kernel-modules-3.14.pl-r0.1.CSCTp11111.admin.x86_64.rpm
cisco-klm-mifpga-0.1-r0.0.admin.x86_64.rpm
kernel-image-3.14.23-wr7.0.0.2-standard-3.14-r0.1.admin.x86_64.rpm
kernel-modules-3.14-r0.1.admin.x86_64.rpm
kernel-image-3.14.23-wr7.0.0.2-standard-3.14.pl-r0.1.CSCTp11111.admin.x86_64.rpm
kernel-3.14.23-wr7.0.0.2-standard-3.14-r0.1.admin.x86_64.rpm
kernel-3.14.23-wr7.0.0.2-standard-3.14.pl-r0.1.CSCTp11111.admin.x86_64.rpm
cisco-klm-mifpga-0.1.pl-r0.0.CSCTp11111.admin.x86_64.rpm
openssh-scp-6.6p1.pl-r0.0.CSCTp11111.admin.x86_64.rpm
openssh-scp-6.6p1-r0.0.admin.x86_64.rpm
...RPM compatibility check [PASS]
```

Following HOST x86_64 rpm(s) will be used for building GISO:

```
ncs5500-sysadmin-hostos-6.2.1.17-r62114I.CSCho77777.host.x86_64.rpm
cisco-klm-mifpga-0.1.pl-r0.0.CSCTp11111.host.x86_64.rpm
kernel-image-3.14.23-wr7.0.0.2-standard-3.14-r0.1.host.x86_64.rpm
kernel-modules-3.14.pl-r0.1.CSCTp11111.host.x86_64.rpm
kernel-modules-3.14-r0.1.host.x86_64.rpm
openssh-scp-6.6p1.pl-r0.0.CSCTp11111.host.x86_64.rpm
cisco-klm-mifpga-0.1-r0.0.host.x86_64.rpm
kernel-3.14.23-wr7.0.0.2-standard-3.14.pl-r0.1.CSCTp11111.host.x86_64.rpm
kernel-3.14.23-wr7.0.0.2-standard-3.14-r0.1.host.x86_64.rpm
openssh-scp-6.6p1-r0.0.host.x86_64.rpm
kernel-image-3.14.23-wr7.0.0.2-standard-3.14.pl-r0.1.CSCTp11111.host.x86_64.rpm
...RPM compatibility check [PASS]
```

Building Golden ISO...

Summary

XR rpms:

```
kernel-modules-3.14-r0.1.xr.x86_64.rpm
kernel-3.14.23-wr7.0.0.2-standard-3.14.pl-r0.1.CSCTp11111.xr.x86_64.rpm
ncs5500-iosxr-fwding-4.0.0.2-r62114I.CSCXr22222.x86_64.rpm
ncs5500-iosxr-infra-4.0.0.3-r62114I.CSCXr44444.x86_64.rpm
ncs5500-iosxr-infra-4.0.0.2-r62114I.CSCXr11111.x86_64.rpm
kernel-image-3.14.23-wr7.0.0.2-standard-3.14-r0.1.xr.x86_64.rpm
cisco-klm-mifpga-0.1-r0.0.xr.x86_64.rpm
cisco-klm-mifpga-0.1.pl-r0.0.CSCTp11111.xr.x86_64.rpm
ncs5500-iosxr-infra-4.0.0.4-r62114I.CSCXr11111.x86_64.rpm
ncs5500-iosxr-fwding-4.0.0.1-r62114I.CSCXr22222.x86_64.rpm
openssh-scp-6.6p1-r0.0.xr.x86_64.rpm
ncs5500-k9sec-3.1.0.2-r62114I.CSCXr33333.x86_64.rpm
ncs5500-k9sec-3.1.0.0-r62114I.x86_64.rpm
kernel-modules-3.14.pl-r0.1.CSCTp11111.xr.x86_64.rpm
ncs5500-k9sec-3.1.0.1-r62114I.CSCXr33333.x86_64.rpm
```

```
openssh-scp-6.6p1.p1-r0.0.CSCTp11111.xr.x86_64.rpm
ncs5500-spirit-boot-1.0.0.3-r62114I.CSCsb77777.x86_64.rpm
ncs5500-iosxr-mgbl-3.0.0.0-r62114I.x86_64.rpm
kernel-image-3.14.23-wr7.0.0.2-standard-3.14.p1-r0.1.CSCTp11111.xr.x86_64.rpm
ncs5500-iosxr-infra-4.0.0.1-r62114I.CSCxr44444.x86_64.rpm
kernel-3.14.23-wr7.0.0.2-standard-3.14-r0.1.xr.x86_64.rpm
```

CALVADOS rpms:

```
ncs5500-sysadmin-shared-6.2.1.18-r62114I.CSCcv22222.x86_64.rpm
ncs5500-sysadmin-hostos-6.2.1.17-r62114I.CSCho77777.admin.x86_64.rpm
ncs5500-sysadmin-system-6.2.1.18-r62114I.CSCcv11111.x86_64.rpm
ncs5500-sysadmin-shared-6.2.1.17-r62114I.CSCcv33333.x86_64.rpm
ncs5500-sysadmin-system-6.2.1.17-r62114I.CSCcv44444.x86_64.rpm
kernel-modules-3.14.p1-r0.1.CSCTp11111.admin.x86_64.rpm
cisco-klm-mifpga-0.1-r0.0.admin.x86_64.rpm
kernel-image-3.14.23-wr7.0.0.2-standard-3.14-r0.1.admin.x86_64.rpm
kernel-modules-3.14-r0.1.admin.x86_64.rpm
kernel-image-3.14.23-wr7.0.0.2-standard-3.14.p1-r0.1.CSCTp11111.admin.x86_64.rpm
kernel-3.14.23-wr7.0.0.2-standard-3.14-r0.1.admin.x86_64.rpm
kernel-3.14.23-wr7.0.0.2-standard-3.14.p1-r0.1.CSCTp11111.admin.x86_64.rpm
cisco-klm-mifpga-0.1.p1-r0.0.CSCTp11111.admin.x86_64.rpm
openssh-scp-6.6p1.p1-r0.0.CSCTp11111.admin.x86_64.rpm
openssh-scp-6.6p1-r0.0.admin.x86_64.rpm
```

HOST rpms:

```
ncs5500-sysadmin-hostos-6.2.1.17-r62114I.CSCho77777.host.x86_64.rpm
cisco-klm-mifpga-0.1.p1-r0.0.CSCTp11111.host.x86_64.rpm
kernel-image-3.14.23-wr7.0.0.2-standard-3.14-r0.1.host.x86_64.rpm
kernel-modules-3.14.p1-r0.1.CSCTp11111.host.x86_64.rpm
kernel-modules-3.14-r0.1.host.x86_64.rpm
openssh-scp-6.6p1.p1-r0.0.CSCTp11111.host.x86_64.rpm
cisco-klm-mifpga-0.1-r0.0.host.x86_64.rpm
kernel-3.14.23-wr7.0.0.2-standard-3.14.p1-r0.1.CSCTp11111.host.x86_64.rpm
kernel-3.14.23-wr7.0.0.2-standard-3.14-r0.1.host.x86_64.rpm
openssh-scp-6.6p1-r0.0.host.x86_64.rpm
kernel-image-3.14.23-wr7.0.0.2-standard-3.14.p1-r0.1.CSCTp11111.host.x86_64.rpm
```

XR Config file:

router.cfg

```
...Golden ISO creation SUCCESS.
```

```
Golden ISO Image Location: /<directory>/ncs5500-goldenk9-x.iso-6.2.1.14I.v1
```

```
Detail logs: <directory>/Giso_build.log-2016-10-01:16:22:48.305211
```

値は次のとおりです。

- -i は mini-x.iso へのパスです
- -r は RPM リポジトリへのパスです
- -c は XR config ファイルへのパスです
- -l はゴールデン ISO ラベルです
- -h はヘルプ メッセージを表示します
- -v は、作成ツール gisobuild.py のバージョンです
- -m は、IOS XR から IOS XR 64 ビットに移行するための移行 tar を構築します

GISO は、指定されたディレクトリ内の各フォルダに配置された RPM を使用して作成され、ログファイル `giso_summary.txt` および `gisobuild.log-<タイムスタンプ>` も含まれています。XR コンフィギュレーション ファイルはディレクトリ内に `router.cfg` として格納されます。



(注) GISO スクリプトは XR 設定の検証をサポートしていません。

次のタスク

ゴールデン ISO をルータにインストールします。

ゴールデン ISO のインストール

ゴールデン ISO (GISO) は、次のアクションを自動的に実行します。

- ホストおよびシステム管理 RPM をインストールします。
- RP でリポジトリと TFTP ブートをパーティションに分割します。
- システム管理モードおよび XR モードでソフトウェア プロファイルを作成します。
- XR RPM をインストールします。 `show instal active` コマンドを使用して RPM のリストを表示します。
- XR 設定を適用します。XR モードで `show running-config` コマンドを使用して確認します。

手順

ステップ 1 次のいずれかのオプションを使用して、ルータに GISO イメージをダウンロードします。

- **PXE ブート** : ルータが起動すると、ブート モードが識別されます。PXE をブート モードとして検出すると、利用可能なすべてのイーサネットインターフェイスが起動し、各インターフェイスで DHCPClient が実行されます。DHClient スクリプトは HTTP または TFTP プロトコルを解析し、GISO がボックスにダウンロードされます。
- **USB ブートまたはディスク ブート** : ブート中に USB モードが検出され、GISO が識別されると、追加の RPM および XR 設定ファイルが抽出されてインストールされます。
- システムのアップグレード時の **システム アップグレード** では、`install add`、`install activate`、または `install replace` コマンドを使用して GISO をインストールできます。

次に、システムをアップグレードするオプションを示します。

- **非 GISO (GISO をサポートしていないイメージ) から GISO イメージへのシステムアップグレード** : システムが GISO をサポートしていないイメージを使用してバージョン 1 を実行している場合、システムは GISO をサポートするイメージのバージョン 2 に直接アップグレードすることは

できません。その代わりに、バージョン 1 をバージョン 2 ミニ ISO にアップグレードし、次にバージョン 2 GISO にアップグレードする必要があります。

- **バージョン 1 GISO からバージョン 2 GISO へのリリースでのシステム アップグレード**：両方の GISO イメージの基本バージョンは同じでラベルが異なる場合、**install add** および **install activate** コマンドは同じバージョンの 2 つのイメージをサポートしません。その代わりに、**install update** コマンドを使用してデルタ RPM のみをインストールします。システムのリロードはデルタ RPM の再起動タイプに基づいています。
- **バージョン 1 GISO からバージョン 2 GISO へのリリース間でのシステム アップグレード**：両方の GISO イメージの基本バージョンが異なります。**install add** および **install activate** コマンド、または **install replace** コマンドを使用して、システムアップグレードを実行します。ルータは、バージョン 2 GISO イメージを使用したアップグレード後にリロードされます。

ステップ 2 システム管理モードで **show install repository all** コマンドを実行し、ホスト、システム管理、および XR の RPM と基本 ISO を表示します。

ステップ 3 **show install package <golden-iso>** コマンドを実行し、RPM のリストおよび GISO に組み込まれているパッケージを表示します。

GISO 内の ISO、SMU、およびパッケージがルータにインストールされます。

ゴールデン ISO への置換のインストール

ゴールデン ISO (GISO) は、単一の操作でソフトウェア メンテナンス アップデート (SMU) の事前定義されたリストを持つバージョンにルータをアップグレードします。ただし、異なる SMU セットを使用した同じバージョンに更新するには、2 段階のプロセスが必要です。

この 2 段階のプロセスを回避するには、**install replace** コマンドを使用して、現在アクティブなバージョンを、新しく追加した GISO のイメージと SMU を含む完全なパッケージに置き換えます。

このプロセスでは、GISO をアップグレードしてデルタ SMU を追加し、使用されていない SMU を手動で非アクティブにする必要があります。さらにこれは、オプションの実行中の一連の RPM のサブセットであるさまざまなオプション RPM を含んでいる GISO にアップグレードする唯一の方法です。たとえば、GISO の V1 は、V1 mini とオプション RPM の V1 mpls、V1 mpls-te、V1 mgbl、および V1 k9sec を含む実行中のバージョンです。GISO の V2 に V2 k9sec が含まれていない場合は、**install replace** を使用して V2 のオプション RPM にアップグレードします。

手順

ステップ 1 **install replace <GISO-location> [commit|noprompt]**

例：

```

Router#install replace harddisk:/<giso-image>.iso
+++++
Install operation 11 started by root:
exec-timeout is suspended.
No install operation in progress at this moment
Label = More_Pkgs
ISO <giso-iso-image>.iso in input package list. Going to upgrade the system to

version <new-giso-image>.
System is in committed state
Current full-label: <giso-image>_R_Commit
Current only-label: R_Commit
Current label: R_Commit
Updating contents of golden ISO
Scheme : localdisk
Hostname : localhost
Username : None
SourceDir : /ws
Collecting software state..
Getting platform
Getting supported architecture
Getting active packages from XR
Getting inactive packages from XR
Getting list of RPMs in local repo
Getting list of provides of all active packages
Getting provides of each rpm in repo
Getting requires of each rpm in repo
Fetching .... <giso-image>.iso
Label within GISO: More_Pkgs
Skipping <platform>-mgb1-3.0.0.0-<release>.x86_64.rpm from GISO as it's active
Adding packages
    <platform>-golden-x-<release>-<Label>.iso
RP/0/RP0/CPU0:Jun 20 14:43:59.349 UTC: sdr_instmgr[1164]: %INSTALL-INSTMGR-2-OPERATION_SUCCESS :

Install operation 12 finished successfully
Install add operation successful
Activating <platform>-golden-x-<release>-<Label>
Jun 20 14:44:05 Install operation 13 started by root:
    install activate pkg <platform>-golden-x-<release>-<Label> replace noprompt
Jun 20 14:44:05 Package list:
Jun 20 14:44:05     <platform>-golden-x-<release>-<Label>.iso
Jun 20 14:44:29 Install operation will continue in the background
exec-timeout is resumed.
Router# Install operation 13 finished successfully
Router: sdr_instmgr[1164]: %INSTALL-INSTMGR-2-OPERATION_SUCCESS :

Install operation 13 finished successfully

```

新しく追加された GISO のバージョンおよびラベルは、現在アクティブなバージョンのバージョンおよびラベルと比較されます。不一致が特定されると、新しいパーティションが作成され、完全なパッケージがインストールされます。インストール後、システムは新しく追加された GISO からイメージおよびパッケージをリロードします。

ステップ 2 show version

例 :

```

Router#show version
Wed Jun 20 15:06:37.915 UTC
Cisco IOS XR Software, Version <new-giso-image>
Copyright (c) 2013-2018 by Cisco Systems, Inc.

```

```
Build Information:
Built By      : <user>
Built On     : <date>
Build Host   : <host-name>
Workspace    : <workspace-name>
Version      : <version>
Location     : <path>
Label       : <label-name>

cisco <platform> () processor
System uptime is 3 hours 51 minutes
```

システムは新しく追加された GISO からイメージおよびパッケージをリロードします。



第 5 章

Cisco IOS XRv 9000 ルータ Smart Licensing

Cisco IOS XR リリース 5.4 以降、Cisco IOS XRv 9000 ルータでは、Cisco Smart Licensing を使用したアクティベーションがサポートされています。Cisco Smart Licensing があると、Cisco IOS XRv 9000 インスタンスにノードロック ライセンスをインストールしなければならないという要件はなくなります。

- [Cisco Smart Licensing](#) (43 ページ)
- [Cisco IOS XRv 9000 ルータのライセンシング モデル](#) (44 ページ)
- [Cisco Smart Licence の設定](#) (46 ページ)
- [Cisco IOS XRv 9000 ルータをシスコ ライセンシング クラウドに登録する](#) (47 ページ)
- [スマート ライセンスの管理](#) (47 ページ)
- [仮想固有デバイス識別子 \(vUDI\)](#) (48 ページ)
- [Cisco Smart License に関する問題のトラブルシューティング](#) (49 ページ)

Cisco Smart Licensing

Cisco Smart Licensing は、時間のかかる手動のライセンス タスクを自動化できるクラウドベースのソフトウェアライセンス管理ソリューションです。このソリューションを使用すると、ライセンスのステータスとソフトウェアの使用傾向を簡単に追跡できます。

Cisco Smart Licensing によって、次の 3 つのコア機能が簡素化されます。

- **購入**：ネットワークにインストールされているソフトウェアを製品アクティベーション キー (PAK) を指定せずに自動的に登録できます。
- **管理**：ライセンスエンタイトルメントの有効化を自動的に追跡できます。また、すべてのノードにライセンスファイルをインストールする必要はありません。組織構造に合わせたライセンスプール (ライセンスの論理的なグループ) を作成できます。Smart Licensing には、すべての Cisco ソフトウェア ライセンスを 1 つの一元化された Web サイトで管理できる集中型ポータルである Cisco Smart Software Manager が用意されています。
- **レポート**：Cisco Smart Licensing では、ポータルを使用することで、購入したライセンスとネットワークに実際に展開された製品を統合して表示できます。このデータを使用すると、購入の意思決定を実際の使用状況に基づいてより適切に行うことができます。

Cisco IOS XRv 9000 ルータは、Cisco Smart Licensing を使用した有効化のみをサポートします。Cisco Smart Licensing があると、Cisco IOS XRv 9000 ルータのインスタンスにノードロック ライセンスをインストールしなければならないという要件はなくなります。代わりに、Cisco IOS XRv 9000 ルータが Cisco Licensing Cloud と（直接、プロキシ経由で、または Smart Licensing Satellite 経由で）通信して、使用される機能とシステムの使用規模に関するレポートを提供します。

Cisco Smart Licensing は、Cisco Smart Call Home 機能を使用して Cisco Smart Software Manager と通信します。Smart Call Home は、Smart Licensing のデフォルト設定に合わせて自動設定されます。Cisco Smart Call Home およびデフォルト以外の設定の詳細については、『Cisco ASR 9000 Series Aggregation Services Router System Management Configuration Guide』の「Configuring Call Home on the Cisco ASR 9000 Series Router」の章を参照してください。

Cisco Smart Licensing は、Cisco Smart Software Manager を使用してライセンスを管理します。Cisco Smart Software Manager にアクセスするには、[ここ](#)をクリックしてください。

手動でスマート ライセンスを更新する方法および Cisco Smart Licensing からデバイスの登録を解除する方法については、[スマート ライセンスの管理 \(47 ページ\)](#) を参照してください。

Cisco Smart Software Manager の詳細については、Smart Software Manager ツールからアクセスできる『Cisco Smart Software Manager User Guide』を参照してください。

Cisco IOS XRv 9000 ルータのライセンスングモデル

Cisco IOS XRv 9000 ルータのライセンスには、デモおよび実稼働モードが含まれています。

表 4: Cisco IOS XRv ルータのライセンス モード

モード	説明
デモ	<ul style="list-style-type: none"> これは、ルータが起動するときのデフォルトのモードです。 クラウドへの接続は必要ありません。 機能レベルは強制されません。 すべてのインターフェイスのスループットに対する 200 Kbps のレート制限。
実稼働	<ul style="list-style-type: none"> このモードには登録が必要です。 強制措置がとられることはありません。

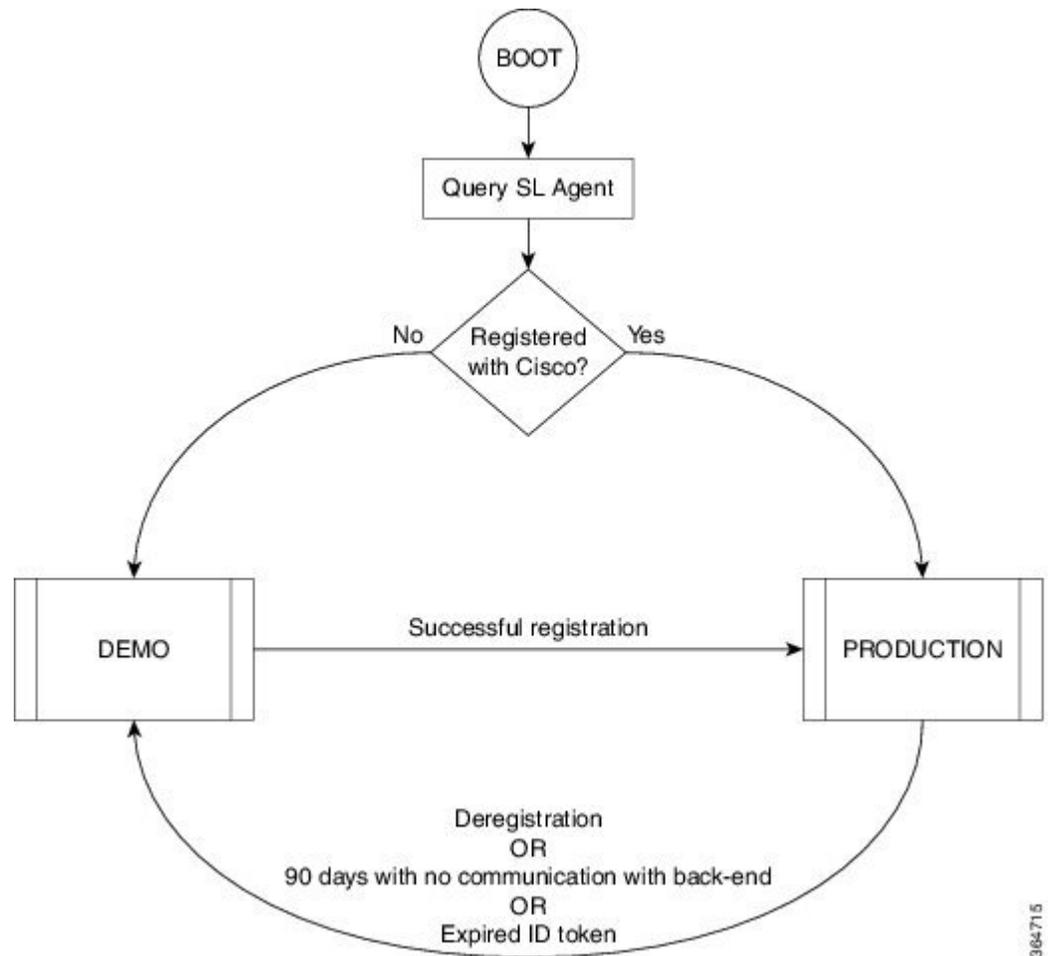
専用のハードウェアを持つその他のシスコ製品（ASR9K、NCS4K、NCS6K）とは異なり、仮想という性質ゆえに、起動した Cisco IOS XRv 9000 ルータは評価モードにはなりません（これを可能にすると、数カ月ごとにソフトウェアを再インストールするだけで、ライセンスを必要とせずに無制限に機能を使用することができるようになるため）。

このためデモモードでは、Cisco IOS XRv 9000 ルータは、システムを導入シナリオでは使用できないようにしながらも、デモ目的としては十分に機能する程度の、スループットスケール制限を課します。



(注) ルータが未登録状態の場合、ライセンスは EVAL (評価) モードになります。評価期間は 90 日間です。

図 3: ライセンス モード



この図は、デモおよび実稼働モードをサポートする、基本的な状態遷移を示しています。

- システムがシスコバックエンドに現在登録されていることを Smart Licensing エージェントの状態が示しているかどうかによって、起動したシステムはデモまたは実稼働モードのいずれかになります。
- デモモードの場合、システムではスループットに 200 kbps のレート制限が課されます。
- 実稼働モードでは、コンプライアンスに従っているかどうかに関係なく、ライセンスの観点から制限が課されたり強制措置がとられることはありません。(これは Smart Licensing

の原理に従っています。各ルータは使用状況をバックエンドにレポートするだけで、コンプライアンスに従っていないアカウントがある場合は、バックエンドによって顧客にライセンスの購入または更新を促すための処理が行われます)。

Cisco IOS XRv 9000 ルータのライセンス PID の詳細については、最新の『Cisco IOS XRv 9000 Router Release Notes』の「License Ordering Information」セクションを参照してください。

評価期間の管理

デモ モードで課される制限のため、Cisco IOS XRv 9000 を制限のない状態で評価したい顧客をサポートする別の手段が必要になります。ここでは、この別の手段の手順について説明します。

1. 顧客がライセンスを細かく分割するための手段として、Smart Licensing のバックエンドではバーチャルアカウントをサポートします。通常、これは主に大規模な顧客に適用されますが、どこでも利用できます。
2. 顧客は、評価目的のみの用途で、ライセンス ポータル内に「バーチャル アカウント」を作成します。
3. シスコの担当者が、バーチャルアカウントに、顧客が関心がある各機能の期間限定のエンタイトルメントを付与します。
4. 顧客は、実稼働の場合と同じ手順でオンルータの登録に進みます（この場合は、使用されるライセンスがバーチャルアカウントからのものであることを除いて）。

オンルータの操作は同一であるため、この手続きにより、顧客が評価フェーズで登録手続きの練習（または自動化、あるいは両方）を行うことができるという、付加的なメリットがあります。

Cisco Smart Licence の設定

Cisco IOS XRv 9000 ルータは、Smart Licensing の実行に十分なデフォルトの Smart Call Home が設定されています。Smart Call Home のデフォルト設定は、組み込み CiscoTAC-1 プロファイルを使用して、Smart Licensing によって内部的にトリガーされます。このデフォルト設定には、トラフィック ポートまたは管理ポートを介して cisco.com にルータが接続されている必要があります。

Cisco クラウドへの IP 接続を確保する以外は、ユーザの介入は必要ありません。

Smart Call Home Gateway ゲートウェイを介した別の Smart Licensing 設定が必要な場合は、『Cisco ASR 9000 Series Aggregation Services Router System Management Configuration Guide』の「Configuring Call Home on the Cisco ASR 9000 Series Router」の章を参照してください。



- (注) シスコに登録された VM を削除する前に、登録を解除して、エンタイトルメントとしてカウントされないようにする必要があります。

Cisco Smart Licensing からのデバイスの登録解除

ルータ インスタンスの Cisco Smart Licensing の登録を削除するには、EXEC モードで **license smart deregister** コマンドを実行します。シスコ スマート ライセンシングのすべての証明書と権限が削除されます。

仮想固有デバイス識別子 (vUDI)

すべてのシスコ プラットフォームは、製品 ID、バージョン、およびシリアル番号で構成される固有デバイス識別子 (UDI) を持っています。物理プラットフォームの場合は、デバイスの製造時にこれらがシャーシに焼き付けられます。仮想プラットフォームの場合は、システムによってシリアル番号が生成され、それを製品 ID およびバージョンと組み合わせて仮想 UDI が作成されます。

ルータの UDI 情報を表示するには、EXEC モードで **show license udi** コマンドを実行します。

```
RP/0/RP0/CPU0:ios#show license udi
Tue Aug 25 09:47:09.780 UTC

Product Information
=====
UDI:
PID:R-IOSXRv9000-IMG,SN:DF855094AA4,SUVI:R-IOSXRv9000-IMGDF855094AA4,UUID:1BB98DDC-3EE1-4A60-95A6-530870AC19D9
```

Cisco IOS XRv 9000 ルータは仮想化されており、仮想ディスク イメージで表されているため、コピーまたは複製することができます。したがって、同じ vUDI を持つインスタンスが複数存在する可能性があります。vUDI はライセンスの一部としてインスタンスを識別するために使用されるため、競合が発生しないように注意する必要があります。ただし、仮想化環境ではハイパーバイザから提供される情報がないとインスタンス自体が機能しないため、ブートアップロジックに加えて **virtual-platform udi reset** コマンドがフェールセーフ機能として用意されています。



- (注) **virtual-platform udi reset** コマンドを実行すると、VM がリロードされます。

ブートアップ時の UDI の動作

Cisco IOS XRv 9000 VM インスタンスが初めて起動したときに、Cisco IOS XRv 9000 ルータによって一意のシリアル番号 (UDI) が作成されます。シリアル番号が作成されなかった場合は、乱数ジェネレータによって VM インスタンスのシリアル番号が生成され、製品 ID および

バージョンと組み合わせて vUDI が作成されます。これは安全なストレージに格納され、ユーザが手動で変更することはできません。

VM をコピーまたは複製すると、その VM に関連付けられている vUDI が重複します。vUDI の重複を抑制するため、ハイパーバイザは起動のたびに VM インスタンスに対して仮想ディスクに格納されている汎用一意識別子 (UUID) を提供します。仮想ディスク イメージからルータが初めて起動するときに、ルータはハイパーバイザによって割り当てられた UUID をその仮想ディスクの内部に格納します。その後の起動では、ハイパーバイザによって割り当てられた新しい UUID が最後の起動時に使用されたものと比較されます。同じハイパーバイザが同じ仮想ディスク イメージを起動する場合、UUID は同じままである必要があります。ただし、ディスク イメージが別のハイパーバイザにコピーされて実行された場合、UUID は異なるものになります。そのため、システムによって新しいシリアル番号が生成され、複製された VM の新しい vUDI が作成されます。

次の場合、ユーザは同じ UDI を検出します。

- 乱数ジェネレータが同じシリアル番号を 2 つ作成した
- ハイパーバイザから UUID が提供されなかった

このような場合、ユーザは管理モードで **virtual-platform udi reset** コマンドを実行して新しい vUDI を生成できます。

virtual-platform udi reset コマンドを実行すると、確認のプロンプトが表示され、要求を確認する一連のシステム メッセージが表示されます。これによって、Smart Licensing が Cisco Smart Software Manager で登録を更新し、新しい vUDI が Cisco IOS XRv 9000 VM インスタンスに割り当てられます。

Cisco Smart License に関する問題のトラブルシューティング

Cisco Smart License に関する問題をトラブルシューティングする場合、以下のコマンドをすべての Smart Licensing プラットフォームで使用できます。これらのコマンドを使用して、使用されているエンタイトルメントやデバイスのコンプライアンスなどを確認できます。

- show license all
- show license status
- show license summary
- show license tech support
- show license udi
- show license usage

以下は、シスコ サポートが使用する Cisco IOS XRv 9000 ルータ プラットフォーム専用のコマンドです。

- show license platform detail
- show license platform summary
- show license platform trace

ライセンス認証ステータス

ライセンス認証ステータスとして、主に次の 4 つの状態が用意されています。

ステータス	説明
登録済み	デバイス登録が完了し、ID 証明書を受信しています。ID 証明書は、シスコのライセンス機関との今後の通信に使用されます。
承認済み	有効なスマートアカウントを使用して登録が完了しました。ライセンスの利用が開始されました。これは、コンプライアンスに従っている状態を示しています。
コンプライアンス違反	利用数がスマートアカウントの使用可能なライセンスを超えています。
承認が期限切れ	デバイスは、Cisco Smart Software Manager (CSSM) と一定期間にわたって通信できていません。通常、90 日後にこの状態になります。デバイスは、登録期間が終了するまで、承認を更新するために 1 時間ごとに CSSM への接続を試行します。



第 6 章

コンソールポートへのアクセス

この章では、Cisco IOS XRv 9000 ルータのコンソールにアクセスする方法を説明します。

- [コンソール マッピング \(51 ページ\)](#)
- [VM コンソール経由の Cisco IOS XRv 9000 ルータへのアクセス \(52 ページ\)](#)
- [仮想シリアルポート経由の Cisco IOS XRv 9000 ルータへのアクセス \(53 ページ\)](#)
- [システム管理コンソールから XR コンソールへのアクセス \(56 ページ\)](#)
- [ユーザープロファイルの作成および権限の割り当て \(57 ページ\)](#)

コンソール マッピング

Cisco IOS XRv 9000 ルータは、VM の電源がオンになったときに起動されます。どのインストール イメージを使用したかに応じて (VGA の有無に応じて)、VM コンソールまたは仮想シリアルポートのコンソールでインストール プロセスをモニタできます。

次の表に、VGA 有りのインストール イメージタイプが使用された場合のコンソール マッピングを示します。

表 5: VGA 有りのイメージタイプのコンソール マッピング

VM デバイス	マッピング
VM コンソール	XR コンソール
最初のシリアルポート	XR 補助
2 番目のシリアルポート	Admin コンソール
3 番目のシリアルポート	Admin 補助
4 番目のシリアルポート	未使用

次の表に、VGA 無しのインストール イメージタイプが使用された場合のコンソール マッピングを示します。

表 6: VGA 無しのイメージタイプのコンソール マッピング

VM デバイス	マッピング
VM コンソール	未使用
最初のシリアルポート	XR コンソール
2 番目のシリアルポート	XR 補助
3 番目のシリアルポート	Admin コンソール
4 番目のシリアルポート	Admin 補助

VM コンソール経由の Cisco IOS XRv 9000 ルータへのアクセス

デフォルトでは、XR コンソールは最初のシリアルポートにマッピングされます。ただし、次の状況では、XR コンソールは VM コンソールにマッピングされます。

- ESXi ハイパーバイザまたは OpenStack プラットフォームで VM を実行していて VGA イメージタイプを使用している場合
- Cisco IOS XRv 9000 ルータの展開に VGA イメージを使用している場合

VGA コンソールには、vSphere Client の [Console] タブでアクセスします。ESXi では、VGA コンソールは、インストールが完了すると自動的に開くコンソールです。VM コンソールに接続するための特定の手順は必要ありません

vga イメージをロードした後、VM コンソールにログインできます。VM コンソールには、直接ログインの利点があります。ただし、VM コンソールを使用している場合、VM コンソールでのコピーが少し難しい場合があります。また、ナビゲーションはレベルに制限されます。シリアルコンソールのコピーオプションとナビゲーションコントロールが改善されました。

Cisco IOS XRv 9000 ルータは、VM の電源がオンになったときに起動されます。どのインストールイメージを使用したかに応じて（VGA の有無に応じて）、VM コンソールまたは仮想シリアルポートのコンソールでインストールプロセスをモニタできます。

インストール手順は、VM コンソールの有無にかかわらず同じです。

適切な GUI インターフェイス経由で VM コンソールにアクセスする方法については、VMware または OpenStack のドキュメンテーションを参照してください。

仮想シリアルポート経由の Cisco IOS XRv 9000 ルータへのアクセス

デフォルトで、Cisco IOS XRv 9000 ルータは VM コンソールを使用してアクセスされます。Cisco IOS XRv 9000 ルータのコンソールポートとして仮想シリアルポートを使用するように VM を設定できます。ハイパーバイザ上に仮想シリアルポートを設定するには、次の項を参照してください。

次に、デフォルトのコンソール設定を示します。

- ボー レート 115200 bps
- パリティなし
- 2 ストップ ビット
- 8 データ ビット



(注) Cisco IOS XRv 9000 ルータのインストール中にこの手順を実行する必要があります。

VMware ESXi のシリアル コンソール アクセスの設定

この手順では、VMware vSphere を使用して VMware ESXi のシリアル コンソール アクセスを設定する方法について説明します。詳細については、VMware vSphere のドキュメンテーションを参照してください。

手順

ステップ 1 VM の電源をオフにします。

ステップ 2 VM を選択し、仮想シリアルポートを設定します。

- a) [Edit Settings] > [Add] を選択します。
- b) [Device Type] > [Serial port] を選択し、[Next] をクリックします。
- c) [Select Port Type] を選択し、[Connect via Network] を選択して、[Next] をクリックします。

ステップ 3 [Select Network Backing] を選択し、次の手順を実行します。

- [Server (VM listens for connection)] を選択します。
- シンタックスとして `telnet://esxi-host-ipaddress:portnumber` を使用し、[Port URI] に入力します。
ここで、*portnumber* は仮想シリアルポートのポート番号です。
- [I/O mode] から、[Yield CPU on poll] を選択します。

- [Next] をクリックします。

ステップ 4 VM の電源をオンにします。

VM の電源がオンになったら、仮想シリアルポート コンソールにアクセスします。仮想シリアルポートにアクセスできない場合は、ステップ 5 を実行します。

ステップ 5 仮想シリアルポートのセキュリティ設定を行います。

- 仮想シリアルポートに [ESXi host] を選択します。
- [Configuration] タブをクリックし、[Security Profile] をクリックします。
- [Firewall] セクションで、[Properties] をクリックし、次に [VM serial port connected over Network] の値を選択します。

次のタスク

これで、Telnet ポート URI (telnet esxi-host-ip-address <portnumber>) を使用して Cisco IOS XR コンソールにアクセスできるようになります。

仮想シリアルポートを設定すると、VMware ESXi コンソールから Cisco IOS XRv 9000 ルータにアクセスすることはできなくなります。

Virsh を使用した KVM のシリアル コンソール アクセスの設定

KVM 環境のシリアル コンソール アクセス設定については、Cisco.com からダウンロードしたインストールイメージに付属するサンプルの Virsh XML ファイルに説明されています。ただし、XML の送信元設定にあるシリアルポート番号を編集する必要があります。

この手順では、4 つのシリアルポートのシリアルポート番号を編集し、そのシリアルポートに Telnet 接続してシリアル コンソールにアクセスする方法を説明します。

手順

ステップ 1 各シリアルポートにサービス値を指定します。各ポートのサービス値を指定する際は、未使用のポートを選択する必要があります。

最初のシリアルポート (ポート 0) には、11768 がポート番号として指定されます。

```
<!-- Use virsh qemu-monitor-command IOS-XRv-9000_vpe_rwe1node10_virsh - -hmp "info chardev" to view
or create serial ports -->
<serial type='tcp'>
  <source mode="bind" host="127.0.0.10" service="11768"/>
  <protocol type="telnet"/>
  <target port="0"/>
</serial>
```

2 番目のシリアルポート（ポート 1）には、12251 がポート番号として指定されます。

```
<serial type='tcp'>
  <source mode="bind" host="127.0.0.10" service="12251"/>
  <protocol type="telnet"/>
  <target port="1"/>
</serial>
```

3 番目のシリアルポート（ポート 2）には、17161 がポート番号として指定されます。

```
<serial type='tcp'>
  <source mode="bind" host="127.0.0.10" service="17161"/>
  <protocol type="telnet"/>
  <target port="2"/>
</serial>
```

4 番目のシリアルポート（ポート 3）には、16998 がポート番号として指定されます。

```
<serial type='tcp'>
  <source mode="bind" host="127.0.0.10" service="16998"/>
  <protocol type="telnet"/>
  <target port="3"/>
</serial>
```

ステップ 2 シリアル コンソールにアクセスするには、そのポートに Telnet 接続し、**telnet localhost <portnumber>** コマンドを使用します。

この例では、ポート番号 12251 が指定された 2 番目のシリアルポートにアクセスする方法を示します。

```
telnet localhost 12251
```

ステップ 3 Virsh コンソールが必要な場合は、**console** のセクションをアンコメントし、最後のシリアルポートをコメントアウトします。

次に例を示します。

```
<!-- <console type='pty'> -->
<!-- <target type='serial' port='0'> -->
<!-- </console> -->
```

QEMU を使用した KVM のシリアル コンソール アクセスの設定

KVM-QEMU コマンドラインを使用した Cisco IOS XRv 9000 ルータへのアクセスは、Telnet で確立できます。

手順

次のサンプル コマンドラインは、Telnet を介して接続できる 4 つのシリアルポートを作成します。

```
-serial telnet:127.0.1.10:10621,nowait,server \  
-serial telnet:127.0.1.10:14713,nowait,server \  
-serial telnet:127.0.1.10:18090,nowait,server \  
-serial telnet:127.0.1.10:17181,nowait,server \  

```

ポートの1つにアクセスするには、**telnet localhost <port-number>** コマンドを使用します。

たとえば、上のサンプル CLI に表示されている最初のポートにアクセスするには、次のようにします。

```
telnet localhost 10621
```

システム管理コンソールから XR コンソールへのアクセス

インバンド接続に問題があり、システム管理コンソールにのみログインできる場合は、次の方法を使用して XR コンソールにアクセスできます。

手順

ステップ 1 root ユーザとしてシステム管理コンソールにログインします。

ステップ 2 **show vm location 0/RP0**

例：

次に、すべての仮想マシン（VM）が表示されたコマンド出力の例を示します。

```
sysadmin-vm:0_RP0# show vm location 0/RP0  
Location: 0/RP0  
-----  
Id                Status           IP Address       HB Sent/Recv  
-----  
sysadmin          running         192.0.0.1        NA/NA  
default-sdr      running         192.0.0.4        6304304/6304304  
default-sdr      running         192.0.0.6        315193/315193
```

ステップ 3 **run ssh ip address**。

例：

次に、最初の **default-sdr** に接続する例を示します。

```
sysadmin-vm:0_RP0# run ssh 192.0.0.4  
Last login: Fri Apr 6 20:53:47 2018 from 192.0.0.1  
[xr-vm_node0_RP0_CPU0:~]$
```

ステップ 4 **exec**

例：

次に、XR ログイン クレデンシアルを使用して XR コンソールにアクセスする例を示します。

```
[xr-vm_node0_RP0_CPU0:~]$exec
User Access Verification

Username: iox

Password:

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

ユーザープロファイルの作成および権限の割り当て

ルータ上の XR およびシステム管理設定へのアクセス権を管理するには、権限を割り当てたユーザープロファイルを作成します。権限はコマンドルールとデータルールを使用して指定します。ユーザー、グループ、コマンドルール、およびデータルールを作成するには、認証、許可、およびアカウントिंग (AAA) コマンドを使用します。aaa コマンドはディザスタリカバリ パスワードを変更する際にも使用します。

ユーザープロファイルの作成と権限の割り当ての詳細については、『[System Setup and Software Installation Guide for Cisco ASR 9000 Series Routers](#)』の「Create User Profiles and Assign Privileges」の章を参照してください。



第 7 章

Cisco IOS XRv 9000 ルータのコントロールプレーンの特徴的な機能

この章では、Cisco XRv 9000 コントロールプレーンの特徴的な機能について説明します。

- [BGP の最適なルート リフレクタ \(59 ページ\)](#)
- [bgp bestpath igp-metric ignore コマンドのサポート \(75 ページ\)](#)
- [マルチホップパスの BFD \(76 ページ\)](#)
- [CVAC : ブートストラップ構成のサポート \(78 ページ\)](#)
- [FlexAlgo の ORR サポート \(81 ページ\)](#)
- [セグメントルーティングフレキシブルアルゴリズムの有効化 \(81 ページ\)](#)
- [ポリシーベースルーティングを使用した IPv4 および IPv6 トラフィックのリダイレクト \(92 ページ\)](#)
- [テレメトリ更新の gNMI バンドル \(95 ページ\)](#)
- [IPv4 GRE トンネルの QoS \(96 ページ\)](#)
- [ネットワークスタックへのアクセス \(98 ページ\)](#)
- [Cisco IOS XR Linux シェルでのアプリケーションホスティング \(103 ページ\)](#)

BGP の最適なルート リフレクタ

表 7: 機能の履歴 (表)

機能名	リリース情報	機能説明
-----	--------	------

BGP ORR 6PE	リリース 7.3.1	<p>この機能が導入されます。特定の ORR テーブルのベストパスとして選択可能なパスがない場合は、デフォルトテーブルのベストパスを ORR グループのベストパスとして割り当てることができます。この機能は、IPv4 ネクストホップおよびデフォルトのフォールバックパスを使用した IPv6 ラベルユニキャストを有効にします。</p> <p>新しいキーワードがこのリリースで追加されました。</p> <ul style="list-style-type: none"> • fallback-default-bestpath
-------------	------------	--

BGP-ORR（最適なルートリフレクタ）により、仮想ルートリフレクタ（vRR）は、ルートリフレクタの（RR）クライアントの観点からベストパスを計算できます。

BGP ORR は次の方法でベストパスを計算します。

1. RR クライアントまたは RR クラスター（RR クライアントのセット）のコンテキストで、SPF を複数回実行します。
2. それぞれの SPF 実行結果を、別個のデータベースに保存します。
3. これらのデータベースを使用して BGP のベストパス判断を処理し、これにより BGP がクライアントの観点から最適なベストパスを使用し、通知できるようにします。

自律システムでは、BGP のルートリフレクタは焦点として機能し、RR が計算したベストパスとルートとそのピア（RR クライアント）にアドバタイズします。RR によってアドバタイズされたベストパスは RR の観点から計算されることになるので、RR の配置は導入に関する重要な検討事項になります。

ネットワーク機能の仮想化（NFV）が主要な技術となっていることから、サービスプロバイダー（SP）は複数のサーバを使用するクラウドで仮想 RR 機能をホストしています。vRR はコントロールプレーンデバイス上で実行でき、トポロジまたは SP データセンター内のどこにでも配置できます。Cisco IOS XRv 9000 ルータは、SP データセンター内の NFV プラットフォーム上の vRR として実装できます。vRR を利用することにより、SP は RR 導入のメモリと CPU 使用率を大幅に拡張できます。RR を最適な配置から移動するには、RR クライアントの観点から最適なパスを計算する ORR 機能を vRR が実装する必要があります。

BGP ORR には次のような利点があります。

- RR クライアントの観点からベストパスを計算します。
- vRR をトポロジまたは SP データセンター内のどこにでも配置できます。
- SP は RR 導入のメモリと CPU 使用率を拡張できます。



- (注) ORR 機能を有効にすると、BGP と RIB のメモリフットプリントが増加します。ネットワーク内に設定されている vRR の数が増えると、ORR は BGP のコンバージェンスに悪影響を及ぼします。

BGP ORR の設定方法

BGP ORR は次のように設定できます。

- パス アドバタイズメント ポリシーを使用しない BGP ORR。
- パス アドバタイズメント ポリシーを使用した BGP ORR。

パス アドバタイズメント ポリシーを使用しない BGP ORR

vRR は次の手順を実行して、BGP ORR ベストパスを計算し、すべての vRR サイトと ORR グループにアドバタイズします。

1. vRR は、グローバルアドレス ファミリー インジケータ (AFI) ポリシーから各 ORR グループの BGP ORR ベストパスを計算します。
2. vRR は、選択された追加パスとともに、計算したベストパスをアドバタイズします。

ORR グループにポリシーを割り当てていない場合、デフォルトでは、vRR はその ORR グループのグローバル AFI のデフォルト選択ポリシーを使用してベストパスを計算します。詳細については、[パス アドバタイズメント ポリシー トポロジを使用しない BGP ORR \(62 ページ\)](#) を参照してください。

パス アドバタイズメント ポリシーを使用した BGP ORR

vRR は次の手順を実行して、BGP ORR ベストパスとバックアップ/追加パスを計算し、すべての vRR サイトと ORR グループにアドバタイズします。

1. **optimal-route-reflection** コマンドを使用して、ORR グループ追加パスポリシーを設定できます。詳細については、[パス アドバタイズメント ポリシーを使用した BGP 最適ルートリフレクタ \(69 ページ\)](#) を参照してください。
2. vRR は、割り当てられたポリシーに基づいて各 ORR グループのベストパスを計算します。
3. vRR は、追加パスルートポリシーを持つ ORR グループ内の PE ルータにベストパスとバックアップ/追加パスをアドバタイズします。



- (注) 必要に応じて、**optimal-route-reflection** コマンドで *addpath-all policy* を使用して、vRR が ORR グループへのすべてのパスをアドバタイズできるようにすることができます。

パスアドバタイズメントポリシートポロジを使用しない BGP ORR

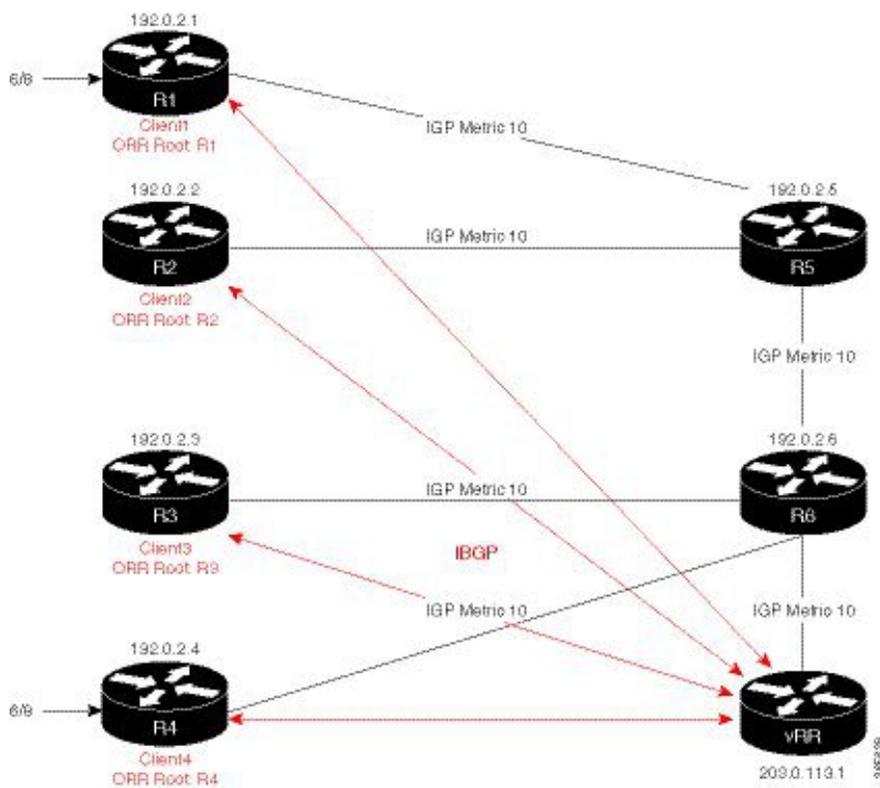
4. vRRは、計算されたベストパスを、すべての既知のパスとともに、リージョン内グループ内のすべてのvRR サイトにアドバタイズします。
5. vRRは、クライアントパスと必要なピアリングパスのみを、リージョン間グループ内のすべてのvRR サイトにアドバタイズします。

パスアドバタイズメントポリシートポロジを使用しない BGP ORR

次のようなパスアドバタイズメントポリシートポロジを使用しない BGP ORR があるとします。

- ルータ R1、R2、R3、R4、R5、R6 がルートリフレクタクライアントである。
- ルータ R1 および R4 が vRR に 6/8 プレフィックスをアドバタイズする。

図 4: パスアドバタイズメントポリシートポロジを使用しない BGP-ORR



vRR は、R1 および R4 からプレフィックス 6/8 を受信します。ネットワークに BGP ORR が設定されていない場合、vRR は RR クライアント R2、R3、R5、R6 の最も近い出力点として R4 を選択し、R4 から学習した 6/8 プレフィックスをこれらの RR クライアント (R2、R3、R5、R6) にリフレクトします。トポロジから、R2 のベストパスが R4 ではなく R1 であるのは明らかです。これは、vRR が RR の観点からベストパスを計算するためです。

BGP ORR がネットワークに設定されると、vRR は R2 の観点からネットワークの最短出力点を計算し、R2 に最も近い出力点は R1 であると判別します。その結果、vRR は R1 から学習した 6/8 プレフィックスを R2 にリフレクトします。

パスアドバタイズメント ポリシーを使用しない BGP ORR の制約事項と制限事項

次に、パスアドバタイズメント ポリシーを使用しない BGP ORR の制約事項と制限事項を示します。

- ORR グループに複数のパスが追加されている場合、vRR はそれらの中からパスを 1 つだけ選択して、vRR サイトのベストパスを計算します。
- vRR は、すべてのクライアントに同じタイプおよびセットの追加パスを計算してアドバタイズします。

パスアドバタイズメント ポリシーを使用しない BGP ORR の設定

手順

パスアドバタイズメント ポリシーを使用しない BGP ORR を設定するには、次の手順を実行します。

ステップ 1 ルータ BGP モードで ORR をグローバルに設定します。

例：

```
Router# configure
Router(config)# router bgp 100
Router(config-bgp)# optimal-route-reflection ipv4 foo 10.1.1.1 10.1.1.2 10.1.1.3
Router(config-bgp)# optimal-route-reflection ipv6 bar abcd::1 abcd::2 abcd::3
```

ステップ 2 アドレスファミリモードで ORR グループを有効にします。

例：

```
Router(config-bgp-route)# address-family ipv4 unicast
Router(config-bgp-route-af)# optimal-route-reflection apply foo
Router(config-bgp-route-af)# exit
Router(config-bgp-route)# address-family ipv6 unicast
Router(config-bgp-route-af)# optimal-route-reflection apply bar
Router(config-bgp-route-af)# allocate-label {all | route-policy <>}
Router(config-bgp-route-af)# commit
```

ステップ 3 ネイバーを ORR クライアントとして指定します。

例：

```
Router# configure
Router(config)# neighbor 2.2.2.2
Router(config-nbr)# remote-as 100
Router(config-nbr)# address-family ipv4 unicast
```

パスアドバタイズメントポリシーを使用しない BGP ORR の設定

```
Router(config-nbr-af)# optimal-route-reflection foo
Router(config-nbr-af)# exit
Router(config-nbr)# address-family ipv6 label-unicast
Router(config-nbr-af)# optimal-route-reflection foo
Router(config-nbr-af)# commit
```

ステップ 4 ORR グループのベストパスがない場合は、デフォルトテーブルのベストパスの選択を有効にします。

例：

```
Router# configure
Router(config)# router bgp 65000
Router(config-bgp)# bgp router-id 10.1.1.1
Router(config-bgp)# address-family ipv4 unicast
Router(config-bgp-af)# optimal-route-reflection fallback-default-bestpath
```

ステップ 5 実行コンフィギュレーションを表示して、設定を確認します。

例：

```
router bgp 100
optimal-route-reflection ipv4 foo 10.1.1.1 10.1.1.2 10.1.1.3
optimal-route-reflection ipv6 bar abcd::1 abcd::2 abcd::3
address-family ipv4 unicast
  optimal-route-reflection apply foo
address-family ipv6 unicast
  optimal-route-reflection apply foo
  allocate-label {all | route-policy <>}
neighbor 2.2.2.2
  remote-as 100
address-family ipv4 unicast
  optimal-route-reflection foo
address-family ipv6 label-unicast
  optimal-route-reflection foo
```

ステップ 6 IPv4 の BGP ORR 設定を確認するには、次の手順を実行します。

- R2 が最適な出力を受信したかどうかを確認し、EXEC モードで **show bgp <prefix>** コマンドを (R2 から) 実行します。上記の例では、R1 および R4 は 6/8 プレフィックスをアドバタイズします。次のように **show bgp 6.0.0.0/8** コマンドを実行します。

例：

```
Router# sh bgp 200.1.1.0/24 path-elements
BGP routing table entry for 200.1.1.0/24
Versions:
  Process      bRIB/RIB SendTblVer
  Speaker      84        84
  Flags: 0x043e3028+0x00000000;
Last Modified: Sep 7 20:21:38.000 for 00:11:45
Paths: (2 available, best #1)
Path-elements: 2
  Path ID: 1
    Gateway metric 0, Version 84
    Path: Nexthop 11.1.1.2, flags 0x4000000001440207
          Neighbor 11.1.1.2, Received Path ID 0
    Flags: 0x00000001
          status: valid
          path type: bestpath
          add-path action:
    Opaque: pelem=0xebf81d10
            net=0xab435c, tblattr=0xa962f64 (ver 84)
            path=0xabf4450, path-tblattr=0xa962f64 (ver 84)
```


(注)

パス #1 は、アップデートグループ 0.1 にアドバタイズされます。R2 はアップデートグループ 0.1 に含まれます。

- c) **show bgp** コマンドをアップデートグループ 0.1 に対して実行し、R2 がアップデートグループ 0.1 に含まれているかどうか確認します。

例：

```
VRR# show bgp update-group 0.1
Thu Apr 28 13:38:18.517 UTC

Update group for IPv4 Unicast, index 0.1:
Attributes:
Neighbor sessions are IPv4
Internal
Common admin
First neighbor AS: 65000
Send communities
Send GSHUT community if originated
Send extended communities
Route Reflector Client
ORR root (configured): g1; Index: 0
4-byte AS capable
Non-labeled address-family capable
Send AIGP
Send multicast attributes
Minimum advertisement interval: 0 secs
Update group desynchronized: 0
Sub-groups merged: 0
Number of refresh subgroups: 0
Messages formatted: 5, replicated: 5
All neighbors are assigned to sub-group(s)
Neighbors in sub-group: 0.2, Filter-Groups num:1
Neighbors in filter-group: 0.2(RT num: 0)
192.0.2.2
```

- d) 次の確認点として、g1 ポリシーを設定した結果 vRR に作成されたテーブルのコンテンツを確認します。R2 の観点からは、R1 に到達するためのコストは 20 で、R4 に到達するためのコストは 30 です。したがって、R2 に最も近い最適な出力は R1 経由になります。

例：

```
VRR# show orrspf database g1
Thu Apr 28 13:39:20.333 UTC

ORR policy: g1, IPv4, RIB tableid: 0xe0000011
Configured root: primary: 192.0.2.2, secondary: NULL, tertiary: NULL
Actual Root: 192.0.2.2, Root node: 2000.0100.1002.0000

Prefix Cost
203.0.113.1 30
192.0.2.1 20
192.0.2.2 0
192.0.2.3 30
192.0.2.4 30
192.0.2.5 10
192.0.2.6 20

Number of mapping entries: 8
```

ステップ7 IPv6 プロバイダーエッジ (6PE) の BGP ORR 設定を確認するには、次の手順を実行します。

- a) **show bgp** コマンドを使用して、BGP ORR の設定を確認します。

例 :

```
show bgp ipv6 labeled-unicast 1111::1/128
Tue Mar 2 10:25:00.748 PST
BGP routing table entry for 1111::1/128
Versions:
Process bRIB/RIB SendTblVer
Speaker 4 4
Last Modified: Mar 2 10:18:53.000 for 00:06:08
Paths: (3 available, best #3)
Advertised IPv6 Labeled-unicast paths to update-groups (with more than one peer):
0.2
Path #1: Received by speaker 0
ORR bestpath for update-groups (with more than one peer):
0.1
Local, (Received from a RR-client)
192.168.0.3 (metric 75) from 192.168.0.3 (192.168.0.3)
Received Label 24007
Origin incomplete, metric 0, localpref 100, valid, internal, add-path, labeled-unicast
Received Path ID 0, Local Path ID 2, version 4
Path #2: Received by speaker 0
Not advertised to any peer
Local, (Received from a RR-client)
192.168.0.4 (metric 190) from 192.168.0.4 (192.168.0.4)
Received Label 24007
Origin incomplete, metric 0, localpref 100, valid, internal, labeled-unicast
Received Path ID 0, Local Path ID 0, version 0
Path #3: Received by speaker 0
Advertised IPv6 Labeled-unicast paths to update-groups (with more than one peer):
0.2
Local, (Received from a RR-client)
192.168.0.5 (metric 65) from 192.168.0.5 (192.168.0.5)
Received Label 24007
Origin incomplete, metric 0, localpref 100, valid, internal, best, group-best, labeled-unicast
Received Path ID 0, Local Path ID 1, version 3
```

- b) **show bgp** コマンドを使用して、ipv6 グループの設定を確認します。

例 :

```
show bgp ipv6 labeled-unicast update-group
Tue Mar 2 10:25:51.308 PST

Update group for IPv6 Labeled-unicast, index 0.1:
Attributes:
Neighbor sessions are IPv4
Internal
Common admin
First neighbor AS: 1
Send communities
Send GSHUT community if originated
Send extended communities
Route Reflector Client
ORR root (configured): orr-grp-1; Index: 0
4-byte AS capable
Send AIGP
Send multicast attributes
Minimum advertisement interval: 0 secs
Update group desynchronized: 0
Sub-groups merged: 0
Number of refresh subgroups: 0
```

パスアドバタイズメントポリシーを使用しない BGP ORR の設定

```

Messages formatted: 1, replicated: 2
All neighbors are assigned to sub-group(s)
Neighbors in sub-group: 0.2, Filter-Groups num:1
Neighbors in filter-group: 0.2(RT num: 0)
192.168.0.2 192.168.0.4

Update group for IPv6 Labeled-unicast, index 0.2:
Attributes:
Neighbor sessions are IPv4
Internal
Common admin
First neighbor AS: 1
Send communities
Send GSHUT community if originated
Send extended communities
Route Reflector Client
4-byte AS capable
Send AIGP
Send multicast attributes
Minimum advertisement interval: 0 secs
Update group desynchronized: 0
Sub-groups merged: 0
Number of refresh subgroups: 0
Messages formatted: 1, replicated: 4
All neighbors are assigned to sub-group(s)
Neighbors in sub-group: 0.1, Filter-Groups num:1
Neighbors in filter-group: 0.1(RT num: 0)
192.168.0.3 192.168.0.5

```

- c) **show bgp** コマンドを使用して、BGP 設定の IPv6 ユニキャストルートを確認します。

例：

```

show bgp ipv6 unicast orr-group all
Tue Mar 2 10:26:41.072 PST
Name Tableid Nbrcnt Index Root
orr-grp-1 0xe0000019 2 0 192.168.0.3

```

- d) BGP スピーカーのグローバル ORR ポリシーグループを確認します。

例：

次の show コマンドは、BGP スピーカーのグローバル ORR ポリシーグループテーブルを表示します。

```

Router# show bgp orr-group global all
Wed Apr 8 16:46:29.929 PDT
Name Policy-afi Global Tableid AFI-count Root
orr-grp-3 IPv4 Yes 0xe0000014 1 1.1.2.1
orr-grp-2 Ipv6 Yes 0xe0800013 0 1::1
orr-grp-1 IPv4 Yes 0xe0000012 2 192.168.0.3

```

次の show コマンドは、特定の ORR 名のグローバル ORR グループエントリの詳細を表示します。

```

Router# show bgp orr-group global orr-grp-1
Wed Apr 8 16:46:51.596 PDT
  ORR Name : orr-grp-1
    policy afi : IPv4
  global Defined : Yes
    tableid : 0xe0000012
    aficnt : 2
  IPv4 unicast used : Yes

```

```
IPv6 unicast used : Yes
      root : 192.168.0.3
```

次の show コマンドは、BPM ORR ポリシーグループテーブルを表示します。

```
Router# show bgp orr-group bpm all
Wed Apr  8 16:49:44.223 PDT
Name          Policy-afi Global  AFI-cnt  Nbr-af-cnt  Root
orr-grp-3     IPv4         Yes    1         0            1.1.2.1
orr-grp-2     IPv6         Yes    0         0            1::1
orr-grp-1     IPv4         Yes    2         4            192.168.0.3
```

次の show コマンドは、特定の ORR 名の BPM ORR グループエントリの詳細を表示します。

```
Router# show bgp orr-group bpm orr-grp-1
Wed Apr  8 16:50:02.437 PDT
  ORR Name : orr-grp-1
    v4 policy : Yes
  global Defined : Yes
    AFI count : 2
  total nbr af cnt : 4
  IPv4 unicast used : Yes
  IPv6 unicast used : Yes
    IPv4 nbr af cnt : 2
    IPv6 nbr af cnt : 2
    root : 192.168.0.3
```

パス アドバタイズメント ポリシーを使用した BGP 最適ルートリフレクタ

表 8: 機能の履歴 (表)

機能名	リリース情報	機能説明

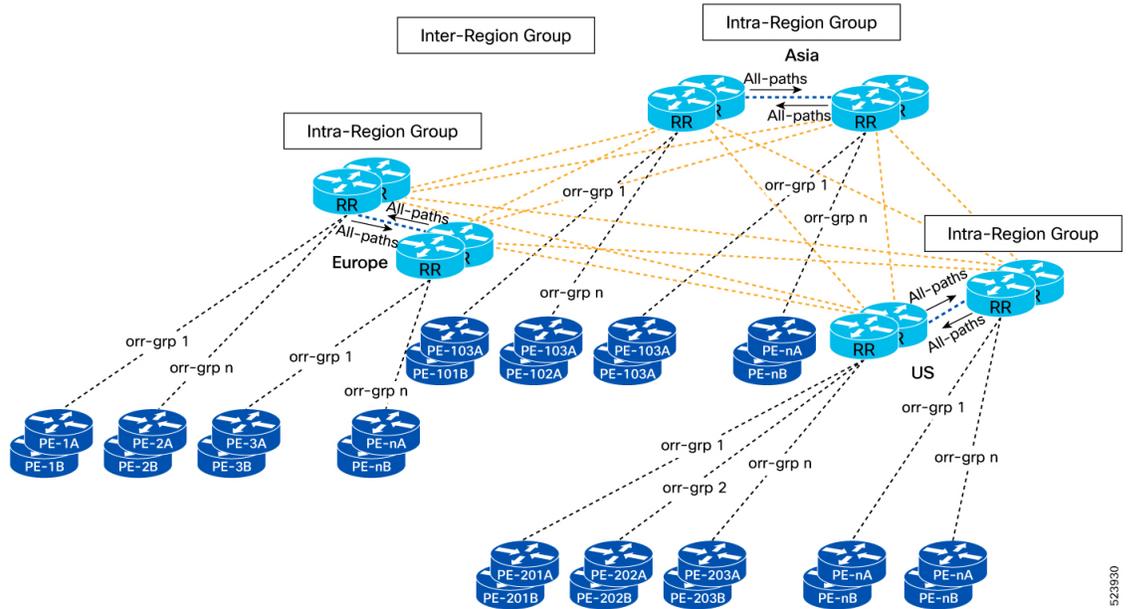
<p>パスアドバタイズメントポリシーを使用した BGP 最適ルータリフレクタ</p>	<p>リリース 24.2.1</p>	<p>この機能を使用すると、個別の最適ルータリフレクタ (ORR) グループに独立したルーティングポリシーを割り当てることができます。仮想ルータリフレクタ (vRR) は、割り当てられたポリシーに基づいて、その ORR グループの追加パスを計算します。これにより、ORR グループごとに異なるベストパスとバックアップパスが作成されます。これは、クライアントの観点から最適なパスです。</p> <p>この機能により、次の変更が導入されました。</p> <p>CLI :</p> <ul style="list-style-type: none"> • <i>route-policy</i> キーワードが optimal-route-reflection コマンドに導入されました。 <p>YANG データモデル :</p> <ul style="list-style-type: none"> • <code>Cisco-IOS-XR-ipv4-bgp-cfg.yang</code> • <code>Cisco-IOS-XR-um-router-bgp-cfg.yang</code> <p>(「GitHub」、「YANG Data Models Navigator」を参照)</p>
--	--------------------	---

BGP-ORR (最適なルータリフレクタ) により、仮想ルータリフレクタ (vRR) は、ルータリフレクタの (RR) クライアントの観点からベストパスを計算できます。詳細については、[BGP の最適なルータリフレクタ \(59 ページ\)](#) を参照してください。

トラフィックフロートポロジ

ネットワークトラフィックが ORR グループから、vRR サイトを通過し、リージョン内グループを通過し、最後にリージョン間グループを通過するトポロジについて考えてみます。[図 5: BGP-ORR ベストパスのアドバタイズ \(71 ページ\)](#) を参照してください。

図 5: BGP-ORR ベストパスのアドバタイズ



次に、[図 5: BGP-ORR ベストパスのアドバタイズ \(71 ページ\)](#) に示すように、BGP ORR 環境内のネットワーク要素の階層構造の定義を示します。

- ORR グループ：複数のルータとプロバイダーエッジ (PE) ルータのセットが接続して ORR グループを形成します。たとえば、`orr-grp n` です。
- vRR サイト：複数の ORR グループが接続して vRR サイトを形成します。
- リージョン内グループ：複数の vRR サイトが接続してリージョン内グループを形成します。たとえば、`Europe` です。
- リージョン間グループ：複数のリージョン内グループが接続してリージョン間グループを形成します。

この機能を使用すると、vRR サイトおよび ORR グループごとに追加パス選択ポリシーを設定できます。vRR は、BGP ORR ベストパスとバックアップ/追加パスを計算して、すべての vRR サイトと ORR グループにアドバタイズします。詳細については、[パスアドバタイズメントポリシーを使用した BGP ORR \(61 ページ\)](#) を参照してください。

次のルータポリシーを設定して、vRR がベストパス、バックアップパス、または追加パスをアドバタイズできるようにすることができます。

- **「Is-destination」**：これは、ベストパス、バックアップパス、または追加パスなどの特定のタイプのパスを指定されたネイバーにアドバタイズするポリシーに割り当てられる用語です。このポリシーは、ORR ネイバーには適用されません。
- **「Set path-selection backup 1 advertise」**：これは、vRR がベストパスとバックアップパスのみを vRR サイトと ORR グループにアドバタイズできるようにするポリシーに割り当てられる用語です。

- 「**Set path-selection all advertise**」：これは、vRR がベストパスと追加パスを vRR サイトと ORR グループにアドバタイズできるようにするポリシーに割り当てられる用語です。

パスアドバタイズメントポリシーを使用した BGP ORR の制約事項と制限事項

次に、パスアドバタイズメントポリシーを使用した BGP ORR の制約事項と制限事項を示します。

- デフォルトでは、vRR は計算されたベストパスとバックアップパスを ORR グループにアドバタイズします。必要に応じて、**additional-paths selection** コマンドで *addpath-all* キーワードを使用して、vRR が ORR グループにすべてのパスをアドバタイズできるようにすることができます。
- デフォルトでは、vRR はすべてのパスをすべての vRR サイトにアドバタイズします。
- ORR グループにポリシーを割り当てておらず、AFI ルータポリシーにデフォルトパスが割り当てられていない場合、vRR はその ORR グループのベストパスを計算しません。
- ルータポリシーの「**Is-destination**」オプションは、ORR route-policy attach-point ではサポートされていません。たとえば、「is-destination」パスを設定した場合、「is-destination」パスは vRR にのみアドバタイズされ、PE にはアドバタイズされません。
- vRR は転送パスの一部ではないため、「**Set path-selection backup 1 install**」オプションは ORR グループをサポートしません。したがって、転送テーブルに ORR およびバックアップパスをインストールすることはできません。ORR グループに対してこのポリシーを設定すると、システムはインストールコマンドを無視し、デフォルトでは、指定された ORR グループにバックアップパスをアドバタイズします。

パスアドバタイズメントポリシーを使用した BGP ORR の設定

パスアドバタイズメントポリシーを使用して BGP ORR を設定するには、次の手順を実行します。

手順

ステップ 1 RPL ポリシー「Is-destination」を設定します。

例：

```
Router# configure
Router(config)# route-policy Bestpath
Router(config-rpl)# if destination is-best-path then
Router(config-rpl)# pass
Router(config-rpl)# else
Router(config-rpl)# drop
Router(config-rpl)# endif
```

```
Router(config-rpl)# end-policy
Router(config-rpl)# exit
```

このポリシーは、指定されたパスがベストパスである場合にのみ、vRR がパスを渡すことを許可します。それ以外の場合は、ルートをドロップします。このポリシーは、vRR が vRR サイトにベストパスのみをアドバタイズすることを許可し、ORR ネイバーには適用されません。

ステップ 2 ベストパスとバックアップパスをアドバタイズするように RPL ポリシーを設定します。

例：

```
Router# configure
Router(config)# route-policy Backup
Router(config-rpl)# set path-selection backup 1 advertise
Router(config-rpl)# end-policy
Router(config-rpl)# exit
```

このポリシーにより、vRR はベストパスとバックアップパスをアドバタイズできます。

ステップ 3 すべてのパスをアドバタイズするように RPL ポリシーを設定します。

例：

```
Router# configure
Router(config)# route-policy addpath-all
Router(config-rpl)# set path-selection all advertise
Router(config-rpl)# end-policy
Router(config-rpl)# exit
Router(config)# route-policy pass
Router(config-rpl)# pass
Router(config-rpl)# end-policy
Router(config-rpl)# exit
Router(config)# route-policy addpath-backup
Router(config-rpl)# set path-selection backup 1 advertise
Router(config-rpl)# end-policy
Router(config-rpl)# exit
Router(config)# end
Router#
```

このポリシーは、vRR が ORR グループにすべてのパスをアドバタイズすることを許可します。

ステップ 4 **optimal-route-reflection** コマンドを使用して ORR グループ追加ポリシーを設定し、**address-family** コマンドを使用してアドレスファミリを ORR グループに割り当てます。

例：

```
Router# configure
Router(config)# router bgp 13
Router(config-bgp)# optimal-route-reflection ipv4 orr-grp-1 1.1.1.1 route-policy addpath-backup
Router(config-bgp-router)# address-family ipv4 unicast
Router(config-bgp-router-af)# additional-paths selection route-policy addpath-all
Router(config-bgp-router-af)# optimal-route-reflection apply orr-grp-1
Router(config-bgp-router-af)# commit
```

ステップ 5 **neighbor-group** コマンドを使用して、BGP のネイバーグループを設定します。

例：

```
Router# configure
Router(config)# neighbor-group PR
Router(config-nbrgrp)# address-family ipv4 unicast
Router(config-nbrgrp-af)# route-reflector-client
Router(config-nbrgrp-af)# optimal-route-reflection orr-grp-1
```

パスアドバタイズメントポリシーを使用した BGP ORR の設定

```
Router(config-nbrgrp-af)# route-policy pass in
Router(config-nbrgrp-af)# commit
```

ステップ 6 neighbor コマンドを使用して、設定された BGP ネイバークラスタを BGP ネイバーに適用します。

例：

```
Router# configure
Router(config)# neighbor 10.10.10.10
Router(config-nbr)# use neighbor-group PR
```

ステップ 7 実行コンフィギュレーションを表示して、設定を確認します。

例：

```
/*Enable to BGP Policies*/
route-policy addpath-all
  set path-selection all advertise
end-policy
route-policy pass
  pass
end-policy
route-policy addpath-backup
  set path-selection backup 1 advertise
end-policy

/*Configure ORR Add-Path and create ORR Groups*/
router bgp 13
  optimal-route-reflection ipv4 orr-grp-1 1.1.1.1 route-policy addpath-backup
  optimal-route-reflection ipv4 orr-grp-2 2.2.2.2
  address-family ipv4 unicast
    additional-paths selection route-policy addpath-all
    optimal-route-reflection apply orr-grp-1
    optimal-route-reflection apply orr-grp-2
  !
!

/* Configure BGP neighbor groups*/
neighbor-group PR
  address-family ipv4 unicast
    route-reflector-client
    optimal-route-reflection orr-grp-1
    route-policy pass in
  !
!
neighbor-group BB
  address-family ipv4 unicast
    route-reflector-client
    route-policy PR_BB_COMM in
  !
!

/*Apply the neighbor groups*/
neighbor 10.10.10.10
  use neighbor-group PR
neighbor 3.3.3.3
  use neighbor-group BB
```

ステップ 8 show bgp コマンドを使用して、追加パスポリシー設定と BGP ORR を確認します。

例：

```
Router# sh bgp 200.1.1.0/24 path-elements
BGP routing table entry for 200.1.1.0/24
Versions:
```


次に、ベストパスの選択を実行するときに内部ゲートウェイプロトコル (IGP) メトリックを無視するようにソフトウェアを設定する例を示します。この例では、ルータ BGP VRF コンフィギュレーション モードでコマンドが設定されています。

```
RP/0/0/CPU0:router#configure
RP/0/0/CPU0:router (config)#router bgp 50000
RP/0/0/CPU0:router (config-bgp)#vrf 1
RP/0/0/CPU0:router (config-bgp-vrf)#bgp bestpath igp-metric ignore
```

マルチホップパスの BFD

表 9: 機能の履歴 (表)

機能名	リリース情報	機能説明
iBGP および eBGP の BFDv4 および BFDv6 マルチホップ	リリース 7.3.1	この機能では、複数ホップ (最大 255 ホップ) 離れた宛先に対する 1 秒未満の転送障害検出が可能になります。この機能は、BFD シングルホップをサポートするすべてのメディアタイプでサポートされます。

BFD マルチホップ (BFD-MH) は、同じサブネット上にない 2 つのアドレス間の BFD セッションです。BFD-MH の例には、PE および CE ループバック アドレス間の BFD セッションや、数 TTL ホップ離れたルータ間の BFD セッションがあります。外部および内部 BGP アプリケーションは BFD マルチホップをサポートします。BFD マルチホップは、複数のネットワークホップにまたがる場合もある任意のパス上の BFD をサポートします。

マルチホップパスの BFD 機能は、BFD シングルホップをサポートするすべてのメディアタイプでサポートされます。



(注) iBGP および eBGP 機能のマルチホップ BFDv4 および BFDv6 は、MPLS/GRE トンネル/SR ではサポートされません。

BFD マルチホップセッションの設定

BFD マルチホップセッションは、クライアントによって指定された送信元アドレスと宛先アドレスの一意のペア間で設定されます。IP 接続された 2 つのエンドポイント間でセッションを設定できます。グローバルルーティング テーブルと VRF テーブルの両方にある IPv4 アドレスがサポートされます。



(注) アグレッシブタイマーは、BFD マルチパスセッションおよびマルチホップセッションに使用することは推奨されません。推奨時間は 100 ミリ秒 x 3 = 300 ミリ秒以上です。

BFD IPv6 マルチホップセッションの設定

BFD を BGP とともに使用すると、BFD セッションタイプ（シングルホップまたはマルチホップ）が BGP 設定に基づいて設定されます。eBGP-multihop キーワードを設定すると、BFD セッションもマルチホップモードで実行されます。それ以外の場合、セッションはシングルホップモードで実行されます。

特定のホップ数を超えるネイバーから送信された BFD パケットをドロップするには、**bfd multihop ttl-drop-threshold** コマンドを使用します。

- eBGP ネイバーの BFD IPv6 マルチホップの設定
- iBGP ネイバーの BFD IPv6 マルチホップの設定
- BGP ネイバーで BFD を有効化

eBGP ネイバーの BFD IPv6 マルチホップの設定

```
Router# configure
Router(config)# bfd multipath include location 0/7/CPU0
Router(config)# router bgp 65001
Router(config-bgp)# neighbor 21:1:1:1:1:1:2 ebgp-multihop 255
Router(config-bgp)# neighbor 21:1:1:1:1:1:2 bfd fast-detect
```

iBGP ネイバーの BFD IPv6 マルチホップの設定

```
Router# configure
Router(config)# bfd multipath include location 0/7/CPU0
Router(config)# router bgp 65001
Router(config-bgp)# neighbor 21:1:1:1:1:1:2
```

BGP ネイバーで BFD を有効化

```
Router# configure
Router(config)# router bgp 120
Router(config-bgp)# bfd minimum-interval 6500
Router(config-bgp)# bfd multiplier 7
Router(config-bgp)# neighbor 172.168.40.24
Router(config-bgp-nbr)# remote-as 2002
Router(config-bgp-nbr)# bfd fast-detect
```

実行コンフィギュレーション

次に、eBGP ネイバーの BFD IPv6 マルチホップの実行コンフィギュレーションを示します。

```
Router# show running-configuration
bfd multipath include location 0/7/CPU0
router bgp 65001
neighbor 21:1:1:1:1:1:2 ebgp-multihop 255
neighbor 21:1:1:1:1:1:2 bfd fast-detect
```

次に、iBGP ネイバーの BFD IPv6 マルチホップの実行コンフィギュレーションを示します。

```
Router# show running-configuration
bfd multipath include location 0/7/CPU0
router bgp 65001
```

```
neighbor 21:1:1:1:1:1:2
```

次に、iBGP ネイバーの BFD IPv6 マルチホップの実行コンフィギュレーションを示します。

```
Router# show running-configuration
```

```
router bgp 120
bfd minimum-interval 6500
bfd multiplier 7
neighbor 172.168.40.24
remote-as 2002
bfd fast-detect
```

確認

```
Router# show bfd session
```

```
Tue Apr 7 06:16:36.982 UTC
```

Src Addr	Dest Addr	VRF Name	H/W	NPU	Local	det	time(int*mult)	State
10.1.1.1	192.0.2.1	default	No	n/a	n/a	150ms	(50ms*3)	UP
10.1.1.2	192.0.2.2	default	No	n/a	n/a	150ms	(50ms*3)	UP
10.1.1.3	192.0.2.3	default	No	n/a	n/a	150ms	(50ms*3)	UP
10.1.1.4	192.0.2.4	default	No	n/a	n/a	150ms	(50ms*3)	UP

```
Router# show bfd ipv6 session
```

```
Tue Apr 7 06:16:45.012 UTC
```

Src Addr	Dest Addr	VRF Name	Local	det	time(int*mult)	State	Echo	Async
2001:DB8::1	2001:DB8:0:ABCD::1	default	0s (0s*0)	150ms	(50ms*3)	UP		
2001:DB8::2	2001:DB8:0:ABCD::2	default	0s (0s*0)	150ms	(50ms*3)	UP		
2001:DB8::3	2001:DB8:0:ABCD::3	default	0s (0s*0)	150ms	(50ms*3)	UP		
2001:DB8::4	2001:DB8:0:ABCD::4	default	0s (0s*0)	150ms	(50ms*3)	UP		

CVAC : ブートストラップ構成のサポート

Cisco Virtual Appliance Configuration (CVAC) は、複数の Cisco 仮想ルータでサポートされるアウトオブバンド構成機能です。CVACは、ハイパーバイザによって提供されるCD-ROM、ディスクイメージ、またはUSBドライブで、仮想ルータ環境に入っている構成を受け取ります。この構成は、起動時に検出されて適用されます。

これにより、ユーザは初期導入の際に新しい仮想ルータをスタートアップ（ブートストラップ）と組み合わせることができるようになり、通常は手動で実施する必要のある基本的な多数の要件（管理IPアドレスなど）の構成が非常に簡単に短時間で行えるようになります。



(注) CVACは、既存の構成（新しいシステムで入力を求められる初期ユーザ名およびパスワードを含む）がない場合に機能します。

Cisco IOS XRv 9000 ルータは、ネイティブのKVM、Openstack Config Drive、およびVirshでCVACを完全にサポートします。VMware ESXiでは、CVACがサポートされません。

ブートストラップ構成 ISO の作成

Cisco IOS XRv 9000 ルータは、単一 CD-ROM ドライブ上のプレーン テキスト設定ファイルをサポートします。

- **iosxr_config.txt** : 標準的な XR 設定を提供します。

このテキスト ファイルは、CVAC が自動的に適用されるのに必要な設定 CLI の簡単な一覧を提供します。この操作は、手動で **copy iosxr_config.txt running-config** コマンドを実行するのと機能的には同じです。

1 つ以上の設定ファイルがある場合は、次のコマンドを使用して、Cisco IOS XRv 9000 ルータに挿入するのに適した ISO イメージを作成できます。

```
mkisofs -output bootstrap.iso -l -V config-1 --relaxed-filenames --iso-level 2
iosxr_config.txt
```

次に、Ubuntu で **mkisofs** コマンドを実行した場合のサンプル出力を示します。

```
Warning: creating filesystem that does not conform to ISO-9660.
I: -input-charset not specified, using utf-8 (detected in locale settings)
Total translation table size: 0
Total rockridge attributes bytes: 0
Total directory bytes: 0
Path table size(bytes): 10
Max brk space used 0
175 extents written (0 MB)
```

CVAC と KVM

CVAC が **config** ファイルを処理するようにするには、**Qemu** コマンドラインに付加的なドライブを（最後のドライブとして）追加します。

```
-drive file=./bootstrap.iso,if=virtio,media=cdrom,index=3
```

設定ファイルが適切に提供され、CVAC が正常に実行された場合、以下の **syslog** メッセージが表示されます。

```
RP/0/0/CPU0:Dec 14 09:10:22.719 : config[1]: %MGBL-CONFIG-6-DB_COMMIT : Configuration
committed by user 'CVAC'
Use 'show configuration commit changes 1000000001' to view the changes.
```

```
RP/0/0/CPU0:Dec 14 09:10:23.619 : cvac[2]: %MGBL-CVAC-5-CONFIG_DONE :
```

ファイル **/disk0:/iosxr_config.txt** から設定が適用されました。

config ファイルからのいずれかの設定が拒否された場合、以下の **syslog** メッセージが追加で表示されます。

```
RP/0/0/CPU0:Dec 14 09:10:23.619 : cvac[2]: %MGBL-CVAC-3-CONFIG_ERROR : Errors were
encountered while applying configs from file /etc/sysconfig/iosxr_config.txt. Please
inspect 'show configuration failed' for details
```

失敗しなかった設定はコミットされます。disk0/cvac.log ファイルにさらに詳しいデバッグ情報が含まれています。

CVAC と Virsh

1. Cisco IOS XRv 9000 ルータに挿入するのに適した ISO イメージを作成し、「ブートストラップ構成 ISO の作成」のセクションで説明されている手順に従います。
2. ルータのイメージと共にダウンロードされる Virsh.xml ファイルには、以下に示す Bootstrap セクションがあります。

```
<!-- BootstrapSection -->
<!-- Example Bootstrap CLI ISO -->
<!-- <disk type='file' device='cdrom'> -->
<!-- <driver name='qemu' type='raw'> -->
<!-- <source file='<ISO with file iosxr_config.txt'> -->
<!-- <target dev='vdc' bus='virtio'> -->
<!-- <readonly/> -->
<!-- <alias name='bootstrap_CLI'> -->
<!-- </disk> -->
```

3. Bootstrap セクションをアンコメントし、source file の参照先として、インスタンスを起動するマシン上のブートストラップ ISO ファイルの絶対パスを指定します。次に例を示します。

```
<!--BootstrapSection -->
<disk type='file' device='cdrom'>
<driver name='qemu' type='raw'>
<source file='/production/bootstrap.iso'>
<target dev='vdc' bus='virtio'>
<readonly/>
<alias name='bootstrap_CLI'>
</disk>
```

CVAC と OpenStack Config-Drive

Config-Drive は、OpenStack でオーケストレーションされる VM を初期設定でブートストラップするメカニズムです。OpenStack Config-Drive については、[ここ](#)をクリックしてください。

Config-Drive のサポートが望ましい場合は、以下のコマンドラインを使用して XR 初期設定のプレーンテキスト ファイルを渡すことができます。

```
config-drive true user-data /iosxr_config.txt file /iosxr_config.txt=/iosxr_config.txt
```

ファイル *iosxr_config.txt* は、XR コマンドが含まれている生のテキスト ファイルです。前のセクションで言及されていた、KVM コマンドラインまたは Virsh で使用される ISO ファイルではありません。

起動時の CVAC

構成無しで新規インストールした VM インスタンスの場合、CVAC の動作は簡単なものになります。それは、パーサーを通じて受け入れられた構成がすべてコミットされるというものです。CVAC は、最後に適用した設定ファイルの署名 (CRC) を保持します。

それ以降のリブート時には、システムに渡される CVAC 設定ファイルの CRC と、最後に適用した CVAC 構成の CRC が照合チェックされます。変更がなければ、何も実行されません。つまり、最初の CVAC による構成の適用後、それ以降の構成の変更は、渡される CVAC 設定ファイルに変更がない場合、システム設定が変更されない (または元に戻されない) ことを意味します。

変更があると、新しい構成が既存の構成を上書きするように適用されます。これにより、最初の CVAC 構成、それ以降の構成の変更、および構成済みのシステムに対する CVAC 構成の追加の変更が可能になります。

すでにコミットされた構成上にパーサーによって読み込まれるだけの構成を、CVAC は合理化しようとしません。パーサーを通過するコマンドはコミットされ、前述したようにエラーがログに記録されます。

FlexAlgo の ORR サポート

表 10: 機能の履歴 (表)

機能名	リリース情報	機能説明
FlexAlgo の ORR サポート	リリース 7.5.1	この機能を使用すると、オペレータは、独自のニーズに応じて IGP 最短パス計算をカスタマイズできます。オペレータは、リンクコストベースの SPF よりも優れた転送を実現するために、カスタムの SR プレフィックス SID を割り当てることができます。結果として、フレキシブルアルゴリズムにより、IGP から到達可能なあらゆる宛先へのトラフィック エンジニアリングに基づくパスを IGP で自動的に計算できます。

セグメントルーティングフレキシブルアルゴリズムの有効化

セグメントルーティングフレキシブルアルゴリズムを使用すると、オペレータは、独自のニーズに応じて IGP 最短パス計算をカスタマイズできます。オペレータは、リンクコストベースの SPF よりも優れた転送を実現するために、カスタムの SR プレフィックス SID を割り当てること

とができます。結果として、フレキシブルアルゴリズムにより、IGPから到達可能なあらゆる宛先へのトラフィック エンジニアリングに基づくパスをIGPで自動的に計算できます。

SR アーキテクチャでは、パスの計算方法を定義するアルゴリズムにプレフィックス SID が関連付けられます。フレキシブルアルゴリズムにより、ユーザーが定義したメトリックタイプと制約の組み合わせに基づいてIGPでパスを計算する、ユーザー定義のアルゴリズムを実現できます。

このマニュアルでは、MPLS データプレーンでセグメント ルーティング フレキシブル アルゴリズムをサポートするための IS-IS 拡張機能について説明します。

フレキシブルアルゴリズムの前提条件

フレキシブルアルゴリズム機能をアクティブ化する前に、ルータでセグメントルーティングを有効にする必要があります。

セグメント ルーティング フレキシブル アルゴリズムの構成要素

このセクションでは、IS-IS でSR フレキシブルアルゴリズム機能をサポートする必須の構成要素について説明します。

フレキシブルアルゴリズムの定義

ネットワーク上のパスを計算するには、考えられる多くの制約を使用する必要があります。一部のネットワークでは、複数のプレーンを展開する必要があります。特定のプレーンを使用することが、単純な形の制約になる場合もあります。より洗練された形の制約には、「RFC7810」で説明されているように、遅延など、一部の拡張メトリックが含まれる可能性があります。さらに高度なケースでは、パスを制限し、特定のアフィニティを持つリンクを回避することも考えられます。また、これらを組み合わせて使用することも可能です。最大限の柔軟性を得られるように、アルゴリズム値とその意味の間のマッピングを定義できます。ドメイン内のすべてのルータで、特定のアルゴリズム値が持つ意味について共通の認識が確立されている場合、アルゴリズムの計算は一貫性のあるものとなり、トラフィックがループすることはありません。フレキシブルアルゴリズムでは、標準ではなく、アルゴリズムの意味を定義します。

フレキシブル アルゴリズム メンバーシップ

アルゴリズムは、IGPによるベストパスの計算方法を定義します。ルータは、ノード機能としてアルゴリズムのサポートをアドバタイズします。ルータは、アルゴリズム値を使用してプレフィックスSIDをアドバタイズします。ルータは、プレフィックスSIDをアルゴリズム自体と緊密に結合します。

アルゴリズムは1つのオクテット値です。ルータは、ユーザー定義値用に値128～255を予約します。ルータは、フレキシブルアルゴリズム表現を使用します。

フレキシブルアルゴリズムの定義のアドバタイズメント

特定のフレキシブルアルゴリズムで計算されたパスについてループフリーの転送を実現するためには、ネットワーク内のすべてのルータでフレキシブルアルゴリズムの同じ定義を共有する必要があります。これは、各フレキシブルアルゴリズムの定義をアドバタイズする1つ以上の専用ルータによって実現されます。このようなアドバタイズメントでは、優先度を設定して、フレキシブルアルゴリズムごとに一貫した1つの定義がすべてのルータで適用されるようにします。

フレキシブルアルゴリズムの定義には以下が含まれます。

- メトリックタイプ
- アフィニティ制約

advertise-definition コマンドを使用して、特定のフレキシブルアルゴリズムの定義をルータからアドバタイズできるようにします。エリア内の少なくとも1つのルータ、または可能であれば冗長性を確保するために2つのルータで、フレキシブルアルゴリズム定義をアドバタイズする必要があります。ルータが有効な定義をアドバタイズしない限り、フレキシブルアルゴリズムは機能しません。

フレキシブルアルゴリズム リンク属性アドバタイズメント

ルータは、フレキシブルアルゴリズムパスの計算中にさまざまなリンク属性を使用しました。たとえば、リンクアフィニティに基づく包含ルールまたは除外ルールは、IETF ドラフトで定義されているように、フレキシブルアルゴリズム定義の一部にすることができます。

フレキシブルアルゴリズムの計算中にルータが使用するリンク属性アドバタイズメントでは、RFC8919 (IS-IS) で定義されているように、アプリケーション固有リンク属性 (ASLA) アドバタイズメントを使用する必要があります。IS-IS では、ルータが ASLA アドバタイズメントで L フラグを設定すると、ルータは代わりにレガシーアドバタイズメント (IS-IS 拡張可用性 TLV) を使用します。

ASLA アドバタイズメントの必須使用は、次のリンク属性に適用されます。

- 最小単方向リンク遅延
- TE デフォルトメトリック
- 管理グループ

フレキシブルアルゴリズムのプレフィックス SID のアドバタイズメント

フレキシブルアルゴリズム固有のパスでトラフィックを転送するため、フレキシブルアルゴリズムに参加するすべてのルータは、ルータがプレフィックスに対してアドバタイズするフレキシブルアルゴリズム固有の SID の MPLS ラベル付きパスをインストールします。ルータがアドバタイズするプレフィックスのみ、フレキシブルアルゴリズム固有のプレフィックス SID がフレキシブルアルゴリズム固有の転送の対象となります。

フレキシブルアルゴリズムパスの計算

ルータは、複数のフレキシブルアルゴリズムのパスを計算できます。このようなフレキシブルアルゴリズムのパスを計算する前に、特定のフレキシブルアルゴリズムをサポートするようにルータを設定します。フレキシブルアルゴリズムを使用する前に、フレキシブルアルゴリズムの有効な定義をルータで確立しておく必要があります。

特定のフレキシブルアルゴリズムの最短パスツリーを計算する場合は、次のようなプロセスになります。

- ルータは、このようなフレキシブルアルゴリズムのサポートをアドバタイズしないすべてのノードをトポロジからプルーニングします。
- フレキシブルアルゴリズムの定義に除外されるアフィニティが含まれている場合、ルータは、そのようなアフィニティのいずれかをアドバタイズするすべてのリンクをトポロジからプルーニングします。
- ルータは、フレキシブルアルゴリズム定義の一部であるメトリックを使用します。ルータが特定のリンクのメトリックをアドバタイズしない場合、ルータはトポロジからリンクをプルーニングします。

ルータは、特定のフレキシブルアルゴリズムのループフリー代替 (LFA) パス、TI-LFA バックアップパス、およびマイクロループ回避パスを計算します。ルータは、このようなフレキシブルアルゴリズムのプライマリパスの計算と同じ制約を使用します。これらのパスでは、バックアップパスまたはマイクロループ回避パスを適用するために、フレキシブルアルゴリズム用にアドバタイズされたプレフィックス SID が使用されます。

フレキシブルアルゴリズムのマイクロループ回避の設定

デフォルトでは、フレキシブルアルゴリズムごとのマイクロループ回避インスタンスは、algo-0 のマイクロループ回避設定に従います。

次のコマンドを使用して、フレキシブルアルゴリズムのマイクロループ回避を無効化できます。

```
router isis instance flex-algo algo microloop avoidance disable
router ospf process flex-algo algo microloop avoidance disable
```

フレキシブルアルゴリズムの LFA/TI-LFA の設定

デフォルトでは、フレキシブルアルゴリズムごとの LFA/TI-LFA インスタンスは、algo-0 の LFA/TI-LFA 設定に従います。

次のコマンドを使用して、フレキシブルアルゴリズムの TI-LFA を無効化できます。

```
router isis instance flex-algo algo fast-reroute disable
router ospf process flex-algo algo fast-reroute disable
```

フレキシブルアルゴリズムパスの転送エントリの組み込み

ルータは、ルータがフレキシブルアルゴリズム用にアドバタイズしたプレフィックス SID を使用して、あらゆるプレフィックスに対するフレキシブルアルゴリズムパスを転送に組み込む必要があります。フレキシブルアルゴリズムのプレフィックス SID が不明な場合、ルータはそのプレフィックスの転送にフレキシブルアルゴリズムパスを組み込みません。

ルータはフレキシブルアルゴリズムパスのMPLSからMPLSへのエントリのみを組み込みます。ルータは、IPからIPまたはIPからMPLSへのエントリを組み込みません。これらは、デフォルトのアルゴリズムと通常のIGPメトリックに基づいて計算されたネイティブIGPパスに従います。

フレキシブルアルゴリズムのプレフィックスSIDの再配布

以前は、ルータはIS-ISから別のIS-ISインスタンスまたはプロトコルへのプレフィックスの再配布をSRアルゴリズム0（通常のSPF）プレフィックスSIDに制限していました。ルータは、SRアルゴリズム1（厳格なSPF）およびSRアルゴリズム128～255（フレキシブルアルゴリズム）のプレフィックスSIDをプレフィックスとともに再配布しませんでした。セグメントルーティングIS-ISフレキシブルアルゴリズムのプレフィックスSIDの再配布機能により、IS-ISインスタンスまたはIS-ISプロトコル間で厳格なSPFおよびフレキシブルアルゴリズムのプレフィックスSIDを再配布できます。厳格な、またはフレキシブルアルゴリズムのSIDを使用するIS-ISルートの再配布を設定すると、ルータはこの機能を自動的に有効にします。

フレキシブルアルゴリズムのプレフィックスメトリック

IS-ISの既存のフレキシブルアルゴリズム機能の制限は、リモートエリアまたはリモートIGPドメインのプレフィックスへのベストパスを計算できないことです。ルータは、IS-ISエリア間またはプロトコルドメイン間でプレフィックスをアドバタイズしますが、既存のプレフィックスメトリックは、フレキシブルアルゴリズムパスの制約を反映しません。エリア間への最適なフレキシブルアルゴリズムパスを計算したり、エリア内でプレフィックスを再配布したりできますが、パスは複数のエリアまたはIGPドメインを通過する全体的なベストパスを表していない場合があります。

フレキシブルアルゴリズムプレフィックスメトリック機能は、IS-ISプレフィックスアドバタイズメントにフレキシブルアルゴリズム固有のプレフィックスメトリックを導入します。プレフィックスメトリックは、複数のエリアまたはドメインにわたる最適なエンドツーエンドのフレキシブルアルゴリズム最適化パスを計算する方法を提供します。



- (注) フレキシブルアルゴリズムの定義は、ドメインまたはエリア間で一貫している必要があります。

フレキシブルアルゴリズムの設定

次のIS-ISコンフィギュレーションサブモードを使用して、フレキシブルアルゴリズムを設定します。

```
router isis instance flex-algo algo
```



- (注) 常にIGPメトリックを使用します。遅延またはTEメトリックを有効にした場合、リンク上でアドバタイズされた遅延またはTEメトリックを、フレキシブルアルゴリズム計算のメトリックとして使用します。

```
router isis instance flex-algo algo affinity { include-any | include-all | exclude-any }
    name1, name2, ...
```

name : アフィニティマップの名前

```
router isis instance flex-algo algo priority priority-value
```

priority value : フレキシブルアルゴリズム定義の選択時に使用される優先順位

IS-IS でアドバタイズされるフレキシブルアルゴリズム定義にフレキシブルアルゴリズムプレフィックスメトリックを含めるには、次のコマンドを設定します。

```
router isis instance flex-algo algo prefix-metric
```

IS-IS でのフレキシブルアルゴリズム定義のアドバタイズメントを有効にするには、次のコマンドを設定します。

```
router isis instance flex-algo algo advertise-definition
```

アフィニティの設定

アフィニティマップを定義する際は、次のコマンドを設定します。アフィニティマップは、拡張管理者グループのビットマスク内の特定のビット位置に名前を関連付けます。

```
router isis instance flex-algo algo affinity-map name bit-position bit number
```

- *name* : アフィニティマップの名前
- *bit number* : 拡張管理者グループのビットマスク内のビット位置

アフィニティをインターフェイスに関連付けるには、次のコマンドを設定します。

```
router isis instance interface type interface-path-id affinity flex-algo name 1, name 2, ...
```

name : アフィニティマップの名前

プレフィックス SID アドバタイズメントの設定

デフォルトおよび厳格な SPF のアルゴリズムのプレフィックス SID をアドバタイズするには、次のコマンドを設定します。

```
router isis instance interface type interface-path-id address-family { ipv4 | ipv6 }
[unicast] prefix-sid [strict-spf | algorithm algorithm-number] [index | absolute] sid
value
```

- *Algorithm-number* : フレキシブルアルゴリズム番号
- *SID value* : SID 値

フレキシブルアルゴリズムの定義の設定

フレキシブルアルゴリズム サブモードでフレキシブルアルゴリズム定義を設定するには、次のコマンドを設定します。

```
router isis instance flex-algo algo metric-type { delay | te }
```

例 : IS-IS フレキシブルアルゴリズムの設定

```
Example: Configuring IS-IS Flexible Algorithm
router isis 1
    affinity-map red bit-position 65
```

```

affinity-map blue bit-position 8
affinity-map green bit-position 201

flex-algo 128
  advertise-definition
  affinity exclude-any red
  affinity include-any blue
!
flex-algo 129
  affinity exclude-any green
!
!
address-family ipv4 unicast
  segment-routing mpls
!
interface Loopback0
  address-family ipv4 unicast
    prefix-sid algorithm 128 index 100
    prefix-sid algorithm 129 index 101
!
!
interface GigabitEthernet0/0/0/0
  affinity flex-algo red
!
interface GigabitEthernet0/0/0/1
  affinity flex-algo blue red
!
interface GigabitEthernet0/0/0/2
  affinity flex-algo blue
!

```

PE 上の BGP ルート : カラーベースのステアリング

SR-TE On Demand Next-Hop (ODN) feature can be used to steer the BGP traffic towards the Flexible Algorithm paths.

The following example configuration shows how to setup BGP steering local policy, assuming two router: R1 (2.2.2.2) and R2 (4.4.4.4), in the topology.

Configuration on router R1:

```

vrf Test
address-family ipv4 unicast
  import route-target
    1:150
  !
  export route-policy SET_COLOR_RED_HI_BW
  export route-target
    1:150
  !
!
!
interface Loopback0
ipv4 address 2.2.2.2 255.255.255.255
!
interface Loopback150
vrf Test
ipv4 address 2.2.2.222 255.255.255.255
!
interface TenGigE0/1/0/3/0
description exr1 to cxr1
ipv4 address 10.0.20.2 255.255.255.0
!
extcommunity-set opaque color129-red-igp

```

```

    129
end-set
!
route-policy PASS
    pass
end-policy
!
route-policy SET_COLOR_RED_HI_BW
    set extcommunity color color129-red-igp
    pass
end-policy
!
router isis 1
    is-type level-2-only
    net 49.0001.0000.0000.0002.00
    log adjacency changes
    affinity-map RED bit-position 28
    flex-algo 128
    priority 228
!
address-family ipv4 unicast
    metric-style wide
    advertise link attributes
    router-id 2.2.2.2
    segment-routing mpls
!
interface Loopback0
    address-family ipv4 unicast
    prefix-sid index 2
    prefix-sid algorithm 128 index 282
!
!
interface TenGigE0/1/0/3/0
    point-to-point
    address-family ipv4 unicast
!
!
router bgp 65000
    bgp router-id 2.2.2.2
    address-family ipv4 unicast
!
    address-family vpnv4 unicast
        retain route-target all
!
neighbor-group RR-services-group
    remote-as 65000
    update-source Loopback0
    address-family ipv4 unicast
!
    address-family vpnv4 unicast
!
neighbor 4.4.4.4
    use neighbor-group RR-services-group
!
vrf Test
    rd auto
    address-family ipv4 unicast
        redistribute connected
!
segment-routing
traffic-eng
    logging

```



```
metric-style wide
advertise link attributes
router-id 4.4.4.4
segment-routing mpls
!
interface Loopback0
address-family ipv4 unicast
prefix-sid index 4
prefix-sid algorithm 128 index 284
prefix-sid algorithm 129 index 294
prefix-sid algorithm 130 index 304
!
!
interface GigabitEthernet0/0/0/0
point-to-point
address-family ipv4 unicast
!
!
interface TenGigE0/1/0/1
point-to-point
address-family ipv4 unicast
!
!
router bgp 65000
bgp router-id 4.4.4.4
address-family ipv4 unicast
!
address-family vpnv4 unicast
!
neighbor-group RR-services-group
remote-as 65000
update-source Loopback0
address-family ipv4 unicast
!
address-family vpnv4 unicast
!
!
neighbor 1.1.1.1
use neighbor-group RR-services-group
!
neighbor 2.2.2.2
use neighbor-group RR-services-group
!
vrf Test
rd auto
address-family ipv4 unicast
redistribute connected
!
neighbor 25.1.1.2
remote-as 4
address-family ipv4 unicast
route-policy PASS in
route-policy PASS out
!
!
!
segment-routing
!
end
```

フレキシブルアルゴリズムの設定

フレキシブルアルゴリズムを設定するには、次の ISIS コンフィギュレーションサブモードを使用します。

```
router isis instance flex-algo algo
```

```
router ospf process flex-algo algo
```

algo : 128 ~ 255 の値

フレキシブルアルゴリズムの定義の設定

フレキシブルアルゴリズム サブモードでフレキシブルアルゴリズム定義を設定するには、次のコマンドを使用します。

- **metric-type delay**



(注) デフォルトでは、通常の IGP メトリックが使用されます。遅延メトリックが有効になっている場合、リンク上でアダプタイズされた遅延が、フレキシブルアルゴリズム計算のメトリックとして使用されます。

- **affinity exclude-any name1, name2, ...**

name : アフィニティマップの名前

- **priority priority value**

priority value : フレキシブルアルゴリズム定義の選択時に使用される優先度

IS-IS でのフレキシブルアルゴリズム定義のアダプタイズメントを有効にするには、次のコマンドを使用します。

```
router isis instance flex-algo algo advertise-definition
```

アフィニティの設定

アフィニティマップを定義する際は、次のコマンドを使用します。アフィニティマップは、拡張管理者グループのビットマスク内の特定のビット位置に名前を関連付けます。

```
router isis instance flex-algo algo affinity-map name bit-position bit number
```

```
router ospf process flex-algo algo affinity-map name bit-position bit number
```

- *name* : アフィニティマップの名前
- *bit number* : 拡張管理者グループのビットマスク内のビット位置

アフィニティをインターフェイスに関連付けるには、次のコマンドを使用します。

```
router isis instance interface type interface-path-id affinity flex-algo name 1, name
2, ...

router ospf process area area interface type interface-path-id affinity flex-algo name
1, name 2, ...
```

name : アフィニティマップの名前

プレフィックス SID アドバタイズメントの設定

デフォルトおよび厳格な SPF のアルゴリズムのプレフィックス SID をアドバタイズするには、次のコマンドを使用します。

```
router isis instance interface type interface-path-id address-family {ipv4 | ipv6}
[unicast] prefix-sid [strict-spf | algorithm algorithm-number] [index | absolute] sid
value

router ospf process area area interface Loopback interface-instance prefix-sid [strict-spf
| algorithm algorithm-number] [index | absolute] sid value
```

- *algorithm-number* : フレキシブルアルゴリズム番号
- *sid value* : SID 値

ポリシーベースルーティングを使用した IPv4 および IPv6 トラフィックのリダイレクト

表 11: 機能の履歴 (表)

機能名	リリース情報	機能説明
ポリシーベースルーティング (PBR) を使用した IPv4 および IPv6 トラフィックのリダイレクト	リリース 7.3.3	この機能を使用すると、IPv4 および IPv6 サブスクライバートラフィックを、当初の宛先ではなく、選択した接続先にリダイレクトできます。この機能により、特殊なパスを介して特定のトラフィックを柔軟に割り当てることができ、音声、ビデオ、およびデータを伝送するサービスネットワークを効率的に管理できます。

音声、ビデオ、およびデータを伝送する今日の統合型ネットワークでは、ルーティングプロトコルからのパスを使用するのではなく、特定のパスを介してトラフィックをルーティングする必要性が高まっています。PBR リダイレクト機能は、サブスクライバートラフィックを元の宛先以外の接続先にリダイレクトすることで、このニーズに対応します。これは、ポリシーベース

ルーティング (PBR) 機能の追加機能であり、パケット転送の決定がルーティングプロトコルではなくポリシー設定に基づいています。

たとえば、サービスプロバイダーは、音声トラフィックが特定の専用リンクを通過し、データトラフィックが通常のルーティングパスを通過するようにしたい場合があります。ポリシーベースのリダイレクト機能は、サービスプロバイダーが事前設定済みの一連の一致基準に基づいてトラフィックを IPv4 または IPv6 ネクストホップアドレスにリダイレクトできるメカニズムを提供します。

一般的な注意事項

- PBR リダイレクトは、IPv4 および IPv6 トラフィックタイプをサポートします。
- PBR リダイレクトと GRE 機能は相互に排他的です。
- PBR リダイレクトは、接続先アドレスと送信元アドレスの一致をサポートします。
- ルータは、PBR リダイレクトのネクストホップアドレスに到達できない場合、一致するすべてのトラフィックをドロップします。
- PBR リダイレクトは、次のリダイレクトタイプをサポートします。
 - IPv4 および IPv6 ネクストホップ。
 - 名前付き VRF からデフォルト VRF
 - デフォルト VRF から名前付き VRF
- PBR リダイレクトは VRF スケーリングをサポートします。
- 次のコマンドを使用して、PBR ポリシーおよび関連するクラスマップを表示できます。
 - **show pbr-pal km policy name vnr interface type sw**
 - **show pbr-pal km policy name info location ID**

設定

PBR で ACL を設定するには、次の設定例を使用します。

```
/* Configure an access list */
Router(config)# ipv4 access-list Test
Router(config-ipv4-acl)# 10 permit ipv4 any host 10.1.1.10
Router(config-ipv4-acl)# 20 permit ipv4 any host 10.2.3.4
Router(config-ipv4-acl)# commit
Router(config-ipv4-acl)# exit

/* Configure a class map for the access list */
Router(config)# class-map type traffic match-any Test A
Router(config-cmap)# match access-group ipv4 Test
Router(config-cmap)# end-class-map
Router(config)# commit

/* Configure an PBR policy map with the class map */
Router(config)# policy-map type pbr Test B
```

```
Router(config-pmap)# class type traffic Test A
Router(config-pmap-c)# redirect nexthop 192.168.10.1
Router(config-pmap-c)# exit
Router(config-pmap)# end-policy-map
```

実行コンフィギュレーション

show コマンドを使用して設定を検証します。

```
Router(config)# show running-config
Building configuration...
!! IOS XR Configuration 0.0.0
!! Last configuration change at Mon Nov  6 17:31:23 2017 by UNKNOWN
!
ipv4 access-list Test
 10 permit ipv4 any host 10.1.1.10
 20 permit ipv4 any host 10.2.3.4
!
!
class-map type traffic match-any Test A
 match access-group ipv4 Test
end-class-map
!
!
policy-map type pbr Test B
 class type traffic Test A
  redirect ipv4 nexthop 192.168.10.1
!
!
end-policy-map
!
```

確認

次の show コマンドは、PBR を使用した IPv4 および IPv6 トラフィックリダイレクトのデバッグ性を向上させるのに役立ちます。

```
Router#show pbr-pal km policy policy1 vmr interface GigabitEthernet 0/0/0/0 sw
Wed Feb 16 16:38:03.096 UTC
KM ifname GigabitEthernet0_0_0_0, ul_ifname NULL, policy info name policy1 pnum = 0
ingress_format = 0 egress_format = 1 km policy flags = 0x00001100 class# = 2 ref# = 1
 num_intfs = 1 is_bvi 0 is_typhoon_tomahawk 0 np_start 255 np_end 255
=====
B : type & id      E : ether type    VO : vlan outer   VI : vlan inner
Q : tos/exp/group X : Reserved      DC : discard class Fl : flags
F2: L2 flags      F4: L4 flags     SP/DP: L4 ports
T : IP TTL        D : DFS class#   L : leaf class#
Pl: Protocol      G : QoS Grp     M : V6 hdr ext.   C : VMR count
=====
policy name policy1 and format type 0
Total Ingress TCAM entries: 2
 |B   Q  T  Fl Pl SP  DP  G  IPv4/v6 SA IPv4/v6 DA
=====
V|0006 00 00 00 11 0000 0000 00 AC020400 00000000
M|FF00 00 00 80 FF 0000 0000 00 FFFFFFF0 00000000
R| C=0 D=0 L=0
V|0006 00 00 00 00 0000 0000 00 00000000 00000000
M|FF00 00 00 00 00 0000 0000 00 00000000 00000000
R| C=1 D=1 L=1
```

```

=====
Total Ingress and Egress TCAM entries: 2

Router#show pbr-pal km policy policy1 info location 0/0/CPU0
Wed Feb 16 16:39:44.503 UTC
KM ifname NULL, ul_ifname NULL, policy info name policy1 pnum = 0 ingress_format = 0
egress_format = 1 km policy flags = 0x00001100 class# = 2 ref# = 1 num_intfs = 1 is_bvi
 0 is_typhoon_tomahawk 0 np_start 255 np_end 255KM policy info
    name policy1
      pnum = 0 ingress_format = 0 egress_format = 1 km policy flags = 0x00001100
class# = 2
  ref# = 1 num_intfs = 1
Interface Details
=====
No.   Interface Name                VMR ID   Ingress
1     GigabitEthernet0/0/0/0         1

```

テレメトリ更新の gNMI バンドル

表 12:機能の履歴 (表)

機能名	リリース情報	説明
gNMIバンドルサイズの拡張	リリース 7.8.1	<p>gRPC ネットワーク管理インターフェイス (gNMI) バンドルを使用すると、ルータは同じクライアント向けの複数の gNMI Update メッセージを単一の gNMI Notification メッセージに内部的にバンドルし、インターフェイスを介してクライアントに送信します。</p> <p>クライアントへの単一の Notification メッセージでより多くの gNMI Update に対応することで、インターフェイスの帯域幅使用率を最適化できるようになりました。gNMI バンドルサイズが 32768 バイトから 65536 バイトに増加し、シスコのネイティブデータモデルを介して gNMI バンドルサイズの設定が可能になりました。</p> <p>以前のリリースでは、最大バンドルサイズは 32768 バイトのみであり、CLI を介してのみ設定できました。</p> <p>この機能では、シスコのネイティブデータモデル Cisco-IOS-XR-telemetry-model-driven-cfg.yang に新しい XPath を導入して、gNMI バンドルサイズを設定します。</p> <p>gNMI バンドルの仕様を確認するには、「Github」リポジトリを参照してください。</p>

gNMI インターフェイスを介して送信するバイト数を減らすために、同じクライアントに関連する複数の gNMI Update メッセージがバンドルされてクライアントに送信され、帯域幅の使用率が最適化されます。

ルータは、複数の gNMI Update メッセージを gNMI SubscribeResponse メッセージの単一の gNMI Notification メッセージに内部的にバンドルします。Cisco IOS XR ソフトウェア リリース 7.8.1 は、最大 65536 バイトの gNMI バンドルサイズをサポートします。

ルータは、同じクライアントの複数のインスタンスをバンドルします。たとえば、ルータは、次のパスのインターフェイス MgmtEth0/RP0/CPU0/0、FourHundredGigE0/0/0/0、FourHundredGigE0/0/0/1などをバンドルします。

- Cisco-IOS-XR-infra-statsd-oper:infra-statistics/interfaces/interface/latest/generic-counters

ルータは、異なるクライアントのメッセージを単一の gNMI Notification メッセージにバンドルしません。次の例を参考にしてください。

- Cisco-IOS-XR-infra-statsd-oper:infra-statistics/interfaces/interface/latest/generic-counters
- Cisco-IOS-XR-infra-statsd-oper:infra-statistics/interfaces/interface/latest/protocols

クライアントパスのコンテナの下にあるデータは、異なるバンドルに分割できません。

gNMI Notification メッセージには、イベントが発生したとき、またはサンプルが取得されたときのタイムスタンプが含まれています。バンドルプロセスでは、バンドルされたすべての Update 値に単一のタイムスタンプが割り当てられます。Notification のタイムスタンプは、バンドルの最初のメッセージです。



- (注)
- ON-CHANGE サブスクリプションモードは、gNMI バンドルをサポートしていません。
 - ルータは、次のシナリオではバンドルサイズを適用しません。
 - (N-1) メッセージ処理の最後で、Notification メッセージのサイズが設定されたバンドルサイズよりも小さい場合、ルータは1つの余分なインスタンスを許可し、バンドルサイズを超える可能性があります。
 - バンドルサイズを超える単一インスタンスのデータ。
 - XPath : network-instances/network-instance/afts はバンドルをサポートしていません。

IPv4 GRE トンネルの QoS

表 13: 機能の履歴 (表)

機能名	リリース情報	機能説明
-----	--------	------

IPv4 GRE トンネルの QoS	リリース 7.3.3	この機能により、サービスプロバイダー ネットワーク内のプロバイダーエッジ (PE) ルータ上で、受信カスタマーtraフィックと送信カスタマーtraフィックの両方に関する QoS を定義して制御できるようになります。
--------------------	------------	---

IPv4 GRE トンネルでの QoS サポートを使用すると、IPv4 GRE インターフェイスにポリシーマップを直接適用できます。これにより、トンネルで集約ポリシングとマーキングが有効になります。トンネルの入力側にポリシーを適用して、トンネルに入るペイロードトラフィックをマーキングおよびポリシングできます。QoS は、トンネルから出力されるトラフィックではサポートされません。

GRE トンネルについては、『Cisco ASR 9000 Series Aggregation Services Router MPLS Layer 3 VPN Configuration Guide』の「Implementing Generic Routing Encapsulation」の章を参照してください。

機能制限

GRE トンネルの QoS には、次の制約事項が適用されます。

- IPv4、IPv6、および MPLS 以外の他のペイロードトラフィックはサポートされていません。
- GRE トンネルの QoS は現在、1 つのラインカードと 1 つのネットワークプロセッサのみをサポートしています。
- GRE サービスポリシーの QoS は、出力方向でサポートされています。ラインカードで出力サービスポリシーを適用し、GRE トンネルで入力ポリシーを適用します。
- GRE サービスポリシーの QoS は、入力方向ではサポートされていません。
- GRE トンネルの QoS 機能は、シェーピングおよびキューイングアクションをサポートしていません。
- GRE トンネルの QoS 機能は、マーキングおよび条件付きマーキングアクションをサポートしていません。
- パーセンテージベースのポリシングが設定されているポリシーマップはアタッチできません。
- GRE トンネルの QoS 機能は、ポリシングアクションのみをサポートしています。ポリシーには、シングルレート、2 カラーポリサー (1R2C) を使用できます。
- 適合トラフィックは送信され、超過アクションはドロップされます。

IPv4 GRE トンネルトラフィックの分類

次の表に、分類のための IPv4 GRE トンネルでのさまざまなペイロードトラフィック QoS フィールドのサポートを示します。

QoS フィールド	分類	
	入力	出力
優先順位	非対応	対応
トンネルの優先順位	非対応	非対応
VLAN	非対応	非対応
サービス クラス (CoS)	非対応	非対応
ドロップ適性インジケータ (DEI)	非対応	非対応
IPv4 L3 フィールド	非対応	対応

例

次に、GRE トンネルでのシングルレート 2 カラーポリサー (1R2C) を使用した 2 レベルの階層型ポリシーのアプリケーション例を示します。

```
Router(config)#class-map match-any mydata
Router(config-cmap)#match mpls experimental topmost 3 4
Router(config-cmap)#end-class-map
Router(config)#class-map match-any mycontrol
Router(config-cmap)#match mpls experimental topmost 1 2
Router(config-cmap)#end-class-map
Router(config)#policy-map child
Router(config-pmap)#class mycontrol
Router(config-pmap-c)#police rate 2 gbps
Router(config-pmap-c-police)#conform-action transmit
Router(config-pmap-c-police)#exceed-action transmit
Router(config-pmap-c-police)#class class-default
Router(config-pmap-c)#police rate 1 gbps
Router(config-pmap-c-police)#conform-action transmit
Router(config-pmap-c-police)#exceed-action transmit
Router(config-pmap-c-police)#end-policy-map
Router(config)#policy-map parent_gre
Router(config-pmap)#class class-default
Router(config-pmap-c)#service-policy child
Router(config-pmap-c)#police rate 5 gbps
Router(config-pmap-c-police)#child-conform-aware
Router(config-pmap-c-police)#end-policy-map
Router(config)#interface tunnel-ipl
Router(config-if)#service-policy output parent_gre
Router(config-if)#ipv4 address 12.0.0.1 255.255.255.0
Router(config-if)#tunnel source TenGigE0/0/0/1
Router(config-if)#tunnel destination 15.1.1.2
```

ネットワークスタックへのアクセス

Cisco IOS XR ソフトウェアは、通信を行うネットワーク スタックとして機能します。ここでは、IOS XR のアプリケーションが内部プロセスやサーバ、または外部デバイスとどのように通信を行うかについて説明します。



- (注) フラグメントパケットのドロップを回避するために、MTU 値を 1514 (デフォルト) 以上に設定することを強く推奨します。

Cisco IOS XR の外部との通信

表 14: 機能の履歴 (表)

機能名	リリース情報	説明
Linux ネットワーキングスタックの仮想 IP アドレス	リリース 7.5.4	仮想 IP アドレスを使用すると、RP スイッチオーバーイベント後に、単一の IP アドレスを現在のアクティブ RP に接続できます。さらに、この機能により、ネットワークスタックは、Linux ネットワーキングスタックを使用するサードパーティ製アプリケーションおよび IOS XR アプリケーションの仮想 IP アドレスをサポートできます。

Cisco IOS XR の外部と通信するには、アプリケーションが loopback0 インターフェイスにマップされた fwdintf インターフェイスアドレス、または設定されたギガビットイーサネットインターフェイスアドレスを使用します。IOS XR のさまざまなインターフェイスについては、[Cisco IOS XR Linux シェルでのアプリケーションホスティング \(103 ページ\)](#) を参照してください。

IOS XR の外部のそれぞれ対応するサーバとの IOS XR 通信に iPerf クライアントまたは Chef クライアントを配備するには、インターフェイスアドレスを XR の送信元アドレスとして設定する必要があります。リモートサーバは、IOS XR のそれぞれ対応するクライアントに到達するためにこのルートアドレスを設定する必要があります。

仮想アドレスは、単一の仮想 IP アドレスを使用して gRPC などの管理ネットワークからルータにアクセスするように設定できます。2 つ以上の RP を持つデバイスでは、仮想アドレスは現在アクティブな管理インターフェイスを参照します。この機能は、現在アクティブな RP の情報がなくても、RP フェールオーバー全体で使用できます。これは、Linux パケットパスに適用されます。

ここでは、外部通信の送信元アドレスとしてギガビットイーサネットインターフェイスアドレスを設定する例を示します。

外部通信用のギガビットイーサネットインターフェイスの使用方法

外部通信用の IOS XR に GigE インターフェイスを設定するには、次の手順を実行します。

1. GigE インターフェイスを設定します。

```
RP/0/RP0/CPU0:ios(config)# interface GigabitEthernet 0/0/0/1
RP/0/RP0/CPU0:ios(config-if)# ipv4 address 192.57.43.10 255.255.255.0
RP/0/RP0/CPU0:ios(config-if)# no shut
RP/0/RP0/CPU0:ios(config-if)# commit
Fri Oct 30 07:51:14.785 UTC
RP/0/RP0/CPU0:ios(config-if)# exit
RP/0/RP0/CPU0:ios(config)# exit
```

2. 設定されたインターフェイスがIOSXRで起動しており、動作可能かどうかを確認します。

```
RP/0/RP0/CPU0:ios# show ipv4 interface brief
Fri Oct 30 07:51:48.996 UTC
```

Interface	IP-Address	Status	Protocol
Loopback0	1.1.1.1	Up	Up
Loopback1	8.8.8.8	Up	Up
GigabitEthernet0/0/0/0	192.164.168.10	Up	Up
GigabitEthernet0/0/0/1	192.57.43.10	Up	Up
GigabitEthernet0/0/0/2	unassigned	Shutdown	Down
MgmtEth0/RP0/CPU0/0	192.168.122.197	Up	Up

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

3. Linux bash シェルに移動し、設定されているインターフェイスが稼働しているかどうかを確認します。

```
/* If you are using Cisco IOS XR Version 6.0.0, run the following command */
RP/0/RP0/CPU0:ios# run ip netns exec tpnns bash
```

```
/* If you are using Cisco IOS XR Version 6.0.2, run the following command */
RP/0/RP0/CPU0:ios# bash
```

```
[xr-vm_node0_RP0_CPU0:~]$ ifconfig
Gi0_0_0_0 Link encap:Ethernet HWaddr 52:46:04:87:19:3c
inet addr:192.164.168.10 Mask:255.255.255.0
inet6 addr: fe80::5046:4ff:fe87:193c/64 Scope:Link
UP RUNNING NOARP MULTICAST MTU:1514 Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:3 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:0 (0.0 B) TX bytes:210 (210.0 B)

Gi0_0_0_1 Link encap:Ethernet HWaddr 52:46:2e:49:f6:ff
inet addr:192.57.43.10 Mask:255.255.255.0
inet6 addr: fe80::5046:2eff:fe49:f6ff/64 Scope:Link
UP RUNNING NOARP MULTICAST MTU:1514 Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:3 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:0 (0.0 B) TX bytes:210 (210.0 B)

Mg0_RP0_CPU0_0 Link encap:Ethernet HWaddr 52:46:12:7a:88:41
inet addr:192.168.122.197 Mask:255.255.255.0
inet6 addr: fe80::5046:12ff:fe7a:8841/64 Scope:Link
UP RUNNING NOARP MULTICAST MTU:1514 Metric:1
RX packets:3 errors:0 dropped:0 overruns:0 frame:0
TX packets:6 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:294 (294.0 B) TX bytes:504 (504.0 B)
```

```

fwd_ew    Link encap:Ethernet HWaddr 00:00:00:00:00:0b
          inet6 addr: fe80::200:ff:fe00:b/64 Scope:Link
          UP RUNNING NOARP MULTICAST MTU:1500 Metric:1
          RX packets:4 errors:0 dropped:0 overruns:0 frame:0
          TX packets:6 errors:0 dropped:1 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:392 (392.0 B) TX bytes:532 (532.0 B)

fwdintf   Link encap:Ethernet HWaddr 00:00:00:00:00:0a
          inet6 addr: fe80::200:ff:fe00:a/64 Scope:Link
          UP RUNNING NOARP MULTICAST MTU:1482 Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2 errors:0 dropped:1 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B) TX bytes:140 (140.0 B)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1 Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING MTU:1500 Metric:1
          RX packets:8 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:672 (672.0 B) TX bytes:672 (672.0 B)

lo:0      Link encap:Local Loopback
          inet addr:1.1.1.1 Mask:255.255.255.255
          UP LOOPBACK RUNNING MTU:1500 Metric:1

```

- Linux bash シェルを終了し、外部通信の送信元アドレスとして GigE インターフェイスを設定します。

```

[xr-vm_node0_RP0_CPU0:~]$ exit

RP/0/RP0/CPU0:ios# config
Fri Oct 30 08:55:17.992 UTC
RP/0/RP0/CPU0:ios(config)# tpa address-family ipv4 update-source gigabitEthernet
0/0/0/1
RP/0/RP0/CPU0:ios(config)# commit
Fri Oct 30 08:55:38.795 UTC

```



- (注) デフォルトでは、fwdintf インターフェイスが外部通信用の loopback0 インターフェイスにマップします。これは、Loopback0 インターフェイスへのルーティングプロセスまたはルータ ID のバインドに似ています。tpa address-family ipv4 update-source コマンドを使用してギガビットイーサネットインターフェイスに fwdintf インターフェイスをバインドすると、インターフェイスがダウンした場合にネットワーク接続が影響を受けます。

- Linux bash シェルを入力し、GigE インターフェイス アドレスが外部通信用に fwdintf インターフェイスによって使用されるかどうかを確認します。

```

/* If you are using Cisco IOS XR Version 6.0.0, run the following command */
RP/0/RP0/CPU0:ios# run ip netns exec tpnns bash

/* If you are using Cisco IOS XR Version 6.0.2, run the following command */
RP/0/RP0/CPU0:ios# bash

```

```
[xr-vm_node0_RP0_CPU0:~]$ ip route
default dev fwdintf scope link src 192.57.43.10
8.8.8.8 dev fwd_ew scope link
192.168.122.0/24 dev Mg0_RP0_CPU0_0 proto kernel scope link src 192.168.122.197
[xr-vm_node0_RP0_CPU0:~]$
```

IOS XR で外部通信が正常にイネーブルになります。

サードパーティ製アプリケーションの水平方向通信

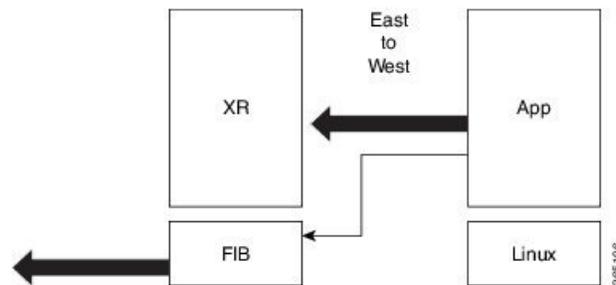
IOS XR での水平方向通信は、コンテナでホストされているアプリケーションがネイティブの XR アプリケーション（XR コントロールプレーン上でホスティング）と対話するためのメカニズムです。

次の図に、IOS XR でホストされているサードパーティ製アプリケーションが XR コントロールプレーンとどのように対話するかを示します。

アプリケーションが IOS XR Forwarding Information Base (FIB) にデータを送信します。アプリケーションは、IOS XR の東側でホストされ、XR コントロールプレーンは西側の地域にあります。したがって、サードパーティ製アプリケーションと XR コントロールプレーン間の通信の形式は水平方向（E-W）通信と呼ばれます。

Chef Client や Puppet Agent などのサードパーティ製アプリケーションは、この通信モードを使用して、IOS XR 上でコンテナ、パケット、およびアプリケーションを設定し、管理します。今後、このサポートが IOS XR に拡張されて、このようなサードパーティ製アプリケーションによって設定、制御される可能性があります。

図 6: IOS XR での水平方向の通信



IOS XR と通信するサードパーティ製アプリケーションでは、Loopback1 インターフェイスを設定する必要があります。次の手順でこの設定方法を説明します。

1. IOS XR の Loopback1 インターフェイスを設定します。

```
RP/0/RP0/CPU0:ios(config)# interface Loopback1
RP/0/RP0/CPU0:ios(config-if)# ipv4 address 8.8.8.8/32
RP/0/RP0/CPU0:ios(config-if)# no shut
RP/0/RP0/CPU0:ios(config-if)# commit
RP/0/RP0/CPU0:ios(config-if)# exit
RP/0/RP0/CPU0:ios(config)#
```

2. Loopback1 インターフェイスの作成を確認します。

```
RP/0/RP0/CPU0:ios# show ipv4 interface brief
Thu Nov 12 10:01:00.874 UTC

Interface                IP-Address      Status          Protocol
Loopback0                1.1.1.1        Up              Up
Loopback1                8.8.8.8       Up             Up
GigabitEthernet0/0/0/0   192.164.168.10 Up              Up
GigabitEthernet0/0/0/1   192.57.43.10   Up              Up
GigabitEthernet0/0/0/2   unassigned     Shutdown       Down
MgmtEth0/RP0/CPU0/0     192.168.122.197 Up              Up
RP/0/RP0/CPU0:ios#
```

3. ネットワークに使用している IOS XR のバージョンに応じて、サードパーティのネットワーク名前空間またはグローバル VRF を入力します。

```
/* If you are using Cisco IOS XR Version 6.0.0, run the following command */
RP/0/RP0/CPU0:ios# run ip netns exec tpnns bash
```

```
/* If you are using Cisco IOS XR Version 6.0.2, run the following command */
RP/0/RP0/CPU0:ios# bash
```

4. Loopback1 インターフェイスアドレスが E-W インターフェイスにマップされているかどうかを確認します。

```
[xr-vm_node0_RP0_CPU0:~]$ ip route
default dev fwdintf scope link src 192.57.43.10
8.8.8.8 dev fwd_ew scope link
192.168.122.0/24 dev Mg0_RP0_CPU0_0 proto kernel scope link src 192.168.122.197
[xr-vm_node0_RP0_CPU0:~]$
```

Cisco IOS XR Linux シェルでのアプリケーションホスティング

Linux は、システム管理者、開発者、およびネットワーク エンジニアが過去 20 ～ 30 年にわたって作成し、テストし、導入してきたアプリケーションやツールのエコシステム全体をサポートします。Linux は、アプリケーションの有無を問わず、安定性、セキュリティ、拡張性、低コストのライセンス、特定のインフラストラクチャのニーズに合わせたアプリケーションのカスタマイズを実現する柔軟性により、サーバのホスティングに適しています。

自動化と統合の簡易性に重点を置く DevOps 形式のワークフローへの注目が高まる中、ネットワーク デバイスは進化し、自動化プロセスをより簡単にする標準的なツールやアプリケーションをサポートする必要があります。標準化された共有ツールのチェーンはスピード、効率、コラボレーションを強化できます。IOS XR は Yocto ベースの Wind River Linux 7 ディストリビューションから開発されています。OS は RPM ベースとなっており、組み込み型システムに最適です。

IOS XR によって、ボックス上での 64 ビット Linux アプリケーションのホスティングが可能となり、次の利点が得られます。

- 設定管理アプリケーションとのシームレスな統合
- ファイル システムへの容易なアクセス

- 操作の簡易性

IOS XR の Linux アプリケーションをホストするには、XR の Linux シェルを理解している必要があります。

一般的な Linux OS は、ネットワーク インターフェイスと OS で共有されるルーティング テーブルエントリ式を提供します。ネットワークの名前空間の導入により、Linux は独立して機能する複数のインスタンスのネットワーク インターフェイスとルーティング テーブルを提供します。



- (注) ネットワークの名前空間のサポートは、Linux OS のディストリビューションによって異なります。アプリケーションのホスティングに使用するディストリビューションがネットワークの名前空間をサポートしていることを確認します。

IOS XR のネットワーク名前空間

ネットワークで使用している Cisco IOS XR のバージョンに応じて、IOS XR Linux シェルにアクセスする方法は 2 つあります。

- Cisco IOS XR バージョン 6.0.0 を使用している場合は、[Cisco IOS XR Linux シェルでのサードパーティのネットワーク名前空間へのアクセス \(108 ページ\)](#) の手順を使用する必要があります。XR Linux シェルにアクセスすると、デフォルトのネットワーク名前空間である XRNNS に移動します。すべてのサードパーティアプリケーション インターフェイスが存在するサードパーティのネットワーク名前空間 (TPNNS) にアクセスするには、この名前空間から移動する必要があります。XR ルータプロンプトでアクセスして表示できるものと、XR Linux シェルでアクセスして表示できるものには違いがあります。
- Cisco IOS XR バージョン 6.0.2 以降を使用している場合は、[Cisco IOS XR Linux シェルでのグローバル VRF へのアクセス \(104 ページ\)](#) の手順を使用する必要があります。XR Linux シェルにアクセスすると、グローバル VRF に名前が変更されたサードパーティのネットワーク名前空間に直接移動します。XR ルータプロンプト自体で `bash` コマンドを実行して、グローバル VRF に保存されているインターフェイスと IP アドレスを表示できます。このバージョンの IOS XR では、ナビゲーションがより高速で直感的になりました。

Cisco IOS XR Linux シェルでのグローバル VRF へのアクセス

Cisco IOS XR バージョン 6.0.2 以降では、サードパーティのネットワーク名前空間 (TPNNS) の名前がグローバル VRF (`global-vrf`) に変更されています。Cisco IOS XR Linux シェルにアクセスする場合は、グローバル VRF を直接入力します。これについては、次の手順で説明します。

1. Linux ボックスから SSH を使用して IOS XR コンソールにアクセスし、ログインします。

```
cisco@host:~$ ssh root@192.168.122.188
root@192.168.122.188's password:
RP/0/RP0/CPU0:ios#
```

IOS XR ルータのプロンプトが表示されます。

2. IOS XR のイーサネット インターフェイスを表示します。

```
RP/0/0/CPU0:ios# show ipv4 interface brief
...
```

Interface	IP-Address	Status	Protocol
Loopback0	1.1.1.1/32	Up	Up
GigabitEthernet0/0/0/0	10.1.1.1/24	Up	Up
...			

```
RP/0/RP0/CPU0:ios# show interfaces gigabitEthernet 0/0/0/0
...
```

```
GigabitEthernet0/0/0/0 is up, line protocol is up
Interface state transitions: 4
Hardware is GigabitEthernet, address is 5246.e8a3.3754 (bia
5246.e8a3.3754)
Internet address is 10.1.1.1/24
MTU 1514 bytes, BW 1000000 Kbit (Max: 1000000 Kbit)
reliability 255/255, txload 0/255, rxload 0/255
Encapsulation ARPA,
Duplex unknown, 1000Mb/s, link type is force-up
output flow control is off, input flow control is off
loopback not set,
Last link flapped 01:03:50
ARP type ARPA, ARP timeout 04:00:00
Last input 00:38:45, output 00:38:45
Last clearing of "show interface" counters never
5 minute input rate 0 bits/sec, 0 packets/sec
5 minute output rate 0 bits/sec, 0 packets/sec
12 packets input, 1260 bytes, 0 total input drops
0 drops for unrecognized upper-level protocol
Received 2 broadcast packets, 0 multicast packets
0 runts, 0 giants, 0 throttles, 0 parity
0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
12 packets output, 1224 bytes, 0 total output drops
Output 1 broadcast packets, 0 multicast packets
```

出力には、GigabitEthernet0/0/0/0 インターフェイスの IP アドレスと MAC アドレスが表示されます。

3. bash -c ifconfig コマンドを実行してネットワーク インターフェイスを表示し、bash コマンドがグローバル VRF で実行されているかどうかを確認します。

```
RP/0/RP0/CPU0:ios# bash -c ifconfig
...
```

```
Gi0_0_0_0 Link encap:Ethernet HWaddr 52:46:04:87:19:3c
inet addr:192.164.168.10 Mask:255.255.255.0
inet6 addr: fe80::5046:4ff:fe87:193c/64 Scope:Link
UP RUNNING NOARP MULTICAST MTU:1514 Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:3 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:0 (0.0 B) TX bytes:210 (210.0 B)

Mg0_RP0_CPU0_0 Link encap:Ethernet HWaddr 52:46:12:7a:88:41
inet addr:192.168.122.197 Mask:255.255.255.0
inet6 addr: fe80::5046:12ff:fe7a:8841/64 Scope:Link
UP RUNNING NOARP MULTICAST MTU:1514 Metric:1
```

```

RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:3 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:0 (0.0 B) TX bytes:210 (210.0 B)

fwd_ew Link encap:Ethernet HWaddr 00:00:00:00:00:0b
inet6 addr: fe80::200:ff:fe00:b/64 Scope:Link
UP RUNNING NOARP MULTICAST MTU:1500 Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:2 errors:0 dropped:1 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:0 (0.0 B) TX bytes:140 (140.0 B)

fwdintf Link encap:Ethernet HWaddr 00:00:00:00:00:0a
inet6 addr: fe80::200:ff:fe00:a/64 Scope:Link
UP RUNNING NOARP MULTICAST MTU:1500 Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:2 errors:0 dropped:1 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:0 (0.0 B) TX bytes:140 (140.0 B)

lo Link encap:Local Loopback
inet addr:127.0.0.1 Mask:255.0.0.0
inet6 addr: ::1/128 Scope:Host
UP LOOPBACK RUNNING MTU:65536 Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:0
RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)

lo:0 Link encap:Local Loopback
inet addr:1.1.1.1 Mask:255.255.255.255
UP LOOPBACK RUNNING MTU:1500 Metric:1

```

次の2つのインターフェイスが存在することで、グローバル VRF に移動していることを確認できます。

`fwd_ew` はサードパーティ製アプリケーションと IOS XR 間の通信（水平方向）に使用するインターフェイスです。

`fwdintf` は IOS XR の外部のサードパーティ製アプリケーションとネットワーク間の通信に使用するインターフェイスです。

4. `bash` コマンドを実行して、Linux シェルにアクセスします。

```

RP/0/RP0/CPU0:ios# bash
Tue Aug 02 13:44:07.627 UTC
[xr-vm_node0_RP0_CPU0:~]$

```

5. （任意）`fwd_ew` インターフェイスと `fwdintf` インターフェイスで使用される IP ルートを表示します。

```

[xr-vm_node0_RP0_CPU0:~]$ ip route
default dev fwdintf scope link src 1.1.1.1
8.8.8.8 dev fwd_ew scope link
192.168.122.0/24 dev Mg0_RP0_CPU0_0 proto kernel scope link src 192.168.122.213

```

IOS XR でグローバル VRF を入力する代替方法

IOS XR へのログイン時に **bash** コマンドを入力せずにグローバル VRF に直接移動するには、次に示すステップで説明する **sshd_operns** サービスを使用します。この手順には、サービスにアクセスするための非ルートユーザの作成が含まれています（ルートユーザはこのサービスにアクセスできません）。



- (注) IOS XR で、インターフェイスをバインドするサービスを開始する前に、インターフェイスが設定され、起動され、動作可能であることを確認します。

インターフェイスが設定された後にのみサービスを開始するには、サービススクリプトに次の関数を含めます。

```
./etc/init.d/operns-functions
operns_wait_until_ready
```

operns_wait_until_ready 関数を追加することによって、サービススクリプトは、1つ以上のインターフェイスが設定されるのを待機してから、サービスを開始するようになります。

1. (任意) リロード時に **operns** サービスを自動的に開始するには、**sshd_operns** サービスを追加し、そのサービスの存在を確認します。

```
bash-4.3# chkconfig --add sshd_operns
bash-4.3# chkconfig --list sshd_operns
sshd_operns    0:off  1:off  2:off  3:on   4:on   5:on   6:off
bash-4.3#
```

2. **sshd_operns** サービスを開始します。

```
bash-4.3# service sshd_operns start
Generating SSH1 RSA host key: [ OK ]
Generating SSH2 RSA host key: [ OK ]
Generating SSH2 DSA host key: [ OK ]
generating ssh ECDSA key...
Starting sshd: [ OK ]
```

```
bash-4.3# service sshd_operns status
sshd (pid 6224) is running...
```

3. ステップ1で作成した非ルートユーザーとして **sshd_operns** セッションにログインします。

```
host@fe-ucs36:~$ ssh devops@192.168.122.222 -p 57722
devops@192.168.122.222's password:
Last login: Tue Sep  8 20:14:11 2015 from 192.168.122.1
XR-vm_node0_RP0_CPU0:~$
```

4. ネットワーク インターフェイスを表示して、グローバル VRF に移動しているかどうかを確認します。

```
[XR-vm_node0_RP0_CPU0:~]$ ifconfig
Gi0_0_0_0 Link encap:Ethernet HWaddr 52:46:04:87:19:3c
inet addr:192.164.168.10 Mask:255.255.255.0
inet6 addr: fe80::5046:4ff:fe87:193c/64 Scope:Link
UP RUNNING NOARP MULTICAST MTU:1514 Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:3 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:0 (0.0 B) TX bytes:210 (210.0 B)
```

```

Mg0_RP0_CPU0_0 Link encap:Ethernet HWaddr 52:46:12:7a:88:41
  inet addr:192.168.122.197 Mask:255.255.255.0
  inet6 addr: fe80::5046:12ff:fe7a:8841/64 Scope:Link
  UP RUNNING NOARP MULTICAST MTU:1514 Metric:1
  RX packets:0 errors:0 dropped:0 overruns:0 frame:0
  TX packets:3 errors:0 dropped:0 overruns:0 carrier:0
  collisions:0 txqueuelen:1000
  RX bytes:0 (0.0 B) TX bytes:210 (210.0 B)

fwd_ew Link encap:Ethernet HWaddr 00:00:00:00:00:0b
  inet6 addr: fe80::200:ff:fe00:b/64 Scope:Link
  UP RUNNING NOARP MULTICAST MTU:1500 Metric:1
  RX packets:0 errors:0 dropped:0 overruns:0 frame:0
  TX packets:2 errors:0 dropped:1 overruns:0 carrier:0
  collisions:0 txqueuelen:1000
  RX bytes:0 (0.0 B) TX bytes:140 (140.0 B)

fwdintf Link encap:Ethernet HWaddr 00:00:00:00:00:0a
  inet6 addr: fe80::200:ff:fe00:a/64 Scope:Link
  UP RUNNING NOARP MULTICAST MTU:1482 Metric:1
  RX packets:0 errors:0 dropped:0 overruns:0 frame:0
  TX packets:2 errors:0 dropped:1 overruns:0 carrier:0
  collisions:0 txqueuelen:1000
  RX bytes:0 (0.0 B) TX bytes:140 (140.0 B)

lo Link encap:Local Loopback
  inet addr:127.0.0.1 Mask:255.0.0.0
  inet6 addr: ::1/128 Scope:Host
  UP LOOPBACK RUNNING MTU:1500 Metric:1
  RX packets:0 errors:0 dropped:0 overruns:0 frame:0
  TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
  collisions:0 txqueuelen:0
  RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)

lo:0 Link encap:Local Loopback
  inet addr:1.1.1.1 Mask:255.255.255.255
  UP LOOPBACK RUNNING MTU:1500 Metric:1

```

これで、アプリケーションをホストするために IOS XR Linux シェルを使用する準備が整いました。

Cisco IOS XR Linux シェルでのサードパーティのネットワーク名前空間へのアクセス

Cisco IOS XR の Linux シェルは、サードパーティ製アプリケーションと内部 XR プロセス間に必要な隔離を実装すると同時に、XR インターフェイスへの必要なアクセスをアプリケーションに提供するサードパーティのネットワーク名前空間 (TPNNS) を提供します。このセクションで説明する手順を使用して、IOS XR Linux シェルにアクセスし、XRNNS (デフォルトの XR ネットワーク名前空間) と TPNNS を移動できます。



(注) この手順は、Cisco IOS XR バージョン 5.3.2 および 6.0.0 にのみ適用されます。他のバージョンの Cisco IOS XR でこの名前空間にアクセスするには、[Cisco IOS XR Linux シェルでのグローバル VRF へのアクセス \(104 ページ\)](#) を参照してください。

次のステップに従って、XR Linux シェルを移動します。

1. Linux ボックスから SSH を使用して IOS XR コンソールにアクセスし、ログインします。

```
cisco@host:~$ ssh root@192.168.122.188
root@192.168.122.188's password:
RP/0/RP0/CPU0:ios#
```

IOS XR ルータのプロンプトが表示されます。

2. IOS XR のイーサネット インターフェイスを表示します。

```
RP/0/0/CPU0:ios# show ipv4 interface brief
...
```

Interface	IP-Address	Status	Protocol
Loopback0	1.1.1.1/32	Up	Up
GigabitEthernet0/0/0/0	10.1.1.1/24	Up	Up
...			

```
RP/0/RP0/CPU0:ios# show interfaces gigabitEthernet 0/0/0/0
...
```

```
GigabitEthernet0/0/0/0 is up, line protocol is up
Interface state transitions: 4
Hardware is GigabitEthernet, address is 5246.e8a3.3754 (bia
5246.e8a3.3754)
Internet address is 10.1.1.1/24
MTU 1514 bytes, BW 1000000 Kbit (Max: 1000000 Kbit)
reliability 255/255, txload 0/255, rxload 0/255
Encapsulation ARPA,
Duplex unknown, 1000Mb/s, link type is force-up
output flow control is off, input flow control is off
loopback not set,
Last link flapped 01:03:50
ARP type ARPA, ARP timeout 04:00:00
Last input 00:38:45, output 00:38:45
Last clearing of "show interface" counters never
5 minute input rate 0 bits/sec, 0 packets/sec
5 minute output rate 0 bits/sec, 0 packets/sec
12 packets input, 1260 bytes, 0 total input drops
0 drops for unrecognized upper-level protocol
Received 2 broadcast packets, 0 multicast packets
0 runts, 0 giants, 0 throttles, 0 parity
0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
12 packets output, 1224 bytes, 0 total output drops
Output 1 broadcast packets, 0 multicast packets
```

出力には、GigabitEthernet0/0/0/0 インターフェイスの IP アドレスと MAC アドレスが表示されます。

3. run コマンドを入力し、IOS XR の Linux bash シェルを起動します。

また、bash プロンプトの表示時に IOS XR のバージョンも確認します。

```
RP/0/RP0/CPU0:ios# run
Wed Oct 28 18:45:56.168 IST

[xr-vm_node0_RP0_CPU0:~]$ uname -a
Linux xr-vm_node0_RP0_CPU0 3.10.19-WR7.0.0.2_standard #1 SMP Mon Jul 6
13:38:23 PDT 2015 x86_64 GNU/Linux
[xr-vm_node0_RP0_CPU0:~]$
```



(注) Linux bash セルを終了し、IOS XR コンソールを起動するには、**exit** コマンドを入力します。

```
[xr-vm_node0_RP0_CPU0:~]$ exit
exit
RP/0/RP0/CPU0:ios#
```

4. **ifconfig** コマンドを実行してネットワーク インターフェイスを見つけます。

```
[xr-vm_node0_RP0_CPU0:~]$ ifconfig
eth0      Link encap:Ethernet  HWaddr 52:46:12:7a:88:41
          inet6 addr: fe80::5046:12ff:fe7a:8841/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:8996  Metric:1
          RX packets:280 errors:0 dropped:0 overruns:0 frame:0
          TX packets:160 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:31235 (30.5 KiB)  TX bytes:20005 (19.5 KiB)

eth-vf0   Link encap:Ethernet  HWaddr 52:54:00:34:29:44
          inet addr:10.11.12.14  Bcast:10.11.12.255  Mask:255.255.255.0
          inet6 addr: fe80::5054:ff:fe34:2944/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:9000  Metric:1
          RX packets:19 errors:0 dropped:0 overruns:0 frame:0
          TX packets:13 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1566 (1.5 KiB)  TX bytes:1086 (1.0 KiB)

eth-vf1   Link encap:Ethernet  HWaddr 52:54:00:ee:f7:68
          inet6 addr: fe80::5054:ff:feee:f768/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:9000  Metric:1
          RX packets:326483 errors:0 dropped:3 overruns:0 frame:0
          TX packets:290174 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:24155455 (23.0 MiB)  TX bytes:215862857 (205.8 MiB)

eth-vf1.1794 Link encap:Ethernet  HWaddr 52:54:01:5c:55:8e
          inet6 addr: fe80::5054:1ff:fe5c:558e/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:8996  Metric:1
          RX packets:10 errors:0 dropped:0 overruns:0 frame:0
          TX packets:13 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:728 (728.0 B)  TX bytes:1234 (1.2 KiB)

eth-vf1.3073 Link encap:Ethernet  HWaddr e2:3a:dd:0a:8c:06
          inet addr:192.0.0.4  Bcast:192.255.255.255  Mask:255.0.0.0
          inet6 addr: fe80::e03a:dfff:fe0a:8c06/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:8996  Metric:1
          RX packets:317735 errors:0 dropped:3560 overruns:0 frame:0
          TX packets:257881 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:18856325 (17.9 MiB)  TX bytes:204552163 (195.0 MiB)

eth-vf1.3074 Link encap:Ethernet  HWaddr 4e:41:50:00:10:01
          inet addr:172.0.16.1  Bcast:172.255.255.255  Mask:255.0.0.0
          inet6 addr: fe80::4c41:50ff:fe00:1001/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:8996  Metric:1
          RX packets:8712 errors:0 dropped:0 overruns:0 frame:0
          TX packets:32267 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:723388 (706.4 KiB)  TX bytes:11308374 (10.7 MiB)

lo        Link encap:Local Loopback
```

```

inet addr:127.0.0.1 Mask:255.0.0.0
inet6 addr: ::1/128 Scope:Host
UP LOOPBACK RUNNING MTU:65536 Metric:1
RX packets:1635360 errors:0 dropped:0 overruns:0 frame:0
TX packets:1635360 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:0
RX bytes:182532711 (174.0 MiB) TX bytes:182532711 (174.0 MiB)

```

```

tap123 Link encap:Ethernet HWaddr c6:13:74:4b:dc:e3
inet6 addr: fe80::c413:74ff:fe4b:dce3/64 Scope:Link
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:13 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:500
RX bytes:0 (0.0 B) TX bytes:998 (998.0 B)

```

出力には、IOS XR が使用する内部インターフェイス (eth0 ~ eth-vf1.3074) が表示されます。これらのインターフェイスは、XR ネットワークの名前空間 (XRNNS) にあり、IOS XR の外部のネットワークとやり取りすることはありません。IOS XR の外部のネットワークとやり取りするインターフェイスは、サードパーティ製ネットワークの名前空間 (TPNNS) にあります。

5. IOS XR の bash シェルで TPNNS に移動します。

```
[XR-vm_node0_RP0_CPU0:~]$ ip netns exec tpnns bash
```

6. TPNNS インターフェイスを表示します。

```

[XR-vm_node0_RP0_CPU0:~]$ ifconfig
Gi0_0_0_0 Link encap:Ethernet HWaddr 52:46:04:87:19:3c
inet addr:192.164.168.10 Mask:255.255.255.0
inet6 addr: fe80::5046:4ff:fe87:193c/64 Scope:Link
UP RUNNING NOARP MULTICAST MTU:1514 Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:3 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:0 (0.0 B) TX bytes:210 (210.0 B)

Mg0_RP0_CPU0_0 Link encap:Ethernet HWaddr 52:46:12:7a:88:41
inet addr:192.168.122.197 Mask:255.255.255.0
inet6 addr: fe80::5046:12ff:fe7a:8841/64 Scope:Link
UP RUNNING NOARP MULTICAST MTU:1514 Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:3 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:0 (0.0 B) TX bytes:210 (210.0 B)

 fwd_ew Link encap:Ethernet HWaddr 00:00:00:00:00:0b
inet6 addr: fe80::200:ff:fe00:b/64 Scope:Link
UP RUNNING NOARP MULTICAST MTU:1500 Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:2 errors:0 dropped:1 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:0 (0.0 B) TX bytes:140 (140.0 B)

 fwdintf Link encap:Ethernet HWaddr 00:00:00:00:00:0a
inet6 addr: fe80::200:ff:fe00:a/64 Scope:Link
UP RUNNING NOARP MULTICAST MTU:1482 Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:2 errors:0 dropped:1 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:0 (0.0 B) TX bytes:140 (140.0 B)

```

```

lo          Link encap:Local Loopback
            inet addr:127.0.0.1  Mask:255.0.0.0
            inet6 addr: ::1/128 Scope:Host
            UP LOOPBACK RUNNING  MTU:1500  Metric:1
            RX packets:0 errors:0 dropped:0 overruns:0 frame:0
            TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:0
            RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

lo:0       Link encap:Local Loopback
            inet addr:1.1.1.1  Mask:255.255.255.255
            UP LOOPBACK RUNNING  MTU:1500  Metric:1

```

出力に表示されるインターフェイスは Linux 環境での IOS XR インターフェイスの複製です (同じ MAC アドレスと IP アドレスを備えています)。

- Gi0_0_0_0 は IOS XR GigabitEthernet 0/0/0/0 インターフェイスです。
- Mg0_RP0_CPU0_0 は XR での管理操作に使用する IOS XR 管理インターフェイスです。
- fwd_ew はサードパーティ製アプリケーションと IOS XR 間の通信 (水平方向) に使用するインターフェイスです。
- fwdintf は IOS XR の外部のサードパーティ製アプリケーションとネットワーク間の通信に使用するインターフェイスです。
- lo:0 は fwdintf インターフェイスを通じたサードパーティ製アプリケーションと外部ネットワーク間の通信に使用する IOS XR loopback0 インターフェイスです。loopback0 インターフェイスは、XR の外部の通信に使用できるように設定する必要があります。また、[Cisco IOS XR の外部との通信 \(99 ページ\)](#) の項で説明したように、アプリケーションは外部通信用の GigE インターフェイスも設定できます。

(no shut コマンドで) 有効になっているすべてのインターフェイスが IOS XR の TPNNNS に追加されます。

7. (任意) fwd_ew インターフェイスと fwdintf インターフェイスで使用される IP ルートを表示します。

```

[xr-vm_node0_RP0_CPU0:~]$ ip route
default dev fwdintf scope link src 1.1.1.1
8.8.8.8 dev fwd_ew scope link
192.168.122.0/24 dev Mg0_RP0_CPU0_0 proto kernel scope link src 192.168.122.213

```

IOS XR でのサードパーティ製ネットワーク名前空間に移動する代替方法

IOS XR へのログイン時に `ip netns exec tpnns bash` コマンドを入力せずに TPNNNS に直接移動するには、次に示すステップで説明する `sshd_tpnns` サービスを使用します。この手順には、サービスにアクセスするための非ルートユーザの作成が含まれています (ルートユーザはこのサービスにアクセスできません)。



- (注) IOS XR で、インターフェイスをバインドするサービスを開始する前に、インターフェイスが設定され、起動され、動作可能であることを確認します。

インターフェイスが設定された後にのみサービスを開始するには、サービススクリプトに次の関数を含めます。

```
. /etc/init.d/tpnns-functions
tpnns_wait_until_ready
```

tpnns_wait_until_ready 関数を追加することによって、サービススクリプトは、1つ以上のインターフェイスが設定されるのを待機してから、サービスを開始ようになります。

1. (任意) リロード時に TPNNS サービスを自動的に開始するには、`sshd_tpnns` サービスを追加し、そのサービスの存在を確認します。

```
bash-4.3# chkconfig --add sshd_tpnns
bash-4.3# chkconfig --list sshd_tpnns
sshd_tpnns    0:off  1:off  2:off  3:on   4:on   5:on   6:off
bash-4.3#
```

2. `sshd_tpnns` サービスを開始します。

```
bash-4.3# service sshd_tpnns start
Generating SSH1 RSA host key: [ OK ]
Generating SSH2 RSA host key: [ OK ]
Generating SSH2 DSA host key: [ OK ]
  generating ssh ECDSA key...
Starting sshd: [ OK ]
```

```
bash-4.3# service sshd_tpnns status
sshd (pid 6224) is running...
```

3. ステップ 1 で作成した非ルートユーザとして `sshd_tpnns` セッションにログインします。

```
host@fe-ucs36:~$ ssh devops@192.168.122.222 -p 57722
devops@192.168.122.222's password:
Last login: Tue Sep  8 20:14:11 2015 from 192.168.122.1
XR-vm_node0_RP0_CPU0:~$
```

4. インターフェイスを表示して、TPNNS に移動していることを確認します。

```
[XR-vm_node0_RP0_CPU0:~]$ ifconfig
Gi0_0_0_0 Link encap:Ethernet  HWaddr 52:46:04:87:19:3c
  inet addr:192.164.168.10  Mask:255.255.255.0
  inet6 addr: fe80::5046:4ff:fe87:193c/64 Scope:Link
  UP RUNNING NOARP MULTICAST  MTU:1514  Metric:1
  RX packets:0 errors:0 dropped:0 overruns:0 frame:0
  TX packets:3 errors:0 dropped:0 overruns:0 carrier:0
  collisions:0 txqueuelen:1000
  RX bytes:0 (0.0 B)  TX bytes:210 (210.0 B)

Mg0_RP0_CPU0_0 Link encap:Ethernet  HWaddr 52:46:12:7a:88:41
  inet addr:192.168.122.197  Mask:255.255.255.0
  inet6 addr: fe80::5046:12ff:fe7a:8841/64 Scope:Link
  UP RUNNING NOARP MULTICAST  MTU:1514  Metric:1
  RX packets:0 errors:0 dropped:0 overruns:0 frame:0
  TX packets:3 errors:0 dropped:0 overruns:0 carrier:0
  collisions:0 txqueuelen:1000
  RX bytes:0 (0.0 B)  TX bytes:210 (210.0 B)
```

```
fw_d_ew  Link encap:Ethernet  HWaddr 00:00:00:00:00:0b
         inet6 addr: fe80::200:ff:fe00:b/64 Scope:Link
         UP RUNNING NOARP MULTICAST  MTU:1500  Metric:1
         RX packets:0 errors:0 dropped:0 overruns:0 frame:0
         TX packets:2 errors:0 dropped:1 overruns:0 carrier:0
         collisions:0 txqueuelen:1000
         RX bytes:0 (0.0 B)  TX bytes:140 (140.0 B)

fw_dintf  Link encap:Ethernet  HWaddr 00:00:00:00:00:0a
         inet6 addr: fe80::200:ff:fe00:a/64 Scope:Link
         UP RUNNING NOARP MULTICAST  MTU:1482  Metric:1
         RX packets:0 errors:0 dropped:0 overruns:0 frame:0
         TX packets:2 errors:0 dropped:1 overruns:0 carrier:0
         collisions:0 txqueuelen:1000
         RX bytes:0 (0.0 B)  TX bytes:140 (140.0 B)

lo        Link encap:Local Loopback
         inet addr:127.0.0.1  Mask:255.0.0.0
         inet6 addr: ::1/128 Scope:Host
         UP LOOPBACK RUNNING  MTU:1500  Metric:1
         RX packets:0 errors:0 dropped:0 overruns:0 frame:0
         TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
         collisions:0 txqueuelen:0
         RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

lo:0      Link encap:Local Loopback
         inet addr:1.1.1.1  Mask:255.255.255.255
         UP LOOPBACK RUNNING  MTU:1500  Metric:1
```

これで、アプリケーションをホストするために IOS XR Linux シェルを使用する準備が整いました。



第 8 章

Cisco IOS XRv 9000 データプレーンの特徴的な機能

- [Early Fast Discard](#) (115 ページ)
- [GTP を使用した CEF ロードバランシング](#) (116 ページ)
- [データプレーンの管理](#) (118 ページ)

Early Fast Discard

表 15: 機能の履歴 (表)

機能名	リリース情報	機能説明
Early Fast Discard	リリース 24.1.1	Cisco IOS XR リリース 24.1.1 以降、Early Fast Discard 機能は廃止され、今後のリリースではサポートされなくなります。Cisco IOS XR リリース 24.1.1 以降では、この機能を使用しないことをお勧めします。

Early Fast Discard (EFD) は、着信トラフィックがルータのキャパシティを超過した状況进行处理するための輻輳保護機能です。通常の輻輳制御はトラフィック マネージャ (TM) および添付された QoS ポリシーによって処理されますが、極端な場合には EFD 機能が有効化されます。EFD 機能は、優先順位の高いトラフィック (キープアライブ、制御、BFD など) にフィルタを適用し、それ以外のトラフィックをデータパス処理の非常に早い段階で破棄することで、制御パケットの重要なフローを維持します。

EFD 機能を有効化するには、設定モードで **early-fast-discard** コマンドを使用します。廃棄するトラフィックは、IP プレシデンス値、MPLS exp 値、VLAN cos 値を設定することで定義します。

Early Fast Discard の設定

使用例

Early Fast Discard はデータプレーン上に設定され、破棄するトラフィックを次の条件に基づいて定義することにより、着信トラフィックフローを管理します。

- IP precedence=4
- MPLS exp=3
- VLAN cos=5

デフォルト値は 6 および ge (greater than or equal to : 「以上」の略) です。

設定

```
Router# configure
Router(config)# hw-module early-fast-discard
Router(config-early-fast-discard)# ip-prec 4 ip-op [lt | ge]
Router(config-early-fast-discard)# mpls-exp 3 mpls-op [lt | ge]
Router(config-early-fast-discard)# vlan-cos 5 vlan-op [lt | ge]
Router(config-early-fast-discard)# exit
```



(注) **no hw-module early-fast-discard** コマンドを使用して、EFD を非アクティブ化します。

実行コンフィギュレーション

```
RP/0/RP0/CPU0:ios#show run hw-module early-fast-discard
Thu Jul 16 15:51:34.672 UTC
hw-module early-fast-discard
 ip-prec 4 ip-op ge
 mpls-exp 3 mpls-op ge
 vlan-cos 5 vlan-op lt
!
```

関連項目

- [Early Fast Discard \(115 ページ\)](#)

GTP を使用した CEF ロードバランシング

セクションでは、フローごとのロードバランシングについて説明します。

フロー単位ロード バランシング

ロード バランシングでは、レイヤ 3（ネットワーク層）およびレイヤ 4（トランスポート層）のルーティング情報に基づいてパケットを複数のリンクに分散させるルータ機能について説明します。ルータが宛先に至るパスを複数検出した場合は、その宛先の複数のエントリでルーティング テーブルが更新されます。

フロー単位のロードバランシングでは、以下の機能が実行されます。

- 着信データ トラフィックは、複数の等コスト接続に均等に分散されます。
- レイヤ 3（ネットワーク層）ロードバランシングの決定は、5 タプルハッシュアルゴリズムでサポートされる IPv4、IPv6、および MPLS のフローに対して行われます。
- 同じハッシュアルゴリズム（3 タプルまたは 5 タプル）が、複数の等コストレイヤ 3（ネットワーク層）パス全体でのロードバランシングに使用されます。レイヤ 3（ネットワーク層）パスは、物理インターフェイスまたはバンドル インターフェイス上にあります。
- `cef load-balancing fields L3 global` コマンドを使用すると、3 タプルハッシュアルゴリズムを選択できます。
- デフォルトでは、5 タプルハッシュアルゴリズムがロードバランシングに使用されます。`cef load-balancing fields L3 global` コマンドを使用すると、3 タプルハッシュアルゴリズムが有効になります。

レイヤ 3 (L3) ルーティング情報

3 タプルロードバランスハッシュ計算には、次のレイヤ 3（ネットワーク層）入力が含まれます。

- 送信元 IP アドレス
- 宛先 IP アドレス
- ルータ ID

レイヤ 4 (L4) ルーティング情報

5 タプルロードバランスハッシュ計算には、3 タプルの入力と、次の追加のレイヤ 4（トランスポート層）入力が含まれます。

- 送信元 IP アドレス
- 宛先 IP アドレス

GTP の Cisco Express Forwarding (CEF) ロードバランシングは、XRv9K プラットフォームではデフォルトで常に有効になっており、無効にすることはできません。したがって、`no cef loadbalancing fields l4 gtp` コマンドは、ハッシュ計算での GTP フィールドの包含を無効化しません。

XRv9k は、GRE ヘッダー + MPLS ラベルの背後にある場合でも GTP ヘッダーを検出でき、GTP フィールドに基づいてロードバランシングを実行できます。

データプレーンの管理

Cisco IOS XRv 9000 ルータのデータプレーンは、ルータの起動後に自動的に開始されます。コントロールプレーンがデータプレーンとの通信を確立しない場合や通信できなくなった場合、コントロールプレーンは自動的にデータプレーンを再起動します。

データプレーンは、メンテナンスとトラブルシューティングのために、管理コンソールから手動で開始、シャットダウン、リロードできます。この表では、データプレーンの管理に必要なコマンドを示します。

表 16: データプレーンの管理コマンド

タスク	使用するコマンド
データプレーンのステータスを確認する。	<pre>show sdr sysadmin-vm:0_RP0# show sdr Thu May 7 18:38:38.996 UTC sdr default-sdr location 0/RP0/VM1 sdr-id 2 IP Address of VM 192.0.0.4 MAC address of VM E2:3A:DD:0A:8C:06 VM State RUNNING start-time 2015-05-07T17:54:39.457822+00:00 Last Reload Reason FIRST_BOOT Reboot Count 1 location 0/RP0/VM2 sdr-id 2 IP Address of VM 192.0.0.6 MAC address of VM E2:3A:DD:0A:8C:06 VM State RUNNING start-time 2015-05-07T18:22:44.136498+00:00 Last Reload Reason FIRST_BOOT Reboot Count 1</pre> <p>データプレーンの場所をメモしておく必要があります。この例では、データプレーンの状態は場所 0/RP0/VM2 の下に表示されています。</p>

タスク	使用するコマンド
データプレーンを開始する	sdr default-sdr location <data-plane-location> start 例 <pre> sysadmin-vm:0_RP0# sdr default-sdr location 0/RP0/VM2 start Mon May 4 17:16:37.867 UTC Start ? [no,yes] yes result start sdr default-sdr location 0/RP0 request acknowledged </pre>
データプレーンをシャットダウンする	sdr default-sdr location <data-plane-location> shut 例 <pre> sysadmin-vm:0_RP0# sdr default-sdr location 0/RP0/VM2 shut Mon May 4 17:12:32.397 UTC Shut ? [no,yes] yes result shutdown sdr default-sdr location 0/RP0 request acknowledged </pre>
データプレーンをリロードする	sdr default-sdr location <data-plane-location> reload 例 <pre> sysadmin-vm:0_RP0# sdr default-sdr location 0/RP0/VM2 reload Mon May 4 17:21:17.390 UTC Reload ? [no,yes] yes result graceful reload sdr default-sdr location 0/RP0 request acknowledged </pre>



(注) 通常の運用では、ユーザはデータプレーンを手動で開始および停止してはいけません。

データプレーンのデバッグ

Cisco IOS XRv 9000 ルータには、データプレーンのステータスと統計をチェックするコマンドのセットが用意されています。これらのコマンドは、次のとおりです。



(注) 次に示すデータプレーンのデバッグコマンドは、一時的にトラフィックの転送を中断することがあります。

- show controller dpa statistics
- show controller dpa fib ipv4|ipv6 [<prefix> | summary]

- **show controller dpa version** : データプレーンのバージョンを表示します。

次に例を示します。

```
RP/0/RP0/CPU0:ios#show controller dpa version
Fri May 29 19:28:16.520 UTC
Image built on 13:29:13 May 29 2015 in workspace /workspacel/shope/ttl_commit
DPA started May 29 18:11:23, up 0 days, 01:16
```

- **show controller dpa logging** : データプレーンログを表示します。デフォルトでは、エラーや重要なイベントのログ情報のみが使用可能です。

次に例を示します。

```
RP/0/RP0/CPU0:SS_Node1#show controllers dpa logging
Mon Jun 29 19:47:33.245 UTC
Jun 29 01:43:32.820: Log File Started
Jun 29 01:43:32.820: DPA_INFO: DPA beginning initialization
Jun 29 01:43:32.823: DPA_INFO: Dataplane Agent enabled
Jun 29 01:43:32.823: DPA_INFO: Image built on 15:02:53 Jun 25 2015
Jun 29 01:43:32.823: DPA_INFO: Table WRED_STR of size 8388480 is being initialized

Jun 29 01:43:32.824: DPA_INFO: Table STATIC_POLICER_STR of size 8192 is being
initialized with data
Jun 29 01:43:32.824: DPA_INFO: Table HASH_DYN_BUCKET_STR of size 3355264 is being
initialized
Jun 29 01:43:32.825: DPA_INFO: Table HASH_BUCKET_STR of size 33552832 is being
initialized
Jun 29 01:43:32.829: DPA_INFO: Table DYN_FREE_BLOCK_STR of size 16777216 is being
initialized
Jun 29 01:43:32.832: DPA_INFO: Table INDEX_Q_STR of size 8192 is being initialized

Jun 29 01:43:32.832: DPA_INFO: Table HASH_HOST_DYN_BUCKET_STR of size 1677504 is
being initialized
```

- **show controller dpa statistics global** : データプレーンの統計を表示します。この統計には、ドロップパケット数、コントロールプレーンから入ってきたパケット、およびコントロールプレーンにパントされたパケットが含まれます。

次に例を示します。

```
RP/0/RP0/CPU0:ios#show control dpa statistics global
Fri May 29 19:27:35.497 UTC
Index  Punt                                     Count
-----
 1575  ARP                                         28
 1677  IFIB                                       1341
 1701  IPv4 FIB                                   5

Index  Inject                                     Count
-----
 267  IPv4 from fabric                          31
 268  IPv4 from fabric multicast                1290
 270  IPv4 from fabric next-hop                 55
 275  Inject to fabric                          1376
 276  Inject to port                             11

Index  Drop                                     Count
-----
 63  Egress uIDB in down state                 1
```

113 IPv6 disabled in uIDB

52

- **show controller dpa fib ipv4|ipv6 [<prefix> | summary]** : データプレーンの FIB エントリを表示します。<prefix> および summary キーワードはオプションです。

次に例を示します。

```
RP/0/RP0/CPU0:ios#show controller dpa fib ipv4
Fri May 29 19:54:40.110 UTC

VRF id: 0
Default prefix 0.0.0.0/0 -> leaf:46423
total number of prefix:35
total_node_allocated:29 leaf_inserts:50 leaf_deletes 15 leaf_replaces: 2

Prefix                leaf_index
224.0.0.0/4           46436 (0xb564)
224.0.0.0/24          46434 (0xb562)
255.255.255.255/32    46429 (0xb55d)
0.0.0.0/32            46433 (0xb561)
10.1.1.1/32           46513 (0xb5b1)
2.2.2.2/32            46510 (0xb5ae)
```

summary キーワードを使用すると、このコマンドは各 vrf テーブルのプレフィックスと運用統計を表示します。次に例を示します。

```
RP/0/RP0/CPU0:R1-PE1#show control dpa fib ipv4 summary

VRF id: 0
Default prefix 0.0.0.0/0 -> leaf:46421
total number of prefix: 27859
allocated nodes:         1089
leaf_inserts:            27922
leaf_deletes             63
leaf_replaces:          173

VRF id: 1
Default prefix 0.0.0.0/0 -> leaf:46444
total number of prefix: 430
allocated nodes:         26
leaf_inserts:            430
leaf_deletes             0
leaf_replaces:          213
```

<prefix> キーワードを使用すると、このコマンドは、すべてのプレフィックスと、そのプレフィックスに一致する vrf テーブルのリストを表示します。次に例を示します。

```
RP/0/RP0/CPU0:ios#show controller dpa fib ipv4 5.11.23.131/32
VRF id: 0
Prefix                leaf_index
5.11.23.131/32        1164818 (0x11c612)

VRF id: 1

VRF id: 2

VRF id: 3
```

- **show controller dpa tm queue <num>** : トラフィック マネージャ キューの内部データを表示します。このデータには、DRR ウェイト、Q-limit、瞬間的なパケットとバイト数が含まれます。

次に例を示します。

```
RP/0/RP0/CPU0:ios#show controller dpa tm queue 1
Fri May 29 19:31:25.556 UTC
Queue 1
  Parent Subport:      0
  Weight:              10
  Q-Limit:             625000
  Packets:             0
  Bytes:               0
```

- **show controller dpa tm subport <num>** : トラフィック マネージャ サブポートの内部データを表示します。このデータには、DRR ウェイト、シェーピングされたレート、キュー設定、および瞬間的なパケットが含まれます。

次に例を示します。

```
RP/0/RP0/CPU0:ios#show controller dpa tm subport 3
Fri May 29 19:44:12.993 UTC
Subport 3
  Parent vPort:       3
  Weight:             10200
  Rate:               776726
  Being Deleted:     no
  Configured:        yes
  Queue 24 pkts:     0 bytes:      0
  Queue 25 pkts:     6 bytes:     8376
  Queue 26 pkts:     0 bytes:      0
  Queue 27 pkts:     0 bytes:      0
  Queue 28 pkts:     0 bytes:      0
  Queue 29 pkts:     0 bytes:      0
  Queue 30 pkts:     0 bytes:      0
  Queue 31 pkts:     0 bytes:      0
  Priority Queues:    1
  Best effort Queues: 7
```

- **show controller dpa tm vport <num>** : トラフィック マネージャ仮想ポートの内部データを表示します。このデータには、そのポートのレートとフロー制御が現在アクティブかどうかが含まれます。

次に例を示します。

```
RP/0/RP0/CPU0:ios#show controller dpa tm vport 0
Fri May 29 19:32:39.447 UTC
vPort 0
  Parent port:       0
  Rate:              95
  Flow control:      0
```



第 9 章

Cisco IOS XRv 9000 アプライアンス

この章では、アプライアンスとしての Cisco IOS XRv 9000 について紹介し、アプライアンスに関連するいくつかの概念について説明します。この章では、アプライアンスの IOS XRv 9000 ソフトウェアをアップグレード、ダウンロード、再インストールするために必要な作業について説明します。



(注) Cisco IOS XRv 9000 アプライアンスは、Cisco IOS XR リリース 6.1.2 で導入されたものです。

- [Cisco IOS XRv 9000 アプライアンスの導入 \(123 ページ\)](#)
- [アプライアンスの物理的接続の概要 \(124 ページ\)](#)
- [アプライアンスの設定 \(127 ページ\)](#)
- [ソフトウェア管理 \(129 ページ\)](#)
- [Cisco IOS XRv 9000 アプライアンス ハードウェアのモニタリング \(134 ページ\)](#)

Cisco IOS XRv 9000 アプライアンスの導入

Cisco IOS XRv 9000 アプライアンスは、適切なライセンスすべてが含まれた、UCS ハードウェアと Cisco XRv 9000 ルータ ソフトウェアのパッケージです。アプライアンス パッケージにより、ハードウェアとソフトウェアの所有権についての運用上の心配をせずに、ネットワークルーティング機能を仮想化できます。

Cisco IOS XRv 9000 アプライアンスは、出荷時にベア メタル UCS サーバ ハードウェアに事前にインストールされた Cisco IOS XRv 9000 ルータ ソフトウェアです。このアプライアンスは、仮想ルートリフレクタとして実行すると、7千万個のルートプレフィックスまで拡張可能な非常に高い拡張性をサポートしています。そのため、ソフトウェア (ハイパーバイザ) のレイヤを追加する必要ありません。

さらに、このアプライアンスはゼロ タッチ プロビジョニング (ZTP) もサポートしており、既存のネットワークに簡単に挿入できます。

アプライアンスの単一 PID には、ハードウェア、ソフトウェア、ライセンス、サービスがすべて含まれています。このアプライアンスの単一 PID により、ソフトウェアとハードウェアで個

別にサービス契約を結ぶ必要がなく、サポートとサービスのエクスペリエンスが簡素化されます。



- (注)
- ライセンシングは無効になっています。
 - ハードウェアの追加および削除はサポートされません。

次の表に、サポートされている UCS サーバとアプライアンス PID を示します。

表 17:

Cisco IOS XR リリース	サポートされている UCS サーバモデル	アプライアンスの単一 PID
リリース 6.1.2 およびリリース 7.2.2 まで	UCS C220 M4S	ASR-XRV9000-APLN
リリース 6.6.2	UCS C220 M5SX (UCSC-C220-M5SX)	XRV9000-APLN-ROUT

次に、デフォルトのコンソール設定を示します。

- ボー レート 115200 bps
- パリティなし
- 2 ストップ ビット
- 8 データ ビット

アプライアンスの物理的接続の概要

アプライアンスの背面パネルビューは UCS サーバと似ています。ただし、UCS サーバで利用可能なインターフェイスの一部は、このアプライアンスでは使用されません。次のトピックでは、アプライアンスのインターフェイスの使用方法与マッピングについて説明します。

UCS M5 ベースのアプライアンスの背面パネルの機能

次の図に、UCS M5 ベースのアプライアンスの背面パネルの機能の概要を示します。

図 7: UCS M5 ベースのアプライアンスの背面パネルの機能

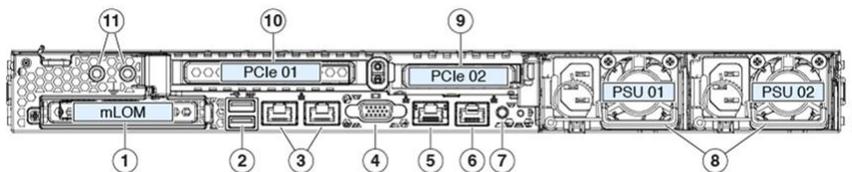


表 18: アプライアンスのインターフェイスのマッピング

	インターフェイスの説明	アプライアンスでの使用法
1	モジュラ LAN-on-motherboard (mLOM) カードベイ (x16 PCIe レーン)	未使用
2	USB 3.0 ポート (2 個)	Admin コンソールにキーボード接続するために使用
3	デュアル 1 Gb/10 Gb イーサネット ポート (LAN1 と LAN2)	LAN1 は XR 管理インターフェイスにマッピングされます。 LAN2 は使用されません。
4	VGA ビデオポート (DB-15 コネクタ)	Admin コンソールにマッピング VGA コネクタは通常の VGA モニタに接続でき、USB キーボードをモニタの USB ポートに接続できます。または、UCS USB/VGA ブレークアウト ケーブルを使用して、サーバの前面で接続することもできます (ケーブルはサーバに同梱)。
5	1 Gb イーサネット専用管理ポート	Cisco Integrated Management Controller (CIMC) にマッピング
6	シリアルポート (RJ-45 コネクタ)	XR コンソールにマッピング シリアルポートは、そのシリアルポート経由でキーボードかビデオへのアクセスを可能にしているデバイスにケーブル接続する必要があります。
7	背面ユニット識別ボタン/LED	CIMC にマッピング
8	電源装置 (2、1+1 として冗長)	-

	インターフェイスの説明	アプライアンスでの使用法
9	PCIe ライザー 2/スロット 2 (x16 レーン)	10 G イーサネット ポート X 8 を含む
10	PCIe ライザー 1/スロット 1 (x16 レーン)	
11	デュアルホール アース ラグ用ネジ穴	必要に応じて使用

UCS M4 ベースのアプライアンスの背面パネルの機能

次の図に、UCS M4 ベースのアプライアンスの背面パネルの機能の概要を示します。

図 8: UCS M4 ベースのアプライアンスの背面パネルの機能

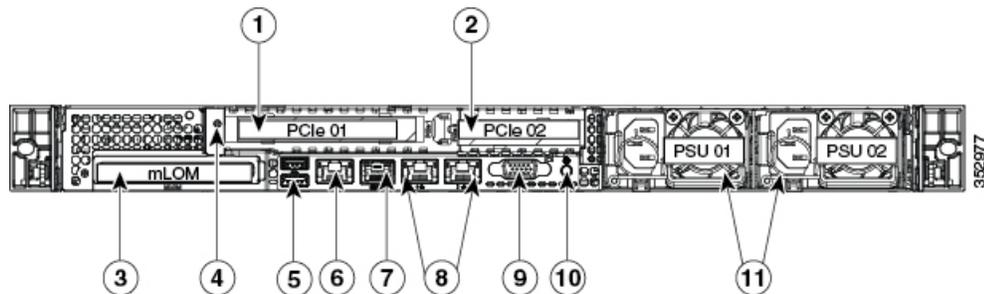


表 19: アプライアンスのインターフェイスのマッピング

	インターフェイスの説明	アプライアンスでの使用法
1	PCIe ライザー 1/スロット 1	10 G イーサネット ポート X 8 を含む
2	PCIe ライザー 2/スロット 2	
3	モジュラ LAN-on-motherboard (mLOM) カードスロット	未使用
4	アースラグの穴 (DC 電源装置の場合)	必要に応じて使用
5	USB 3.0 ポート (2 個)	Admin コンソールにキーボード接続するために使用
6	1 Gb イーサネット専用管理ポート	Cisco Integrated Management Controller (CIMC) にマッピング

	インターフェイスの説明	アプライアンスでの使用法
7	シリアルポート (RJ-45 コネクタ)	XR コンソールにマッピング シリアルポートは、そのシリアルポート経由でキーボードかビデオへのアクセスを可能にしているデバイスにケーブル接続する必要があります。
8	デュアル 1 Gb イーサネット ポート (LAN1 と LAN2)	LAN1 は XR 管理インターフェイスにマッピングされます。 LAN2 は使用されません。
9	VGA ビデオ ポート (DB-15)	Admin コンソールにマッピング VGA コネクタは通常の VGA モニタに接続でき、USB キーボードをモニタの USB ポートに接続できます。または、UCS USB/VGA ブレークアウト ケーブルを使用して、サーバの前面で接続することもできます (ケーブルはサーバに同梱)。
10	背面ユニット識別ボタン/LED	CIMC にマッピング
11	電源装置 (最大 2 台、1+1 冗長)	-

インターフェイスの一覧と物理マッピング

PCIe02 アダプタは PCIe01 とは物理的に逆向きに挿入します。そのため、PCIe02 インターフェイスの最後の 4 つのポートは逆向きになっています。したがって、これらのポートの物理的な XR ポートマッピングは、左から右に次の表に示すようになります。

0	1	2	3	7	6	5	4
PCIe01				PCIe02			

アプライアンスの設定

アプライアンスは次の 3 通りの方法で設定できます。

CLI を使用した手動設定

手動設定の開始方法：

1. シリアルポートを介して XR コンソール (またはコントローラ) に接続します。



- (注) vga イメージを使用する場合、アプライアンスのインストール中は、VM の電源投入後に XR コンソールにアクセスできません。したがって、インストールには非 vga アプライアンス イメージを使用することをお勧めします。

2. 管理者パスワードを使用して XR コンソールにログインします。



- (注) デフォルトのログイン情報はユーザー名として <root>、パスワードとして <lab> を使用できます。それでもログインできない場合は、ユーザー名として <root>、パスワードとして <Cisco123> を使用します。

3. CLI を使用してルータを手動で設定します。

特定の IOS XR 設定の詳細については、『[ASR 9000 System Management Configuration Guide](#)』を参照してください。

特定の IOS XR 設定 CLI の詳細については、『[ASR 9000 System Management Command Reference](#)』を参照してください。

IOS XRv 9000 では、IOS XR でサポートされている一部の機能をサポートしていません。IOS XRv 9000 ルータでサポートされている機能については、最新の『[IOS XRv 9000 Router Release Notes](#)』を参照してください。

ゼロタッチ プロビジョニングを介した自動設定

ゼロタッチプロビジョニング (ZTP) は、iPXE を使用してルータでソフトウェアをインストールした後の自動プロビジョニングに役立ちます。

ZTP の自動プロビジョニングでは以下の手順を実行します。

- 設定：設定ファイルをダウンロードして実行します。ファイルの最初の行に `!!` が含まれている必要があります。IOS XR が含まれている必要があります。
- スクリプト：スクリプト ファイルをダウンロードして実行します。これらのスクリプト ファイルには、タスクを完了するためのプログラムによるアプローチが含まれています。たとえば IOS XR コマンドを使用して作成されたスクリプトは、パッチアップグレードを実行します。ファイルの最初の行に `#!` が含まれている必要があります。/bin/bash or #!/bin/sh が含まれている必要があります。



- (注) ZTP は管理インターフェイスでのみサポートされます。

ZTP を使用した自動プロビジョニングの詳細については、「[Zero Touch Provisioning](#)」を参照してください。

CVAC と USB を使用した自動設定

Cisco IOS XRv 9000 アプライアンスは、CVAC を使用した自動設定をサポートしています。標準 XR 設定を持つプレーンテキスト設定ファイルの **iosxr_config.txt** を USB ドライブに保存して CVAC に提供し、アプライアンスを起動する必要があります。これは、他の設定（初期ユーザ名およびパスワードを含む）が設定されていない場合にのみ機能します。

CVAC を使用してアプライアンスを起動する方法の詳細は、[CVAC : ブートストラップ構成のサポート \(78 ページ\)](#) を参照してください。

ソフトウェア管理

IOS XRv 9000 アプライアンスは、IOS XR ベースの製品であるため、多くのソフトウェア管理機能を IOS XR から継承しています。このセクションでは、IOS XRv 9000 ルータ ソフトウェアのアップグレード、ダウングレード、および再インストールに必要な概念とタスクについて説明します。



(注) FPD 関連のコマンドは、IOS XRv 9000 アプライアンスではサポートされていません。これには **fpd auto-update** コマンドが含まれています。

Unified Computing System を介したソフトウェア管理

アプライアンスデバイスには、IOS XRv 9000 ソフトウェアがプレインストールされています。次のいずれかの方法により、必要なソフトウェアバージョン（リリース 6.1.1 以降）を使用し、いつでもデバイスにイメージを再適用できます。

- CIMC を使用した OS の再インストール
- USB ポートからの OS の再インストール
- PXE インストール サーバを使用した OS の再インストール



(注) OS を再インストールすると、既存の設定とシステム情報がすべて削除されます。

OS をインストールした後は、「アプライアンスの設定」の項に従って基本設定を実行してください。

CIMC を使用した IOS XRv 9000 ソフトウェアの再インストール

Cisco Integrated Management Controller (CIMC) は物理デバイスの管理に使用されるもので、Web ブラウザからアクセスできます。次の目的で CIMC を使用します。

- リモートからアプライアンスの電源をオン/オフする。

- リモートからコンソールにアクセスする。
- ソフトウェアを再インストールする。
- ファームウェアをアップグレードする。

CIMC を使用して、IOS XRv 9000 ソフトウェアをリモートからアプライアンスに再インストールできます。デフォルトで、CIMC はアプライアンスに GigE 専用ポートがあります。Web ブラウザから CIMC にアクセスするには、CIMC ポートに IP アドレスを設定する必要があります。CIMC ポートを設定するオプションは、デバイスの電源をオンにしている間に VGA コンソールで利用できます。

CIMC ポートに IP アドレスを設定した後、Web ブラウザから CIMC にログインし、KVM（キーボード、ビデオ、マウス）コンソールを使用します。

UCS ベアメタルサーバーで Cisco IOS XRv バージョン 7.0.1 以降に更新すると、KVM コンソールは使用できなくなります。代わりに、XR コンソールと Calvados コンソールの両方にアクセスできる telnet ポートを使用する必要があります。

サーバに物理的に接続された CD/DVD ドライブまたはフロッピー ドライブを使用する代わりに、KVM コンソールは仮想メディアを使用します。これは、仮想 CD/DVD ドライブまたはフロッピー ドライブにマップされる実際のディスク ドライブまたはディスク イメージファイルです。

KVM コンソールの起動に関する詳細については、「[KVM Console](#)」を参照してください。



-
- (注) ソフトウェアのインストールおよび再インストールには、IOS XRv 9000 ソフトウェアの ISO バージョンを使用する必要があります。
-

CIMC を使用した OS の再インストール

次の手順に従って、M4 および M5 UCS ベースのアプライアンスに OS を再インストールします。

始める前に

- 必要な ISO イメージファイル（リリースバージョン 6.1.1 以降）をマシンにダウンロードします。
- OS をインストールするには、管理者権限を持つユーザとしてログインする必要があります。
- CIMC の最新バージョンを実行している必要があります。

手順

ステップ 1 OS インストールの ISO ディスク イメージファイルをコンピュータにコピーします。

ステップ 2 CIMC が開いていない場合は、ログインします。

ステップ 3 [Navigation] ペインで [Launch KVM] をクリックします。

ステップ 4 Java ベースの KVM か、または HTML ベースの KVM を選択します。

Java ベースの KVM と HTML ベースの KVM の GUI は似ています。Java ベースの KVM と HTML ベースの KVM のコンソールは、KVM コンソールと総称されています。

[KVM Console] が別ウィンドウで開きます。

ステップ 5 KVM コンソールで [Virtual Media] > [Activate Virtual Device] を選択します。

ステップ 6 [Virtual Media] > [Map CD/DVD] を選択します。次に、ローカルに保存されている ISO インストールディスクイメージを参照し、[Map Device] をクリックします。

ステップ 7 KVM コンソールで [Power] > [Reset System (warm boot)] を選択します。

サーバが再起動する際に、インストールプロセスが開始されます。インストールプロセスが完了したら、「アプライアンスの設定」の項を参照してデバイスを設定します。コンソールアクセスを保持するには、COM0 が SOL にマッピングされているかどうかを確認します。したがって、SOL を無効にするか、SOL を COM1 にマッピングできます。その後、物理シリアルポートが XR コンソールにマッピングされます。

USB ポートからの OS の再インストール

アプライアンスは、どの USB ポートからでもオペレーティングシステムを起動できます。ただし、USB ポートから BIOS を起動する前に、いくつかのガイドラインを考慮する必要があります。

- BIOS のインストールプロセスには、ブート可能 USB ドライブが必要です。「ブート可能 USB ドライブの作成」の項を参照してください。
- ブート順序の設定を保持するために、内部 USB ポートを使って BIOS を起動することをお勧めします。
- USB ポートから OS を起動する前に、そのポートを有効にしておく必要があります。



(注) デフォルトでは、USB ポートは無効になっています。USB ポートが無効化している場合、そこから BIOS を起動する前に有効にする必要があります。

- USB ポートから BIOS を起動した後、その USB ソースからサーバが毎回ブートするよう、セカンドレベルのブート順序を設定する必要があります。

次の手順で、USB ポートからオペレーティングシステムをインストールします。

1. アプライアンスの電源を再投入します。
2. 起動プロセス中に [USB Boot Option] を選択して続行します。

3. システムは、USB ドライブからハードディスク ドライブにイメージをインストールし、それから再起動します。



- (注) メモリ サイズが大きい USB ドライブは起動しません。そのため、8 GB の USB ドライブを使用することを推奨します。

PXE インストール サーバを使用した OS の再インストール

始める前に

- VLAN 経由でサーバに到達できることを確認します。
- OS をインストールするには、admin 権限を持つユーザとしてログインする必要があります。

手順

ステップ 1 PXE のブート順序を最初に設定します。

ステップ 2 サーバをリブートします。

VLAN で PXE インストール サーバを使用できる場合は、サーバが再起動するとインストールプロセスが開始します。通常、PXE インストールは自動化されており、追加のユーザ入力が必要としません。残りのインストールプロセスについては、インストールしている OS のインストーレーションガイドを参照してください。

次のタスク

OS のインストールが完了したら、LAN のブート順序を元の設定にリセットします。

ブート可能 USB ドライブの作成

ブート可能 USB ドライブを作成するには、UNetbootin という外部のオープン ソース ソフトウェアが必要になります。

始める前に

- 必要な Cisco IOS XRv ISO 9000 のインストール ファイルを、ラップトップまたはサーバにダウンロードします。
- UNetbootin アプリを <https://unetbootin.github.io/> からダウンロードします。

手順

ステップ 1 コンピュータに OS インストール ディスク イメージ ファイルをコピーします。

(注)

再インストールする場合は、IOS XRv 9000 ソフトウェア インストレーション ファイルの ISO バージョンを使用することを推奨します。

ステップ 2 USB ディスクを fat32 形式にフォーマットします。

ステップ 3 UNetbootin を実行し、ISO インストール ファイルをロードします。

ステップ 4 USB ディスクをビルドします。次のリンク先にある手順を参照してください。 <https://unetbootin.github.io/>

ステップ 5 デフォルト オプションとして *Panini-no-issu* 起動メニュー項目を使用するために、USB の `syslinux.cfg` ファイルを編集します。

デフォルトで、ユーザが選択する必要がある項目の一覧が BIOS に表示されます。

Mac OS ユーザの場合は、ターミナルを使用してマウント ポイントに移動し、Vi エディタを使用してファイルを編集します。例：`/Volumes/MYDISK`。

IOS XR を使用したソフトウェア管理

IOS XRv 9000 ソフトウェアは、次のいずれかの方法でアップグレードとダウングレードを実行できます。

- IOS XR CLI コマンド
- ZTP bash スクリプト (インストール コマンド)
- IOS XR がサポートする管理性インターフェイス

アップグレード手順とダウングレード手順の詳細については、アップグレードに関するドキュメントを参照してください。ソフトウェア イメージと一緒に使用できます。

CLI を使用したソフトウェアアップグレード

始める前に

- 必要な ISO イメージ ファイルをマシンにダウンロードします。

手順

ステップ 1 `install commit`

例 :

```
router# install commit
```

アプライアンスにインストールされている、現在のバージョンの IOS XRv 9000 ソフトウェアを確定します。

ステップ 2 install add source <filepath>

例 :

```
router# install add source tftp://192.0.2.4/fakepath/xrv9k-fullk9-x.iso
```

アプライアンスにインストールする必要がある ISO ディスク イメージ ファイルの場所を特定します。

ステップ 3 install activate <filename>

例 :

```
router# install activate xrv9k-fullk9-x.iso
```

IOS XRv 9000 の新しいイメージ バージョンを有効化します。ルータが再起動します。

ステップ 4 show version

例 :

```
router# show version
```

インストールされた新しいイメージ バージョンを確認します。

ステップ 5 install commit

例 :

```
router# install commit
```

新しいバージョンを確定します。

Cisco IOS XRv 9000 アプライアンス ハードウェアのモニタリング

Cisco IOS XRv 9000 アプライアンスのハードウェアのモニタリングを使用すると、従来のハードウェア ルータ上で表示されるものと同じ方法でアプライアンスのハードウェア環境のパラメータを表示できます。情報の取得に関するインターフェイスに基づいて、アプライアンスのハードウェア情報が以下の 3 つのセクションにグループ化されます。

- ハードウェア環境のモニタリング — これには、電源装置、ファン、電圧、電流、および温度の情報が含まれます。また、ハードウェア障害警告とアラーム情報も含まれます。
- ホスト OS レベルのモニタリング — これには、プロセッサ、コア、メモリ、および HDD 使用率の情報が含まれます。

- SFP 光モニタリング — これには、光診断と SFP OIR（活性挿抜）のモニタリング情報が含まれます。

ハードウェア環境のモニタリング

Cisco IOS XRv 9000 アプライアンスでは、システムが継続的にハードウェアをモニタして、電力消費に関する情報を収集し、ハードウェア障害をレポートします。システム管理モードで以下のコマンドを使用してこれらの情報を表示することができます。

タスク	使用するコマンド
シャーシのファンの情報を表示する	<pre>sysadmin-vm:0_RP0# show environment fan Sun Nov 26 20:00:46.373 UTC</pre> <hr/> <pre> Fan speed (rpm) Location FRU Type FAN_0 FAN_1 FAN_2 FAN_3 FAN_4 FAN_5 0/FT0 XRV-FAN-C220M4= 7700 7500 7700 7700 7700 7500</pre> <p>Cisco IOS XRv 9000 アプライアンスには 6 つのファンがあります。このファンは OIR をサポートしていないため、ファンを交換するにはアプライアンスをシャットダウンする必要があります。</p> <p>他のハードウェアプラットフォームとは異なり、アプライアンス上で実行されている Cisco IOS XR ソフトウェアではファン速度は管理していません。代わりに、UCS Cisco Integrated Management Controller (CIMC) システムで制御しています。</p>

タスク	使用するコマンド
電源トレイの情報を表示する	

タスク	使用するコマンド
	<pre> sysadmin-vm:0_RP0# show environment power </pre> <hr/> <pre> CHASSIS LEVEL POWER INFO: 0 </pre> <hr/> <pre> Total output power capacity (-) : 0W + 0W Total output power required : 0W Total power input : 0W Total power output : 108W </pre> <pre> Power Shelf 0: </pre> <hr/> <pre> Power Supply -----Input----- -----Output--- Status Module Type Volts Amps Volts Amps </pre> <hr/> <pre> 0/PT0-PM0 Cisco 0.0 0.0 12.1 4.0 OK 0/PT0-PM1 Cisco 0.0 0.0 12.0 5.0 OK </pre> <pre> Total of Power Shelf 0: 0W/ 0.0A 108W/ 9.0A </pre> <hr/> <pre> Location Card Type Power Power Status Allocated Used Watts Watts </pre> <hr/> <pre> 0/0 R-IOXRv9000-LC-A 0 - 0/RP0 R-IOXRv9000-RP-A 0 - 0/FT0 XRv-FAN-C220M4= 0 - </pre> <p>上記のコマンドの出力で、強調表示されているフィールドの情報は（電源モジュールと出力）のみが Cisco IOS XRv 9000 アプライアンスで意味のある測定値です。合計の電源出力は、各電源モジュールの電源出力（電源出力 = ボルト * アンペア）の合計です。</p>

タスク	使用するコマンド																																																																																																																																												
	<p>アプライアンスにはボルト/アンペアのセンサーがなく、アプライアンスの電力容量、所要電力、割り当てられた電力、使用電力のデータはありません。</p>																																																																																																																																												
温度情報を表示する	<pre>sysadmin-vm:0_RP0# show environment temperature</pre> <table border="1"> <thead> <tr> <th colspan="2">Location</th> <th colspan="3">TEMPERATURE</th> <th>Value</th> <th>Crit</th> </tr> <tr> <th>Major</th> <th>Minor</th> <th>Minor</th> <th>Major</th> <th>Crit</th> <th rowspan="2">Sensor</th> <th rowspan="2">Sensor</th> </tr> <tr> <th>(Lo)</th> <th>(Hi)</th> <th>(Hi)</th> <th>(Hi)</th> <th>(Hi)</th> <th>(deg C)</th> <th>(Lo) (Lo)</th> </tr> </thead> <tbody> <tr> <td colspan="7">0/RP0</td> </tr> <tr> <td>-5</td> <td>0</td> <td>40</td> <td>45</td> <td>50</td> <td>27</td> <td>-10</td> </tr> <tr> <td>-5</td> <td>0</td> <td>80</td> <td>85</td> <td>90</td> <td>37</td> <td>-10</td> </tr> <tr> <td>-5</td> <td>0</td> <td>60</td> <td>70</td> <td>80</td> <td>34</td> <td>-10</td> </tr> <tr> <td>-5</td> <td>0</td> <td>60</td> <td>70</td> <td>80</td> <td>34</td> <td>-10</td> </tr> <tr> <td>-5</td> <td>0</td> <td>60</td> <td>70</td> <td>80</td> <td>35</td> <td>-10</td> </tr> <tr> <td>-5</td> <td>0</td> <td>60</td> <td>70</td> <td>80</td> <td>38</td> <td>-10</td> </tr> <tr> <td>-5</td> <td>0</td> <td>92</td> <td>97</td> <td>100</td> <td>39</td> <td>-10</td> </tr> <tr> <td>-5</td> <td>0</td> <td>92</td> <td>97</td> <td>100</td> <td>46</td> <td>-10</td> </tr> <tr> <td>-5</td> <td>-1</td> <td>65</td> <td>85</td> <td>90</td> <td>33</td> <td>-10</td> </tr> <tr> <td>-5</td> <td>-1</td> <td>65</td> <td>85</td> <td>90</td> <td>0</td> <td>-10</td> </tr> <tr> <td>-5</td> <td>-1</td> <td>65</td> <td>85</td> <td>90</td> <td>0</td> <td>-10</td> </tr> <tr> <td colspan="7">...</td> </tr> <tr> <td colspan="7">0/PT0-PM0</td> </tr> <tr> <td>-5</td> <td>-1</td> <td>60</td> <td>65</td> <td>70</td> <td>33</td> <td>-10</td> </tr> <tr> <td colspan="7">0/PT0-PM1</td> </tr> <tr> <td>-5</td> <td>-1</td> <td>60</td> <td>65</td> <td>70</td> <td>28</td> <td>-10</td> </tr> </tbody> </table> <p>(注) このメモリ スロットには DDR メモリが装着されていないため、メモリ スロットの温度測定値はほとんどが0ではありません。</p>	Location		TEMPERATURE			Value	Crit	Major	Minor	Minor	Major	Crit	Sensor	Sensor	(Lo)	(Hi)	(Hi)	(Hi)	(Hi)	(deg C)	(Lo) (Lo)	0/RP0							-5	0	40	45	50	27	-10	-5	0	80	85	90	37	-10	-5	0	60	70	80	34	-10	-5	0	60	70	80	34	-10	-5	0	60	70	80	35	-10	-5	0	60	70	80	38	-10	-5	0	92	97	100	39	-10	-5	0	92	97	100	46	-10	-5	-1	65	85	90	33	-10	-5	-1	65	85	90	0	-10	-5	-1	65	85	90	0	-10	...							0/PT0-PM0							-5	-1	60	65	70	33	-10	0/PT0-PM1							-5	-1	60	65	70	28	-10
Location		TEMPERATURE			Value	Crit																																																																																																																																							
Major	Minor	Minor	Major	Crit	Sensor	Sensor																																																																																																																																							
(Lo)	(Hi)	(Hi)	(Hi)	(Hi)			(deg C)	(Lo) (Lo)																																																																																																																																					
0/RP0																																																																																																																																													
-5	0	40	45	50	27	-10																																																																																																																																							
-5	0	80	85	90	37	-10																																																																																																																																							
-5	0	60	70	80	34	-10																																																																																																																																							
-5	0	60	70	80	34	-10																																																																																																																																							
-5	0	60	70	80	35	-10																																																																																																																																							
-5	0	60	70	80	38	-10																																																																																																																																							
-5	0	92	97	100	39	-10																																																																																																																																							
-5	0	92	97	100	46	-10																																																																																																																																							
-5	-1	65	85	90	33	-10																																																																																																																																							
-5	-1	65	85	90	0	-10																																																																																																																																							
-5	-1	65	85	90	0	-10																																																																																																																																							
...																																																																																																																																													
0/PT0-PM0																																																																																																																																													
-5	-1	60	65	70	33	-10																																																																																																																																							
0/PT0-PM1																																																																																																																																													
-5	-1	60	65	70	28	-10																																																																																																																																							

タスク	使用するコマンド
電圧情報を表示する	<pre> sysadmin-vm:0_RP0# show environment voltage Sun Nov 26 20:00:32.333 UTC ----- Location VOLTAGE Value Crit Minor Minor Crit Sensor (Hi) (Hi) ----- 0/RP0 Board (P12V_V_MOIN) 12036 10148 10797 13157 13806 Board (P12V_AUX_V_MOIN) 12095 10148 10797 13157 13806 Board (P12V_STBY_V_MOIN) 12064 10150 10788 13166 13804 Board (P5V_V_MOIN) 5005 4301 4535 5452 5687 Board (P5V_AUX) 5026 4319 4555 5428 5688 Board (P3V3_V_MOIN) 3376 2848 3008 3584 3744 Board (P3V3_AUX) 3312 2842 3014 3580 3737 Board (P3V_BAT_V_MOIN) 2995 2246 2543 3588 3760 Board (P1V8_AUX) 1794 1591 1677 1911 1981 Board (P1V5_AUX) 1489 1326 1404 1599 1677 Board (P1V2_AUX) 1193 1061 1123 1279 1342 ... 上記の電圧測定値は UCS マザーボードからのものです。 </pre>

タスク	使用するコマンド																																												
ハードウェア障害の情報を表示する	<pre>sysadmin-vm:0_RP0# show logging i envmon</pre> <pre>Mon Oct 2 09:38:06.390 UTC 0/RP0/ADMIN0:Oct 1 16:58:44.394 : envmon[2332]: %PKT_INFRA-FM-6-FAULT_INFO : Power Module insertion :INFO :0/PT0-PM0: 0/RP0/ADMIN0:Oct 2 09:26:37.657 : envmon[2332]: %PKT_INFRA-FM-6-FAULT_INFO : Power Module insertion :INFO :0/PT0-PM1: 0/RP0/ADMIN0:Oct 2 09:37:03.605 : envmon[2332]: %PKT_INFRA-FM-6-FAULT_INFO : Power Module removal :INFO :0/PT0-PM1: 0/RP0/ADMIN0:Oct 2 09:37:50.221 : envmon[2332]: %PKT_INFRA-FM-6-FAULT_INFO : Power Module insertion :INFO :0/PT0-PM1:</pre> <p>上記のコマンドを実行する前に、電源モジュール（0/PT0-PM1）を取りはずして再度挿入します。電源モジュールは OIR に対応しています。</p> <p>上記のコマンドの出力では、強調表示されている行が、電源モジュールの取り外しと挿入情報をキャプチャしています。</p> <p>(注) 上記のコマンドの出力の最初の2つの挿入はシステムブートです。</p>																																												
アラームを表示する	<pre>sysadmin-vm:0_RP0# show alarms</pre> <pre>Thu Oct 19 12:28:59.400 UTC</pre> <hr/> <pre>Active Alarms</pre> <hr/> <table border="1"> <thead> <tr> <th>Location</th> <th>Severity</th> <th>Group</th> <th>Set time</th> </tr> </thead> <tbody> <tr> <td>0/PT0-PM0</td> <td>major</td> <td>environ</td> <td>10/19/17</td> </tr> <tr> <td>12:27:34</td> <td>Power Module Error</td> <td>(PM_OUTPUT_STAGE_OT).</td> <td></td> </tr> <tr> <td>0/PT0-PM0</td> <td>major</td> <td>environ</td> <td>10/19/17</td> </tr> <tr> <td>12:27:34</td> <td>Power Module Shutdown</td> <td>(PM_OC_SHUTDOWN).</td> <td></td> </tr> <tr> <td>0/PT0-PM1</td> <td>major</td> <td>environ</td> <td>10/19/17</td> </tr> <tr> <td>12:27:34</td> <td>Power Module Fault</td> <td>(PM_VOUT_VOLT_OOR).</td> <td></td> </tr> <tr> <td>0/RP0</td> <td>major</td> <td>environ</td> <td>10/19/17</td> </tr> <tr> <td>12:27:34</td> <td>Processor (P1_TEMP_SENS):</td> <td>temperature alarm.</td> <td></td> </tr> <tr> <td>0/RP0</td> <td>major</td> <td>environ</td> <td>10/19/17</td> </tr> <tr> <td>12:27:40</td> <td>Board (P3V3_AUX):</td> <td>low voltage alarm.</td> <td></td> </tr> </tbody> </table>	Location	Severity	Group	Set time	0/PT0-PM0	major	environ	10/19/17	12:27:34	Power Module Error	(PM_OUTPUT_STAGE_OT).		0/PT0-PM0	major	environ	10/19/17	12:27:34	Power Module Shutdown	(PM_OC_SHUTDOWN).		0/PT0-PM1	major	environ	10/19/17	12:27:34	Power Module Fault	(PM_VOUT_VOLT_OOR).		0/RP0	major	environ	10/19/17	12:27:34	Processor (P1_TEMP_SENS):	temperature alarm.		0/RP0	major	environ	10/19/17	12:27:40	Board (P3V3_AUX):	low voltage alarm.	
Location	Severity	Group	Set time																																										
0/PT0-PM0	major	environ	10/19/17																																										
12:27:34	Power Module Error	(PM_OUTPUT_STAGE_OT).																																											
0/PT0-PM0	major	environ	10/19/17																																										
12:27:34	Power Module Shutdown	(PM_OC_SHUTDOWN).																																											
0/PT0-PM1	major	environ	10/19/17																																										
12:27:34	Power Module Fault	(PM_VOUT_VOLT_OOR).																																											
0/RP0	major	environ	10/19/17																																										
12:27:34	Processor (P1_TEMP_SENS):	temperature alarm.																																											
0/RP0	major	environ	10/19/17																																										
12:27:40	Board (P3V3_AUX):	low voltage alarm.																																											

ホストレベルのモニタリング情報

アプライアンスおよび XRv 9000 VM のホスト OS レベルの使用率情報をモニタすることができます。システム管理モードで以下の show コマンドを使用して、情報を表示します。

タスク	使用するコマンド
CPU 情報を表示する	<pre> sysadmin-vm:0_RP0# show virtual-platform cpu System CPU utilization Linux 3.14.23-WR7.0.0.2_standard (host) 11/27/17 _x86_64_ (16 CPU) 02:27:49 CPU %usr %nice %sys %iowait %irq %soft %steal %guest %gnice %idle 02:27:49 all 4.06 0.00 0.00 4.66 0.01 0.00 0.06 0.00 0.00 0.00 91.21 02:27:49 0 0.84 0.00 1.72 0.02 0.00 0.30 0.00 0.00 0.00 97.12 02:27:49 1 2.08 0.00 2.31 0.01 0.00 0.10 0.00 0.00 0.00 95.50 02:27:49 2 0.99 0.00 1.73 0.01 0.00 0.05 0.00 0.00 0.00 97.22 ... 02:27:49 14 2.40 0.00 1.64 0.00 0.00 0.00 0.00 0.00 0.00 95.96 02:27:49 15 1.24 0.00 1.41 0.00 0.00 0.00 0.00 0.00 0.00 97.35 </pre>
ディスク情報を表示する	<pre> sysadmin-vm:0_RP0# show virtual-platform disk System Disk Utilization Filesystem 1K-blocks Used Available Use% Mounted on /dev/mapper/panini_vol_grp-host_lv0 991512 425304 498624 47% / ... /dev/mapper/panini_vol_grp-host_data_scratch_lv0 2007248 3036 1884200 1% /misc/scratch /dev/mapper/panini_vol_grp-host_data_config_lv0 95088 44 87876 1% /misc/config /dev/mapper/panini_vol_grp-host_data_log_lv0 479560 8080 435640 2% /var/log none 512 0 512 0% /mnt /dev/loop5 6060604 1330192 4399508 24% /lxc_rootfs/panini_vol_grp-xr_lv0 </pre>
メモリ情報を表示する	<pre> sysadmin-vm:0_RP0# show virtual-platform memory System Memory Usage MemTotal: 131982032 kB MemFree: 109636132 kB MemAvailable: 111675924 kB ... HugePages_Total: 12 ... Hugepagesize: 1048576 kB ... </pre> <p>上記のコマンドの出力で、MemFree 情報は、Cisco IOS XRv 9000 システムがメモリの枯渇状況にあるかどうかを確認するのに有用です。Hugepages フィールドの値は、VPE の問題のトリアージに役立ちます。</p>

タスク	使用するコマンド
プロセッサ情報を表示する	<pre>sysadmin-vm:0_RP0# show virtual-platform processor System Processor Information processor : 0 vendor_id : GenuineIntel cpu family : 6 ... flags : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx pdpe1gb rdtscp lm constant_tsc arch_perfmon pebs bts rep_good nopl xtopology nonstop_tsc aperfmperf eagerfpu pni pclmulqdq dtes64 monitor ds_cpl vmx smx est tm2 ssse3 fma cx16 xtpr pdcm pcid dca sse4_1 sse4_2 x2apic movbe popcnt tsc_deadline_timer aes xsave avx f16c rdrand lahf_lm abm 3dnowprefetch ida arat epb xsaveopt pln pts dtherm tpr_shadow vnmi flexpriority ept vpid fsgsbase tsc_adjust bmi1 hle avx2 smep bmi2 erms invpcid rtm rdseed adx smap ... processor : 1 ... 上記のコマンドはアプライアンス内の16個のコアの詳細情報を表示します。フラグの情報は、CPUがXRv9000 VMハイパーバイザ設定で適切に設定されているかどうかを確認するのに有用です。</pre>

UCS Cisco Integrated Management Controller (CIMC) を介した環境モニタリング

6.4.1 より前のリリースには、アプライアンスにXRベースの環境モニタリング機能はありません。

ただし、アプライアンスはUCSサーバ上に構築されているため、サーバの環境モニタリングは、CIMCを介してSNMPまたはIPMIのインターフェイスから直接実行することもできます。CIMCを介してSNMPまたはIPMIモニタリングを設定し、使用方法については、Cisco UCSサーバのドキュメントを参照してください。

SFP 光モニタリング情報

Cisco IOS XRv 9000 は、NIC に接続されている SFP から光接続の健全性情報を抽出します。この情報には、ベンダー名、製品番号、現在の受容電力および送出電力が含まれます。情報を表示するには、XR EXEC モードで **show controllers <interface> physical** コマンドを使用します。

```
RP/0/RP0/CPU0:SS_Node1# show controllers TenGigE 0/0/0/1 physical
SFP EEPROM port:1
  Xcvr Type: SFP
  Xcvr Code: SFP-10G-SR
  Encoding: 64B66B
  Bit Rate: 10300 Mbps
  Link Reach 50u fiber: 80 meter
  Link Reach 62.5u fiber: 20 meter
```

```
Vendor Name: CISCO-JDSU
Vendor OUI: 00.01.9c
Vendor Part Number: PLRXPL-SC-S43-CS (rev.: 1 )
Laser wavelength: 850 nm (fraction: 0.00 nm)
Optional SFP Signal: Tx_Disable, Tx_Fault, LOS
Vendor Serial Number: JUS1734G1L5
Date Code (yy/mm/dd): 13/10/13 lot code:
Diagnostic Monitoring: DOM, Int. Cal.,
Enhanced Options: SW RX LOS Mon., SW TX Fault Mon, SW TX Disable, Alarm/Warning
Flags
...
Temperature: 28.445
Voltage: 3.300 Volt
.
```

SFP OIR（活性挿抜）情報は、5 秒ごとにすべての SFP のステータスをポーリングすることでモニタされます。状態の変更がキャプチャされ、Syslog メッセージに反映されます。

```
RP/0/RP0/CPU0:SS_Node1# show logging | i envmon
```




第 10 章

Amazon Web Services (AWS) での Cisco IOS XRv 9000 ルータの展開

この章では、Amazon Web Services での Cisco IOS XRv 9000 ルータの展開の概要を示します。



(注) AWS で Cisco IOS XRv 9000 ルータを展開できるのは Cisco IOS XR リリース 6.3.1 以降です。

- [はじめに \(145 ページ\)](#)
- [Amazon Web Services 用の Cisco IOS XRv 9000 ルータ AMI オプション \(146 ページ\)](#)
- [Cisco IOS XRv 9000 ルータ時間課金 AMI \(146 ページ\)](#)
- [サポートされている Cisco IOS XR テクノロジー \(146 ページ\)](#)
- [Amazon Web Services での Cisco IOS XRv 9000 ルータの展開 \(148 ページ\)](#)
- [AWS での Cisco IOS XRv 9000 ルータの起動に関する情報 \(148 ページ\)](#)
- [Cisco IOS XRv 9000 ルータ AMI の起動 \(150 ページ\)](#)

はじめに

Cisco IOS XRv 9000 ルータは、パブリックおよびプライベートクラウドソリューションのために Amazon Web Service (AWS) で展開することができます。AWS での実装およびインストールは、サポートされている他のハイパーバイザとは異なります。Cisco IOS XRv 9000 ルータは、次の AWS プラットフォームでサポートされています。

- AWS Virtual Private Cloud (VPC)
- AWS Elastic Compute Cloud (EC2)

詳細については、次の場所にある AWS VPC ドキュメントを参照してください。

<http://aws.amazon.com/documentation/vpc/>

Amazon Web Services 用の Cisco IOS XRv 9000 ルータ AMI オプション

AWS 用の Cisco IOS XRv 9000 ルータは、[AWS Marketplace](#) で Amazon マシンイメージ (AMI) として購入され、起動されます。



(注) この Cisco IOS XRv 9000 ソフトウェアは通常の IOS XR アップグレード手順でアップグレードできます。IOS XR Release 6.3.1 より前のリリースにダウングレードすることはできません。

Cisco IOS XRv 9000 ルータ 時間課金 AMI

AWS 用の Cisco IOS XRv 9000 ルータは、[AWS Marketplace](#) で Amazon マシンイメージ (AMI) として購入され、起動されます。ここでは、時間課金 AMI について説明します。

AWS Marketplace から直接起動される Cisco IOS XRv 9000 ルータ 時間課金 AMI には、次の条件が適用されます。

- Cisco IOS XRv 9000 ルータ AMI を使用すると Amazon Web Services (AWS) によって時間単位で課金されます。この時間単位の使用料は、AWS から請求される VPC 使用料に追加されます。
- 時間単位で課金される Cisco IOS XRv 9000 ルータを使用するために各種ライセンスの購入、インストール、または設定の必要はありません。

サポートされていない IOS XR 機能については、「サポートされている Cisco IOS XR テクノロジー」を参照してください。

サポートされている IOS XR 機能については、[Cisco IOS XRv 9000 ルータのリリース ノートの Web サイト](#)にある最新のリリース ノートを参照してください。

サポートされている Cisco IOS XR テクノロジー

Cisco IOS XRv 9000 ルータを AWS インスタンスで展開する場合、サポートされる Cisco IOS XR テクノロジーの数は他のハイパーバイザよりも少なくなります。一部のテクノロジーは Amazon クラウドでサポートされていないために利用できないことがあります。

AWS インスタンスでの Cisco IOS XRv 9000 ルータの展開には次の制限が適用されます。

- サポートされていない機能の CLI コマンドが Cisco IOS XRv 9000 ルータで表示される場合がありますが、シスコによるテストでは、これらのサポートされていない機能は AWS 展開では機能しないことが判明しています。

- ルーティング プロトコルは、トンネル経由でのみサポートされます。

AWS インスタンスで Cisco IOS XRv 9000 ルータを展開する場合にサポートされる Cisco IOS XR テクノロジーの一覧を次に示します。

- 単一仮想ルータ VM : vPE
- ルーティング : AWS 内のスタティックおよび BGP に限定されるが、GREv4 トンネルによりダイナミック ルーティングのサポートが可能
- IPv4、IPv6 L3 フォワーディング
- IPv4/IPv6 ペイロードによる GREv4
- e1000
- QOS : IPv4/IPv6 QOS (ポリシング/マーキング/H-QOS/出力 TM) 、階層型ポリサー (適合認識)
- IPv4/6 ACL (チェーン接続)
- 厳密な IPv4/IPv6 uRPF
- LPTS ベースの CoPP
- EFD DOS 保護
- IPSLA

AWS インスタンスで Cisco IOS XRv 9000 ルータを展開する場合にサポートされない Cisco IOS XR テクノロジーの一覧を次に示します。

- 仮想ルート リフレクタ (vRR)
- アプリケーション ホスティング
- バンドル
- BFD IPv4 と BFD IPv6 のシングル ホップ (スタティックおよび BGP)
- VM シリアルポートおよびコンソール アクセス
- VLAN

Cisco IOS XRv 9000 ルータのテクノロジーパッケージに含まれる機能の詳細については、『[Cisco IOS XRv 9000 ルータ設置およびコンフィギュレーションガイド](#)』を参照してください。

AWS 展開での Cisco IOS XR テクノロジーのサポートには次の警告が適用されます。

- Amazon クラウド内の Cisco IOS XRv 9000 ルータノード間で HSRP を設定することはできません。Amazon は、VPC 内のホストで HSRP を実行することを許可していません。Amazon AWS は、VPC 内のすべてのブロードキャストトラフィックとマルチキャストトラフィックをブロックします。

Amazon Web Services での Cisco IOS XRv 9000 ルータの展開

AWS で Cisco IOS XRv 9000 ルータ の起動を試みる前に、次の前提条件を確認してください。

- Amazon Web Services アカウントを持っている必要があります。
- AWS では FireFox が他のブラウザよりも安定して動作するため推奨されています。
- Cisco IOS XRv 9000 ルータ AWS コンソールにアクセスするには、SSH クライアント (Windows 場合の PuTTY、Macintosh の場合はターミナルなど) が必要です。
- Cisco IOS XRv 9000 ルータ用に導入するインスタンス タイプを決定します。詳細については、次の項を参照してください。
- ワンクリック起動による AMI の起動を予定している場合は、まず仮想プライベートクラウド (VPC) を作成する必要があります。詳細については、[Amazon Virtual Private Cloud \(VPC\) ドキュメント](#)を参照してください。

AWS での Cisco IOS XRv 9000 ルータの起動に関する情報

Cisco IOS XRv 9000 ルータ AMI の起動は AWS Marketplace から直接行います。

Amazon EC2 インスタンスと Amazon VPC インスタンスのどちらで Cisco IOS XRv 9000 ルータを展開するかを決定してください。

Amazon VPC インスタンスを使用する場合は、[手動起動による Cisco IOS XRv 9000 ルータ AMI の起動 \(151 ページ\)](#) を参照してください。この項では、インスタンスを起動するためにキーペアを生成または既存のキー ペアを使用する必要があることについても説明しています。

VPC 内のジャンボフレームには制限があります。『[Network Maximum Transmission Unit \(MTU\) for Your EC2 Instance](#)』を参照してください。

サポートされているインスタンスタイプ

Amazon マシンイメージは、さまざまなインスタンスタイプをサポートしています。インスタンスタイプによって、インスタンスのサイズと必要なメモリ容量が決まります。

表 20: 機能の履歴 (表)

機能名	リリース	説明
Amazon EC2 M5 インスタンスでの Elastic Network Adapter (ENA) による拡張ネットワークワーキング機能	リリース 7.3.3	Amazon Elastic Compute Cloud (Amazon EC2) M5 インスタンスで Elastic Network Adapter (ENA) を使用してルータを起動すると、高いネットワークスループットを実現できます。Amazon EC2 M5 インスタンスは、より多くの CPU コア、より高速なディスク速度、およびより広いネットワーク帯域幅を提供し、ネットワークパフォーマンスを向上させます。

Cisco IOS XRv 9000 ルータについては、次の AMI インスタンスタイプがサポートされています。

表 21: サポートされている AMI インスタンスタイプ

インスタンスタイプ	vCPU	メモリ (GB)	NIC の最大数	IPv4 および IPv6 アドレスの最大数
m4.xlarge	4	16	4	15
m4.2xlarge	8	32	4	15
m4.4xlarge	16	64	8	30
m4.10xlarge	40	160	8	30
c4.xlarge	4	7.5	4	15
c4.2xlarge	8	15	2	15
c4.4xlarge	16	30	8	30
c4.8xlarge	36	60	8	30

Cisco IOS XRv 9000 ルータについては、次の追加の AWS EC2 M5 インスタンスタイプがサポートされています。

- m5
- m5n
- c5
- c5n

インスタンスタイプごとにサポートされるネットワーク インターフェイスの最大数、およびネットワーク インターフェイスごとのプライベート IPv4 アドレスと IPv6 アドレスの最大数の詳細については、『[AWS ユーザーガイド](#)』を参照してください。

これらの AWS インスタンスタイプの詳細については、Amazon Web Services ドキュメントの「[Amazon インスタンスタイプ](#)」のページを参照してください。

Cisco IOS XRv 9000 ルータ AMI の起動

Cisco IOS XRv 9000 ルータ AMI を起動するには、次の手順を実行します。

最初に、[Cisco IOS XRv 9000 ルータ AMI の選択 \(150 ページ\)](#) を参照してください。

Amazon EC2 または VPC インスタンスを使用している場合は、[手動起動による Cisco IOS XRv 9000 ルータ AMI の起動 \(151 ページ\)](#) を参照してください。

次に、[Elastic IP アドレスと Cisco IOS XRv 9000 ルータ インスタンスの関連付け \(153 ページ\)](#) および [SSH を使用した IOS XRv 9000 インスタンスへの接続 \(154 ページ\)](#) を参照してください。

Cisco IOS XRv 9000 ルータ AMI の選択

Cisco IOS XRv 9000 ルータ AMI を選択するには、次の手順に従います。

手順

ステップ 1 [Amazon Web Services Marketplace](#) にログインします。

ステップ 2 AWS Marketplace で「Cisco IOS XRv 9000」を検索します。次のような AMI の一覧が表示されます。

- Cisco IOS XRv 9000 Demo Version (hourly billing)

ステップ 3 展開する予定の Cisco IOS XRv 9000 ルータ AMI を選択します。

AMI 情報のページが表示され、サポートされているインスタンス タイプと、AWS によって課金される時間単位の使用料が表示されます。ユーザの地域における価格設定の詳細を選択します。

[Continue] をクリックします。

ステップ 4 AWS の電子メールアドレスとパスワードを入力または新しいアカウントを作成します。

「EC2 で起動するページ」が表示されます。

手動起動による Cisco IOS XRv 9000 ルータ AMI の起動

手順

-
- ステップ 1** [Launch with EC2] ページのドロップダウン リストから [Region] を選択します。
- ステップ 2** [Select a Version] ペインから Cisco IOS XRv 9000 ルータのリリース バージョンを選択します。ユーザの地域における時間単位の使用料が [Pricing Details] に表示されます。
- ステップ 3** ユーザの地域用の [Launch with EC2 Console] ボタンをクリックします。インスタンス タイプを選択するためのウィンドウが表示されます。サポートされているインスタンス タイプの [General purpose] タブを選択します。インスタンス タイプを選択します。[Next: Configure Instance Details] ボタンをクリックします。
- ステップ 4** インスタンスの詳細を設定します。
- ネットワークのドロップダウン リストからネットワークを選択します。[Subnet] ドロップダウンから、IOS XRv 9000 を展開する VPC サブネットを選択します。これによりインスタンスの可用性ゾーンが決定されることに注意してください。
- [Auto-assign Public IP] はデフォルト設定のままにしてください。
- 最初に [Instance Details] 画面で 2 つのインターフェイスを作成できます。その後、さらにインターフェイスを追加するには、**Network Interfaces** で [Add Device] をクリックします。サポートされるインターフェイスの最大数は、インスタンス タイプによって異なります。
- AWS から利用できる追加オプションを選択します。
- ステップ 5** (任意) [Next: Add Storage] ボタンをクリックします。
- ステップ 6** (任意) デフォルトのハードドライブ設定はそのままにします。
- (注)
AWS で Cisco IOS XRv 9000 ルータを使用する場合、仮想ハードドライブのサイズ (46 GB EBS) は変更できません。
- (任意) [Next: Add Tags] ボタンをクリックします。
- ステップ 7** (任意) 必要に応じてタグ情報を入力します。
- (任意) [Next: Configure Security Groups] ボタンをクリックします。
- ステップ 8** (任意) 次のいずれかを選択します。
- [Create a new Security Group]
 - [Select an existing Security Group]

Cisco IOS XRv 9000 ルータへのコンソールアクセスには SSH が必要です。また、Cisco IOS XRv 9000 ルータでは、セキュリティグループで少なくとも TCP/22 がブロックされないようにする必要があります。これらの設定は、Cisco IOS XRv 9000 ルータを管理するために使用されます。

[Review and Launch] ボタンをクリックします。

ステップ 9 Cisco IOS XRv 9000 ルータ インスタンス情報を確認します。

[Launch] をクリックします。

ステップ 10 プロンプトが表示されたら、キーペア情報を入力します。キーペアは、AWS に保存された公開キーと、インスタンスへのアクセスの認証に使用される秘密キーで構成されます。次のいずれかを実行します。

- a) 既存のキーペアを選択します。または、
- b) 次の手順を実行して新しいキーを作成します。
 - ユーザ独自の公開キーをアップロードします。
 - AWS で新しいキーペアを作成します。

[Create Key Pair.] をクリックし、キーペア名を入力して [Create] をクリックします。キーペアが作成されたら、続行する前に Amazon から秘密キーをダウンロードしていることを確認します。新しく作成された秘密キーには一度しかアクセスできません。キーペアがダウンロードされたら、[Close] をクリックします。

(注)

AWS セキュリティポリシーでは、秘密キーの権限レベルを 400 に設定する必要があります。この値を .pem ファイル用に設定するには、UNIX シェルターミナル画面を開き、**chmod 400 pem-file-name** コマンドを入力します。

ステップ 11 [Launch Instance] をクリックします。

AMI インスタンスの展開には約 10 分かかります。メニューの [Instances] リンクをクリックすることによりステータスを表示できます。

[State] に **Running** と表示され、[Status Checks] に **passed** と表示されるまで待ちます。

この時点で、Cisco IOS XRv 9000 ルータ AWS インスタンスが起動され、ソフトウェア設定の準備が完了します。[Elastic IP アドレスと Cisco IOS XRv 9000 ルータ インスタンスの関連付け \(153 ページ\)](#) および [SSH を使用した IOS XRv 9000 インスタンスへの接続 \(154 ページ\)](#) に進んでください。

デイゼロ設定

デイゼロ設定（ブートストラップ設定とも呼ばれる）は、ルータが初めて起動するときに適用される設定です。デイゼロ設定は CLI（コマンドラインインターフェイス）としてユーザデータボックスに入力する必要があります。次に例を示します。

```
username root
group root-lr
group cisco-support
```

```
secret 5 $1$920D$OrPQMgw1/3WdUe5R3RpLP/
!
interface TenGigE 0/0/0/0
ipv4 address 192.0.2.2/255.255.255.0
no shutdown
!
router static
address-family ipv4 unicast
0.0.0.0/0 192.0.2.2
!
!
ssh server v2
ssh server vrf default
```



(注) ユーザ独自のユーザ データ ボックスを使用する場合は、SSH を使用してボックスに接続するためにユーザ名を設定する必要があります。

Elastic IP アドレスと Cisco IOS XRv 9000 ルータ インスタンスの関連付け

SSH 接続を使用して管理コンソールにアクセスするには、Cisco IOS XRv 9000 ルータのインターフェイスと、VPC で作成された Elastic IP アドレスを関連付ける必要があります。次の操作を行ってください。

手順

ステップ 1 [Services] > [EC2] > [Instances] ページで Cisco IOS XRv 9000 インスタンスを選択します。

ステップ 2 表示されるネットワークインターフェイスで、[eth0] をクリックします。

ポップアップ ウィンドウが表示され、**eth0** インターフェイスに関する詳細情報が表示されます。

インターフェイスのプライベート IP アドレスを書き留めておきます。

ステップ 3 **Interface ID** をコピーします。

ステップ 4 [EC2 Dashboard] > [Network & Security] で、[Elastic IPs] をクリックします。

ステップ 5 一覧から IP を関連付ける Elastic IP を選択します。

ステップ 6 [Actions] ドロップダウンから、[Associate address] を選択します。

ステップ 7 [Associate Address] ページで、次の手順を実行します。

- a) [Resource Type] として [Network Interface] を選択します。
- b) ステップ 3 でコピーしたインターフェイス ID を [Network Interfaces] フィールドに貼り付けます。
- c) AWS によって割り当てられている **プライベート IP** アドレスをドロップダウンから選択し、[Allow Elastic IP to be reassociated if already attached] をオンにします。
- d) [Associate] をクリックします。

このアクションにより、Elastic IP アドレス (Amazon Elastic IP) がネットワーク インターフェイスのプライベート IP アドレスと関連付けられます。これで、このインターフェイスを使用して管理コンソールにアクセスできるようになります。[SSH を使用した IOS XRv 9000 インスタンスへの接続 \(154 ページ\)](#) を参照してください。

SSH を使用した IOS XRv 9000 インスタンスへの接続

AWS 上の Cisco IOS XRv 9000 ルータ インスタンスへのコンソールアクセスには SSH が必要です。Cisco IOS XRv 9000 ルータ AMI にアクセスするには、次の手順に従います。

手順

ステップ 1 Cisco IOS XRv 9000 ルータのステータスによって動作中であることが示されたら、そのインスタンスを選択します。

ステップ 2 次の UNIX シェル コマンドを入力して、SSH を使用して Cisco IOS XRv 9000 ルータ コンソールに接続します。

```
ssh -i pem-file-name root@[public-ipaddress | DNS-name]
```

(注)

初めてインスタンスにアクセスするときは **root** としてログインする必要があります。

.pem ファイルに保存されている秘密キーは、Cisco IOS XRv 9000 ルータ インスタンスへのアクセスを認証するために使用されます。

ステップ 3 Cisco IOS XRv 9000 ルータの設定を開始します。



第 11 章

付録

付録は、いくつかの参照トピックで構成されています。

- [仮想マシンの要件 \(155 ページ\)](#)
- [Cisco IOS XRv 9000 ルータを使用した仮想化のメリット \(158 ページ\)](#)
- [Cisco IOS XRv 9000 ルータ アーキテクチャ：ハードウェア プラットフォームとの相違点 \(159 ページ\)](#)
- [プラットフォームおよびシスコ ソフトウェア イメージのサポート情報 \(161 ページ\)](#)
- [VMware ESXi のサポート情報 \(162 ページ\)](#)
- [OpenStack での KVM のサポート \(168 ページ\)](#)

仮想マシンの要件

Cisco IOS XRv 9000 ルータは、仮想マシンでのみ実行されます。ここでは、ルータの仮想マシンの要件について説明します。

仮想マシン

仮想マシン (VM) は、オペレーティングシステム (OS) またはプログラムをインストールおよび実行できるコンピューティング環境のソフトウェア実装です。VMは通常、物理コンピューティング環境をエミュレートしますが、CPU、メモリ、ハードディスク、ネットワーク、およびその他のハードウェアリソースが必要です。これらは仮想化層によって管理され、仮想化層がこれらのリソースに対する要求を基盤となる物理ハードウェアへの要求に変換します。

VMを展開するために、Open Virtualization Archive (OVA) ファイルを使用できます。OVA ファイルは、新しいVMのパラメータとリソース割り当て要件の詳細定義を提供することにより、VMの展開プロセスを合理化します。

OVA ファイルは記述子 (.ovf) ファイル、ストレージ (.vmdk) ファイル、およびマニフェスト (.mf) ファイルで構成されます。

- **ovfファイル**：記述子ファイル、つまり、パッケージに関するすべてのメタデータで構成された、拡張子 .ovf の xml ファイル。これは、すべての製品の詳細、仮想ハードウェア要件、およびライセンスをエンコードします。

- vmdk ファイル：VM から 1 つの仮想ディスクをエンコードするファイル形式。
- mf ファイル：パッケージング時に生成される SHA キーを保存するオプションファイル。

.iso ファイルを使用して Cisco IOS XRv 9000 ルータをインストールし、ハイパーバイザに手動で VM を作成することもできます。

ハイパーバイザのサポート

ハイパーバイザは、単一のハードウェア ホスト マシンを複数のオペレーティング システムで共有できるようにします。各オペレーティング システムはホストのプロセッサ、メモリ、およびその他のリソースを専有するのようになりますが、ハイパーバイザは、実際にはオペレーティング システムで必要なリソースのみを制御して割り当て、オペレーティング システム (VM) が相互に干渉しないことを保証します。

Cisco IOS XRv 9000 ルータのインストールは、選択した Type 1 (ネイティブ、ベア メタル) ハイパーバイザ上でサポートされます。インストールは VMware Fusion、VMware Player、Virtual Box などの Type 2 (ホスト型) ハイパーバイザではサポートされていません。次の表に、最新の Cisco IOS XR ソフトウェア リリースでサポートされるハイパーバイザのバージョンを示します。

表 22: ハイパーバイザバージョンのサポートマトリックス

Cisco IOS XR のバージョン	VMWare ESXi	カーネルベース仮想マシン (KVM)
リリース 7.3.1	バージョン 6.7 以降	次のものに基づいた Linux KVM <ul style="list-style-type: none"> • Red Hat Enterprise Linux 7.1、7.2、7.3、7.4 • Ubuntu 14.04.03 LTS • Ubuntu 16.04 LTS • CentOS 7、7.1、7.2、7.3、7.4 • Openstack 10

ハイパーバイザで使用できる機能はタイプによって異なります。特定のバージョンでは、一部のハイパーバイザ機能がサポートされないことがあります。リストされているハイパーバイザバージョンは、Cisco IOS XRv 9000 ルータで正式にテストされたサポート対象のバージョンです。詳細については、次の項を参照してください。

- [VMware ESXi のサポート情報](#)

ハイパーバイザ NIC の要件

ハイパーバイザでサポートされる NIC の種類と NIC の最大数は、どのリリースの Cisco IOS XR を使用しているかによって異なります。一部の Cisco IOS XR ソフトウェアバージョンとハイパーバイザは、VM の電源を切ることなく NIC を追加および削除できる機能もサポートしています。この機能は、NIC のホット追加/削除と呼ばれます。

この表に、それぞれの VM インスタンスでサポートされる NIC を示します。

表 23: Cisco IOS XRv 9000 ルータの NIC サポート

Cisco IOS XR リリース	5.4	6.0.x、6.1.x、6.2.x、6.3.x、6.4.x、6.5.x	24.1.1
VMware ESXi			
サポートされる NIC の種類	E1000	E1000、トラフィックインターフェイス専用の VMXNET Generation 3 (VMXNET3)。	E810 (SR-IOV VF のみ)、X710、XXV710
VM インスタンスあたりの最大 NIC 数	11 (管理用に 1 つ、2 つは予約済み、トラフィック用に 8 つ)	11 (管理用に 1 つ、2 つは予約済み、トラフィック用に 8 つ)	11 (管理用に 1 つ、2 つは予約済み、トラフィック用に 8 つ)
NIC のホット追加/削除のサポート	非対応	非対応	非対応
Single Root I/O Virtualization (SR-IOV) のサポート	非対応	非対応	対応
物理 OIR のサポート	非対応	非対応	VM の電源を切り、OIR プロセスを完了し、VM の電源を入れる必要があります。
KVM			
サポートされる NIC の種類	VirtIO、ixgbe/ixgbev	VirtIO、ixgbe/ixgbev	VirtIO、ixgbe/ixgbev
VM インスタンスあたりの最大 NIC 数	11 (管理用に 1 つ、2 つは予約済み、トラフィック用に 8 つ)	11 (管理用に 1 つ、2 つは予約済み、トラフィック用に 8 つ)	11 (管理用に 1 つ、2 つは予約済み、トラフィック用に 8 つ)

NIC のホット追加/ 削除のサポート	非対応	非対応	非対応
Single Root I/O Virtualization (SR-IOV) のサ ポート	非対応	非対応	非対応

KVM 環境の Cisco IOS XRv 9000 ルータでサポートされる物理 NIC の詳細については、[KVM のインストール要件 \(19 ページ\)](#) の項を参照してください。

Cisco IOS XRv 9000 ルータとハイパーバイザの制限事項

Cisco IOS XRv 9000 ルータには次の制限があります。

- Cisco IOS XRv 9000 ルータ インターフェイスの帯域幅はすべての仮想化インターフェイスに関して、ハイパーバイザの物理 NIC 帯域幅に関係なく、デフォルトで 1 GB です。
- Cisco IOS XRv 9000 ルータが仮想化されたインターフェイスまたは仮想機能（物理パススルーではない）を使用していて、そのインターフェイスが物理ルータに直接接続されてそのルータの接続インターフェイスがダウンした場合、その変化は Cisco IOS XRv 9000 ルータには反映されません。これは、Cisco IOS XRv 9000 ルータが実際にはハイパーバイザの vSwitch に接続され、vSwitch のアップリンク ポートがルータの物理インターフェイスに接続されているからです。これは想定されている動作です。
- Cisco IOS XRv 9000 ルータは 64 ~ 9216 バイトの MTU 範囲を提供しますが、68 バイトの最小 MTU 値を使用することをお勧めします。ただし、VMWare ESXi vSwitches は最大 9000 バイトのフレーム サイズをサポートします。

Cisco IOS XRv 9000 ルータを使用した仮想化のメリット

Cisco IOS XRv 9000 ルータは、仮想化による次のメリットをクラウド環境で実現します。

表 24: 仮想化のメリット

利点	説明
ハードウェアへの非依存性	Cisco IOS XRv 9000 ルータは仮想マシン上で動作するため、仮想化プラットフォームでサポートされている任意の x86 ハードウェアでサポートされます。

利点	説明
リソースの共有	Cisco IOS XRv 9000 ルータで使用されるリソースはハイパーバイザによって管理されており、VM 間で共有できます。VM サーバが特定の VM に割り当てるハードウェア リソースの量は、サーバ上の別の VM に再割り当てできます。
導入の柔軟性	サーバ間で容易に VM を移動できます。したがって、ある物理的な場所にあるサーバから別の物理的な場所にあるサーバに、ハードウェア リソースを移動せずに Cisco IOS XRv 9000 ルータを移動できます。

Cisco IOS XRv 9000 ルータ アーキテクチャ：ハードウェア プラットフォームとの相違点

従来のシスコ ハードウェア ルータ プラットフォームとは異なり、Cisco IOS XRv 9000 ルータは、x86 マシン上で個別に実行される仮想ルータです。その結果、Cisco IOS XRv 9000 ルータ アーキテクチャには、ハードウェア ベースのルータ プラットフォームとの違いをもたらす独自の属性があります。

次の表で、Cisco IOS XRv 9000 ルータが Cisco ASR 9000 シリーズ ルータと異なる重要な分野を比較します。

表 25: Cisco IOS XRv 9000 シリーズ ルータ アーキテクチャと Cisco ASR 9000 シリーズ ルータとの相違点

機能	Cisco ASR 9000 シリーズ	Cisco IOS XRv 9000 シリーズ
分散ルーティング	1 つの RP と複数の LC で構成される分散ルーティングシステム。複数の LC はファブリック経由で相互接続されます。	1 つの RP と 1 つの LC の組み合わせで構成される一元化されたルーティングシステム。仮想化プラットフォームであるため、LC やファブリックはありません。
コントロールプレーンとデータプレーンの分離	コントロールプレーンとデータプレーンが同一シャーシに配置。	アーキテクチャとして、コントロールプレーンとデータプレーンの分離をサポート。データプレーン OIR をサポートします。

機能	Cisco ASR 9000 シリーズ	Cisco IOS XRv 9000 シリーズ
インターフェイス名	ラインインターフェイスは LC 上にホストされます。ラインインターフェイスの名前は、シャーシ内のインターフェイスの場所を示します。たとえば、Tenge 0/0/0/0 は、LC スロット 0 の最初のポートです。	ラインインターフェイスは RP 上にホストされます。ラインインターフェイスの名前は、特定のタイプのインスタンスを表します。たとえば、Tenge 0/0/0/0 は Tenge インターフェイスの最初のインスタンスです。
クラスタ	1つの論理ルータとして、ASR9000 ルータのクラスタをサポートします。	サポート対象外
サテライト インターフェイス	サテライト インターフェイスをサポートします。	サポート対象外
コントロールプレーンの冗長性	アクティブおよびスタンバイ RP をサポートします。	サポート対象外
動的なリソース割り当て	リソースは固定されています。	メモリと CPU はインストール中に動的に割り当てることができます。
物理リソース	ハードウェア プラットフォームのアーキテクチャによって管理されます。	メモリと CPU は VM プロビジョニング中に割り当てることができますが、変更を適用するには再起動する必要があります。
サポートされているコンソールタイプ	物理シリアルポート	<ul style="list-style-type: none"> • VGA コンソール • シリアルポート (デフォルト)
ROMMON	サポートあり	Cisco IOS XRv 9000 は ROMMON を含みませんが、GRUB を使用して、類似してはいるもののより制限された機能を提供します。
ISSU	対応	サポート対象外
インターフェイス モジュール	プラグ着脱可能なインターフェイス モジュールのインストールをサポートします。	サポート対象外

機能	Cisco ASR 9000 シリーズ	Cisco IOS XRv 9000 シリーズ
ポートの動的な追加/削除	サポートあり	サポートあり。ただし、VM をリロードする必要があります。 (注) VMware ESXi および KVM 環境のインターフェイスを追加または削除する前に、VM の電源をオフにしてください。

プラットフォームおよびシスコソフトウェアイメージのサポート情報

シスコのソフトウェアには、特定のプラットフォームに対応したソフトウェアイメージで構成されるフィーチャセットが含まれています。特定のプラットフォームで使用できるフィーチャセットは、リリースに含まれるシスコソフトウェアイメージによって異なります。個々のリリースで使用できるソフトウェアイメージのフィーチャセットを特定したり、特定の Cisco IOS XR ソフトウェア イメージで機能が使用できるかどうかを確認したりする場合は、Cisco Feature Navigator、Software Advisor、またはソフトウェアのリリース ノートを参照してください。

Cisco Feature Navigator

[Cisco Feature Navigator](#) を使用すると、プラットフォームおよびソフトウェアイメージのサポート情報を検索できます。Cisco Feature Navigator を使用すると、特定のソフトウェアリリース、フィーチャセット、またはプラットフォームをサポートする Cisco IOS XR ソフトウェア イメージを判別できます。Cisco Feature Navigator にアクセスするのに、Cisco.com のアカウントは必要ありません。

Software Advisor

ある機能が特定の Cisco IOS XR リリースでサポートされているかどうかを確認する場合、その機能に関するソフトウェアマニュアルを検索する場合、またはルータにインストールされている Cisco IOS XR ソフトウェアの最小要件を確認する場合は、Cisco.com の [Software Advisor ツール](#) を使用できます。このツールにアクセスするには、Cisco.com の登録ユーザである必要があります。

ソフトウェア リリース ノート

Cisco IOS XR ソフトウェア リリース ノートには、次の情報が含まれます。

- プラットフォーム サポート

- メモリに関する推奨事項
- 新機能
- 重大度 1 および 2 の未解決および解決済みの注意事項

リリースノートには、最新のリリースに固有の情報が記載されています。これらの情報には、以前のリリースに記載済みの機能に関する情報が含まれていないことがあります。機能の累積的な情報については、Cisco Feature Navigator を参照してください。

詳細については、『[Cisco IOS XRv 9000 Router Release Notes](#)』 ページを参照してください。

VMware ESXi のサポート情報

Cisco IOS XRv 9000 ルータは、VMware ESXi ハイパーバイザで実行されます。1 つの VMware ESXi ハイパーバイザを使用して複数の VM を実行できます。VM を作成して管理するには、VMware vSphere Client GUI を使用します。

VMware vSphere Client が VMware vCenter Server VM を作成、構成、管理するためのアプリケーションです。Cisco IOS XRv 9000 ルータは、データストアにある仮想ディスクから起動できます。VMware vSphere Client を使用して、Cisco IOS XRv 9000 ルータの起動やシャットダウンなどの基本的な管理作業を実行できます。

VMware vCenter Server は、vSphere 環境を管理し、単一のコンソールからデータセンターのすべてのホストと VM を統合管理できます。

この表は、Cisco IOS XRv 9000 ルータでサポートされている VMware 仮想マシンのベンダーツールを示しています。

表 26: VMware 仮想マシンの要件

Cisco IOS XRv 9000	サポートされているツールと要件	サポートされている vSwitch
リリース 7.3.1	VMware vSphere Client 5.5 または 6.0 を実行する PC VMware ESXi 6.7 以降のバージョンを実行しているサーバ VMware vCenter インストールツール	VMware 標準および分散スイッチ

サポートされている VMware 機能と操作

VMware では、仮想アプリケーションを管理したり、複製、移行、シャットダウン、復帰などの操作を実行したりするためのさまざまな機能と操作がサポートされています。

これらの操作の一部では、VMの実行時状態が保存され、再起動時に復元されます。実行時状態にトラフィック関連状態が含まれていると、実行時状態を回復したり再生したりするときに、ユーザコンソールに追加のエラー、統計情報、メッセージが表示される場合があります。設定のみに基づいて回復される保存状態の場合は、これらの機能と動作を問題なく使用できません。

この表は、Cisco IOS XRv 9000 ルータの Cisco IOS XR ソフトウェア最新リリースでサポートされている VMware 機能と操作を示しています。VMware 機能と操作の詳細については、VMware のドキュメンテーションを参照してください。

表 27: サポートされている VMware 機能と操作 : ストレージオプション (vCenter Server と vSphere Client の両方)

エンティティ	ステータス	説明
ローカルストレージ	サポートあり	ローカルストレージは ESXi ホスト内にある内部ハードディスクにあります。ローカルストレージデバイスは複数ホストにまたがる共有をサポートしません。ローカルストレージデバイス上のデータストアは1台のホストによってのみアクセスできます。
外部ストレージターゲット	サポートあり	外部ストレージ、つまり、ストレージエリアネットワーク (SAN) に Cisco IOS XRv 9000 ルータを展開できます。
USB ストレージのマウントまたはパススルー	サポート対象外	Cisco IOS XRv 9000 ルータに USB スティックを接続し、ストレージデバイスとして使用できます。VMware ESXi では、Cisco IOS XRv 9000 ルータに USB コントローラを追加してディスクデバイスを割り当てる必要があります。

次の表は、Cisco IOS XR ソフトウェア最新リリースでサポートされている (またはサポートされていない) 機能を示しています。

表 28: サポートされている VMware 機能と操作 : 一般機能 (vCenter Server のみ)

エンティティ	ステータス	説明
複製	サポートあり	仮想マシンまたはテンプレートを複製したり、仮想マシンをテンプレートに複製したりできます。
移行	サポート対象外	データストレージがまだ共有ストレージの同じ場所にある間でも、仮想マシンの状態全体および必要に応じた設定ファイルが新しいホストに移動されます。

エンティティ	ステータス	説明
VMotion	サポート対象外	VM の実行中に、ある物理サーバから別の物理サーバに VM を移動できます。
テンプレート	サポートあり	テンプレートを仮想マシンとして複製することにより、テンプレートを使用して新しい仮想マシンを作成します。

この表は、vCenter Server と vSphere Client の両方の Cisco IOS XR ソフトウェア最新リリースでサポートされている VMware 機能と操作を示しています。

表 29: サポートされている VMware 機能と操作 : 操作 (vCenter Server と vSphere Client の両方)

エンティティ	ステータス	説明
電源オン	サポートあり	仮想マシンの電源を投入し、ゲストオペレーティングシステムがインストールされていればゲストオペレーティングシステムをブートします。
電源オフ	サポートあり	電源が再度オンになるまで仮想マシンを停止します。電源オフオプションは、「強制」電源オフを行います。これは、物理マシンの電源コードを引き抜くことに相当し、常に機能します。
シャットダウン	サポート対象外	シャットダウン（「安全な」電源オフ）は、VMware ツールを使用してゲストオペレーティングシステムのグレースフルシャットダウンを実行します。特定の状況（VMware ツールがインストールされていない場合や、ゲストオペレーティングシステムが停止している場合など）では、正常にシャットダウンできないことがあり、電源オフオプションを使用する必要があります。
一時停止	サポート対象外	仮想マシンを一時停止します。
リセット/再起動	サポートあり	仮想マシンを停止し、再起動（リブート）します。
OVF の作成	サポートあり	OVF パッケージは自己完結型パッケージに仮想マシンの状態をキャプチャします。ローカルコンピュータにエクスポートして OVF ファイルを作成できます。

エンティティ	ステータス	説明
OVA の作成	サポートあり	1 つの .ova ファイルに OVF テンプレートをパッケージングする単一ファイル (OVA) です。これにより、OVF パッケージを明示的に Web サイトからダウンロードするか USB キーを使用して持ち運ぶ必要がある場合に、OVF パッケージを単一のファイルとして配布できるようになります。

この表は、Cisco IOS XR ソフトウェア最新リリースでサポートされている VMware 機能と操作 (ネットワーク機能) を示しています。

表 30: サポートされている VMware 機能と操作 : ネットワーク機能

エンティティ	ステータス	説明
カスタム MAC アドレス	サポートあり	vCenter Server と vSphere Client の両方から。仮想ネットワーク アダプタの MAC アドレスを手動で設定できます。
分散 vSwitch	サポートあり	vCenter Server からのみ。vCenter Server データセンター上の 1 台の vSphere 分散型スイッチで、データセンターのすべての関連ホストに対するネットワーク トラフィックを処理できます。
分散リソーススケジューラ	サポート対象外	ホスト間の自動ロード バランシングを提供します。
NIC ロード バランシング	サポート対象外	vCenter Server と vSphere Client の両方から。ロード バランシング ポリシーとフェールオーバー ポリシーにより、アダプタが故障した場合に、アダプタ間でネットワーク トラフィックを分散する方法と、トラフィックを再ルーティングする方法を指定できます。

エンティティ	ステータス	説明
NIC チーミング	サポート対象外	<p>vCenter Server と vSphere Client の両方から。各仮想スイッチが NIC チームを形作る 2 個のアップリンク アダプタに接続する環境をセットアップできます。これにより、NIC チームでは、メンバーの一部または全体をまたがって、物理ネットワークと仮想ネットワーク間のトラフィックの負荷を共有するか、ハードウェア障害やネットワーク障害の発生時にパッシブ フェールオーバーを提供します。</p> <p>(注) NIC チーミングにより、多数の ARP パケットが Cisco IOS XRv 9000 ルータにフラッディングし、CPU が過負荷になるおそれがあります。この状況を回避するには、ARP パケットの数を減らし、アクティブ-アクティブではなく、アクティブ-スタンバイとして NIC チーミングを実装します。</p>
vSwitch	サポートあり	<p>vCenter Server と vSphere Client の両方から。vSwitch はレイヤ 2 物理スイッチの仮想化バージョンです。vSwitch では、仮想マシン間でトラフィックを内部的にルーティングでき、外部ネットワークにリンクできます。vSwitch を使用すると、複数ネットワークアダプタの帯域幅を組み合わせ、このアダプタ間で通信トラフィックを分散できます。物理 NIC フェールオーバーを処理するように vSwitch を設定することもできます。</p>

この表は、Cisco IOS XR ソフトウェア最新リリースでサポートされていない VMware 機能および操作（ハイアベイラビリティ）を示しています。

表 31: サポートされていない VMware 機能と操作 : ハイ アベイラビリティ

エンティティ	ステータス	説明
VM レベルのハイ アベイラビリティ	サポート対象外	オペレーティング システムの障害をモニタするために、VM レベルのハイ アベイラビリティでは、VMware ハイ アベイラビリティ クラスタのハートビート情報をモニタします。ユーザ指定の間隔までに、指定した仮想マシンからハートビートが受信されていないと、障害が検出されます。VM レベルのハイ アベイラビリティは、VMware vCenter Server を使用して VM のリソース プールを作成することによって有効化されます。
ホストレベルのハイアベイラビリティ	サポート対象外	物理サーバをモニタするために各サーバ上のエージェントでは、ハートビートが失われたときに、リソース プール内の他のサーバにある影響を受けるすべての仮想マシンの再起動を自動的に開始できるように、リソースプール内の他のサーバとのハートビートを維持します。ホスト レベルのハイアベイラビリティは、サーバまたはホストのリソース プールを作成し、vSphere でハイアベイラビリティを有効にすることによって有効になります。
耐障害性	サポート対象外	ハイ アベイラビリティを使用することで、ESXi ホストの耐障害性が有効になります。Cisco IOS XRv 9000 ルータを実行する VM の耐障害性を有効にすると、クラスタ内の別のホストにセカンダリ VM が作成されます。プライマリ ホストが停止すると、セカンダリ ホストの VM が Cisco IOS XRv 9000 ルータのプライマリ VM を引き継ぎます。



(注) Cisco IOS XRv 9000 ルータは、アクティブ/スタンバイ コントロールプレーン冗長性をサポートしていません。

OpenStack での KVM のサポート

Cisco IOS XRv 9000 ルータでは、OpenStack 環境での KVM のインストールがサポートされています。OpenStack でのサポートには、Cisco.com のダウンロード ページから入手可能な qcow2 インストール ファイルが必要です。

サポートされている OpenStack および Red Hat Enterprise Linux バージョンの詳細については、最新の [Cisco IOS XR ソフトウェア向け Cisco IOS XRv 9000 ルータ リリース ノート](#) 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。