Cisco ASR 9000 シリーズ アグリゲーション サービス ルータ リリース 6.0.x ルーティング設定ガイド

初版：2016 年 4 月 28 日

最終更新：2016 年 5 月 6 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/

目 次

はじめに

リリース 6.1.2 以降、シスコは 64 ビット Linux ベースの IOS XR オペレーティング システムのサポートを導入しています。32 ビット環境と 64 ビット環境の間で、広範な機能パリティが維持されます。特に明記されていない限り、このドキュメントの内容は両方の環境に適用されます。Cisco IOS XR 64 ビットの詳細については、ドキュメント『Release Notes for Cisco ASR 9000 シリーズルータ, Release 6.1.2』を参照してください。

『*Routing Configuration Guide for Cisco ASR 9000 Series Routers*』の「はじめに」には、次の項が含まれています。

マニュアルの変更履歴

日付	変更点
2016 年 4 月	このマニュアルの初版

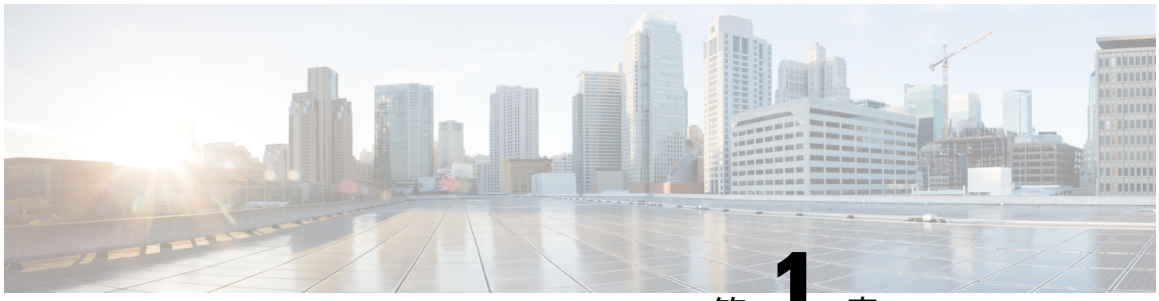
通信、サービス、およびその他の情報

- 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 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、シスコ サービス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、シスコ サポート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 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 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 にアクセスしてください。

Cisco Bug Search Tool

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。



第 1 章

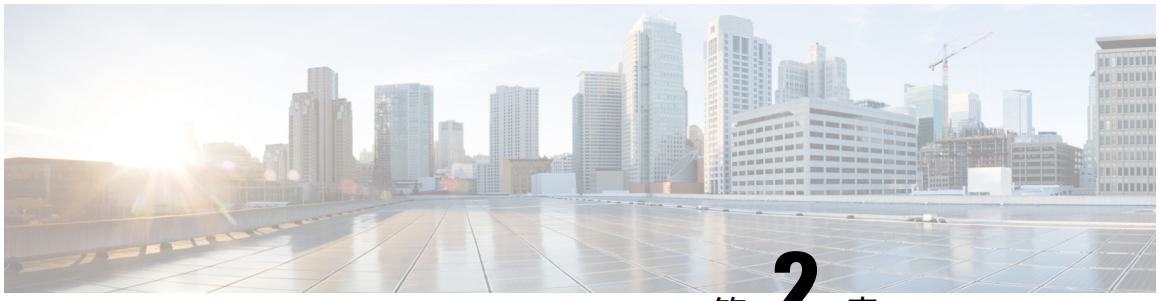
新規および変更されたルーティング機能

次の表では、 *Routing Configuration Guide for Cisco ASR 9000 Series Routers*における新機能および変更された機能に関する情報を要約し、その参照先を示しています。

• 新規および変更されたルーティング機能に関する情報 （1 ページ）

新規および変更されたルーティング機能に関する情報

機能	説明	導入または変更されたリリース	参照先
BGP と OSPF での過剰なパントフロートラップ	この機能が導入されました。	リリース 6.0.1	「*BGP* の実装」の章
BGP 用 64-ECMP のサポート	この機能が導入されました。	リリース 6.0.1	「*BGP* の実装」の章



第 2 章

BGP の実装

ボーダー ゲートウェイ プロトコル（BGP）は、自律システム間にループフリーのドメイン間ルーティングを作成可能なエクステリア ゲートウェイ プロトコル（EGP）です。自律システムは、単一の技術管理に基づくルータのまとまりです。自律システム内のルータは、複数の内部ゲートウェイ プロトコル（IGP）を使用して自律システム内のルーティング情報を交換し、EGP を使用して自律システム外でパケットをルーティングします。

ここでは、Cisco IOS XR ソフトウェアでの BGP の概念と設定情報を説明します。

（注） BGP の詳細とこのモジュールに示す BGP コマンドの詳細な説明については、このモジュールの関連資料（219 ページ）の項を参照してください。設定作業の実行中に必要になることのある他のコマンドのドキュメントを見つけるには、Cisco ASR 9000 シリーズ ルータ ソフトウェア マスター コマンド索引で、オンライン検索してください。

BGP の実装の機能履歴

リリース	変更内容
リリース 3.7.2	この機能が導入されました。
リリース 3.9.0	次の機能がサポートされました。 • BGP プレフィックス独立コンバージェンス ユニパス プライマリ バックアップ • BGP Local Label Retention • 4 バイト自律システム番号の asplain 表記 • BGP ノンストップルーティング • BGP コマンドに対するコマンドライン インターフェイス（CLI）の一貫性 • L2VPN アドレス ファミリ コンフィギュレーション モード

リリース	変更内容
リリース 4.0.0	次の機能がサポートされました。 • BGP Add Path アドバタイズメント • 累積 iGP（AiGP） • プレルート • IPv4 BGP-Policy Accounting • IPv6 uRPF
リリース 4.1.0	5000 BGP NSR セッションのサポートの追加
リリース 4.1.1	次の機能が追加されました。 • BGP Accept Own • 不等コストの連続ロードバランシングに対する BGP DMZ リンク帯域幅
リリース 4.2.0	次の機能がサポートされました。 • 選択的 VRF ダウンロード • BGP Multi-Instance/Multi-AS • BGP の BFD マルチホップ サポート • BGP のエラー処理 分散 BGP（bgp 分散スピーカー）の設定のサポートが削除されました。
リリース 4.2.1	次の機能がサポートされました。 • グローバル プレフィックス用 BGP 3107 PIC アップデート • RIB および FIB 用 BGP プレフィックス独立コンバージェンス • RPKI に基づく BGP プレフィックスの発信元検証
リリース 4.2.3	BGP 属性のフィルタリング機能が追加されました。
リリース 4.3.0	アップデート生成のための BGP-RIB のフィードバック メカニズム機能が追加されました。
リリース 4.3.1	次の機能がサポートされていました。 • BGP VRF ダイナミック ルートのリーク **label-allocation-mode** コマンドは **label mode** コマンドに名前が変更されています。

リリース	変更内容
リリース 4.3.2	次の機能がサポートされました。 • ネイバー単位のリンク帯域幅
リリース 5.3.1	次の機能がサポートされました。 • L3VPN iBGP-PE-CE の設定 • 送信元ベースのフロータグ • 過剰パスの破棄
リリース 5.3.2	次の機能がサポートされました。 • グレースフルメンテナンス • ネイバー単位の TCP MSS • BGP DMZ 総帯域幅
リリース 6.0.1	次の機能がサポートされました。 • 過剰パントフロートラップ処理 • BGP 用 64-ECMP

BGP の実装の前提条件

適切なタスク ID を含むタスク グループに関連付けられているユーザ グループに属している必要があります。このコマンド リファレンスには、各コマンドに必要なタスク ID が含まれます。ユーザ グループの割り当てが原因でコマンドを使用できないと考えられる場合、AAA 管理者に連絡してください。

BGP の実装に関する概要

BGP を実装するには、次の概念を理解する必要があります。

BGP 機能の概要

BGP はトランスポート プロトコルとして TCP を使用します。2 台の BGP ルータが互いの間に TCP 接続を形成し（ピア ルータ）、接続パラメータを開いて確認するためにメッセージを交換します。

BGP ルータはネットワーク到達可能性情報を交換します。この情報は、主に、宛先ネットワークに到達するためにルートで経由する必要のあるフル パス（BGP 自律システム番号）を示します。この情報は、ループ フリーである自律システムや、ルーティング動作に制限が適用されるルーティング ポリシーを表すグラフの作成に役立ちます。

TCP 接続を確立して BGP ルーティング情報を交換している 2 台のルータは、ピアまたはネイバーと呼ばれます。BGP ピアは最初に BGP ルーティング テーブル全体を交換します。この交換の後、ルーティング テーブルが変更されたとき差分更新が送信されます。BGP は BGP テーブルのバージョン番号を保存します。これはすべての BGP ピアで同一です。ルーティング情報の変更によって BGP がテーブルを更新するたびに、バージョン番号は変更されます。BGP ピア間の接続が維持されていることを確認するキープアライブ パケットが送信され、エラーまたは特殊な状態に応じて通知パケットが送信されます。

（注） `address-family ipv4 rtfilter` コマンドを使用して、RTC（ルートターゲット制約）を有効にするだけです。BGP EVPN の RTC を有効にするためには個別の設定は必要ありません。

（注） マルチ プロトコル ラベル スイッチング（MPLS）レイヤ 3 バーチャル プライベート ネットワーク（VPN）情報を配信するように BGP を設定する方法については、『*Cisco ASR 9000 Series Aggregation Services Router MPLS Configuration Guide*』を参照してください。

BGP による双方向フォワーディング検出（BFD）のサポートについては、『*Cisco ASR 9000 Series Aggregation Services Router Interface and Hardware Configuration Guide*』および『*Cisco ASR 9000 Series Aggregation Services Router Interface and Hardware Command Reference*』を参照してください。

BGP ルータ ID

ネイバー間に BGP セッションを確立するには、BGP にルータ ID を割り当てる必要があります。ルータ ID は、BGP セッションが確立されると、OPEN メッセージに含めて BGP ピアに送信されます。

BGP は次の方法（プリファレンス順）でルータ ID の取得を試みます。

- ルータ コンフィギュレーション モードで **bgp router-id** コマンドを使用して設定されたアドレスを使用する。
- 保存されたループバック アドレス設定を使用してルータがブートされた場合に、システムのループバック インターフェイス上の最大の IPv4 アドレスを使用する。
- 保存された設定に存在しない場合に、設定される最初のループバック アドレスのプライマリ IPv4 アドレスを使用する。

このいずれの方法でもルータ ID を取得できない場合、BGP はルータ ID を持たず、BGP ネイバーとのピアリング セッションを確立できません。そのような場合は、エラー メッセージがシステム ログに記録され、**show bgp summary** コマンドでは、ルータ ID として 0.0.0.0 が表示されます。

ルータ ID を取得した BGP では、さらに適したルータ ID が使用可能になっても、同じルータ ID の使用を続行します。この使用方法によって、いずれの BGP セッションでも不要なフラッピングが発生しないようにします。一方、現在使用中のルータ ID が無効になった場合（インターフェイスがダウンするか、設定が変更されたことによる）、BGP では新しいルータ ID を選択し（上記のルールを使用）、確立したすべてのピアリングセッションをリセットします。

（注） ルータ ID の不要な変更（およびそれによる BGP セッションのフラッピング）を避けるために、**bgp router-id** コマンドを設定することを、強く推奨します。

BGP 最大プレフィックス：過剰パスの破棄

IOS XR BGP の最大プレフィックス機能では、特定のアドレスファミリのネイバーから受信されるプレフィックスの数に上限が課されます。受信されるプレフィックスの数が設定した最大数を超えると、停止通知がネイバーに送信された後、BGPセッションが終了します（これはデフォルト動作です）。手動によるクリアがユーザによって実行されるまで、セッションはダウンしたままになります。セッションは、**clear bgp** コマンドを使用して再開できます。**restart** キーワードを指定した **maximum-prefix** コマンドを使用して、セッションが自動的に起動されるまでの期間を設定できます。プレフィックスの上限はユーザが設定できます。ユーザがそのアドレスファミリに対するプレフィックスの最大数を設定していない場合は、デフォルトの制限値が使用されます。デフォルトの制限については、BGP のデフォルト制限 （8 ページ）を参照してください。

過剰パスの破棄

追加パスを廃棄するオプションが、最大プレフィックス設定に追加されました。過剰パスの破棄オプションを設定すると、プレフィックスが設定した最大値を超えた場合に、ネイバーから受信された過剰なプレフィックスはすべて廃棄されます。ただし、この廃棄によってセッション フラップが発生することはありません。

過剰パスの破棄オプションの利点は次のとおりです。

- BGP のメモリ フットスタンプが制限されます。
- パスが設定された制限を超えるとピアのフラッピングが停止します。

過剰パスの破棄設定が削除されると、BGP は更新機能をサポートしている場合にルート更新メッセージをネイバーに送信します。それ以外の場合、セッションはフラップします。

同じ回線で、最大プレフィックス値が変更された場合のアクションを次に示します。

- 最大値が単独で変更されると、必要に応じてルート更新メッセージが送信されます。
- 新しい最大値が現在のプレフィックス カウント ステートよりも大きい場合、新しいプレフィックス ステートが保存されます。
- 新しい最大値が現在のプレフィックス カウント ステートより小さい場合、新しく設定されたステートの値に一致するように、既存のプレフィックスが一部削除されます。

どのプレフィックスを削除するかを制御する方法は現在ありません。

詳細な設定手順については、過剰パスの破棄の設定 （108 ページ）を参照してください。

機能制限

これらの制約事項は、過剰パスの破棄機能に適用されます。

- ルータがプレフィックスを廃棄すると、ネットワークの残りと一致せず、ルーティング ループが起きる可能性があります。
- プレフィックスが廃棄されると、スタンバイおよびアクティブ状態の BGP セッションが別のプレフィックスを廃棄する可能性があります。その結果、NSR スイッチオーバーによって BGP テーブルの矛盾が生じます。
- 過剰パスの破棄設定は、ソフト再設定構成と共存できません。

BGP のデフォルト制限

Cisco IOS XR BGP では、ルータに設定できるネイバーの最大数、および特定のアドレス ファミリのピアから受け入れるプレフィックスの最大数に制限を設定しています。この制限は、ルータにとって、ローカルまたはリモート ネイバーのいずれかの設定ミスに起因する、リソースの枯渇に対する予防措置となります。BGP 設定には、次の制限が適用されます。

- 設定できるピアのデフォルトの最大数は 4000 です。このデフォルトは、**bgp maximum neighbor** コマンドを使用して変更できます。制限の範囲は 1 ～ 15000 です。 最大制限値を超えてさらにピアを設定しようとしたり、現在設定されているピアの数未満の最大制限値を設定しようとしたりすると失敗します。
- アドバタイズメントによりピアが BGP をフラッディングしないようにするために、サポートされているアドレス ファミリごとに、1 つのピアから受け入れるプレフィックスの数に対する制限が課されます。デフォルトの制限値は、該当するアドレス ファミリのピアに対して maximum-prefix *limit* コマンドを設定することにより、上書きできます。ユーザがそ

のアドレス ファミリに対するプレフィックスの最大数を設定していない場合は、次のデフォルト制限値が使用されます。

- IPv4 ユニキャスト：1048576
- IPv4 ラベル付きユニキャスト：131072
- IPv4 トンネル：1048576
- IPv6 ユニキャスト：524288
- IPv6 ラベル付きユニキャスト：131072
- IPv4 マルチキャスト：131072
- IPv6 マルチキャスト：131072
- IPv4 MVPN：2097152
- VPNv4 ユニキャスト：2097152
- IPv4 MDT：131072
- VPNv6 ユニキャスト：1048576
- L2VPN EVPN：2097152

特定のアドレス ファミリのピアから受信したプレフィックスの数が、このアドレス ファミリに対する最大制限値（デフォルト設定またはユーザ設定のいずれかによる）を超えると、停止通知メッセージがそのネイバーに送信され、このネイバーとのピアリングが終了されます。

特定のアドレスファミリのネイバーとのピアリングが確立され、そのネイバーから一定数のプレフィックスをすでに受信した後で、そのネイバーのプレフィックスの最大数が設定されていることがあります。設定されたプレフィックスの最大数が、アドレスファミリのネイバーからすでに受信したプレフィックスの数よりも小さい場合は、設定直後に停止通知メッセージがそのネイバーに送信され、そのネイバーとのピアリングが終了されます。

BGP ネクスト ホップ トラッキング

ネクストホップ情報が変更されると、BGPはルーティング情報ベース（RIB）から通知を受信します（イベント駆動型の通知）。BGP は RIB からネクスト ホップ情報を取得して次の処理を行います。

- ネクスト ホップが到達可能であるかどうかを確認する。
- ネクスト ホップへの完全再帰 IGP メトリックを見つける（最適パス計算で使用）。
- 受信したネクスト ホップを検証する。
- 発信ネクスト ホップを計算する。

- ネイバーの到達可能性および接続を確認する。

BGP は、次のいずれかのイベントが発生したときに通知を受けます。

- ネクスト ホップが到達不能になった。
- ネクスト ホップが到達可能になった。
- ネクスト ホップへの完全な繰り返し IGP メトリックが変更される。
- ファースト ホップの IP アドレスまたはファースト ホップのインターフェイスが変更される。
- ネクスト ホップが接続された。
- ネクスト ホップが接続解除された。
- ネクスト ホップがローカル アドレスになった。
- ネクスト ホップが非ローカル アドレスになった。

（注） 到達可能性および再帰メトリック イベントは、最適パスの再計算をトリガーします。

RIB からのイベント通知は、クリティカルおよび非クリティカルとして分類されます。クリティカルおよび非クリティカルイベントの通知は、別々のバッチで送信されます。ただし、非クリティカルイベントが保留中であり、クリティカルイベントを読み込む要求がある場合は、非クリティカル イベントがクリティカル イベントとともに送信されます。

- クリティカル イベントは、ネクスト ホップの到達可能性（到達可能と到達不能）、接続性（接続と非接続）、および局在性（ローカルと非ローカル）に関係があります。これらのイベントの通知は遅延しません。
- 非クリティカルイベントには、IGP メトリックの変更のみが含まれます。これらのイベントは 3 秒の間隔で送信されます。メトリック変更イベントは最後の 1 つが送信されてから 3 秒後にバッチ処理され、送信されます。

クリティカルおよび非クリティカル イベントのネクスト ホップ トリガー遅延は、nexthop trigger-delay コマンドを使用して、クリティカルおよび非クリティカル イベントの最小バッチ間隔を指定するように設定できます。トリガー遅延は、アドレス ファミリに依存します。

BGP ネクストホップ トラッキング機能では、次の特性を持つルートを持つネクスト ホップだけを BGP ルートの解決に使用するように指定することができます。

- 集約ルートを回避するために、プレフィックスの長さは指定された値よりも長くなっている。
- 振動につながる可能性のあるネクストホップの解決に BGP ルートが使用されないように、選択したリストにソース プロトコルが含まれている。

このルート ポリシーのフィルタリングが可能なのは、RIB により、ネクスト ホップを解決するルートのソース プロトコル、およびこのルートに関連付けられているマスクの長さが特定されるからです。nexthop route-policy コマンドは、ルート ポリシーを指定するために使用します。

ネクストホップ接続点を使用したネクストホップのルートポリシーのフィルタリングについては、『Cisco ASR 9000 シリーズ アグリゲーション サービス ルータ ルーティング設定ガイド』の「Cisco ASR 9000 シリーズ ルータ での ルーティング ポリシー言語の実装」のモジュールを参照してください。

範囲を指定した IPv4/VPNv4 テーブル ウォーク

処理するアドレス ファミリを判別するために、ネクスト ホップと関連付けられたゲートウェイ コンテキストを逆参照し、次に、ゲートウェイ コンテキストを調べてそのゲートウェイ コンテキストを使用しているアドレス ファミリを判別することにより、ネクスト ホップ通知が受信されます。IPv4 ユニキャストと VPNv4 ユニキャスト アドレス ファミリは、RIB 内の IPv4 ユニキャスト テーブルに登録されるため、同じゲートウェイ コンテキストを共有します。その結果、RIB から IPv4 ユニキャスト ネクスト ホップ通知を受信したときは、グローバル IPv4 ユニキャスト テーブルと VPNv4 テーブルの両方が処理されます。ネクスト ホップでマスクを保持することで、そのネクスト ホップが、IPv4 ユニキャストまたは VPNv4 ユニキャスト、あるいはその両方に属しているかどうかを示します。この範囲を指定したテーブル ウォークにより、適切なアドレス ファミリ テーブル内に処理が限定されます。

アドレス ファミリ処理の並べ替え

Cisco IOS XR ソフトウェア では、アドレス ファミリの数値に基づいてアドレス ファミリ テーブルを探索します。ネクスト ホップ通知バッチを受信すると、アドレス ファミリ処理の順序が、次の順序に並べ替えられます。

- IPv4 トンネル
- VPNv4 ユニキャスト
- IPv4 ラベル付きユニキャスト
- IPv4 ユニキャスト
- IPv4 マルチキャスト
- IPv6 ユニキャスト

ネクスト ホップ処理の新規スレッド

spkr プロセスの critical-event スレッドでは、ネクスト ホップ、双方向フォワーディング検出（BFD）、および高速外部フェールオーバー（FEF）の通知のみを処理します。この critical-event スレッドによって、BGP コンバージェンスは、大量の時間を必要とするおそれのある他のイベントによる悪影響が確実に受けなくなります。

show、clear、debug コマンド

show bgp nexthops コマンドは、ネクスト ホップ通知に関する統計情報、この通知の処理に費やした時間、およびRIBに登録されている各ネクストホップに関する詳細を表示します。**clear bgp nexthop performance-statistics** コマンドは、モニタリングを容易にするために、ネクストホップの **show** コマンドの処理部分に関する累積統計情報をクリアします。**clear bgp nexthop registration** コマンドは、ネクスト ホップを RIB に非同期的に登録します。ネクストホップの **show** コマンドおよび **clear** コマンドについては、 *Routing Command Reference for Cisco ASR 9000 Series Routers*の「*BGP Commands on* Cisco ASR 9000 シリーズ ルータ 」のモジュールを参照してください。

debug bgp nexthop コマンドは、ネクスト ホップ処理の情報を表示します。**out** キーワードを指定すると、RIB に登録されている BGP のネクスト ホップに関するデバッグ情報のみが表示されます。 **in** キーワードを指定した場合は、RIB から受信したネクストホップ通知に関するデバッグ情報が表示されます。 **out** キーワードでは、RIB に送信されたネクストホップ通知に関するデバッグ情報が表示されます。『 *Cisco ASR 9000 Series Aggregation Services Router Routing Debug Command Reference*』の「*BGP Debug Commands on Cisco ASR 9000 Series Aggregation Services Router* 」のモジュールを参照してください。

BGP の自律システム番号形式

自律システム番号（ASN）は、自律システム（AS）を識別するために使用されるグローバルに一意な識別子であり、これにより、AS では、ネイバー AS との間で外部ルーティング情報を交換できるようになります。一意の ASN は、BGP ルーティングで使用するために各 AS に割り当てられます。BGP では、ASN を 2 バイトの番号および 4 バイトの番号としてエンコードします。

2 バイト自律システム番号形式

2 バイト ASN は asplain 表記で表されます。2 バイトの範囲は 1 ～ 65535 です。

4 バイト自律システム番号形式

2 バイト自律システム番号（ASN）がいつか枯渇するときに備えて、BGP では 4 バイト ASN をサポートしています。4 バイト ASN は、asplain 表記と asdot 表記の両方で表されます。

asplain 表記での 4 バイト ASN のバイトの範囲は 1 ～ 4294967295 です。AS は 4 バイトの 10 進数として表されます。4 バイト ASN の asplain 表現は draft-ietf-idr-as-representation-01.txt で定義されています。

asdot 形式の 4 バイト ASN の場合は、4 バイトの範囲は 1.0 ～ 65535.65535 で、次の形式になります。

high-order-16-bit-value-in-decimal . low-order-16-bit-value-in-decimal

BGP の 4 バイト ASN 機能は、4 バイト AS 番号をサポートしていない BGP スピーカーをまたがって、4 バイトをベースとする AS パス情報を伝播するために使用されます。ASN のサイズを 2 バイトから 4 バイトに拡張するための情報については、draft-ietf-idr-as4bytes-12.txt を参照してください。AS は 4 バイトの 10 進数として表されます。

as-format コマンド

as-format コマンドは、ASN 表記を asdot に設定します。**as-format** コマンドを設定していない場合のデフォルト値は asplain です。

BGP の設定

Cisco IOS XR ソフトウェアでの BGP は、特定のネイバーに対するすべての設定を、ネイバー設定の下の 1 箇所にまとめる必要のある、ネイバーベースの設定モデルに従っています。ネイバー間での設定の共有と、アップデート メッセージの共有のいずれについても、ピア グループはサポートされていません。ピア グループの概念は、BGP 設定でテンプレートとして使用する一連の設定グループおよびネイバー間でアップデート メッセージを共有するために自動生成されるアップデート グループによって置き換えられました。

コンフィギュレーション モード

BGP コンフィギュレーションは、モードにグループ化されています。ここではいくつかの BGP コンフィギュレーション モードの開始方法について説明します。現行のモードで **?** コマンドを入力すると、そのモードで使用可能なコマンドを表示できます。

ルータ コンフィギュレーション モード

次に、ルータ コンフィギュレーション モードを開始する例を示します。

```
RP/0/RSP0/cpu 0: router# configuration
RP/0/RSP0/cpu 0: router(config)# router bgp 140
RP/0/RSP0/cpu 0: router(config-bgp)#
```

ルータ アドレス ファミリ コンフィギュレーション モード

次に、ルータ アドレス ファミリ コンフィギュレーション モードを開始する例を示します。

```
RP/0/RSP0/cpu 0: router(config)# router bgp 112
RP/0/RSP0/cpu 0: router(config-bgp)# address-family ipv4 unicast
RP/0/RSP0/cpu 0: router(config-bgp-af)#
```

ネイバー コンフィギュレーション モード

次に、ネイバー コンフィギュレーション モードを開始する例を示します。

```
RP/0/RSP0/cpu 0: router(config)# router bgp 140
RP/0/RSP0/cpu 0: router(config-bgp)# neighbor 10.0.0.1
RP/0/RSP0/cpu 0: router(config-bgp-nbr)#
```

ネイバー アドレス ファミリ コンフィギュレーション モード

次に、ネイバー アドレス ファミリ コンフィギュレーション モードを開始する例を示します。

```
RP/0/RSP0/cpu 0: router(config)# router bgp 112
RP/0/RSP0/cpu 0: router(config-bgp)# neighbor 10.0.0.1
RP/0/RSP0/cpu 0: router(config-bgp-nbr)# address-family ipv4 unicast
RP/0/RSP0/cpu 0: router(config-bgp-nbr-af)#
```

VRF コンフィギュレーション モード

次に、VPN ルーティングおよび転送（VRF）コンフィギュレーション モードを開始する例を示します。

```
RP/0/RSP0/cpu 0: router(config)# router bgp 140
RP/0/RSP0/cpu 0: router(config-bgp)# vrf vrf_A
RP/0/RSP0/cpu 0: router(config-bgp-vrf)#
```

VRF アドレス ファミリ コンフィギュレーション モード

次に、VRF アドレス ファミリ コンフィギュレーション モードを開始する例を示します。

```
RP/0/RSP0/cpu 0: router(config)# router bgp 112
RP/0/RSP0/cpu 0: router(config-bgp)# vrf vrf_A
RP/0/RSP0/cpu 0: router(config-bgp-vrf)# address-family ipv4 unicast
RP/0/RSP0/cpu 0: router(config-bgp-vrf-af)#
```

VRF アドレスファミリでの復元力のある *CE* 単位のラベルモードの設定

VRF アドレスファミリに復元力のある CE 単位のラベルモードを設定するには、次のタスクを実行します。

（注） 復元力のある CE 単位の 6PE ラベル割り当ては、CRS-1 ルータと CRS-3 ルータではサポートされていません。ASR 9000 ルータでのみサポートされています。

手順の概要

1. **configure**
2. **router bgp***as-number*
3. **vrf***vrf-instance*
4. **address-family {ipv4 | ipv6} unicast**
5. **label mode per-ce**
6. 次のいずれかを実行します。
   - **end**
   - **commit**

手順の詳細

ステップ 1 **configure**

例：

```
RP/0/RSP0/cpu 0: router# configure
RP/0/RSP0/cpu 0: router(config)#
```

グローバル コンフィギュレーション モードを開始します。

ステップ 2 **router bgp***as-number*

例：

```
RP/0/RSP0/cpu 0: router(config)# router bgp 666
RP/0/RSP0/cpu 0: router(config-bgp)#
```

自律システム番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。

ステップ 3 **vrf***vrf-instance*

例：

```
RP/0/RSP0/cpu 0: router(config-bgp)# vrf vrf-pe
RP/0/RSP0/cpu 0: router(config-bgp-vrf)#
```

VRF インスタンスを設定します。

ステップ 4 **address-family {ipv4 | ipv6} unicast**

例：

```
RP/0/RSP0/cpu 0: router(config-bgp-vrf)# address-family ipv4 unicast
RP/0/RSP0/cpu 0: router(config-bgp-vrf-af)#
```

IPv4 または IPv6 のいずれかのアドレス ファミリ ユニキャストを指定し、アドレス ファミリのコンフィギュレーション サブモードを開始します。

ステップ 5 **label mode per-ce**

例：

```
RP/0/RSP0/cpu 0: router(config-bgp-vrf-af)# label mode per-ce
RP/0/RSP0/cpu 0: router(config-bgp-vrf-af)#
```

復元力のある CE 単位のラベルモードを設定します。

ステップ 6 次のいずれかを実行します。

- **end**
- **commit**

例：

```
RP/0/RSP0/cpu 0: router(config-bgp-vrf-af)# end
```

または

```
RP/0/RSP0/cpu 0: router(config-bgp-vrf-af)# commit
```

設定変更を保存します。

- **end** コマンドを実行すると、変更をコミットするように要求されます。

```
Uncommitted changes found, commit them before exiting(yes/no/cancel)?[cancel]:
```

  - **yes** と入力すると、実行コンフィギュレーション ファイルに変更が保存され、コンフィギュレーション セッションが終了して、ルータが EXEC モードに戻ります。
  - **no** と入力すると、コンフィギュレーションセッションが終了して、ルータが EXEC モードに戻ります。変更はコミットされません。
  - **cancel** と入力すると、現在のコンフィギュレーションセッションが継続します。コンフィギュレーション セッションは終了せず、設定変更もコミットされません。

- 実行コンフィギュレーションファイルに変更を保存し、コンフィギュレーションセッションを継続するには、**commit** コマンドを使用します。

ルートポリシーを使用した復元力のある *CE* 単位のラベルモードの設定

ルートポリシーを使用して復元力のある CE 単位のラベルモードを設定するには、次のタスクを実行します。

（注） 復元力のある CE 単位の 6PE ラベル割り当ては、CRS-1 ルータと CRS-3 ルータではサポートされていません。ASR 9000 ルータでのみサポートされています。

手順の概要

1. **configure**
2. **route-policy** *policy-name*
3. **set label mode per-ce**
4. 次のいずれかを実行します。
   - **end**
   - **commit**

手順の詳細

ステップ 1 **configure**

例：

```
RP/0/RSP0/cpu 0: router# configure
RP/0/RSP0/cpu 0: router(config)#
```

グローバル コンフィギュレーション モードを開始します。

ステップ 2 **route-policy** *policy-name*

例：

```
RP/0/RSP0/cpu 0: router(config)# route-policy route1
RP/0/RSP0/cpu 0: router(config-rpl)#
```

ルート ポリシーを作成し、ルート ポリシー コンフィギュレーション モードを開始します。

ステップ 3 **set label mode per-ce**

例：

```
RP/0/RSP0/cpu 0: router(config-rpl)# set label mode per-ce
RP/0/RSP0/cpu 0: router(config-rpl)#
```

復元力のある CE 単位のラベルモードを設定します。

ステップ 4 次のいずれかを実行します。

- **end**
- **commit**

例：

```
RP/0/RSP0/cpu 0: router(config-rpl)# end
```

または

```
RP/0/RSP0/cpu 0: router(config-rpl)# commit
```

設定変更を保存します。

- **end** コマンドを実行すると、変更をコミットするように要求されます。

```
Uncommitted changes found, commit them before exiting(yes/no/cancel)?[cancel]:
```

  - **yes** と入力すると、実行コンフィギュレーション ファイルに変更が保存され、コンフィギュレーション セッションが終了して、ルータが EXEC モードに戻ります。
  - **no** と入力すると、コンフィギュレーションセッションが終了して、ルータが EXEC モードに戻ります。変更はコミットされません。

- **cancel** と入力すると、現在のコンフィギュレーションセッションが継続します。コンフィギュレーション セッションは終了せず、設定変更もコミットされません。

- 実行コンフィギュレーションファイルに変更を保存し、コンフィギュレーションセッションを継続するには、**commit** コマンドを使用します。

VRF ネイバー コンフィギュレーション モード

次に、VRF ネイバー コンフィギュレーション モードを開始する例を示します。

```
Router(config)# router bgp 140
Router(config-bgp)# vrf vrf_A
Router(config-bgp-vrf)# neighbor 11.0.1.2
Router(config-bgp-vrf-nbr)#
```

VRF ネイバー アドレス ファミリ コンフィギュレーション モード

次に、VRF ネイバー アドレス ファミリ コンフィギュレーション モードを開始する例を示します。

```
RP/0/RSP0/cpu 0: router(config)# router bgp 112
RP/0/RSP0/cpu 0: router(config-bgp)# vrf vrf_A
RP/0/RSP0/cpu 0: router(config-bgp-vrf)# neighbor 11.0.1.2
RP/0/RSP0/cpu 0: router(config-bgp-vrf-nbr)# address-family ipv4 unicast
RP/0/RSP0/cpu 0: router(config-bgp-vrf-nbr-af)#
```

VPNv4 アドレス ファミリ コンフィギュレーション モード

次に、VPNv4 ネイバー アドレス ファミリ コンフィギュレーション モードを開始する例を示します。

```
RP/0/RSP0/cpu 0: router(config)# router bgp 152
RP/0/RSP0/cpu 0: router(config-bgp)# address-family vpnv4 unicast
RP/0/RSP0/cpu 0: router(config-bgp-af)#
```

L2VPN アドレス ファミリ コンフィギュレーション モード

次に、L2VPN ネイバー アドレス ファミリ コンフィギュレーション モードを開始する例を示します。

```
RP/0/RSP0/cpu 0: router(config)# router bgp 100
RP/0/RSP0/cpu 0: router(config-bgp)# address-family l2vpn vpls-vpws
RP/0/RSP0/cpu 0: router(config-bgp-af)#
```

ネイバーサブモード

Cisco IOS XR BGP では、ネイバーサブモードを使用することにより、 **neighbor** キーワードおよびネイバーアドレスによってすべての設定にプレフィックスを付けることなく、設定を入力できます。

- Cisco IOS XR ソフトウェアにはネイバー用のサブモードがあり、このモードではすべてのコマンドに「neighbor *x.x.x.x*」というプレフィックスを付ける必要がなくなります。

  Cisco IOS XR ソフトウェアでの設定は次のとおりです。

```
RP/0/RSP0/cpu 0: router(config-bgp)# neighbor 192.23.1.2
RP/0/RSP0/cpu 0: router(config-bgp-nbr)# remote-as 2002
RP/0/RSP0/cpu 0: router(config-bgp-nbr)# address-family ipv4 unicast
```

- ネイバー コンフィギュレーション サブモード内のアドレス ファミリ コンフィギュレーションサブモードは、アドレス ファミリ固有のネイバー設定の入力に使用できます。Cisco IOS XR ソフトウェアでの設定は次のとおりです。

```
RP/0/RSP0/cpu 0: router(config-bgp)# neighbor 2002::2
RP/0/RSP0/cpu 0: router(config-bgp-nbr)# remote-as 2023
RP/0/RSP0/cpu 0: router(config-bgp-nbr)# address-family ipv6 unicast
RP/0/RSP0/cpu 0: router(config-bgp-nbr-af)# next-hop-self
RP/0/RSP0/cpu 0: router(config-bgp-nbr-af)# route-policy one in
```

- ネイバーアドレス ファミリ コンフィギュレーション サブモードで、ネイバー固有の IPv4、IPv6、VPNv4、または VPNv6 コマンドを入力する必要があります。Cisco IOS XR ソフトウェア での設定は次のとおりです。

```
RP/0/RSP0/cpu 0: router(config)# router bgp 109
RP/0/RSP0/cpu 0: router(config-bgp)# neighbor 192.168.40.24
RP/0/RSP0/cpu 0: router(config-bgp-nbr)# remote-as 1
RP/0/RSP0/cpu 0: router(config-bgp-nbr)# address-family ipv4 unicast
RP/0/RSP0/cpu 0: router(config-bgp-nbr-af)# maximum-prefix 1000
```

- VRF ネイバー アドレス ファミリ コンフィギュレーション サブモードで、ネイバー固有の IPv4 および IPv6 コマンドを入力する必要があります。Cisco IOS XR ソフトウェア での設定は次のとおりです。

```
RP/0/RSP0/cpu 0: router(config)# router bgp 110
RP/0/RSP0/cpu 0: router(config-bgp)# vrf vrf_A
RP/0/RSP0/cpu 0: router(config-bgp-vrf)# neighbor 11.0.1.2
RP/0/RSP0/cpu 0: router(config-bgp-vrf-nbr)# address-family ipv4 unicast
RP/0/RSP0/cpu 0: router(config-bgp-vrf-nbr-af)# route-policy pass all in
```

コンフィギュレーション テンプレート

af-group、**session-group**、および **neighbor-group** コンフィギュレーション コマンドは、Cisco IOS XR ソフトウェアでのネイバー設定にテンプレートのサポートを提供します。

af-group コマンドは、アドレスファミリ固有のネイバーコマンドを IPv4、IPv6、または IPNv4、アドレスファミリ内でグループ化するために使用します。同じアドレス ファミリ コンフィギュレーションを持つネイバーは、アドレス ファミリ固有の設定のアドレス ファミリ グループ（af-group）の名前を使用できます。ネイバーは、**use** コマンドを使用してアドレスファミリグループから設定を継承します。ネイバーがアドレス ファミリ グループを使用するように設定してある場合、ネイバーでは（デフォルトで）アドレス ファミリ グループから設定全体を継承します。ただし、そのネイバーに対して明示的に設定されている項目がある場合、ネイバーでは、設定の一部をアドレス ファミリ グループから継承しません。アドレス ファミリ グループ コンフィギュレーションは、BGP ルータ コンフィギュレーション モードで入力します。次に、アドレス ファミリ グループ コンフィギュレーション モードを開始する例を示します。

```
RP/0/RSP0/cpu 0: router(config)# router bgp 140
RP/0/RSP0/cpu 0: router(config-bgp)# af-group afmcast1 address-family ipv4 unicast
RP/0/RSP0/cpu 0: router(config-bgp-afgrp)#
```

アドレスファミリに依存しないコンフィギュレーションをネイバーが継承してくるセッショングループを作成するには、**session-group** コマンドを使用します。ネイバーは、**use** コマンドを使用してセッショングループから設定を継承します。ネイバーがセッション グループを使用するように設定してある場合、ネイバーでは（デフォルトで）セッション グループの設定全体を継承します。そのネイバーに直接設定されている場合、ネイバーでは一部の設定をセッション グループから継承しません。次に、セッション グループ コンフィギュレーション モードを開始する例を示します。

```
RP/0/RSP0/cpu 0: router# router bgp 140
RP/0/RSP0/cpu 0: router(config-bgp)# session-group session1
RP/0/RSP0/cpu 0: router(config-bgp-sngrp)#
```

neighbor-group コマンドを使用すると、1 つ以上のネイバーに同一の設定を適用しやすくなります。ネイバー グループにはセッション グループとアドレス ファミリ グループを含めることができ、またネイバーに対する全体的な設定を含めることができます。ネイバーグループを設定すると、**use** コマンドを使用してネイバーはグループの設定を継承できます。ネイバー グループを使用するようにネイバーを設定してある場合、ネイバーでは、ネイバー グループの BGP 設定全体を継承します。

次に、ネイバー グループ コンフィギュレーション モードを開始する例を示します。

```
RP/0/RSP0/cpu 0: router(config)# router bgp 123
RP/0/RSP0/cpu 0: router(config-bgp)# neighbor-group nbrgroup1
RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp)#
```

次に、ネイバー グループ アドレス ファミリ コンフィギュレーション モードを開始する例を示します。

```
RP/0/RSP0/cpu 0: router(config)# router bgp 140
RP/0/RSP0/cpu 0: router(config-bgp)# neighbor-group nbrgroup1
RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp)# address-family ipv4 unicast
RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp-af)#
```

- ただし、そのネイバーに対して明示的に設定されている項目がある場合、ネイバーでは、設定の一部をネイバー グループから継承しません。また、セッション グループまたはアドレス ファミリ グループも使用されている場合は、ネイバー グループの設定の一部が非表示になることがあります。

Cisco IOS XR ソフトウェア での設定のグループ化は、次の効果を持ちます。

- セッション グループ レベルでコマンドを入力すると、アドレス ファミリに依存しないコマンドが定義されます（ネイバー サブモードでの同じコマンドと同様）。
- アドレス ファミリ グループ レベルでコマンドを入力すると、指定したアドレス ファミリに対するアドレス ファミリ依存のコマンドが定義されます（ネイバー アドレス ファミリ コンフィギュレーション サブモードでの同じコマンドと同様）。
- ネイバーグループレベルでコマンドを入力すると、アドレスファミリに依存しないコマンドと、アドレスファミリ依存するコマンドが各アドレスファミリに定義され（使用可能なすべての **neighbor** コマンドと同様）、アドレスファミリグループのコマンドとセッショングループのコマンドに **use** コマンドが定義されます。

テンプレート継承ルール

Cisco IOS XR ソフトウェアの場合、BGP ネイバーまたはグループは、他の設定グループから設定を継承します。

アドレス ファミリに依存しない設定

- ネイバーは、セッション グループおよびネイバー グループから継承できます。
- ネイバー グループは、セッション グループおよび他のネイバー グループから継承できます。
- セッション グループは、他のセッション グループから継承できます。
- セッション グループとネイバー グループを使用しているネイバーの場合は、セッション グループでの設定が、ネイバー グループのグローバル アドレス ファミリ設定よりも優先されます。

アドレス ファミリ依存の設定

- アドレス ファミリ グループは、他のアドレス ファミリ グループから継承できます。
- ネイバー グループは、アドレス ファミリ グループおよび他のネイバー グループから継承できます。
- ネイバーは、アドレス ファミリ グループおよびネイバー グループから継承できます。

設定グループ継承ルールは、次のように優先順位付けされます。

1. 項目がネイバーに直接設定されている場合は、その値が使用されます。次の例では、ネイバー グループとネイバー設定の両方にアドバタイズメント間隔が設定されており、ネイバー設定からのアドバタイズメント間隔が使用されています。

```
RP/0/RSP0/cpu 0: router(config)# router bgp 140
RP/0/RSP0/cpu 0: router(config-bgp)# neighbor-group AS_1
RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp)# advertisement-interval 15
RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp)# exit
RP/0/RSP0/cpu 0: router(config-bgp)# neighbor 10.1.1.1
RP/0/RSP0/cpu 0: router(config-bgp-nbr)# remote-as 1
RP/0/RSP0/cpu 0: router(config-bgp-nbr)# use neighbor-group AS_1
RP/0/RSP0/cpu 0: router(config-bgp-nbr)# advertisement-interval 20
```

show bgp neighbors コマンドからの次の出力は、使用されたアドバタイズメント間隔が 20 秒であることを示しています。

```
RP/0/RSP0/cpu 0: router# show bgp neighbors 10.1.1.1

BGP neighbor is 10.1.1.1, remote AS 1, local AS 140, external link
 Remote router ID 0.0.0.0
  BGP state = Idle
  Last read 00:00:00, hold time is 180, keepalive interval is 60 seconds
  Received 0 messages, 0 notifications, 0 in queue
  Sent 0 messages, 0 notifications, 0 in queue
  Minimum time between advertisement runs is 20 seconds

 For Address Family: IPv4 Unicast
  BGP neighbor version 0
  Update group: 0.1
  eBGP neighbor with no inbound or outbound policy; defaults to 'drop'
  Route refresh request: received 0, sent 0
  0 accepted prefixes
  Prefix advertised 0, suppressed 0, withdrawn 0, maximum limit 524288
  Threshold for warning message 75%

  Connections established 0; dropped 0
  Last reset 00:00:14, due to BGP neighbor initialized
  External BGP neighbor not directly connected.
```

2. 上記と異なり、セッション グループまたはネイバー グループから継承する設定と、ネイバー上での直接設定のある項目の場合は、ネイバー上の設定が使用されます。セッション グループまたはアドレス ファミリ グループから継承するように設定されている一方で、直接設定されている値のないネイバーの場合は、セッション グループまたはアドレス ファミリ グループにある値が使用されます。次の例では、ネイバー グループとセッション グループにアドバタイズメント間隔が設定されており、セッション グループからのアドバタイズメント間隔値が使用されています。

```
RP/0/RSP0/cpu 0: router(config)# router bgp 140
RP/0/RSP0/cpu 0: router(config-bgp)# session-group AS_2
RP/0/RSP0/cpu 0: router(config-bgp-sngrp)# advertisement-interval 15
RP/0/RSP0/cpu 0: router(config-bgp-sngrp)# exit
RP/0/RSP0/cpu 0: router(config-bgp)# neighbor-group AS_1
RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp)# advertisement-interval 20
RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp)# exit
RP/0/RSP0/cpu 0: router(config-bgp)# neighbor 192.168.0.1
RP/0/RSP0/cpu 0: router(config-bgp-nbr)# remote-as 1
RP/0/RSP0/cpu 0: router(config-bgp-nbr)# use session-group AS_2
RP/0/RSP0/cpu 0: router(config-bgp-nbr)# use neighbor-group AS_1
```

show bgp neighbors コマンドからの次の出力は、使用されたアドバタイズメント間隔が 15 秒であることを示しています。

```
RP/0/RSP0/cpu 0: router# show bgp neighbors 192.168.0.1

BGP neighbor is 192.168.0.1, remote AS 1, local AS 140, external link
 Remote router ID 0.0.0.0
  BGP state = Idle
  Last read 00:00:00, hold time is 180, keepalive interval is 60 seconds
  Received 0 messages, 0 notifications, 0 in queue
  Sent 0 messages, 0 notifications, 0 in queue
  Minimum time between advertisement runs is 15 seconds

 For Address Family: IPv4 Unicast
  BGP neighbor version 0
  Update group: 0.1
  eBGP neighbor with no inbound or outbound policy; defaults to 'drop'
  Route refresh request: received 0, sent 0
  0 accepted prefixes
  Prefix advertised 0, suppressed 0, withdrawn 0, maximum limit 524288
  Threshold for warning message 75%

  Connections established 0; dropped 0
  Last reset 00:03:23, due to BGP neighbor initialized
  External BGP neighbor not directly connected.
```

3. 上記の例と異なり、ネイバー グループを使用し、セッション グループもアドレス ファミリ グループも使用しないネイバーの場合は、直接または継承によってネイバー グループから設定値を取得できます。次の例では、ネイバーに直接設定されておらず、セッション グループを使用していないため、ネイバー グループからのアドバタイズメント間隔が使用されます。

```
RP/0/RSP0/cpu 0: router(config)# router bgp 150
RP/0/RSP0/cpu 0: router(config-bgp)# session-group AS_2
RP/0/RSP0/cpu 0: router(config-bgp-sngrp)# advertisement-interval 20
RP/0/RSP0/cpu 0: router(config-bgp-sngrp)# exit
RP/0/RSP0/cpu 0: router(config-bgp)# neighbor-group AS_1
RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp)# advertisement-interval 15
RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp)# exit
RP/0/RSP0/cpu 0: router(config-bgp)# neighbor 192.168.1.1
RP/0/RSP0/cpu 0: router(config-bgp-nbr)# remote-as 1
RP/0/RSP0/cpu 0: router(config-bgp-nbr)# use neighbor-group AS_1
```

show bgp neighbors コマンドからの次の出力は、使用されたアドバタイズメント間隔が 15 秒であることを示しています。

```
 RP/0/RSP0/cpu 0: router# show bgp neighbors 192.168.1.1

BGP neighbor is 192.168.2.2, remote AS 1, local AS 140, external link
 Remote router ID 0.0.0.0
  BGP state = Idle
  Last read 00:00:00, hold time is 180, keepalive interval is 60 seconds
  Received 0 messages, 0 notifications, 0 in queue
  Sent 0 messages, 0 notifications, 0 in queue
  Minimum time between advertisement runs is 15 seconds

 For Address Family: IPv4 Unicast
```

```
 BGP neighbor version 0
 Update group: 0.1
 eBGP neighbor with no outbound policy; defaults to 'drop'
 Route refresh request: received 0, sent 0
 Inbound path policy configured
 Policy for incoming advertisements is POLICY_1
 0 accepted prefixes
 Prefix advertised 0, suppressed 0, withdrawn 0, maximum limit 524288
 Threshold for warning message 75%

 Connections established 0; dropped 0
 Last reset 00:01:14, due to BGP neighbor initialized
 External BGP neighbor not directly connected.
```

同じルールを説明するために、次の例では、アドバタイズメント間隔に 15（セッション グループから）および25（ネイバー グループから）を設定する方法を示します。セッション グループのアドバタイズメント間隔設定は、ネイバー グループの設定よりも優先されます。インバウンド ポリシーには、ネイバー グループから POLICY_1 が設定されます。

```
RP/0/RSP0/cpu 0: routerconfig)# router bgp 140
RP/0/RSP0/cpu 0: router(config-bgp)# session-group ADV
RP/0/RSP0/cpu 0: router(config-bgp-sngrp)# advertisement-interval 15
RP/0/RSP0/cpu 0: router(config-bgp-sngrp)# exit
RP/0/RSP0/cpu 0: router(config-bgp)# neighbor-group ADV_2
RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp)# advertisement-interval 25
RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp)# address-family ipv4 unicast
RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp-af)# route-policy POLICY_1 in
RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp-af)# exit
RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp)# exit
RP/0/RSP0/cpu 0: router(config-bgp)# exit
RP/0/RSP0/cpu 0: router(config-bgp)# neighbor 192.168.2.2
RP/0/RSP0/cpu 0: router(config-bgp-nbr)# remote-as 1
RP/0/RSP0/cpu 0: router(config-bgp-nbr)# use session-group ADV
RP/0/RSP0/cpu 0: router(config-bgp-nbr)# use neighbor-group ADV_2
```

show bgp neighbors コマンドからの次の出力は、使用されたアドバタイズメント間隔が 15 秒であることを示しています。

```
RP/0/RSP0/cpu 0: router# show bgp neighbors 192.168.2.2

BGP neighbor is 192.168.2.2, remote AS 1, local AS 140, external link
 Remote router ID 0.0.0.0
  BGP state = Idle
  Last read 00:00:00, hold time is 180, keepalive interval is 60 seconds
  Received 0 messages, 0 notifications, 0 in queue
  Sent 0 messages, 0 notifications, 0 in queue
  Minimum time between advertisement runs is 15 seconds

 For Address Family: IPv4 Unicast
  BGP neighbor version 0
  Update group: 0.1
  eBGP neighbor with no inbound or outbound policy; defaults to 'drop'
  Route refresh request: received 0, sent 0
  0 accepted prefixes
  Prefix advertised 0, suppressed 0, withdrawn 0, maximum limit 524288
  Threshold for warning message 75%

  Connections established 0; dropped 0
  Last reset 00:02:03, due to BGP neighbor initialized
```

```
  External BGP neighbor not directly connected.
```

4. 指定しない場合は、デフォルト値が使用されます。次の例では、ネイバー設定とネイバーグループ設定のいずれも使用するようにネイバーに設定されていないため、ネイバー 10.0.101.5 のアドバタイズメントの最小実行時間間隔は、30 秒（デフォルト）に設定されています。

```
RP/0/RSP0/cpu 0: router(config)# router bgp 140
RP/0/RSP0/cpu 0: router(config-bgp)# neighbor-group AS_1
RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp)# remote-as 1
RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp)# exit
RP/0/RSP0/cpu 0: router(config-bgp)# neighbor-group adv_15
RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp)# remote-as 10
RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp)# advertisement-interval 15
RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp)# exit
RP/0/RSP0/cpu 0: router(config-bgp)# neighbor 10.0.101.5
RP/0/RSP0/cpu 0: router(config-bgp-nbr)# use neighbor-group AS_1
RP/0/RSP0/cpu 0: router(config-bgp-nbr)# exit
RP/0/RSP0/cpu 0: router(config-bgp)# neighbor 10.0.101.10
RP/0/RSP0/cpu 0: router(config-bgp-nbr)# use neighbor-group adv_15
```

show bgp neighbors コマンドからの次の出力は、使用されたアドバタイズメント間隔が 30 秒であることを示しています。

```
RP/0/RSP0/cpu 0: router# show bgp neighbors 10.0.101.5

BGP neighbor is 10.0.101.5, remote AS 1, local AS 140, external link
 Remote router ID 0.0.0.0
  BGP state = Idle
  Last read 00:00:00, hold time is 180, keepalive interval is 60 seconds
  Received 0 messages, 0 notifications, 0 in queue
  Sent 0 messages, 0 notifications, 0 in queue
  Minimum time between advertisement runs is 30 seconds

 For Address Family: IPv4 Unicast
  BGP neighbor version 0
  Update group: 0.2
  eBGP neighbor with no inbound or outbound policy; defaults to 'drop'
  Route refresh request: received 0, sent 0
  0 accepted prefixes
  Prefix advertised 0, suppressed 0, withdrawn 0, maximum limit 524288
  Threshold for warning message 75%
Connections established 0; dropped 0
  Last reset 00:00:25, due to BGP neighbor initialized
  External BGP neighbor not directly connected.
```

グループが他のグループから設定を継承する場合に使用される継承ルールは、グループから継承するネイバーに対して適用されるルールと同じです。

継承した設定の表示

BGP によって継承された設定を表示するには、次の **show** コマンドを使用します。

show bgp neighbors

show bgp neighbors コマンドは、ネイバーの BGP 設定に関する情報を表示する場合に使用します。

- このネイバーで使用されるセッショングループ、ネイバーグループ、またはアドレスファミリグループから継承するすべての設定など、ネイバーの有効な設定を表示するには、**configuration** キーワードを使用します。
- このネイバーで設定を継承できる、セッショングループ、ネイバーグループ、およびアドレスファミリグループを表示するには、**inheritance** キーワードを使用します。

次に示す **show bgp neighbors** コマンドの例は、この設定例に基づいています。

```
RP/0/RSP0/cpu 0: router(config)# router bgp 142
RP/0/RSP0/cpu 0: router(config-bgp)# af-group GROUP_3 address-family ipv4 unicast
RP/0/RSP0/cpu 0: router(config-bgp-afgrp)# next-hop-self
RP/0/RSP0/cpu 0: router(config-bgp-afgrp)# route-policy POLICY_1 in
RP/0/RSP0/cpu 0: router(config-bgp-afgrp)# exit
RP/0/RSP0/cpu 0: router(config-bgp)# session-group GROUP_2
RP/0/RSP0/cpu 0: router(config-bgp-sngrp)# advertisement-interval 15
RP/0/RSP0/cpu 0: router(config-bgp-sngrp)# exit
RP/0/RSP0/cpu 0: router(config-bgp)# neighbor-group GROUP_1
RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp)# use session-group GROUP_2
RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp)# ebgp-multihop 3
RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp)# address-family ipv4 unicast
RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp-af)# weight 100
RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp-af)# send-community-ebgp
RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp-af)# exit

RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp)# exit
RP/0/RSP0/cpu 0: router(config-bgp)# neighbor 192.168.0.1
RP/0/RSP0/cpu 0: router(config-bgp-nbr)# remote-as 2
RP/0/RSP0/cpu 0: router(config-bgp-nbr)# use neighbor-group GROUP_1
RP/0/RSP0/cpu 0: router(config-bgp-nbr)# address-family ipv4 unicast
RP/0/RSP0/cpu 0: router(config-bgp-nbr-af)# use af-group GROUP_3
RP/0/RSP0/cpu 0: router(config-bgp-nbr-af)# weight 200
```

次に、 **inheritance** キーワードを指定した **show bgp neighbors** コマンドの出力例を示します。この例は、ネイバーが、ネイバー グループ GROUP_1 からセッション パラメータを継承すること、ネイバー グループ GROUP_1 はセッション グループ GROUP_2 から継承していることを示します。このネイバーは、IPv4 ユニキャスト パラメータをアドレス ファミリ グループ GROUP_3 から継承し、IPv4 マルチキャスト パラメータをネイバー グループ GROUP_1 から継承します。

```
RP/0/RSP0/cpu 0: router# show bgp neighbors 192.168.0.1 inheritance

  Session:        n:GROUP_1 s:GROUP_2
  IPv4 Unicast:   a:GROUP_3
  IPv4 Multicast: n:GROUP_1
```

次に、 **configuration** キーワードを指定した **show bgp neighbors** コマンドの出力例を示します。この例は、設定の各項目の継承元を示すか、ネイバーへの直接設定（継承元が [] ）を示し

ます。たとえば、**ebgp-multihop 3** コマンドはネイバーグループ GROUP_1 から継承されており、**next-hop-self** コマンドはアドレスファミリグループ GROUP_3 から継承されています。

```
RP/0/RSP0/cpu 0: router# show bgp neighbors 192.168.0.1 configuration

neighbor 192.168.0.1
 remote-as 2                    []
 advertisement-interval 15      [n:GROUP_1 s:GROUP_2]
 ebgp-multihop 3                [n:GROUP_1]
 address-family ipv4 unicast    []
  next-hop-self                 [a:GROUP_3]
  route-policy POLICY_1    in   [a:GROUP_3]
  weight 200                    []
 address-family ipv4 multicast [n:GROUP_1]
  default-originate             [n:GROUP_1]
```

show bgp af-group

アドレスファミリグループを表示するには、**show bgp af-group** コマンドを使用します。

- このアドレスファミリグループで使用されるアドレスファミリグループから継承したすべての設定など、アドレスファミリグループの有効な設定を表示するには、**configuration** キーワードを使用します。
- このアドレスファミリグループで設定を継承できるアドレスファミリグループを表示するには、**inheritance** キーワードを使用します。
- このアドレスファミリグループから設定を継承するネイバー、ネイバーグループ、アドレスファミリ グループを表示するには、**users** キーワードを使用します。

次に示す **show bgp af-group** コマンドの例は、この設定例に基づいています。

```
RP/0/RSP0/cpu 0: router(config)# router bgp 140
RP/0/RSP0/cpu 0: router(config-bgp)# af-group GROUP_3 address-family ipv4 unicast
RP/0/RSP0/cpu 0: router(config-bgp-afgrp)# remove-private-as
RP/0/RSP0/cpu 0: router(config-bgp-afgrp)# route-policy POLICY_1 in
RP/0/RSP0/cpu 0: router(config-bgp-afgrp)# exit
RP/0/RSP0/cpu 0: router(config-bgp)# af-group GROUP_1 address-family ipv4 unicast
RP/0/RSP0/cpu 0: router(config-bgp-afgrp)# use af-group GROUP_2
RP/0/RSP0/cpu 0: router(config-bgp-afgrp)# maximum-prefix 2500 75 warning-only
RP/0/RSP0/cpu 0: router(config-bgp-afgrp)# default-originate
RP/0/RSP0/cpu 0: router(config-bgp-afgrp)# exit
RP/0/RSP0/cpu 0: router(config-bgp)# af-group GROUP_2 address-family ipv4 unicast
RP/0/RSP0/cpu 0: router(config-bgp-afgrp)# use af-group GROUP_3
RP/0/RSP0/cpu 0: router(config-bgp-afgrp)# send-community-ebgp
RP/0/RSP0/cpu 0: router(config-bgp-afgrp)# send-extended-community-ebgp
RP/0/RSP0/cpu 0: router(config-bgp-afgrp)# capability orf prefix both
```

次に、 **show bgp af-group** コマンドで **configuration** キーワードを指定した場合の出力例を示します。この例では、各設定項目がどこから継承されたかを表しています。**default-originate** コマンドは、このアドレスファミリグループで直接設定されています（[] で示されています）。**remove-private-as** コマンドは、アドレスファミリグループ GROUP_2 から継承されています。アドレスファミリグループ GROUP_2 は、アドレスファミリグループ GROUP_3 から継承されています。

```
RP/0/RSP0/cpu 0: router# show bgp af-group GROUP_1 configuration

af-group GROUP_1 address-family ipv4 unicast
  capability orf prefix-list both           [a:GROUP_2]
  default-originate                         []
  maximum-prefix 2500 75 warning-only       []
  route-policy POLICY_1 in                  [a:GROUP_2 a:GROUP_3]
  remove-private-AS                         [a:GROUP_2 a:GROUP_3]
  send-community-ebgp                       [a:GROUP_2]
  send-extended-community-ebgp              [a:GROUP_2]
```

次に、 **users** キーワードを指定した **show bgp af-group** コマンドの出力例を示します。

```
RP/0/RSP0/cpu 0: router# show bgp af-group GROUP_2 users

IPv4 Unicast: a:GROUP_1
```

次に、 **inheritance** キーワードを指定した **show bgp af-group** コマンドの出力例を示します。これは、指定されたアドレス ファミリ グループ GROUP_1 は、GROUP_2 アドレス ファミリ グループを直接使用しており、さらに GROUP_2 で GROUP_3 アドレス ファミリ グループを使用していることを示しています。

```
RP/0/RSP0/cpu 0: router# show bgp af-group GROUP_1 inheritance

IPv4 Unicast: a:GROUP_2 a:GROUP_3
```

show bgp session-group

セッショングループを表示するには、**show bgp session-group** コマンドを使用します。

- このセッショングループで使用されるセッショングループから継承したすべての設定など、セッショングループの有効な設定を表示するには、**configuration** キーワードを使用します。
- このセッショングループで設定を継承できるセッショングループを表示するには、**inheritance** キーワードを使用します。
- このセッショングループから設定を継承するセッショングループ、ネイバーグループ、ネイバーを表示するには、**users** キーワードを使用します。

show bgp session-group コマンドの出力は、次のセッショングループ設定に基づいています。

```
RP/0/RSP0/cpu 0: router(config)# router bgp 113
RP/0/RSP0/cpu 0: router(config-bgp)# session-group GROUP_1
RP/0/RSP0/cpu 0: router(config-bgp-sngrp)# use session-group GROUP_2
RP/0/RSP0/cpu 0: router(config-bgp-sngrp)# update-source Loopback 0
RP/0/RSP0/cpu 0: router(config-bgp-sngrp)# exit
RP/0/RSP0/cpu 0: router(config-bgp)# session-group GROUP_2
RP/0/RSP0/cpu 0: router(config-bgp-sngrp)# use session-group GROUP_3
RP/0/RSP0/cpu 0: router(config-bgp-sngrp)# ebgp-multihop 2
RP/0/RSP0/cpu 0: router(config-bgp-sngrp)# exit
```

```
RP/0/RSP0/cpu 0: router(config-bgp)# session-group GROUP_3
RP/0/RSP0/cpu 0: router(config-bgp-sngrp)# dmz-link-bandwidth
```

次に、EXEC コンフィギュレーション モードで **configuration** キーワードを指定した **show bgp session-group** コマンドの出力例を示します。

```
RP/0/RSP0/cpu 0: router# show bgp session-group GROUP_1 configuration

session-group GROUP_1
 ebgp-multihop 2          [s:GROUP_2]
 update-source Loopback0 []
 dmz-link-bandwidth       [s:GROUP_2 s:GROUP_3]
```

次に示す **inheritance** キーワードを指定した **show bgp session-group** の出力例では、GROUP_1 セッショングループが GROUP_3 セッショングループと GROUP_2 セッショングループからセッションパラメータを継承することを示しています。

```
RP/0/RSP0/cpu 0: router# show bgp session-group GROUP_1 inheritance

Session: s:GROUP_2 s:GROUP_3
```

次に示す **users** キーワードを指定した **show bgp session-group** の出力例では、GROUP_1 セッショングループと GROUP_2 セッショングループが GROUP_3 セッショングループからセッションパラメータを継承することを示しています。

```
RP/0/RSP0/cpu 0: router# show bgp session-group GROUP_3 users

Session: s:GROUP_1 s:GROUP_2
```

show bgp neighbor-group

ネイバーグループを表示するには、**show bgp neighbor-group** コマンドを使用します。

- このネイバーグループで使用されるネイバーグループから継承したすべての設定など、ネイバーグループの有効な設定を表示するには、**configuration** キーワードを使用します。
- このネイバーファミリグループで設定を継承できるアドレスファミリグループ、セッショングループ、およびネイバーグループを表示するには、**inheritance** キーワードを使用します。
- このネイバーグループから設定を継承するネイバーおよびネイバーグループを表示するには、**users** キーワードを使用します。

この例は、次のグループ設定に基づいています。

```
RP/0/RSP0/cpu 0: router(config)# router bgp 140
RP/0/RSP0/cpu 0: router(config-bgp)# af-group GROUP_3 address-family ipv4 unicast
RP/0/RSP0/cpu 0: router(config-bgp-afgrp)# remove-private-as
RP/0/RSP0/cpu 0: router(config-bgp-afgrp)# soft-reconfiguration inbound
RP/0/RSP0/cpu 0: router(config-bgp-afgrp)# exit
RP/0/RSP0/cpu 0: router(config-bgp)# af-group GROUP_2 address-family ipv4 unicast
RP/0/RSP0/cpu 0: router(config-bgp-afgrp)# use af-group GROUP_3
```

```
RP/0/RSP0/cpu 0: router(config-bgp-afgrp)# send-community-ebgp
RP/0/RSP0/cpu 0: router(config-bgp-afgrp)# send-extended-community-ebgp
RP/0/RSP0/cpu 0: router(config-bgp-afgrp)# capability orf prefix both
RP/0/RSP0/cpu 0: router(config-bgp-afgrp)# exit
RP/0/RSP0/cpu 0: router(config-bgp)# session-group GROUP_3
RP/0/RSP0/cpu 0: router(config-bgp-sngrp)# timers 30 90
RP/0/RSP0/cpu 0: router(config-bgp-sngrp)# exit
RP/0/RSP0/cpu 0: router(config-bgp)# neighbor-group GROUP_1
RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp)# remote-as 1982
RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp)# use neighbor-group GROUP_2
RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp)# address-family ipv4 unicast
RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp-af)# exit
RP/0/RSP0/cpu 0: router(config-nbrgrp)# exit
RP/0/RSP0/cpu 0: router(config-bgp)# neighbor-group GROUP_2
RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp)# use session-group GROUP_3
RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp)# address-family ipv4 unicast
RP/0/RSP0/cpu 0: routerconfig-bgp-nbrgrp-af)# use af-group GROUP_2
RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp-af)# weight 100
```

次に、 **configuration** キーワードを指定した **show bgp neighbor-group** コマンドの出力例を示します。構成セット ソースが各コマンドの右側に表示されます。上記の出力で、リモート自律システムは、ネイバー グループ GROUP_1 に直接設定されており、送信コミュニティ設定はネイバー グループ GROUP_2 から継承されています。ネイバー グループ GROUP_2 では、アドレス ファミリ グループ GROUP_3 から設定を継承しています。

```
RP/0/RSP0/cpu 0: router# show bgp neighbor-group GROUP_1 configuration

   neighbor-group GROUP_1
    remote-as 1982                  []
    timers 30 90                    [n:GROUP_2 s:GROUP_3]
    address-family ipv4 unicast     []
     capability orf prefix-list both [n:GROUP_2 a:GROUP_2]
     remove-private-AS              [n:GROUP_2 a:GROUP_2 a:GROUP_3]
     send-community-ebgp            [n:GROUP_2 a:GROUP_2]
     send-extended-community-ebgp   [n:GROUP_2 a:GROUP_2]
     soft-reconfiguration inbound   [n:GROUP_2 a:GROUP_2 a:GROUP_3]
     weight 100                     [n:GROUP_2]
```

次の例は、**inheritance** キーワードを指定した場合の **show bgp neighbor-group** コマンドの出力を示しています。この出力は、指定したネイバー グループ GROUP_1 が、ネイバー グループ GROUP_2 からセッション（アドレス ファミリ独立）設定パラメータを継承していることを示しています。ネイバー グループ GROUP_2 はセッション グループ GROUP_3 からセッション パラメータを継承しました。また、GROUP_1 ネイバー グループは GROUP_2 ネイバー グループから IPv4 ユニキャスト設定パラメータを継承し、さらに GROUP_2 ネイバー グループが GROUP_2 アドレス ファミリ グループから継承し、GROUP_2 アドレス ファミリ グループ自体は GROUP_3 アドレス ファミリ グループから継承していることも示しています。

```
RP/0/RSP0/cpu 0: router# show bgp neighbor-group GROUP_1 inheritance

    Session:      n:GROUP-2 s:GROUP_3
    IPv4 Unicast: n:GROUP_2 a:GROUP_2 a:GROUP_3
```

次に、 **users** キーワードを指定した **show bgp neighbor-group** コマンドの出力例を示します。この出力は、GROUP_1 ネイバー グループが GROUP_2 ネイバー グループからセッション（アドレス ファミリ独立）設定パラメータを継承していることを示しています。GROUP_1 ネイバー グループは GROUP_2 ネイバー グループから IPv4 ユニキャスト設定パラメータも継承しています。

```
RP/0/RSP0/cpu 0: router# show bgp neighbor-group GROUP_2 users

Session:      n:GROUP_1
IPv4 Unicast: n:GROUP_1
```

デフォルトのアドレス ファミリはない

BGP では、デフォルト アドレス ファミリの概念に対応していません。アドレス ファミリを BGP でアクティブにするには、このアドレス ファミリを BGP ルータ コンフィギュレーションで明示的に設定する必要があります。同様に、このアドレス ファミリの BGP セッションをアクティブにするには、ネイバーでそのアドレス ファミリを明示的に設定する必要があります。ネイバーを設定するために、BGP ルータ コンフィギュレーション レベルでアドレス ファミリを設定する必要はありません。ただし、ネイバーにアドレス ファミリを設定するには、BGP ルータ コンフィギュレーション レベルでそのアドレス ファミリを設定する必要があります。

ネイバーアドレスファミリの組み合わせ

デフォルトの VRF の場合、Cisco IOS XR ソフトウェア リリース 6.2.x 以降では、IPv4 ユニキャスト アドレス ファミリと IPv4 ラベル付きユニキャスト アドレス ファミリの両方が同じネイバーでサポートされています。

デフォルト以外の VRF では、IPv4 ユニキャスト アドレス ファミリと IPv4 ラベル付きユニキャスト アドレス ファミリはどちらも同じネイバーでサポートされません。ただし、次のエラーが発生した場合は、 Cisco ASR 9000 シリーズルータでこの設定が受け入れられます。

```
bgp[1051]: %ROUTING-BGP-4-INCOMPATIBLE_AFI : IPv4 Unicast and IPv4 Labeled-unicast Address
 families together are not supported under the same neighbor.
```

1 つの BGP セッションに IPv4 ユニキャストと IPv4 ラベル付きユニキャスト AFI/SAF の両方がある場合、ルーティング動作は非決定的になります。したがって、プレフィックスが正しくアドバタイズされない場合があります。プレフィックスが正しくアドバタイズされないと、到達可能性の問題が発生します。このような到達可能性の問題を回避するには、IPv4 ユニキャストまたは IPv4 ラベル付きユニキャスト アドレス ファミリのいずれかを介してプレフィックスをアドバタイズするルート ポリシーを明示的に設定する必要があります。

ルーティング ポリシーの強制適用

外部 BGP（eBGP）ネイバーには、インバウンドおよびアウトバウンドのポリシーを設定する必要があります。ポリシーが設定されていない場合、そのネイバーからのルートは受け入れられず、いずれのルートもそのネイバーにアドバタイズされません。この付加的なセキュリティ

手段によって、設定を誤って省略した場合に、ルートが偶然受け入れられたり、アドバタイズされたりすることが決してなくなります。

（注） この制約は eBGP ネイバー（このルータと異なる自律システムに属すネイバー）だけに適用されます。内部 BGP（iBGP）ネイバー（同じ自律システム内のネイバー）の場合は、ポリシーがなければ、すべてのルートが受け入れられるか、アドバタイズされます。

次の例では、すべてのルートが変更なしで許可およびアドバタイズされる場合に、eBGP ネイバーに対して単純な pass-all ポリシーが設定されています。

```
RP/0/RSP0/cpu 0: router(config)# route-policy pass-all
RP/0/RSP0/cpu 0: router(config-rpl)# pass
RP/0/RSP0/cpu 0: router(config-rpl)# end-policy
RP/0/RSP0/cpu 0: router(config)# commit
```

ネイバーに pass-all ポリシーを適用するには、ネイバー アドレス ファミリ コンフィギュレーション モードで **route-policy (BGP)** コマンドを使用します。次の例は、ネイバー 192.168.40.42 からの受信と、このネイバーに対するすべての IPv4 ユニキャスト ルートのアドバタイズを、すべての IPv4 ユニキャスト ルートに許可する方法を示します。

```
RP/0/RSP0/cpu 0: router(config)# router bgp 1
RP/0/RSP0/cpu 0: router(config-bgp)# neighbor 192.168.40.24
RP/0/RSP0/cpu 0: router(config-bgp-nbr)# remote-as 21
RP/0/RSP0/cpu 0: router(config-bgp-nbr)# address-family ipv4 unicast
RP/0/RSP0/cpu 0: router(config-bgp-nbr-af)# route-policy pass-all in
RP/0/RSP0/cpu 0: router(config-bgp-nbr-af)# route-policy pass-all out
RP/0/RSP0/cpu 0: router(config-bgp-nbr-af)# commit
```

すべてのアクティブ アドレス ファミリに対するインバウンドとアウトバウンドの両方のポリシーを持っていない eBGP ネイバーを表示するには、**show bgp summary** コマンドを使用します。次の例の出力では、該当する eBGP ネイバーが感嘆符（!）によって示されています。

```
RP/0/RSP0/cpu 0: router# show bgp all all summary

Address Family: IPv4 Unicast
============================

BGP router identifier 10.0.0.1, local AS number 1
BGP generic scan interval 60 secs
BGP main routing table version 41
BGP scan interval 60 secs
BGP is operating in STANDALONE mode.

Process         RecvTblVer    bRIB/RIB  SendTblVer
Speaker                41          41          41

Neighbor        Spk    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  St/PfxRcd
10.0.101.1        0     1     919     925       41    0    0 15:15:08       10
10.0.101.2        0     2       0       0        0    0    0 00:00:00 Idle


Address Family: IPv4 Multicast
```

```
==============================

BGP router identifier 10.0.0.1, local AS number 1
BGP generic scan interval 60 secs
BGP main routing table version 1
BGP scan interval 60 secs
BGP is operating in STANDALONE mode.

Process         RecvTblVer    bRIB/RIB  SendTblVer
Speaker                 1           1           1

Some configured eBGP neighbors do not have both inbound and
outbound policies configured for IPv4 Multicast address family.
These neighbors will default to sending and/or receiving no
routes and are marked with '!' in the output below. Use the
'show bgp neighbor <nbr_address>' command for details.

Neighbor        Spk    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  St/PfxRcd
10.0.101.2        0     2       0       0        0    0    0 00:00:00 Idle!


Address Family: IPv6 Unicast
============================

BGP router identifier 10.0.0.1, local AS number 1
BGP generic scan interval 60 secs
BGP main routing table version 2
BGP scan interval 60 secs
BGP is operating in STANDALONE mode.

Process         RecvTblVer    bRIB/RIB  SendTblVer
Speaker                 2           2           2

Neighbor        Spk    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  St/PfxRcd
2222::2           0     2     920     918        2    0    0 15:15:11        1
2222::4           0     3       0       0        0    0    0 00:00:00 Idle


Address Family: IPv6 Multicast
==============================

BGP router identifier 10.0.0.1, local AS number 1
BGP generic scan interval 60 secs
BGP main routing table version 1
BGP scan interval 60 secs
BGP is operating in STANDALONE mode.

Process         RecvTblVer    bRIB/RIB  SendTblVer
Speaker                 1           1           1

Some configured eBGP neighbors do not have both inbound and
outbound policies configured for IPv6 Multicast address family.
These neighbors will default to sending and/or receiving no
routes and are marked with '!' in the output below. Use the
'show bgp neighbor <nbr_address>' command for details.

Neighbor        Spk    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  St/PfxRcd
2222::2           0     2     920     918        0    0    0 15:15:11        0
2222::4           0     3       0       0        0    0    0 00:00:00 Idle!
```

テーブル ポリシー

BGP のテーブル ポリシー機能を使用すると、ルートのトラフィック索引の値をグローバル ルーティング テーブルにインストールされるときに設定できます。この機能を有効にするには **table-policy** コマンドを使用します。また BGP ポリシーアカウンティング機能もサポートされています。

BGP ポリシー アカウンティングでは、BGP ルートに設定されたトラフィック索引を使用してさまざまなカウンタをトラックします。テーブルポリシーの使用法の詳細については、『*Routing Configuration Guide for Cisco ASR 9000 Series Routers*』の「 *Implementing Routing Policy on Cisco ASR 9000 Series Router* 」のモジュールを参照してください。BGP ポリシーアカウンティングの詳細については、『*IP Addresses and Services Command Reference for Cisco ASR 9000 Series Routers*』の「 *Cisco Express Forwarding Commands on Cisco ASR 9000 Series Router* 」のモジュールを参照してください。

テーブル ポリシーを使用すると、一致基準に基づいて RIB からのルートをドロップすることもできます。この機能は特定のアプリケーションにおいて有用ですが、BGP がグローバル ルーティングおよびフォワーディング テーブルにインストールしていないネイバーに対して、BGP がルートをアドバタイズするところに、簡単にルーティング「ブラック ホール」が作成されてしまうため、注意して使用する必要があります。

アップデート グループ

BGP アップデート グループ機能には、アウトバウンド ポリシーを共有し、アップデート メッセージを共有できるネイバーのアップデート グループをダイナミックに計算し、最適化する新しいアルゴリズムが含まれています。BGP アップデート グループ機能では、アップデート グループ レプリケーションはピア グループ コンフィギュレーションから分離されるため、ネイバー コンフィギュレーションのコンバージェンス時間が短縮され、柔軟性が高まります。

この機能を使用するには、次の概念を理解しておく必要があります。

関連トピック

BGP アップデートの生成およびアップデート グループ （34 ページ）
BGP アップデート グループ （34 ページ）

BGP アップデートの生成およびアップデート グループ

BGP アップデート グループ機能により、BGP アップデートの生成がネイバー設定から分離されます。BGP アップデート グループ機能により、アウトバウンド ルーティング ポリシーに基づいて BGP アップデート グループ メンバーシップを動的に計算するアルゴリズムが導入されます。この機能に対してネットワーク オペレータによる設定は不要です。アップデート グループをベースとするメッセージ生成は自動的かつ個別に行われます。

BGP アップデート グループ

設定の変更があった場合、ルータでは、アップデート グループ メンバーシップを自動的に再計算し、変更を適用します。

BGP アップデート グループの生成を最適化するには、ネットワーク オペレータは、類似するアウトバウンド ポリシーを持つネイバーのアウトバウンド ルーティング ポリシーを同じものにしておくことを推奨します。この機能には、BGP アップデート グループを監視するためのコマンドが含まれます。

BGP コスト コミュニティ

BGP コスト コミュニティは非過渡的な拡張コミュニティ属性で、内部 BGP（iBGP）およびコンフェデレーション ピアへ渡されますが、外部 BGP（eBGP）ピアへは渡されません。コスト コミュニティ機能により、コスト値を特定のルートに割り当てることで、ローカル ルート プリファレンスをカスタマイズし、最適パス選択プロセスに反映させることができます。拡張コミュニティ形式は、最適パスアルゴリズムの異なるポイントでの最適パスの決定に影響する標準の挿入ポイント（POI）を定義します。

コスト コミュニティ属性は、ルート ポリシーで **set extcommunity cost** コマンドを設定することにより、内部ルートに適用されます。**set extcommunity cost** コマンドについては、『*Cisco ASR 9000 Series Aggregation Services Router Routing Command Reference*』の「*Routing Policy Language Commands on Cisco ASR 9000* シリーズ ルータ 」のモジュールを参照してください。cost community set 句は、コスト コミュニティ ID 番号（0 ～ 255）およびコスト コミュニティ番号（0 ～ 4294967295）を使用して設定されます。コストコミュニティ番号によってパスの優先度が判断されます。最も低いコストコミュニティ番号を持つパスが優先されます。コストコミュニティ番号を具体的に設定していないパスには、デフォルトのコストコミュニティ番号である 2147483647（0 ～ 4294967295 の中央値）が割り当てられ、最適パス選択プロセスにより評価されます。2つのパスが同じコストコミュニティ番号を使用して設定されている場合、パス選択プロセスでは最も低いコスト コミュニティ ID のパスが優先されます。このコスト拡張コミュニティ リンク属性は、拡張コミュニティ交換がイネーブルなとき、iBGP ピアに伝播します。

次のコマンドには **route-policy** キーワードが含まれています。このキーワードは、cost community set 句で設定されるルート ポリシーを適用するために使用できます。

- **aggregate-address**
- **redistribute**
- **network**

BGP コスト コミュニティはどのように最適パス選択プロセスに影響するか

BGP最適パス選択プロセスは、挿入ポイント（POI）においてコストコミュニティ属性の影響を受けます。デフォルトでは、POI は、内部ゲートウェイ プロトコル（IGP）メトリック比較に従います。同一の宛先に向かう複数のパスを受信したとき、BGP では最適パス選択プロセスを使用して、いずれのパスが最適パスであるのかを決定します。最良パスは BGP により自動的に決定され、ルーティングテーブルにインストールされます。複数の等コストパスが使用可能な場合、POI で個別のパスにプリファレンスを割り当てることができます。ローカルの最適パス選択で POI が有効でない場合は、コスト コミュニティ属性は暗黙的に無視されます。

コスト コミュニティは、最初に POI で、次にコミュニティ ID でソートされます。コスト コミュニティ属性を使用して、同一の POI に対し複数のパスを設定できます。最も低いコスト コミュニティ ID を持つパスが最優先で検討されます。つまり、特定の POI に対するすべてのコスト コミュニティ パスは、最も低いコスト コミュニティを持つパスから検討されていきます。コスト コミュニティ コストを持たないパス（評価中の POI およびコミュニティ ID）には、デフォルトのコミュニティ コスト値（2147483647）が割り当てられます。コスト コミュニティ値が等しい場合、コスト コミュニティ比較は、その POI で次に低いコミュニティ ID に進みます。

最も低いコスト コミュニティを持つパスを選択するには、両方のパスのコスト コミュニティを同時に探索します。これを行うには、コストコミュニティのチェーンにポインタを 2 つ設定し、各パスに 1 つずつ割り当て、POI に対する探索の各ステップでコミュニティ ID の順に両方のポインタを次のコストコミュニティに進め、最良のパスが選ばれたとき、または比較して順位が付かなくなったときに終了します。探索の各ステップで、次のチェックが実行されます。

```
If neither pointer refers to a cost community,
     Declare a tie;

  Elseif a cost community is found for one path but not for the other,
     Choose the path with cost community as best path;
  Elseif the Community ID from one path is less than the other,
     Choose the path with the lesser Community ID as best path;
  Elseif the Cost from one path is less than the other,
     Choose the path with the lesser Cost as best path;
  Else Continue.
```

（注） パスにコストコミュニティ属性が設定されていない場合、最適パス選択プロセスはそのパスにデフォルトのコスト値（最大値 4294967295 の半分である 2147483647）が割り当てられているものと見なします。

POI でコストコミュニティ属性を適用することで、ローカルの自律システムまたはコンフェデレーションにおける任意の部分にあるピアを起点とするか、このピアで学習したパスに、値を割り当てることができるようになります。コストコミュニティは、最適パス選択プロセス中の「タイブレーカー」として使用できます。同一の自律システムまたはコンフェデレーションにおける別個の等コストパスに対し、コストコミュニティのインスタンスを複数設定できます。たとえば、複数の等コスト出口ポイントがあるネットワークにおいて、特定の出口パスに、より低いコスト コミュニティ値を適用すれば、その出口パスは BGP 最適パス選択プロセスにより優先されることになります。マルチエグジット IGP ネットワークにおけるルート プリファレンスの反映 （38 ページ）に記載されているシナリオを参照してください。

（注） BGP では、コストコミュニティの比較がデフォルトで有効になっています。比較を無効にするには、**bgp bestpath cost-community ignore** コマンドを使用します。

BGP 最適パス選択処理については、BGP 最適パス アルゴリズム （43 ページ）を参照してください。

集約ルートおよびマルチパスに対するコスト コミュニティのサポート

BGP コスト コミュニティ機能では、集約ルートおよびマルチパスをサポートしています。コスト コミュニティ属性は、いずれかのルートのタイプに適用できます。コスト コミュニティ属性は、コスト コミュニティ属性を伝送するコンポーネント ルートから集約ルートまたはマルチパス ルートに渡されます。一意の ID のみが渡され、いずれの個別コンポーネント ルートについても、最大のコストのみが、ID ごとの集約に対して適用されます。複数のコンポーネント ルートに同一の ID が含まれる場合は、設定されている最大のコストがルートに適用されます。たとえば、次の 2 つのコンポーネント ルートは、インバウンド ルート ポリシーを使用してコスト コミュニティ属性が設定されています。

- 10.0.0.1
  - POI=IGP
  - コスト コミュニティ ID=1
  - コスト番号=100
- 192.168.0.1
  - POI=IGP
  - コスト コミュニティ ID=1
  - コスト番号=200

これらのコンポーネント ルートを集約するか、マルチパスとして設定した場合は、コスト値 200 が最大のコストであるため、この値がアドバタイズされます。

1 つ以上のコンポーネント ルートがコスト コミュニティ属性を伝送しない場合、またはこれらのコンポーネント ルートに異なる ID が設定されている場合は、デフォルト値（2147483647）が、集約ルートまたはマルチパス ルートに対してアドバタイズされます。たとえば、次の 3 つのコンポーネント ルートは、インバウンド ルート ポリシーを使用してコスト コミュニティ属性が設定されています。ただし、これらのコンポーネント ルートには 2 つの異なる ID が設定されています。

- 10.0.0.1
  - POI=IGP
  - コスト コミュニティ ID=1
  - コスト番号=100
- 172.16.0.1
  - POI=IGP
  - コスト コミュニティ ID=2

- コスト番号=100

- 192.168.0.1
  - POI=IGP
  - コスト コミュニティ ID=1
  - コスト番号=200

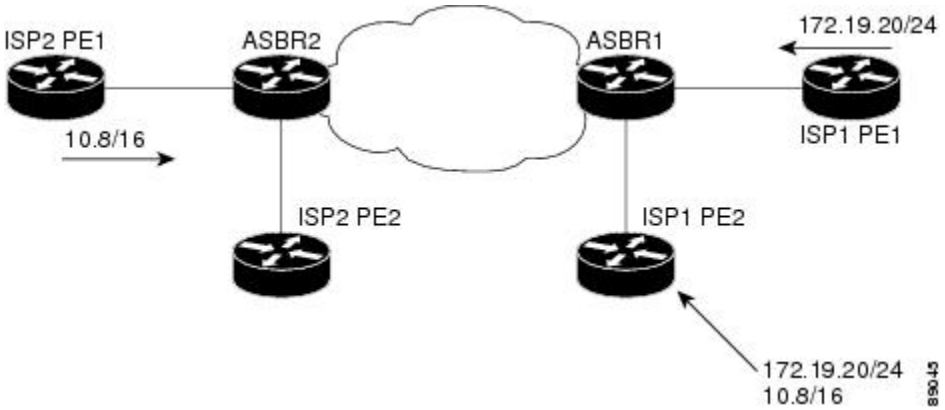
アドバタイズされる単一のパスには、次のような集約コスト コミュニティなどがあります。

{POI=IGP, ID=1, Cost=2147483647} {POI-IGP, ID=2, Cost=2147483647}

マルチエグジット IGP ネットワークにおけるルート プリファレンスの反映

次の図は、エッジに 2 つの自律システム境界ルータ（ASBR）がある IGP ネットワークを示します。各 ASBR は、ネットワーク 10.8/16 に対して等コスト パスを持ちます。

図 *1:* マルチエグジット ポイントの *IGP* ネットワーク



BGP では、両パスは等しいと見なされます。マルチパス ロード シェアリングが設定されている場合は、ルーティングテーブルへの両方のパスが組み込まれ、トラフィックの負荷を分散するために使用されます。マルチパス ロードバランシングが設定されていない場合、BGP により最初に最適パスであると学習されたパスが選択され、ルーティングテーブルに組み込まれます。この動作は、一部の条件下では望ましくない場合があります。たとえば、パスは最初に ISP1 PE2 から学習されますが、ISP1 PE2 と ASBR1 間のリンクは低速リンクです。

コスト コミュニティ属性のコンフィギュレーションを使用して ASBR2 が学習したパスにより低いコスト コミュニティ値を適用することで、BGP 最適パス選択プロセスに影響を与えることができます。たとえば、次のコンフィギュレーションは ASBR2 に適用されています。

```
RP/0/RSP0/cpu 0: router(config)# route-policy ISP2_PE1
RP/0/RSP0/cpu 0: router(config-rpl)# set extcommunity cost (1:1)
```

上記のルート ポリシーでは、コスト コミュニティ番号値の 1 がルート 10.8.0.0 に適用されます。デフォルトでは、ASBR1 で学習したパスにはコスト コミュニティ番号 2147483647 が割り当てられます。ASBR2 から学習したパスのコスト コミュニティ番号の方が小さいため、このパスが優先されます。

バックドア リンクを持つ EIGRP MPLS VPN PE-CE に対する BGP コスト コミュニティ サポート

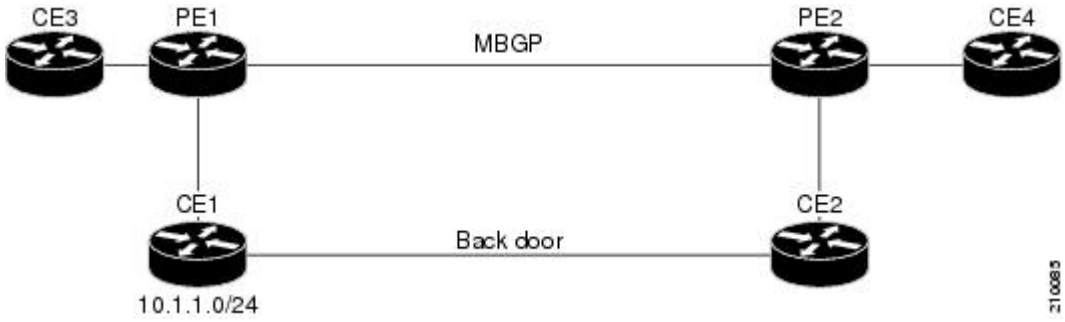
バックドア リンクの方が先に学習された場合、BGP では、EIGRP MPLS VPN トポロジのバックドア リンクを優先します。（バックドア リンクまたはルートは、リモート サイトとメイン サイトの間の VPN の外部に設定される接続で、たとえば、リモート サイトを企業ネットワークへ接続する WAN 専用線などがあります）。

BGP コスト コミュニティの「準最適パス」挿入ポイント（POI）機能は、VPN およびバックドア リンクが混在する EIGRP VPN ネットワーク トポロジをサポートしています。この POI は BGP に再配布される EIGRP ルートに自動的に適用されます。「準最適パス」POI は、EIGRP のルート タイプおよびメトリックを伝送します。この POI は、BGP がその他のあらゆる比較ステップの前にこの POI を検討するように影響を与えておくことで、最適パス計算プロセスに作用します。設定は必要ありません。PE、CE、またはバックドアルータに Cisco IOS XR ソフトウェア がインストールされている場合、この機能は、EIGRP VPN サイトについて自動的に有効にされます。

EIGRP MPLS VPN の設定については、 *MPLS Configuration Guide for Cisco ASR 9000 Series RoutersMPLS Configuration Guide for Cisco NCS 560 Series Routers*を参照してください。

図 *2:* コスト コミュニティを使用してバックドア リンクをサポートする方法を示すネットワーク

この図は、コスト コミュニティを使用して、ネットワークのバックドア リンクをサポートする方法を示します。



次に、PE1 におけるイベント シーケンスを示します。

1. PE1 では、仮想ルーティングおよび転送（VRF）インスタンスを実行している CE1 から EIGRP を介して IPv4 プレフィックス 10.1.1.0/24 を学習します。EIGRP は最適パスを選択して、RIB に組み込みます。コスト拡張コミュニティのエンコードと、RIB に対するこの情報の追加も行います。

2. ルートは BGP に再配布されます（IGP-to-BGP 再配布が設定されていることを想定）。BGP では、再配布プロセスを介して、ルートからコスト拡張コミュニティも受け取ります。
3. 新しく再配布されたプレフィックスの最適パスを BGP が判別すると、そのパスは PE ピア（PE2）にアドバタイズされます。
4. PE2 では、BGP VPNv4 プレフィックス route_distinguisher:10.1.1.0/24 をコスト コミュニティとともに受信します。CE2 では、おそらくは、EIGRP を介して PE2 に同じプレフィックスをアドバタイズします（CE1 と CE2 の間にバックドア リンクがあるため）。通常、PE2 BGP では、再配布プロセスによって、コスト コミュニティ値とともに CE ルートをすでに学習しています。
5. PE2 には、BGP のパスが 2 つあります。マルチパス BGP を介するコスト コミュニティ cost1 のパス（PE1）と、EIGRP ネイバーを介するコスト コミュニティ cost2 の別のパス（CE2）です。
6. PE2 では、拡張された BGP 最適パス計算を実行します。
7. PE2 は、適切なコスト コミュニティ値を渡して、RIB に最適パスを組み込みます。
8. PE2 RIB には、10.1.1.0/24 に対するパスが 2 つあります。EIGRP によって追加されたコスト コミュニティ cost2 のパスと、BGP によって追加されたコスト コミュニティ cost1 の別のパスです。両方のルートパスがコストコミュニティを持つため、RIB では、まずコストを比較します。BGP パスのコストコミュニティの方が低いため、これが選択されて、RIB にダウンロードされます。
9. PE2 RIB では、VRF を介して BGP パスを EIGRP に再配布します。パスが 2 個あるため EIGRP は拡散更新アルゴリズム（DUAL）を実行し、BGP 再配布パスを選択します。
10. PE2 EIGRP は、このパスを CE2 にアドバタイズします。これにより、このパスは、MPLS ネットワークを介してトラフィックを送信するために、このプレフィックス対して使用されるネクスト ホップになります。

ルーティング情報ベースへのルートの追加

最適パス計算の後で、ソーシングされていないパスが最適パスになった場合、BGP では、このルートをルーティング情報ベース（RIB）に追加し、他の IGP 拡張コミュニティと一緒にコスト コミュニティを渡します。

パスを含むルートがプロトコルによって RIB に追加される場合、RIB では、現在の最適パスを調べてルートを確認し、追加されたパスを調べてコスト拡張コミュニティを確認します。コスト拡張コミュニティが見つかった場合、RIB では、コストコミュニティの設定を比較します。比較して順位が付く場合は、適切な最適パスが選択されます。比較して順位が付かない場合、RIB では、最適パス アルゴリズムの残りの手順に進みます。現在の最適パスと追加されたパスのいずれにもコスト コミュニティがない場合、RIB では、最適パス アルゴリズムの残りの手順を続行します。BGP 最適パス アルゴリズムについては、BGP 最適パス アルゴリズム （43 ページ）を参照してください。

BGP DMZ 総帯域幅

BGP は、内部 BGP（iBGP）ピアへのルートをアドバタイズするときに、外部 BGP（eBGP）マルチパスの *dmz-link bandwidth* 値の集約をサポートしています。

帯域幅を集約するための明示的なコマンドはありません。帯域幅は、次の条件を満たしている場合に集約されます。

- ネットワークにはマルチパスがあり、すべてのマルチパスにはリンク帯域幅の値があります。
- next-hop-self に設定されたネクストホップ属性。指定されたネイバーにアドバタイズされるすべてのルートのネクストホップ属性をローカルルータのアドレスに設定します。
- dmz-link bandwidth の値を変更する可能性があるアウトバウンドポリシーは設定されていません。

（注）

- マルチパス（eBGP または iBGP）のいずれかの *dmz-link bandwidth* 値が不明な場合、ベストパスを含むすべてのマルチパスの *dmz-link* 値はルーティング情報ベース（RIB）にダウンロードされません。
- iBGP マルチパスの *dmz-link bandwidth* 値は、集約時に考慮されません。
- 集約値でアドバタイズされるルートは、ベストパスまたは追加パスにすることができます。
- 追加パスは、ネクストホップが維持されるため、DMZ リンクの帯域幅集約には適していません。追加パスの next-hop-self の設定はサポートされていません。
- VPNv4 および VPNv6 afi の場合、f *dmz link-bandwidth* 値はアウトバウンド ルート ポリシーを使用し、ルートテーブルを指定するか、または **additive** キーワードを使用して設定されます。また、これによってピアの受信端でルートがインポートされなくなります。

```
extcommunity-set bandwidth dmz_ext
   1:8000
 end-set
 !
 route-policy dmz_rp_vpn
   set extcommunity bandwidth dmz_ext additive     <<< 'additive' keyword.
   pass
 end-policy
```

例

ネットワーク内の内部ルータに接続されたルタ 1 とルータ 2 の 2 台のルータについて検討してみましょう。ルータ 1 は、2 つの異なる ISP から 50 と 20 の帯域幅をアドバタイズします。ルータ 2 は、2 つの異なる ISP から 60 と 30 の帯域幅をアドバタイズします。ベストパスアルゴリズムを使用すると、内部ルータに対してルータ 1 は 50 の帯域幅をアドバタイズし、ルータ 2 は 60 の帯域幅をアドバタイズします。これによ

り、トラフィックフローが削減されます。ただし、帯域幅を集約することで、ルータ 1 は 70（50 + 20）の帯域幅をアドバタイズし、ルータ 2 は 90（60 + 30）の帯域幅をアドバタイズします。これにより、トラフィックフローが増加します。

BGP DMZ 総帯域幅の設定：例

次に、ボーダー ゲートウェイ プロトコルの緩衝地帯（BGP DMZ）リンク帯域幅の設定例を示します。トポロジ、R1---(iBGP)---R2---(iBGP)---R3 について検討してみましょう。

1. R1 では次のようになります。

```
bgp: prefix p/n has:
path 1(bestpath)       with LB value 100
path 2(ebgp multipath) with LB value 30
path 3(ebgp multipath) with LB value 50
```

   ベストパスが R2 にアドバタイズされると、集約された DMZ リンクの帯域幅の値 180 を送信します。パス 1、2、および 3 の集約値。

2. R2 では次のようになります。

```
bgp: prefix p/n has:
path 1(bestpath)       with LB value 60
path 2(ebgp multipath) with LB value 200
path 3(ebgp multipath) with LB value 50
```

   ベストパスが R3 にアドバタイズされると、集約された DMZ リンクの帯域幅の値 310 を送信します。パス 1、2、および 3 の集約値。

3. R3 では次のようになります。

```
bgp: prefix p/n has:
path 1(bestpath)       with LB 180 {learned from R1}
path 2(ibgp multipath) with LB 310 {learned from R2}
```

ポリシーベースのリンク帯域幅の設定：例

次に、ポリシーベースの DMZ リンク帯域幅を設定する例を示します。リンク帯域幅の拡張コミュニティは、ネイバーインまたはネイバーアウトのポリシー接続点で、パスごとに設定できます。*dmz-link-bandwidth* ノブは、eBGP ネイバー コンフィギュレーション モードで設定されます。この特定のネイバーから受信したすべてのパスは、iBGP ピアに送信されるときに、リンク帯域幅拡張コミュニティでマークされます。

1. インバウンドまたはアウトバウンドのルートポリシーを設定します。

```
extcommunity-set bandwidth dmz_ext
  1:1290400000
end-set
!
route-policy dmz_rp
  set extcommunity bandwidth dmz_ext
  pass
end-policy
!

 neighbor 10.0.101.1
  remote-as 1001
  address-family ipv4 unicast
```

```
 route-policy dmz_rp in          <<< Inbound route-policy.
 route-policy pass out
!
```

2. BGP ネイバーで *dmz-link-bandwidth* を設定します。

```
neighbor 10.0.101.2
  remote-as 1001
  dmz-link-bandwidth              <<< Under neighbor.
  address-family ipv4 unicast
   route-policy pass in
   route-policy pass out
  !
```

ポリシーベースの拡張コミュニティセットの詳細については、『*Cisco ASR 9000* シリーズ アグリゲーション サービス ルータ ルーティング設定ガイド』の「ルーティング ポリシーの実装」の章を参照してください。

BGP 用 64-ECMP のサポート

IOS XR では、BGP に最大 64 の等コストマルチパス（ECMP）ネクストホップを設定できます。過負荷状態のルータが 64 を超える LSP のトラフィックをロードバランシングできる場合、ネットワークに 64-ECMP が必要です。

BGP 最適パス アルゴリズム

BGP ルータは、通常は同じ宛先に対する複数のパスを受信します。BGP の最適パス アルゴリズムは、IP ルーティング テーブルに格納し、トラフィックの転送に使用する最適なパスを決めるものです。この項では、インターネット技術特別調査委員会（IETF）のネットワーク ワーキング グループによる draft-ietf-idr-bgp4-24.txt 資料の 9.1 項で指定されている BGP 最適パス アルゴリズムの Cisco IOS XR ソフトウェア実装について説明します。

BGP 最適パス アルゴリズムは、次の 3 つのパートに分かれて実行されます。

- パート 1：2 つのパスを比較して、いずれが優れているのかを判別します。
- パート 2：すべてのパスを順に処理し、全体として最適なパスを選択するためにパスを比較する順序を決定します。
- パート 3：新しい最適パスを使用するに足るだけの差が新旧の最適パスにあるかどうかを判別します。

（注） 比較演算が推移的ではないため、パート 2 で決定された比較の順序は重要です。つまり、3 つのパス、A、B、C がある場合、A と B を比較したときに A の方が優れていて、B と C と比較したときに B の方が優れている場合、A と C を比較したときに必ずしも A が優れているとは限りません。この非推移性は、Multi Exit Discriminator（MED）が、すべてのパス間ではなく、同じネイバー自律システム（AS）からのパス間のみで比較されるために生じます。

パスのペアの比較

2 つのパスを比較して、優れたパスを判別するには、次の手順を実行します。

1. いずれかのパスが無効な場合（可能な最大MED値を持つパス、到達不能なネクストホップを持つパスなど）、もう一方のパスが選択されます（そのパスが有効な場合）。
2. パスの準最適パス コスト コミュニティが等しくない場合は、準最適パス コスト コミュニティの低いパスが最適パスとして選択されます。
3. パスの重みが等しくない場合は、重みが最大のパスが選択されます。

（注） 重みは完全にルータにローカルであり、weight コマンドまたはルーティング ポリシーを使用して設定できます。

4. パスのローカル プリファレンスが等しくない場合は、ローカル プリファレンスが高い方のパスが選択されます。

（注） パスとともにローカル プリファレンス属性を受信したか、ルーティング ポリシーによって設定された場合は、その値が、この比較で使用されます。それ以外の場合は、デフォルト ローカル プリファレンス値の 100 が使用されます。デフォルト値は、bgp default local-preference コマンドを使用して変更できます。

5. パスの 1 つが再配布されたパス、つまり **redistribute** コマンドまたは **network** コマンドによるパスの場合は、そのパスが選択されます。それ以外の場合、パスの 1 つがローカルで作成された集約パスのとき、つまり aggregate-address コマンドによるパスのときは、そのパスが選択されます。

（注） ステップ 1 ～ ステップ 4 では、RFC 1268 の「Path Selection with BGP」を実装します。

6. パス間で AS パスの長さが異なる場合は、AS パスの短い方のパスが選択されます。このステップは、bgp bestpath as-path ignore コマンドが設定されている場合は省略されます。

（注） AS パスの長さを計算する場合は、コンフェデレーション セグメントは無視され、AS セットは 1 としてカウントされます。

（注） eiBGP は、内部および外部の BGP マルチパス ピアを指定します。eiBGP では、内部および外部のパスを同時に使用できます。

7. パス間で起点が異なる場合は、起点の値が低い方のパスが選択されます。内部ゲートウェイ プロトコル（IGP）は EGP よりも低く、EGP は INCOMPLETE より低いと見なされます。

8. 該当する場合は、パスの MED が比較されます。等しくない場合は、MED の低いパスが選択されます。

   このステップが実行されるかどうかに影響するコンフィギュレーション オプションは多数あります。一般に、MED はパスが両方のパスが同じ AS にあるネイバーから受信された場合に比較され、それ以外の場合は MED 比較はスキップされます。ただし、この動作は特定のコンフィギュレーション オプションによって変更され、考慮すべきいくつかの場合があります。

   bgp bestpath med always コマンドが設定されている場合、MED 比較は、パス内のネイバー AS にかかわらず、常に実行されます。それ以外の場合、MED 比較は、次のように、比較する 2 つのパスの AS パスによって異なります。

   - パスに AS パスがない場合、または AS パスが AS_SET で始まる場合、パスは内部と見なされ、MED は他の内部パスと比較されます。
   - AS パスが AS_SEQUENCE で開始されている場合、ネイバー AS は、シーケンスの最初の AS 番号であり、MED は、同じネイバー AS を持つ他のパスと比較されます。
   - AS パスがコンフェデレーション セグメントのみを含むか、コンフェデレーション セグメントで開始されて AS_SET が続く場合、MED は、他のいずれのパスとも比較されません。ただし、**bgp bestpath med confed** コマンドが設定されている場合を除きます。その場合、パスは内部であると見なされ、MED は他の内部パスと比較されます。
   - AS パスがコンフェデレーション セグメントとそれに続く AS_SEQUENCE で開始している場合、ネイバー AS は AS_SEQUENCE の最初の AS 番号であり、MED は同じネイバー AS を持つ他のパスと比較されます。

（注） パスとともに MED 属性を受信しなかった場合、MED は 0 であると見なされます。ただし、**bgp bestpath med missing-as-worst** コマンドが設定されている場合を除きます。この場合、MED 属性が受信されていない場合、MED は最高値と見なされます。

9. パスの 1 つを外部ピアから受信し、もう 1 つを内部（またはコンフェデレーション）ピアから受信した場合は、外部ピアからのパスが選択されます。

10. パスのネクスト ホップへの IGP メトリックが異なる場合、IGP メトリックが小さい方のパスが選択されます。

11. パスの IP コスト コミュニティが等しくない場合は、IP コスト コミュニティの低いパスが最適パスとして選択されます。

12. ステップ 1 ～ステップ 10 ですべてのパス パラメータが一致している場合は、ルータ ID が比較されます。送信元属性付きでパスを受信した場合は、この属性が比較対象のルー

タ ID として使用されます。それ以外の場合は、パスの受信元ネイバーのルータ ID が使用されます。パス間でルータ ID が異なる場合は、ルータ ID の小さい方のパスが選択されます。

（注） 送信元をルータ ID として使用する場合は、2 つのパスが同じルータ ID を持つことがあります。同じピア ルータと 2 つの BGP セッションを持つこともでき、したがって、同じルータ ID を持つ 2 つのパスを受信することがあります。

13. パス間でクラスタ長が異なる場合は、クラスタ長の小さい方のパスが選択されます。クラスタ リスト属性なしでパスを受信した場合、クラスタの長さは 0 であると見なされます。
14. 最後に、IP アドレスの小さいネイバーから受信したパスが選択されます。ローカル生成されたパス（たとえば、再配布されたパス）は、ネイバー IP アドレスが 0 であると見なされます。

比較の順序

BGP 最適パス アルゴリズム実装のパート 2 では、パスの比較順序を決定します。比較順序は次のように決定されます。

1. 各グループ内のすべてのパス間で MED を比較できるように、パスがグループ分けされます。2 つのパス間で MED を比較できるかどうかは、#unique_73 と同じルールを使用して決定されます。通常、この比較の結果は、ネイバー AS ごとに 1 グループになります。**bgp bestpath med always** コマンドが設定されている場合は、パスを含む 1 グループだけがあります。
2. 各グループ内の最適パスが決定されます。最適パスは、グループ内のすべてのパスを反復処理し、その時点までの最適なパスを追跡することによって決定されます。各パスが、この時点までの最適なパスと比較され、より適していれば新しいこの時点までの最適なパスになって、グループ内の次のパスと比較されます。
3. ステップ 2 の各グループから選択した最適パスで構成される、パスのセットを形成します。このパス セットに対してステップ 2 と同様の比較を繰り返すことによって、全体としての最適パスを選択します。

最適パスの変更の抑制

実装のパート 3 では、最適パスの変更を抑制するかどうか、つまり、新しい最適パスを使用するのか、既存の最適パスの使用を続行するのかを決定します。最適パス選択アルゴリズムが任意性を持つ部分まで、新規の最適パスと一致している場合は（ルータ ID が同一であることが前提）、引き続き既存の最適パスを使用できます。既存の最適パスの使用を続行すると、ネットワークでのチャーンを回避できます。

（注） この抑制動作は、IETF ネットワーキング ワーキング グループの draft-ietf-idr-bgp4-24.txt 資料に準拠していませんが、IETF ネットワーキング ワーキング グループの draft-ietf-idr-avoid-transition-00.txt 資料に指定されています。

この抑制動作は、**bgp bestpath compare-routerid** コマンドを設定してオフにできます。このコマンドを設定すると、新しい最適パスが常に既存の最適パスよりも優先されます。

それ以外の場合は、次の手順を使用して、最適パスの変更を抑制するかどうかが決定されます。

1. 既存の最適パスが有効でなくなった場合は、変更を抑制できません。
2. 既存または新規の最適パスを内部（またはコンフェデレーション）ピアから受信したか、ローカルで生成した（再配布によるなど）場合は、変更を抑制できません。つまり、抑制は、両方のパスを外部ピアから受信した場合のみ可能です。
3. パスを同じピアから受信した場合（通常はパスのルータ ID が同一）は、変更を抑制できません。ルータ ID は、#unique_73 のルールを使用して計算されます。
4. パスの重み、ローカル プリファレンス、起点、またはネクスト ホップへの IGP メトリックが異なる場合は、変更を抑制できません。このすべての値は、#unique_73 のルールを使用して計算されます。
5. パスの AS パス長が異なり、**bgp bestpath as-path ignore** コマンドが設定されていない場合は、変更を抑制できません。この場合もやはり、AS パスの長さは、#unique_73 のルールを使用して計算されます。
6. パスの MED を比較でき、MED が異なる場合は、変更を抑制できません。MED を比較できるかどうかは、#unique_73 で説明されている MED 値の計算とまったく同じルールによって判定されます。
7. ステップ 1 ～ステップ 6 のすべてのパス パラメータに該当しない場合は、変更を抑制できます。

アドミニストレーティブ ディスタンス

アドミニストレーティブディスタンスは、ルーティング情報源の信頼性を示す評価基準です。通常は、値が大きいほど、信頼性の格付けが下がります。BGP のアドミニストレーティブディスタンスを指定する方法については、*Routing Command Reference for Cisco ASR 9000 Series Routers* の「BGP コマンド」のモジュールを参照してください。

一般的にルートは複数のプロトコルによって検出されます。アドミニストレーティブディスタンスは、複数のプロトコルから学習したルートを区別するために使用されます。最もアドミニストレーティブ ディスタンスが低いルートが IP ルーティング テーブルに組み込まれます。BGP はデフォルトで、表 1 : デフォルトの BGP アドミニストレーティブ ディスタンス （48 ページ）のアドミニストレーティブ ディスタンスを使用します。

表 *1:* デフォルトの *BGP* アドミニストレーティブ ディスタンス

ディスタンス	デフォルト値	機能
外部	20	eBGP から学習したルートに適用されます。
内部	200	iBGP から学習したルートに適用されます。
ローカル	200	ルータを起点とするルートに適用されます。

（注） ディスタンスは BGP パス選択アルゴリズムに影響しませんが、BGP で学習されたルートを IP ルーティング テーブルに組み込むかどうかを左右します。

通常、eBGP を介して学習されたルートは、ディスタンス（20）を理由として IP ルーティング テーブルに組み込まれます。ただし、2 つの AS には IGP-learned バックドア ルートと eBGP-learned のルートがあります。ポリシーは、IGP-learned パスを優先パスとして使用し、IGP パスが停止しているときに eBGP-learned パスを使用するなどの内容になります。図 3 : バックドアの例 （48 ページ） を参照してください。

図 *3:* バックドアの例

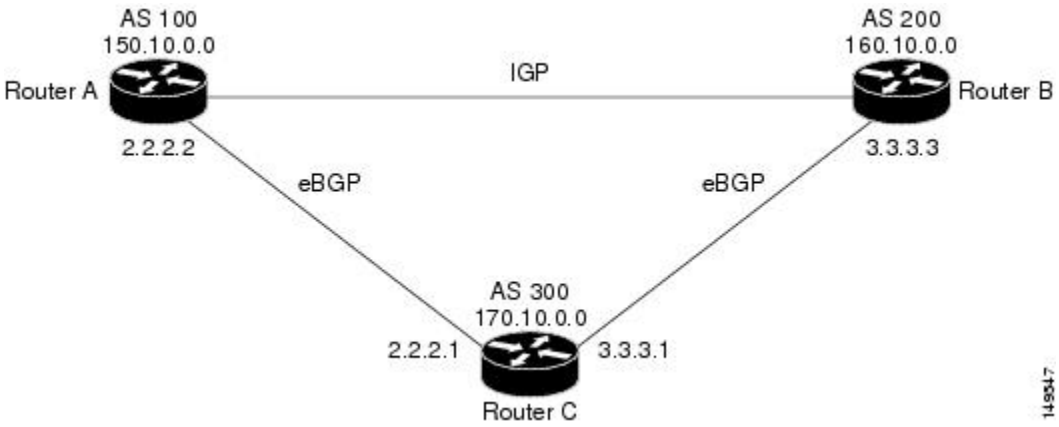


図 3 : バックドアの例 （48 ページ） では、ルータ A と C、ルータ B と C が eBGP を実行しています。ルータ A および B は、IGP を実行しています（ルーティング情報プロトコル（RIP）、Enhanced Interior Gateway Routing Protocol（IGRP）、Enhanced IGRP、または Open Shortest Path First（OSPF）など）。RIP、IGRP、Enhanced IGRP、および OSPF のデフォルト ディスタンスは、それぞれ、120、100、90、および 110 です。これらの距離はすべて eBGP のデフォルト ディスタンス（20）よりも長くなります。通常は、ディスタンスの一番小さいルートが優先されます。

ルータ A は、160.10.0.0 に関するアップデートを、eBGP と IGP の 2 つのルーティング プロトコルから受信します。eBGP のデフォルトのディスタンスが IGP のデフォルトのディスタンスよりも低いので、ルータ A はルータ C からの eBGP-learned ルートを選択します。ルータ A に

ルータ B（IGP）からの 160.10.0.0 について学習させる場合は、BGP バック ドアを確立します。を参照してください。

次の例では、ネットワーク バック ドアが設定されています。

```
RP/0/RSP0/cpu 0: router(config)# router bgp 100
RP/0/RSP0/cpu 0: router(config-bgp)# address-family ipv4 unicast
RP/0/RSP0/cpu 0: router(config-bgp-af)# network 160.10.0.0/16 backdoor
```

ルータ A では、eBGP-learned ルートをローカルとして扱い、ディスタンス 200 で IP ルーティング テーブルに組み込みます。このネットワークは Enhanced IGRP を介しても学習しているため（ディスタンスは 90）、Enhanced IGRP ルートは、IP ルーティング テーブルに正常に組み込まれ、トラフィックの転送に使用されます。Enhanced IGRP-learned ルートが停止すると、eBGP-learned ルートが IP ルーティング テーブルに組み込まれ、トラフィックの転送に使用されます。

Although BGP ではネットワーク 160.10.0.0 をローカル エントリとして扱いますが、通常、ローカル エントリをアドバタイズするようにネットワーク 160.10.0.0 をアドバタイズすることはありません。

マルチプロトコル BGP

マルチプロトコル BGP は、BGP の拡張バージョンで、複数のネットワーク層プロトコル、および IP マルチキャスト ルートに関するルーティング情報を伝送します。BGP は、ユニキャスト ルーティングのセットと、マルチキャスト ルーティングのセットの 2 つのルート セットを伝送します。マルチキャスト ルーティングと関連付けられたルートは、データ分散ツリーを構築するためにプロトコル独立マルチキャスト（PIM）機能で使用されます。

マルチプロトコル BGP は、トラフィックの種類別に使用するリソースを制限するなどの目的で、マルチキャスト トラフィック専用のリンクが必要な場合に役立ちます。マルチプロトコル BGP を使用すると、マルチキャスト ルーティング トポロジとは異なるユニキャスト ルーティング トポロジによって、ネットワークおよびリソースの制御を向上できます。

BGP でドメイン間マルチキャスト ルーティングを実行する唯一の方法は、ユニキャスト ルーティングに対応できる BGP インフラストラクチャを使用することでした。通常は、すべてのマルチキャスト トラフィックを 1 つのネットワーク アクセス ポイント（NAP）で交換します。これらのルータがマルチキャスト対応でないか、マルチキャスト トラフィックのフローに適用するさまざまなポリシーがある場合は、マルチプロトコル BGP なしでマルチキャスト ルーティングをサポートできません。

（注） ユニキャストとマルチキャストの両方のネットワーク層到達可能性情報（NLRI）を交換する BGP ピアを設定することはできますが、マルチ プロトコル BGP クラウドと BGP クラウドを接続することはできません。つまり、マルチプロトコル BGP ルートを BGP に再配布できません。

図 4 : 不一致のユニキャスト ルートおよびマルチキャスト ルート （50 ページ）に、一致しておらず、したがって、マルチプロトコル BGP なしでは実現できない、単純なユニキャストとマルチキャストのトポロジを示します。

自律システム 100、200、および 300 は、FDDI リングである 2 つの NAP にそれぞれ接続しています。1 つはユニキャスト ピアリング（ユニキャスト トラフィックの交換）に使用されます。Multicast Friendly Interconnect（MFI）リングは、マルチキャスト ピアリング（マルチキャスト トラフィックの交換）に使用されます。各ルータは、ユニキャストおよびマルチキャスト対応です。

図 *4:* 不一致のユニキャスト ルートおよびマルチキャスト ルート

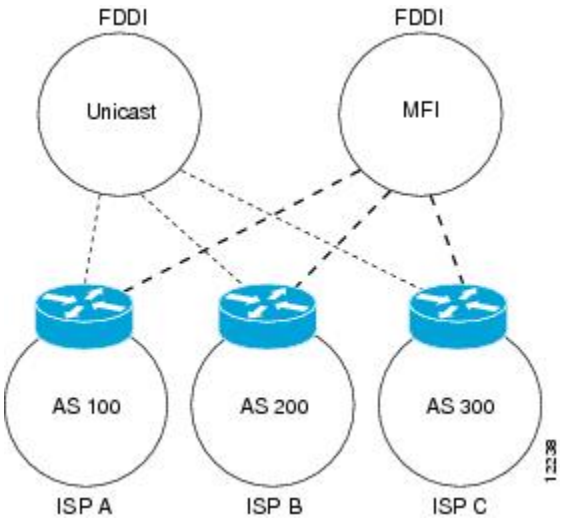


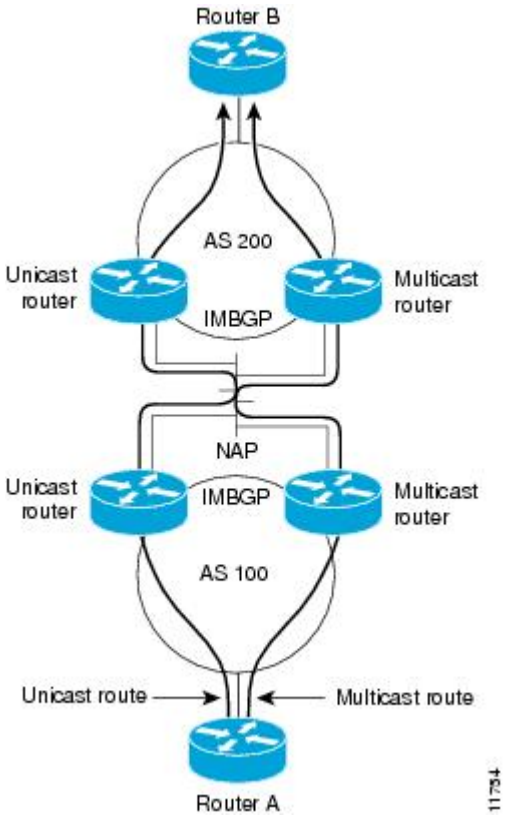
図 5 : マルチキャスト BGP 環境 （51 ページ） は、ユニキャストだけに対応したルータおよびマルチキャストだけに対応したルータのトポロジです。左側にある 2 つのルータはユニキャストだけに対応しています（マルチキャスト ルーティングをサポートしていないか、マルチキャスト ルーティングを実行するよう設定されていない）。右側にある 2 つのルータはマルチキャストだけに対応したルータです。ルータ A および B は、ユニキャストおよびマルチキャスト ルーティングの両方をサポートしています。ユニキャストだけに対応したルータおよびマルチキャストだけに対応したルータは、1 つの NAP に接続されています。

図 5 : マルチキャスト BGP 環境 （51 ページ） では、ユニキャスト トラフィックだけがルータ A からユニキャスト ルータを経由してルータ B との間を行き来できます。マルチキャスト トラフィックは、このパス上を流れることができないため、別のルーティング テーブルが必要です。マルチキャスト トラフィックは、ルータ A からマルチキャスト ルータを経由してルータ B との間を行き来するパスを使用します。

図 5 : マルチキャスト BGP 環境 （51 ページ） に、ルータ A からルータ B へのユニキャスト ルートおよびマルチキャスト ルートを別々に持つマルチプロトコル BGP 環境を示します。マルチプロトコル BGP では、これらのルートが不一致であることが許可されています。この図では、両方の自律システムに内部マルチプロトコル BGP（IMBGP）が設定されている必要があります。

PIM などのマルチキャスト ルーティング プロトコルでは、マルチキャスト BGP データベースを使用して、マルチキャスト対応の送信元に対する Reverse Path Forwarding（RPF）検索を実行します。したがって、マルチキャスト トポロジ上ではパケットの送信と受け入れが可能ですが、ユニキャスト トポロジ上ではできません。

図 *5 :* マルチキャスト *BGP* 環境



ルート ダンプニング

ルート ダンプニングは、インターネットワーク上でのフラッピング ルートの伝搬を最小限に抑える BGP 機能です。ルートの状態が使用可能、使用不可能、使用可能、使用不可能という具合に、繰り返し変化する場合、ルートはフラッピングと見なされます。

たとえば、自律システム 1、自律システム 2、および自律システム 3 の 3 つの BGP 自律システムがあるネットワークについて考えます。自律システム 1 のネットワーク A へのルートがフラッピングする（利用できなくなる）と仮定します。ルート ダンプニングがない状況では、自律システム 1 から自律システム 2 への eBGP ネイバーは、取り消しメッセージを自律システム 2 に送信します。次に自律システム 2 内の境界ルータは、取り消しメッセージを自律システム 3 に伝播します。ネットワーク A へのルートが再出現したとき、自律システム 1 は自律システム 2 に、自律システム 2 は自律システム 3 にアドバタイズメント メッセージを送信します。ネットワーク A へのルートが利用可能になったり不可になったりを繰り返す場合、取り消しメッセージおよびアドバタイズメント メッセージが多数送信されます。ルート フラッピング

は、インターネットに接続されたインターネットワークでの問題です。インターネットのバックボーンでルートのフラッピングが生じると、通常、多くのルートに影響を与えるからです。

フラッピングの最小化

ルート ダンプニング機能は、次のようにしてフラッピングの問題を最小限に抑えます。ここでも、ネットワーク A へのルートがフラッピングしたと仮定します。（ルート ダンプニングがイネーブルになっている）自律システム 2 内のルータは、ネットワーク A にペナルティ 1000 を割り当てて、履歴状態に移行させます。自律システム 2 内のルータは、引き続きネイバーにルートのステータスをアドバタイズします。ペナルティは累積されます。ルート フラップが非常に頻繁に発生し、ペナルティが設定可能な抑制制限を超える場合は、フラップの発生回数に関係なく、ルータはネットワーク A へのルートのアドバタイズを停止します。このようにして、ルート ダンプニングが発生します。

ネットワーク A に課されたペナルティは再使用制限に達するまで減衰し、達すると同時にそのルートは再びアドバタイズされます。再使用制限の半分の時点で、ネットワーク A へのルートのダンプニング情報が削除されます。

（注） ルート ダンプニングがイネーブルの場合は、リセットによってルートが取り消されるときでも、BGP ピアのリセットにペナルティは適用されません。

BGP ルーティング ドメイン コンフェデレーション

iBGP メッシュを削減する方法の 1 つとして、ある自律システムを複数の副自律システムに分割し、単一のコンフェデレーションにグループ化することがあげられます。外部からは、このコンフェデレーションは単一の自律システムであるかのように見えます。各自律システムは内部で完全にメッシュ化されていて、同じコンフェデレーション内の他の自律システムとの間には数本の接続があります。異なる自律システム内にあるピアは eBGP セッションを持ちますが、ルーティング情報は iBGP ピアと同様な方法で交換されます。具体的には、ネクスト ホップ、MED、およびローカル プリファレンス情報は維持されます。この機能により、自律システムすべてに対して単一の IGP を保持できます。

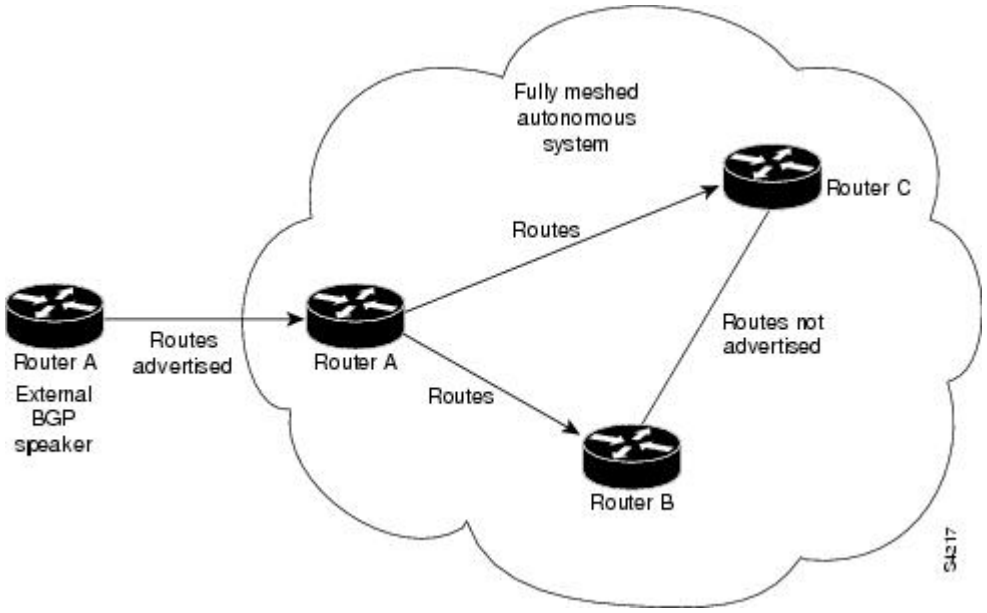
BGP ルート リフレクタ

BGP を使用するには、すべての iBGP スピーカーが完全メッシュ化されている必要があります。ただし、iBGP スピーカーの数が多い場合、この要件には適切な拡張性がありません。コンフェデレーションを設定する代わりに、ルート リフレクタ設定を使用すると iBGP メッシュを削減できます。

図 6 : 完全メッシュ化された 3 つの iBGP スピーカー （53 ページ） に、3 つの iBGP スピーカー（ルータ A、B、C）を持つ、単純な iBGP 設定の例を示します。ルート リフレクタがない場合、ルータ A は外部ネイバーからルートを受け取ると、そのルートをルータ B と C の両方にアドバタイズする必要があります。ルータ B と C は iBGP が学習したルートを他の iBGP ス

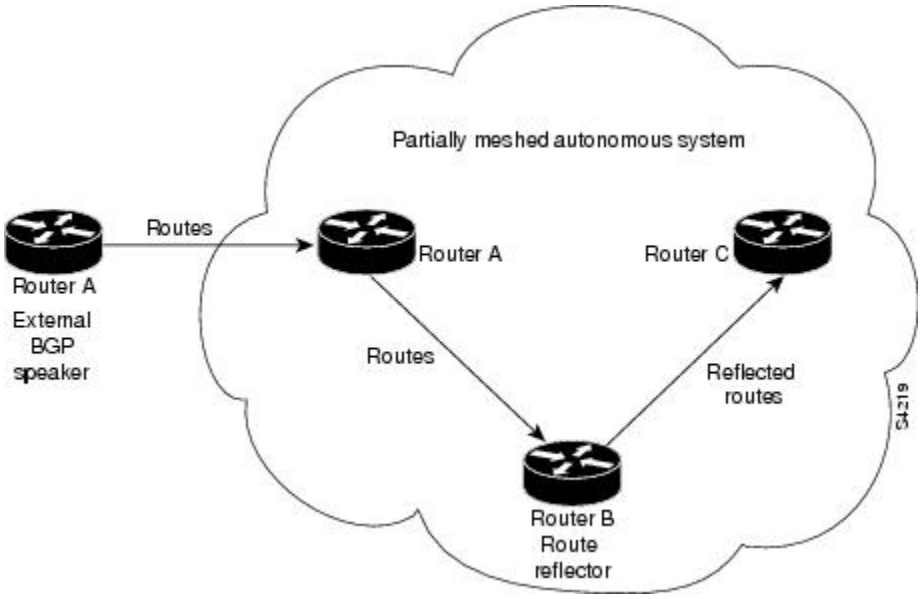
ピーカーに再アドバタイズしません。これは、これらのルータが内部ネイバーから他の内部ネイバーに学習したルートを渡さないことで、ルーティング情報のループを防ぐためです。

図 *6*:完全メッシュ化された *3* つの *iBGP* スピーカー



ルートリフレクタがある場合は、学習したルートをネイバーに渡す方法があるため、すべてのiBGP スピーカーを完全にメッシュ化する必要はありません。このモデルでは、iBGP が学習したルートを一連の iBGP ネイバーに渡す役割を持つルート リフレクタとして、1 つの iBGP ピアを設定しています。図 7 : ルート リフレクタのある単純な BGP モデル （54 ページ） では、ルータ B がルートリフレクタとして設定されています。ルータ A からアドバタイズされたルートをルートリフレクタが受信すると、ルータ C にアドバタイズします。逆の場合も同じです。このスキームにより、ルータ A とルータ C 間の iBGP セッションは不要になります。

図 *7:* ルート リフレクタのある単純な *BGP* モデル



ルート リフレクタの内部ピアは、次の 2 種類のグループに分けられます。クライアントのピアと、自律システム内の他の全ルータ（非クライアント ピア）です。ルート リフレクタは、これらの 2 つのグループ間でルートを反映させます。ルート リフレクタおよびそのクライアント ピアは、クラスタを形成します。非クライアント ピアは相互に完全メッシュ構造にする必要がありますが、クライアント ピアはその必要はありません。クラスタ内のクライアントは、クラスタ外の iBGP スピーカーとは通信しません。

図 *8 :* より複雑な *BGP* ルート リフレクタのモデル

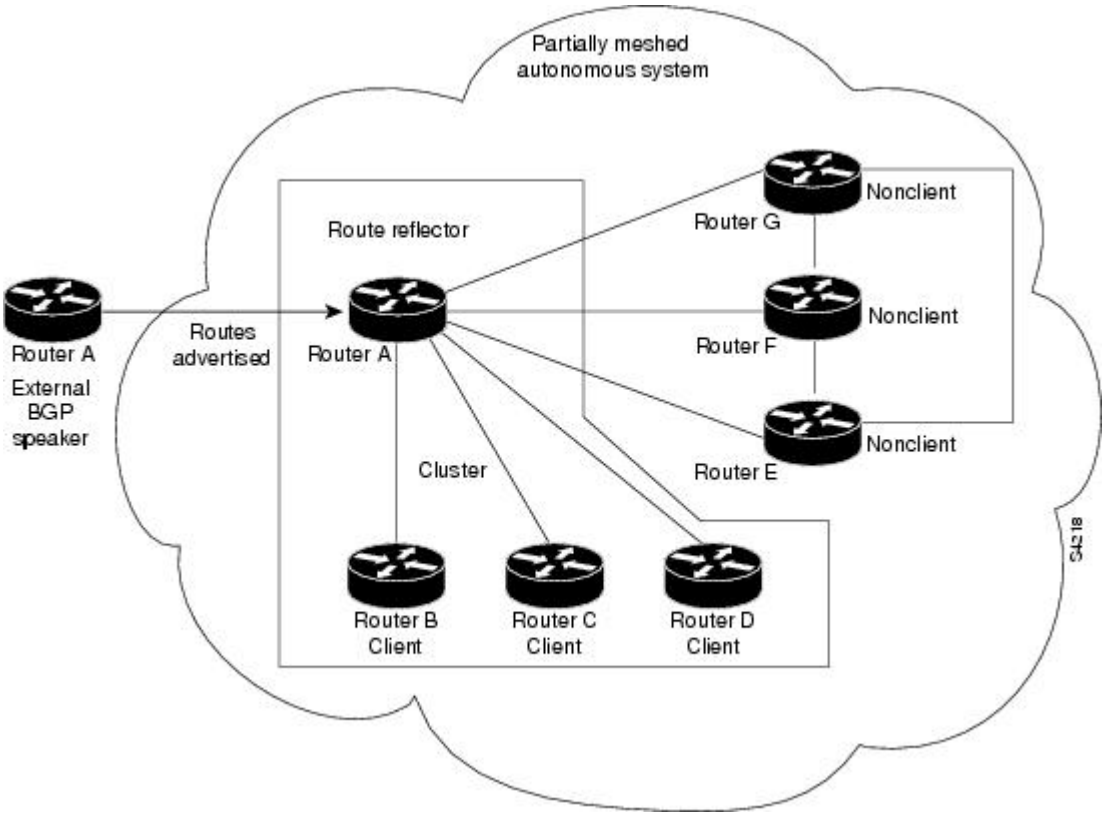


図 8 : より複雑な BGP ルート リフレクタのモデル （55 ページ） に、より複雑なルート リフレクタのスキームを示します。ルータ A は、ルータ B、C、および D があるクラスタ内のルート リフレクタです。ルータ E、F、および G は完全にメッシュ化された非クライアント ルータです。

ルート リフレクタがアドバタイズされたルートを受信すると、ネイバーに応じて、次のようなアクションを取ります。

- 外部 BGP スピーカーからのルートをすべてのクライアントおよび非クライアント ピアにアドバタイズします。
- 非クライアント ピアからのルートをすべてのクライアントにアドバタイズします。
- クライアントからのルートをすべてのクライアントおよび非クライアント ピアにアドバタイズします。したがって、クライアントを完全メッシュ構造にする必要はありません。

ルート リフレクタ対応の BGP スピーカーとともに、ルート リフレクタの概念に対応していない BGP スピーカーを併用することもできます。これらは、クライアントまたは非クライアント グループのメンバとなることができます。したがって、旧 BGP モデルからルート リフレクタ モデルへ、簡単に順次移行できます。たとえば、最初に、ルート リフレクタおよびいくつかのクライアントを持つ単一のクラスタを作成します。他のすべての iBGP スピーカーはルート リフレクタに対して非クライアント ピアとすることができ、クラスタを作成して徐々に追加します。

自律システムは複数のルート リフレクタを持つことができます。ルート リフレクタは、他のルートリフレクタを他の iBGP スピーカーと同様に扱います。ルートリフレクタは、他のルート リフレクタをクライアント グループまたは非クライアント グループに含むように設定できます。単純な設定では、バックボーンを多数のクラスタに分割してもかまいません。各ルートリフレクタは、非クライアント ピアとして他のルート リフレクタとともに設定されます（このため、すべてのルートリフレクタは完全メッシュ化されます）。クライアントは、所属するクラスタのルート リフレクタとだけ、iBGP セッションを維持するように設定されます。

通常、クライアントのクラスタには、ルートリフレクタが 1 つ存在します。その場合、クラスタはルートリフレクタのルータ ID で識別されます。冗長性を向上させ、シングル ポイント障害を避けるために、クラスタは複数のルートリフレクタを含むことがあります。この場合、クラスタ内のすべてのルート リフレクタにクラスタ ID を設定し、ルートリフレクタが同一クラスタ内のルートリフレクタからのアップデートを識別できるようにする必要があります。クラスタに機能を提供しているルートリフレクタはすべて完全メッシュ化され、同一のクライアントおよび非クライアント ピアのセットを持っている必要があります。

デフォルトでは、ルートリフレクタのクライアントは完全メッシュ化されている必要はなく、クライアントからのルートは他のクライアントに反映されます。ただし、クライアントが完全メッシュ化されている場合は、ルートリフレクタはルートをクライアントに反映する必要はありません。

iBGP が学習したルートが反映されるため、ルーティング情報がループする場合があります。ルート リフレクタ モデルには、ルーティングのループを防ぐ、次のようなメカニズムがあります。

- 送信元 ID は、任意で非過渡的な BGP 属性です。これは 4 バイトの属性で、ルート リフレクタにより作成されます。この属性は、ローカル自律システムのルートの送信元のルータ ID を保持します。したがって、設定ミスによりルーティング情報が送信元に戻ってくる場合、その情報は無視されます。
- クラスタ リストは任意で非過渡的な BGP 属性です。これは、ルートが渡したクラスタ ID のシーケンスです。ルートリフレクタでは、クライアントから非クライアント ピアにルートを反映するとき（およびその逆のとき）、ローカル クラスタ ID をクラスタ リストに付加します。クラスタ リストが空の場合は、新規のクラスタ リストが作成されます。ルートリフレクタでは、この属性を使用して、設定ミスによりルーティング情報が同じクラスタにループ バックしているかどうかを識別できます。クラスタ リストにローカル クラスタ ID が見つかった場合、そのアドバタイズメントは無視されます。

RPL：プレフィックスが is-best-path/is-best-multipath の場合

ボーダー ゲートウェイ プロトコル（BGP）ルータは、同じ宛先への複数のパスを受信します。標準として、デフォルトでは、BGP ベストパスアルゴリズムが IP ルーティングテーブルにインストールする最適なパスを決定します。これはトラフィックの転送に使用されます。

BGP は、最初の有効なパスを現在のベストパスとして割り当てます。次に、BGP は、ベストパスとリスト内の次のパスとを比較します。このプロセスは、BGP が有効なパスのリストの最後に到達するまで継続されます。これには、ベストパスの決定に使用されるすべてのルールが

含まれます。指定されたアドレスプレフィックスに複数のパスがある場合、BGPは次のように処理します。

- ベストパス選択ルールに従って、パスの 1 つをベストパスとして選択します。
- 転送テーブルにベストパスをインストールします。各 BGP スピーカーは、ピアへのベストパスのみをアドバタイズします。

（注） ベストパスのみを送信するアドバタイズメントルールは、そのピアに対して BGP スピーカ上に存在する宛先の完全なルーティング状態を伝達しません。

BGPスピーカがピアのいずれかからパスを受信した後、ピアがそのパスをパケットの転送に使用します。他のすべてのピアは、このピアから同じパスを受信します。これにより、BGPネットワークでの一貫したルーティングが実現します。リンク帯域幅使用率を向上させるには、ほとんどの BGP 実装では、特定の条件を満たす追加パスをマルチパスとして選択し、それらを転送テーブルにインストールします。このような着信パケットは、ベストパスとマルチパス上でロードバランシングされます。ピアにアドバタイズされていない転送テーブルにパスをインストールできます。RR ルートリフレクタは、ベストパスとマルチパスを検出します。このようにして、ルートリフレクタはベストパスとマルチパスに異なるコミュニティを使用します。この機能を使用すると、RR または境界ルータによって実行されるローカルの決定を BGP で通知できます。この新機能を使用した場合は、コミュニティストリングを使用して RR によって選択されました（たとえば、is-best-path の場合は community 100:100）。コントローラは、どのベストパスがすべての R に送信されるかを確認します。ボーダー ゲートウェイ プロトコル ルータは、同じ宛先への複数のパスを受信します。ベストパスの計算を実行している間は、1 つのベストパスが存在し、場合によっては同等のパスおよび同等でない若干数のパスが存在します。したがって、abest-path と is-equal-best-path の要件です。

BGP のベストパスアルゴリズムは、IP ルーティングテーブル内でベストパスを決定し、トラフィックの転送に使用します。RPL内のこの機能拡張により、決定を行うためのポリシーを作成できます。ベストパスのローカル選択のためのコミュニティストリングの追加。BGP追加パス（Add Path）の導入により、BGP はベストパスよりも多くを通知するようになりました。BGP はベストパスと、ベストパスと同等のパス全体を通知できます。これは、BGP マルチパスルールとすべてのバックアップパスに従っています。

RPLネクストホップ破棄設定を使用したリモートトリガ型ブラックホールのフィルタリング

リモートトリガ型ブラックホール（RTBH）フィルタリングは、保護されたネットワークに入る前に望ましくないトラフィックをドロップする機能を提供する技術です。RTBH フィルタリングは、null0 インターフェイスに転送することによって、送信元アドレスまたは宛先アドレスのいずれかに基づいて、ネットワークのエッジで望ましくないトラフィックをすばやくドロップする方法を提供します。宛先アドレスに基づく RTBH フィルタリングは、一般に宛先

ベースの RTBH フィルタリングと呼ばれます。一方、送信元アドレスに基づく RTBH フィルタリングは、送信元ベースの RTBH フィルタリングと呼ばれます。

RTBH フィルタリングは、セキュリティツールキットの多くの技術の 1 つであり、次の方法でネットワークセキュリティを強化するために一緒に使用できます。

- DDoS 攻撃とワーム攻撃を効果的に軽減する
- 攻撃下でターゲットを宛先とするすべてのトラフィックを隔離する
- ブロックリスト フィルタリングの適用

宛先ベースの RTBH フィルタリングの設定

RTBH は、**set next-hop discard** コマンドを使用して、ネクストホップで望ましくないトラフィックを破棄するルートポリシー（RPL）を定義することによって実装されます。

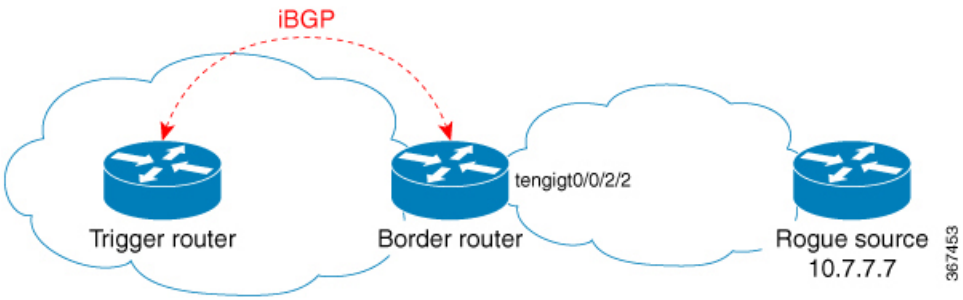
RTBH フィルタリングは、対象のプレフィックスのネクストホップをヌルインターフェイスに設定します。対象を宛先とするトラフィックは、入力時にドロップされます。

set next-hop discard 設定は、ネイバー インバウンド ポリシーで使用されます。この設定がパスに適用されている場合、プライマリネクストホップは実際のパスに関連付けられますが、Null0 に設定されたネクストホップで RIB が更新されます。受信したプライマリネクストホップが到達不能であっても、RTBH パスは到達可能と見なされ、ベストパス選択プロセスの候補となります。RTBH パスは、通常の BGP アドバタイズメントルールに基づいて、受信したネクストホップまたは nexthop-self のいずれかを持つ他のピアに再度アドバタイズされます。

RTBH フィルタリングの一般的な展開シナリオでは、アクセスおよび集約ポイントで内部ボーダー ゲートウェイ プロトコル（iBGP）を実行し、トリガーとして動作するようにネットワーク オペレーション センター（NOC）で個別のデバイスを設定する必要があります。トリガー側のデバイスは、iBGP 更新をエッジに送信します。これにより、望ましくないトラフィックが null0 インターフェイスに転送され、ドロップされます。

次に、不正ルータが境界ルータにトラフィックを送信しているトポロジを示します。

図 *9: RTBH* フィルタリングを実装するためのトポロジ



トリガールータに適用される設定

特殊なタグでマークされた静的ルートにコミュニティを設定し、BGP に適用する静的ルート再配布ポリシーを設定します。

```
route-policy RTBH-trigger
  if tag is 777 then
    set community (1234:4321, no-export) additive
    pass
  else
    pass
  endif
  end-policy

router bgp 65001
 address-family ipv4 unicast
  redistribute static route-policy RTBH-trigger
 !
 neighbor 192.168.102.1
  remote-as 65001
  address-family ipv4 unicast
   route-policy bgp_all in
   route-policy bgp_all out
```

ブラックホール化させる必要がある送信元プレフィックスの特殊なタグを使用して静的ルートを設定します。

```
router static
 address-family ipv4 unicast
 10.7.7.7/32 Null0 tag 777
```

ボーダールータに適用される設定

トリガールータのコミュニティセットと一致するルートポリシーを設定し、次のように set next-hop discard を設定します。

```
route-policy RTBH
  if community matches-any (1234:4321) then
    set next-hop discard
  else
    pass
  endif
end-policy
```

次のように、ルートポリシーを iBGP ピアに適用します。

```
router bgp 65001
 address-family ipv4 unicast
 !
 neighbor 192.168.102.2
  remote-as 65001
  address-family ipv4 unicast
   route-policy RTBH in
   route-policy bgp_all out
```

確認

境界ルータで、プレフィックス 10.7.7.7/32 に Nexthop-discard というフラグが付けられます。

```
RP/0/RSP0/CPU0:router#show bgp
BGP router identifier 10.210.0.5, local AS number 65001
BGP generic scan interval 60 secs
BGP table state: Active
Table ID: 0xe0000000   RD version: 12
```

```
BGP main routing table version 12
BGP scan interval 60 secs

Status codes: s suppressed, d damped, h history, * valid, > best
              i - internal, r RIB-failure, S stale, N Nexthop-discard
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network            Next Hop            Metric LocPrf Weight Path
N>i10.7.7.7/32        192.168.102.2            0    100      0 ?
```

```
RP/0/RSP0/CPU0:router#show bgp 10.7.7.7/32
BGP routing table entry for 10.7.7.7/32
Versions:
  Process           bRIB/RIB  SendTblVer
  Speaker                12          12
Last Modified: Jul  4 14:37:29.048 for 00:20:52
Paths: (1 available, best #1, not advertised to EBGP peer)
  Not advertised to any peer
  Path #1: Received by speaker 0
  Not advertised to any peer
  Local
    192.168.102.2 (discarded) from 192.168.102.2 (10.210.0.2)
      Origin incomplete, metric 0, localpref 100, valid, internal best, group-best
      Received Path ID 0, Local Path ID 1, version 12
      Community: 1234:4321 no-export
```

```
RP/0/RSP0/CPU0:router#show route 10.7.7.7/32

Routing entry for 10.7.7.7/32
  Known via "bgp 65001", distance 200, metric 0, type internal
  Installed Jul 4 14:37:29.394 for 01:47:02
  Routing Descriptor Blocks
    directly connected, via Null0
      Route metric is 0
  No advertising protos.
```

show コマンドのデフォルトのアドレス ファミリ

show コマンドのほとんどは、アドレスファミリ（AFI）およびサブアドレスファミリ（SAFI）の引数を使用します（AFI および SAFI については、RFC 1700 および RFC 2858 を参照してください）。Cisco IOS XR ソフトウェアパーサーには、afi および safi を設定して、**show** コマンドの実行時には指定する必要がないようにする機能があります。次のパーサー コマンドがあります。

- **set default-afi** { **ipv4** | **ipv6** | **all** }
- **set default-safi** { **unicast** | **multicast** | **all** }

パーサーでは、デフォルト afi 値が **ipv4** に、デフォルト safi 値が **unicast** に自動的に設定されます。デフォルトの afi 値を **ipv4** から変更する、あるいはデフォルトの safi 値を **unicast** から変更する場合、使用する必要があるのはパーサーコマンドのみです。**show** コマンドに指定された **afi** または **safi** キーワードは、パーサーコマンドを使用して設定した値を上書きします。afi および safi に現在設定されている値を確認するには、次の **show default-afi-safi-vrf** コマンドを使用します。

TCP Maximum Segment Size

最大セグメントサイズ（MSS）は、コンピュータまたは通信デバイスが単一のフラグメント化されていないTCPセグメントで受信できるデータの最大量です。すべてのTCPセッションは、単一のパケットで転送可能なバイト数に関する制限によってバインドされます。この制限がMSS です。TCP は、パケットを IP レイヤに渡す前に、送信キューでパケットをチャンクに分割します。

TCP MSS 値は、インターフェイスの最大伝送ユニット（MTU）に依存します。これは、1 つのインスタンスでプロトコルによって送信可能なデータの最大長です。最大 TCP パケット長は、TCP セットアップ プロセス中に、送信元デバイスのアウトバウンド インターフェイスの MTU と宛先デバイスによって知らされる MSS の両方によって決まります。MSS が MTU に近づくほど、BGP メッセージの転送がより効率的になります。データフローの各方向に異なる MSS 値を使用できます。

ネイバー単位の TCP MSS

ネイバー単位の TCP MSS 機能を使用すると、ネイバーごとに一意の TCP MSS プロファイルを作成できます。ネイバー単位の TCP MSS は、ネイバーグループとセッショングループの 2 つのモードでサポートされています。以前は、TCP MSS 設定は、BGP 設定のグローバルレベルでのみ使用できるようになっていました。

ネイバー単位の TCP MSS 機能では、以下を行えます。

- ネイバー単位の TCP MSS 設定を有効にする。
- **inheritance-disable** コマンドを使用して、ネイバーグループまたはセッショングループの特定のネイバーの TCP MSS を無効にする。
- TCP MSS 値の設定を解除する。設定解除時に、プロトコル制御ブロック（PCB）の TCP MSS 値がデフォルト値に設定されます。

（注） デフォルトの TCP MSS 値は 536（オクテット単位）または 1460（バイト単位）です。MSS のデフォルトの 1460 は、TCP がパケットを IP レイヤに渡す前に、送信キュー内のデータを 1460 バイトのチャンクにセグメント化することを意味します。

ネイバー単位の TCP MSS を設定するには、ネイバー単位、ネイバーグループまたはセッショングループの設定で **tcp mss** コマンドを使用します。

詳細な設定手順については、ネイバー単位の TCP MSS の設定 （109 ページ）を参照してください。

ネイバー単位の TCP MSS を無効にする詳細な手順については、ネイバー単位の TCP MSS の無効化 （111 ページ）を参照してください。

MPLS VPN Carrier Supporting Carrier

Carrier Supporting Carrier（CSC）は、サービス プロバイダーの 1 つが別のサービス プロバイダーに自社のバックボーン ネットワークのセグメントの使用を許可する状況を記述した用語です。他のプロバイダーにバックボーン ネットワークのセグメントを提供するサービス プロバイダーは、バックボーン キャリアと呼ばれます。バックボーン ネットワークのセグメントを使用するサービス プロバイダーは、カスタマー キャリアと呼ばれます。

バックボーン キャリアは、ボーダー ゲートウェイ プロトコル/マルチプロトコル ラベル スイッチング（BGP/MPLS）VPN サービスを提供します。カスタマー キャリアは、次のいずれかになります。

- インターネット サービス プロバイダー（ISP）（定義上、ISP は VPN サービスを提供しません）
- BGP/MPLS VPN サービス プロバイダー

BGP をイネーブルにするように CSC ネットワークを設定して、バックボーン キャリア プロバイダー エッジ（PE）ルータとカスタマー キャリア カスタマー エッジ（CE）ルータ間のルートおよび MPLS ラベルを、複数パスを使用して転送できます。BGP を使用して IPv4 ルートと MPLS ラベル ルートを配布する利点を次に示します。

- BGPは、VPNルーティング/転送（VRF）テーブル内で内部ゲートウェイ プロトコル（IGP）およびラベル配布プロトコル（LDP）の代わりをします。BGPを使用して、ルートおよび MPLS ラベルを配布できます。2 つではなく単一のプロトコルを使用すると、設定およびトラブルシューティングが簡単になります。
- BGP は、2 つの ISP を接続する場合の優先ルーティング プロトコルです。主な理由は、そのルーティング ポリシーと拡張性です。ISP では、通常、2 つのプロバイダー間で BGP を使用します。この機能を使用すると、これらの ISP は BGP を使用できます。

BGP を使用した MPLS VPN CSC の設定の詳細については、 *MPLS Configuration Guide for Cisco ASR 9000 Series RoutersMPLS Configuration Guide for Cisco NCS 560 Series Routers*の「*Implementing MPLS Layer 3 VPNs on* Cisco ASR 9000 シリーズ ルータ 」のモジュールを参照してください。

BGP キーチェーン

BGP キーチェーンを使用すると、2 つの BGP ピア間のキーチェーン認証がイネーブルになります。BGP のエンドポイントは、どちらも draft-bonica-tcp-auth-05.txt を順守する必要があり、一方のエンドポイントのキーチェーンと、もう一方のエンドポイントのパスワードは機能しません。

キーチェーン管理の詳細については、 *System Security Configuration Guide for Cisco ASR 9000 Series Routers*を参照してください。

BGP では、認証にこのキーチェーンを使用して、ヒットレス キー ロールオーバーを実装できます。キー ロールオーバーの仕様は時間に基づいているため、ピア間で時計のずれがあるとロールオーバーのプロセスに影響します。許容値の指定を設定できるため、承認時間枠をその

分だけ（前後に）拡張できます。この承認時間枠により、アプリケーション（ルーティングプロトコルおよび管理プロトコルなど）のヒットレス キー ロールオーバーが容易になります。

キーのロール オーバーは、エンドポイントでのキーチェーン設定の不一致が原因でセッショントラフィック（送信または受信）で使用する共通のキーがない場合を除き、BGP セッションには影響しません。

BGP ノンストップ ルーティング

ボーダー ゲートウェイ プロトコル（BGP）のノンストップ ルーティング（NSR）とステートフル スイッチオーバー（SSO）機能を使用すると、すべての bgp ピアリングで BGP 状態を維持し、サービスを中断させるおそれのあるイベントの実行中にも連続的なパケット転送を行えるようになります。NSR の下では、サービスを中断するおそれのあるイベントは、ピア ルータに表示されません。プロトコル セッションは中断されず、ルーティング ステートはプロセスの再起動とスイッチオーバーをまたがって維持されます。

BGP NSR では、次のイベントの際のノンストップ ルーティングを実現します。

- ルート プロセッサ スイッチオーバー
- BGP または TCP でのプロセスのクラッシュまたはプロセス障害

（注） BGP NSR は、デフォルトで有効になっています。BGP NSR を無効にするには、**nsr disable** コマンドを使用します。また、無効になっている BGP NSR を有効に戻すには、**no nsr disable** コマンドを使用します。

プロセスのクラッシュまたはプロセス障害が発生した場合、NSR は **nsr process-failures switchover** コマンドが設定されている場合にのみ維持されます。アクティブなインスタンスのプロセス障害が発生した場合は、**nsr process-failures switchover** により復旧処理としてフェールオーバーが設定され、スタンバイ ルート プロセッサ（RP）またはスタンバイ分散型ルートプロセッサ（DRP）にスイッチ オーバーが行われることで、NSR が維持されます。コンフィギュレーション コマンドの一例として、RP/0/RSP0/CPU0:router(config) # nsr process-failures switchover があります。

nsr process-failures switchover コマンドは、BGP または TCP プロセスがクラッシュした場合に NSR セッションと BGP セッションの両方を維持します。この設定を行わないと、BGP プロセスまたは TCP プロセスがクラッシュした場合に BGP ネイバー セションがフラップします。この設定は、BGP ネイバーのフラップが予想される場合に BGP プロセスまたは TCP プロセスが再起動する場合は役立ちません。

ルート プロセッサ スイッチオーバーおよびインサービス システムのアップグレード（ISSU）の間、NSR は TCP と BGP の両方のステートフル スイッチオーバー（SSO）によって実現されます。

NSR では、ネットワーク内の他のルータ上でソフトウェア アップグレードを強要せず、NSR をサポートするためにピア ルータは必要ありません。

障害に起因するルート プロセッサ スイッチオーバーが発生した場合、TCP 接続および BGP セッションはトランスペアレントにスタンバイ ルート プロセッサに移行され、スタンバイ ルート プロセッサがアクティブになります。既存のプロトコル ステートは、アクティブになるスタンバイ ルート プロセッサ上で維持されて、ピアによるプロトコル ステートのリフレッシュは不要です。

ソフト再設定やポリシーの変更などのイベントにより、BGP の内部状態が変化することがあります。このようなイベントの際に、アクティブとスタンバイの BGP プロセスの間でステートの一貫性を確保するために、同期ポイントとして機能する、ポストイットの概念が導入されています。

BGP NSR には次の機能があります。

- NSR 関連のアラームおよび通知
- 設定され、動作している NSR の状態は、個別に追跡される
- NSR 統計情報の収集
- **show** コマンドを使用した NSR 統計情報の表示
- XML スキーマのサポート
- アクティブとスタンバイのインスタンス間のステート同期を検証する監査メカニズム
- NSR をイネーブルおよびディセーブルにする CLI コマンド
- 5000 NSR セッションのサポート

BGP Local Label Retention

プライマリ PE-CE リンクが故障した場合、BGP では、プライマリ パスに対応するルートおよびこのルートのローカル ラベルを取り消し、デフォルトでは、ルーティング情報ベース（RIB）および転送情報ベース（FIB）にバックアップ パスをプログラムします。

ただし、プライマリ PE のすべての内部ピアがバックアップ パスを新しい最適パスとして使用するように再コンバージェンスするまで、トラフィックは、プライマリ パスに割り当てられたローカル ラベルとともに、引き続きプライマリ PE に転送されます。したがって、プライマリ パスに前に割り当てられていたローカル ラベルは、再コンバージェンス後、設定可能な期間、プライマリ PE 上で保持する必要があります。BGP Local Label Retention 機能を使用すると、ローカル ラベルを指定期間保持できます。時間を指定していない場合、ローカル ラベルは、デフォルト値の 5 分間保持されます。

retain local-label コマンドを使用すると、ネットワークがコンバージェンスされるまで、ローカル ラベルを保持できます。

BGP コマンドに対するコマンドライン インターフェイス（CLI）の一貫性

Cisco IOS XR リリース 3.9.0 以降、ボーダー ゲートウェイ プロトコル（BGP）コマンドでは、**disable** キーワードを使用して、機能を無効にします。キーワード **inheritance-disable** では、親レベルからの機能プロパティの継承が無効になります。

BGP の追加パス

ボーダーゲートウェイ プロトコル（BGP）の追加パス機能では、1 つのプレフィックスに対して複数のパスを送信できるように、BGP スピーカーの BGP プロトコル機械を変更します。これにより、ネットワークに「パスの多様性」が生まれます。追加パスにより、エッジルータでの BGP プレフィックス独立コンバージェンス（PIC）が可能になります。

（注） BGP 追加パス機能は、VRF ではサポートされていません。

BGP 追加パスでは、iBGP ネットワーク内の追加パス アドバタイズメントが可能になり、プレフィックスに対する次のタイプのパスがアドバタイズされます。

- バックアップ パス：高速コンバージェンスおよび接続の回復をイネーブルにします。
- グループ最適パス：ルート振動を解決します。
- すべてのパス：iBGP フル メッシュをエミュレートします。

（注） 追加パスは、MDT、トンネル、および L2VPN アドレス ファミリと eBGP ピアリングでは、サポートされていません。

iBGP マルチパス ロード シェアリング

ローカル ポリシーが設定されていないボーダー ゲートウェイ プロトコル（BGP）対応ルータが複数のネットワーク層到達可能性情報（NLRI）を同じ宛先の内部 BGP（iBGP）から受信すると、このルータは 1 つの iBGP パスを最適パスとして選択します。この最適パスは、次にこのルータの IP ルーティング テーブルに組み込まれます。

iBGP のマルチパス ロード シェアリング機能を使用すると、BGP 対応ルータでは、複数の iBGP パスを宛先への最適パスとして選択できます。この最適パスまたはマルチパスは、次にこのルータの IP ルーティング テーブルに組み込まれます。

eBGP から取得した到達可能性情報を持つ複数の境界 BGP ルータがあり、ローカル ポリシーが適用されていない場合、境界ルータでは、eBGPパスを最適パスとして選択します。境界ルータでは、この最適パスをISPネットワークの内部にアドバタイズします。コアルータの場合、同じ宛先に対し複数のパスがある場合がありますが、1つのパスのみを最適パスとして選択し、そのパスを転送用に使用します。iBGPマルチパス ロードシェアリングでは、複数の等距離パス間でロード シェアリングを可能にする機能が追加されます。

複数の iBGP の最適パスを設定すると、ルータでは、特定のサイトを宛先とするトラフィックを均等に負担できるようになります。

iBGP のマルチパス ロード シェアリング機能は、サービス プロバイダー バックボーンを持つマルチプロトコルラベルスイッチング（MPLS）バーチャルプライベートネットワーク（VPN）と同様に機能します。

同じ宛先への複数のパスをマルチパスと見なすには、次の基準を満たす必要があります。

- すべての属性が同じである必要があります。属性には、重み、ローカルプリファレンス、自律システム パス（長さだけでなく属性全体）、発信元コード、Multi Exit Discriminator（MED）、および Interior Gateway Protocol（IGP）距離が含まれます。
- 各マルチパスのネクスト ホップ ルータが異なっている必要があります。

基準を満たしていて、複数のパスがマルチパスと見なされても、BGP対応ルータは、引き続きマルチパスの 1 つをベスト パスに指定し、このベスト パスをそのネイバーにアドバタイズします。

（注） マルチパスの変更後、eiBGPマルチパス候補の評価中にIGPメトリックは考慮されず、また、最適でないパスを使用できます。

内部および外部の BGP マルチパスが設定されている Carrier Supporting Carrier（CSC）ネットワークでは、VRF 単位のラベルモードはサポートされていません。

VRF単位のラベルモードは、ループを引き起こす可能性があるため、eiBGPマルチパスがあるBGPPICエッジには使用できません。プレフィックス単位のラベルのみが、VRF単位のラベルモードをサポートしています。

BGP 選択的マルチパス

従来の BGP マルチパス機能を使用すると、同じ宛先への並列パスを受信するルータは、ルーティングテーブルに複数のパスをインストールできます。デフォルトでは、このマルチパス機能は設定されているすべてのピアに適用されます。BGP選択的マルチパスでは、選択したピアのみにマルチパス機能を適用できます。

複数のパスを受信する BGP ルータは、**maximum-paths ... selective** オプションを使用して設定されます。複数のパスを共有する iBGP/eBGP ネイバーは、**multipath** オプションを使用して設定され、BGP ルータ上にネイバーとして追加されます。

（注） マルチパスをアドバタイズする前にマルチホップ情報を上書きしないようにするには、**next-hop-unchanged multipath** コマンドを使用します。

BGP 選択的マルチパスの使用時には、次の動作に注意してください。

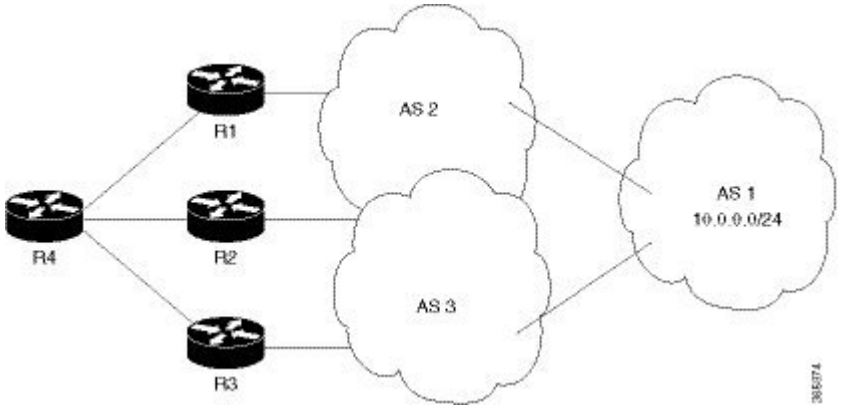
- BGP選択的マルチパスは、ベストパスの計算には影響しません。ベストパスは、マルチパスのセットに常に含まれています。
- VPN プレフィックスの場合、PE パスは「常に」マルチパスの対象となります。

maximum-paths コマンドと **multipath** コマンドについては、『*Cisco ASR 9000 Series Aggregation Services Router Routing Command Reference*』を参照してください。

トポロジ

次の図に、この項で使用する設定を図示したトポロジの例を示します。

図 *10 : BGP* 選択的マルチパス



ルータ R4 は、ルータ R1、R2、および R3 から同じ宛先への並列パスを受信します。ルータ R1 と R2 がルータ R4 上の選択的マルチパスネイバーとして設定されている場合、これらのルータからの並列パスだけがルータ R4 のルーティングテーブルにインストールされます。

コンフィギュレーション

（注） この機能を設定する前に、ルータ上で実行されている iBGP/eBGP を使用してネットワークトポロジを設定します。

ルータ R4 上に BGP 選択的マルチパスを設定するには、次の手順を実行します。

1. トポロジ内の選択した複数のパスを受け入れるようにルータ R4 を設定します。

```
/* To configure selective multipath for iBGP/eBGP
```

```
RP/0/RSP0/cpu 0: router(config)# router bgp 1
RP/0/RSP0/cpu 0: router(config-bgp)# address-family ipv4 unicast
RP/0/RSP0/cpu 0: router(config-bgp-af)# maximum-paths ibgp 4 selective
RP/0/RSP0/cpu 0: router(config-bgp-af)# maximum-paths ebgp 5 selective
RP/0/RSP0/cpu 0: router(config-bgp-af)# commit

/* To configure selective multipath for eiBGP
RP/0/RSP0/cpu 0: router(config)# router bgp 1
RP/0/RSP0/cpu 0: router(config-bgp)# address-family ipv4 unicast
RP/0/RSP0/cpu 0: router(config-bgp-af)# maximum-paths eibgp 6 selective
RP/0/RSP0/cpu 0: router(config-bgp-af)# commit
```

2. ルータ R4 のネイバーを設定します。

ルータ R1（1.1.1.1）および R2（2.2.2.2）は、**multipath** オプションを使用してネイバーとして設定されます。

ルータ R3（3.3.3.3）は **multipath** オプションを使用せずにネイバーとして設定されているため、このルータからのルートをマルチパスとして選択することはできません。

```
RP/0/RSP0/cpu 0: router(config-bgp)# neighbor 1.1.1.1
RP/0/RSP0/cpu 0: router(config-bgp-nbr)# address-family ipv4 unicast
RP/0/RSP0/cpu 0: router(config-bgp-nbr-af)# multipath
RP/0/RSP0/cpu 0: router(config-bgp-nbr-af)# commit

RP/0/RSP0/cpu 0: router(config-bgp-nbr)# neighbor 2.2.2.2
RP/0/RSP0/cpu 0: router(config-bgp-nbr)# address-family ipv4 unicast
RP/0/RSP0/cpu 0: router(config-bgp-nbr-af)# multipath
RP/0/RSP0/cpu 0: router(config-bgp-nbr-af)# commit

RP/0/RSP0/cpu 0: router(config-bgp-nbr)# neighbor 3.3.3.3
RP/0/RSP0/cpu 0: router(config-bgp-nbr)# address-family ipv4 unicast
RP/0/RSP0/cpu 0: router(config-bgp-nbr-af)# commit
```

BGP 選択的マルチパス機能が正常に設定されました。

累積内部ゲートウェイ プロトコル属性

累積内部ゲートウェイ プロトコル（AiGP）属性は、オプションで非推移的な BGP パス属性です。AiGP 属性の属性タイプ コードは、IANA によって割り当てられます。AiGP 属性の値フィールドは、タイプ、長さ、値（TLV）の要素として定義されます。AiGP TLV には、累積 IGP メトリックが含まれます。

AiGP 機能は 3107 ネットワークに必要であり、パスに関連付けられた距離を計算する現在の OSPF の動作をシミュレートします。OSPF/LDP では、プレフィックスおよびラベル情報をローカル領域だけに入れて伝送します。次に、BGP では、エリア境界にある BGP にルートを再配布することにより、すべてのリモートエリアにプレフィックスおよびラベルを伝送します。次に、ルートおよびラベルが、LSP を使用してアドバタイズされます。ルートのネクストホップはローカル ルータに対する各 ABR で変更されます。これによって、エリア境界を越えて OSPF ルートをリークする必要がなくなります。各コア リンクで使用可能な帯域幅が OSPF コストにマップされます。したがって、BGP では、各 PE 間でこのコストを正しく伝送する必要があります。この機能は、AiGP を使用して実現されています。

IPv6 プロバイダー エッジの VRF ごとおよび CE ごとのラベル

IPv6 のための VRF ごとおよび CE ごとのラベルの機能により、デフォルト VRF ごとまたは CE ネクスト ホップごとにラベルを割り当てることにより、ラベル スペースを節約できるようになります。

デフォルトでは、すべての IPv6 プロバイダー エッジ（6PE）ラベルは、プレフィックスごとに割り当てられます。VRF インスタンスに属する各プレフィックスは1つのラベルを使ってアドバタイズされます。これは、パケットのカスタマー エッジ（CE）ネクスト ホップを決定するために、VRF フォワーディング テーブルでさらにルックアップが行われる原因になります。

ただし、**per-ce** キーワードまたは **per-vrf** キーワードを指定して **label mode** コマンドを使用すると、PE ルータ上での追加のルックアップが回避され、ラベル スペースが節約されます。

一意のカスタマー エッジ（CE）ピア ルータからアドバタイズされたすべてのルートで同じラベルを使用するように指定するには、**per-ce** キーワードを使用します。一意の VRF からアドバタイズされたすべてのルートで同じラベルを使用するように指定するには、**per-vrf** キーワードを使用します。

Cisco ASR 9000 の A9K-SIP-700 での IPv4 BGP ポリシー アカウンティング

ボーダー ゲートウェイ プロトコル（BGP）ポリシー アカウンティングは、異なるピア間で送受信される IP トラフィックを測定および分類します。ポリシーアカウンティングは個々の入力または出力インターフェイス単位で有効になります。IP トラフィックを識別するために、コミュニティ リスト、自律システム番号、または自律システム パスなどのパラメータに基づくカウンタが割り当てられます。

BGP ポリシー アカウンティングを使用して、通過するルートに基づいてトラフィックのアカウンティングを行うことができます。サービスプロバイダーは、すべてのトラフィックをカスタマー別に識別してアカウンティングを実施し、それに応じて課金できます。

BGP ポリシーアカウンティングと、BGP ポリシーアカウンティングの設定方法については、『*Cisco ASR 9000 Series Aggregation Services Router IP Addresses and Services Configuration Guide*』の「*Implementing Cisco Express Forwarding*」のモジュールを参照してください。

Cisco ASR 9000 の A9K-SIP-700 での IPv6 ユニキャスト ルーティング

Cisco ASR 9000 の A9K-SIP-700 には、すべてのインターネット プロトコル バージョン 6（IPv6）ユニキャスト機能が備わっています。

IPv6 ユニキャスト アドレスは、単一ノード上の単一インターフェイスの識別子です。ユニキャスト アドレスに送信されたパケットは、そのアドレスが示すインターフェイスに配信されます。Cisco IOS XR ソフトウェアでは、次の IPv6 ユニキャスト アドレス タイプがサポートされます。

- 集約可能グローバル アドレス
- サイトローカル アドレス

- リンクローカル アドレス
- IPv4 互換 IPv6 アドレス

IPv6 ユニキャスト アドレッシングの詳細については、『*Cisco ASR 9000 Series Aggregation Services Router IP Addresses and Services Configuration Guide*』の「*Implementing Network Stack IPv4 and IPv6*」モジュールを参照してください。

Cisco ASR 9000 の A9K-SIP-700 での IPv6 uRPF サポート

ユニキャスト IPv6 リバース パス転送（uRPF）は、検証可能な IP 送信元アドレスを欠いている IP パケットを廃棄することにより、不正な形式の IP 送信元アドレスまたはスプーフィングされた IP 送信元アドレスがネットワークに侵入した場合に生じる問題を軽減します。ユニキャスト RPF はシスコ エクスプレス フォワーディング（CEF）テーブルで逆ルックアップを実行することで、この処理を行います。このため、uRPF が可能になるのは、ルータで CEF が有効になっている場合だけです。

IPV6 uRPF を有効にするには、インターフェイス コンフィギュレーション モードで **ipv6 verify unicast source reachable-via {any | rx} [allow-default] [allow-self-ping]** コマンドを使用します。

IPv6 uRPF の詳細については、 *IP Addresses and Services Command Reference for Cisco ASR 9000 Series Routers*の「*Implementing Cisco Express Forwarding*」のモジュールを参照してください。

BGP の AS パスからのプライベート AS 番号の削除および置換

プライベート自律システム番号（ASN）は、グローバルに一意な AS 番号を保護するために、インターネット サービス プロバイダー（ISP）およびお客様のネットワークで使用されます。プライベート AS 番号は一意でないため、グローバル インターネットへのアクセスには使用できません。AS 番号はルーティング アップデートの eBGP AS パスに表示されます。プライベート ASN を使用している場合にグローバル インターネットにアクセスするには、AS パスからプライベート ASN を削除する必要があります。

パブリックな AS 番号は、InterNIC によって割り当てられ、グローバルに一意です。範囲は 1 ～ 64511 です。プライベート AS 番号は、グローバルに一意な AS 番号（有効な範囲は 64512 ～ 65535）を保護するために使用されます。プライベート AS 番号はグローバル BGP ルーティング テーブルにリークできません。プライベート AS 番号は一意ではなく、BGP 最適パスの計算には一意の AS 番号が必要であるからです。そのため、ルートが BGP ピアに伝播される前に、AS パスからプライベート AS 番号を削除する必要がある可能性があります。

外部 BGP（eBGP）では、グローバルなインターネットへのルーティングで、グローバルに一意な AS 番号を使用する必要があります。プライベート AS 番号（これは一意でない）を使用すると、グローバルなインターネットにアクセスできません。BGP の AS パスからプライベート ASN を削除および交換する機能によって、プライベート AS に属するルータがグローバルなインターネットにアクセスできるようになりました。ネットワーク管理者は、発信アップデート メッセージに含まれる AS パスからプライベート AS を削除するようにルータを設定します。場合によっては、これらの番号をローカル ルータの ASN で置き換えて、AS パス長が変化しないようにします。

AS パスからプライベート ASN を削除および交換する機能は、次のように拡張されました。

- **remove-private-as** コマンドでは、次の処理が行われます。
  - AS パスにパブリックとプライベートの両方の ASN が含まれる場合も、AS パスからプライベート AS 番号を削除します。
  - AS パスにプライベート AS 番号のみが含まれる場合も、プライベート AS 番号を削除します。このコマンドは eBGP ピアのみに適用され、その場合、eBGP ピアではローカル ルータの AS 番号が AS パスに付加されるため、長さ 0 の AS パスにはなることはありません。
  - AS パスでコンフェデレーションセグメントの前にプライベート ASN が出現する場合でも、プライベート AS 番号を削除します。
- **replace-as** コマンドは、パスから削除されるプライベート AS 番号をローカル AS 番号に置き換えることで、AS パスを同じ長さに保ちます。

この機能は、アドレス ファミリ コンフィギュレーション モードでネイバーに適用できます。そのため、アドレスファミリ内のネイバーにこの機能を適用すると、アウトバウンドの更新メッセージのみが影響を受けます。

プライベート AS 番号が削除または置換されたことを確認するには、**show bgp neighbors** コマンドおよび **show bgp update-group** コマンドを使用します。

選択的 VRF ダウンロード

選択的 VRF ダウンロード（SVD）機能を使用すると、ラインカード経由でのトラフィックの転送に必要なラインカードに、これらのプレフィックスおよびラベルだけをダウンロードできるようになります。

統合エッジ MSE プラットフォームにおける要件を満たすために、VRF の数、VRF インターフェイスの数、およびプレフィックス容量が増大しています。コンバージェンスのタイミングは、ラインカードのエンジンによって異なります。コンバージェンスのタイミングを決定する重要な要因の 1 つが、プレフィックスとそれに関連付けられたデータ構造を操作およびプログラムするのにかかる時間です。プレフィックスとラベルの数が少ないほど、コンバージェンスのタイミングが向上します。VRF ルートの選択的ダウンロードを有効にすると、SVD ではレイヤ 3 VPN（L3VPN）のスケーラビリティが高くなり、コンバージェンスの問題が緩和されます。

選択的 VRF ダウンロードでのラインカードのロールとフィルタ

選択的 VRF ダウンロード（SVD）コンテキストでは、ラインカードに次のロールがあります。

- コア LC：コアに接するインターフェイス（他の P/PE に接続するインターフェイス）のみを持つラインカード
- カスタマー LC：カスタマーに接するインターフェイス（異なる VRF の CE に接続するインターフェイス）を 1 つ以上持つラインカード

ラインカードでは、次のプレフィックスを処理します。

- ローカル プレフィックス：設定された VRF コンテキスト内のルータに接続されている CE から受信するプレフィックス
- リモート プレフィックス：別の PE から受信され、設定されている VRF にインポートされたプレフィックス

これらのフィルタは、ラインカード タイプごとに適用できます。

- ラベルや IP フォワーディングを正しく設定できるように、コア LC には、すべてのローカル プレフィックスおよび VRF ラベルが必要です。
- カスタマー LC には、接続されているすべての VRF と、接続されている VRF に依存関係がある他の VRF に対するローカルおよびリモートのプレフィックスが必要です。これはインポートおよびエクスポートの RT コンフィギュレーションに基づきます。VRF「A」に VRF「B」からインポートされたルートがある場合、VRF「A」のインポートされたルートは、VRF「B」にあるネクスト ホップを指します。ルート解決のためには、VRF「A」インターフェイスを持つ各ラインカードに VRF「B」ルートをダウンロードする必要があります。
- ラインカードにコアに接するインターフェイスとカスタマーに接するインターフェイスの両方がある場合は、フィルタリングを実行する必要はありません。このようなラインカードには、すべてのテーブルとすべてのルートがあります。これらのラインカードには「標準」というロールがあります。すべての RP および DRP は、標準ロールがあります。
- L3VPN のルートを正しく解決するために、すべてのノードに IPv4 のデフォルト テーブルがある必要があります。ただし、ラインカードに IPv6 インターフェイスがない場合は、すべての IPv6 テーブルとルートをフィルタで除外できます。このような場合、このラインカードは IPv6 AFI に「関係していない」と見なすことができます。この後、このラインカードは IPv6 をサポートしていないように動作します。

選択的 VRF ダウンロードの無効化

デフォルトでは、選択的 VRF ダウンロード（SVD）機能は無効になっています。SVD を有効にするには、**svd platform enable** コマンドを管理コンフィギュレーションモードで設定し、**reload location all** コマンドを使用してシャーシをリロードします。すでに有効になっている SVD を無効にするには、**no svd platform enable** コマンドを使用し、**reload location all** コマンドでシャーシをリロードします。

SVD の使用または不使用によるラインカードにダウンロードされたルートの計算

選択的 VRF ダウンロードオプションを使用したか、または使用しなかった場合にラインカードにダウンロードされるルートの数は、次に示すラインカードタイプ別にダウンロードされたテーブルとルートの総数に従って計算できます。

次の表に、各 SVD カードタイプのラインカードにダウンロードされたルートとテーブルの総数をまとめます。SVD なしの行の数値の差異によって、節減数を計算できます。

表 *2:* ラインカードタイプ別にダウンロードされたテーブルとルートの総数

カード タイプ	ダウンロードされたテーブル	ダウンロードされたルート
カスタマー	(o+Y)	(o+Y)R
コア	n	nxR
SVD なし	n	nR

- n は、存在する VRF の合計数です。
- o は、カード上で直接プロビジョニング/設定された VRF の数です（n は o 以上）。
- R は VRF ごとのルートの数です。
- x は、SVD ローカルとルート総数の比率です。
- Y は、直接プロビジョニングされた VRF（o）に依存する VRF の数です（Y は 0 以上）。

次に、計算の例を示します。

顧客はシステムに 100 の VRF を設定していて、ラインカードは 5 枚使用しています。IPv4 アドレスファミリの場合、4 枚のラインカードが同等の VRF 分布でカスタマー向けに動作していますが、1 枚はコア向けです。テーブル間の依存関係は存在しません。この例では、n = 100、o = 25、x = 3/10、Y = 0、R = 1000 となっています。

ダウンロードされたルートの数は次のとおりです。

- SVD なし：(nR) = 100,000
- カスタマー向けカード：(o + Y) R = 25,000
- コア向けカード：(nxR) = 30,000

この例では、SVD 機能によって 70 %近く削減されています。

存在する VRF の総数（n）は、RSP カード上で **show cef tables summary location** *node-id* コマンドを使用して検出できます。

```
RP/0/RSP0/cpu 0: router#show cef tables summary location 0/rsp0/cpu0

Role change timestamp     : Apr  3 07:21:46.759
Current Role              : Core
No. of times Eod received : 2
Eod received              : Apr  3 07:21:46.980

No. of Tables                 :       106
No. of Converged Tables       :       106
No. of Deleted Tables         :         0
No. of Bcdl Subscribed Tables :       106
No. of Marked Tables          :         0
```

ラインカード上でプロビジョニングされている VRF の数（o）は、**show cef tables summary location 0/0/cpu0**の「No. Of Tables」フィールドから導出されます。これにより、ラインカード 0/0/cpu0 に固有のテーブルが提供されます。

VRF あたりのルート（R）は、**show cef tables location** *node-id* コマンドを使用して検出できます。

```
RP/0/RSP0/cpu 0: router#show cef tables location 0/1/CPU0
Sat Apr  6 01:22:32.471 UTC

Codes:   L - SVD Local Routes, R - SVD Remote Routes
         T - Total Routes
         C - Table Converged,  D - Table Deleted
         M - Table Marked,     S - Table Subscribed

Table           Table ID             L          R           T C D M S
default         0xe0000000           9          3          23 Y N N Y
**nVSatellite   0xe0000010           1          0           6 Y N N Y
cdn             0xe0000011           0          0           5 Y N N Y
oir             0xe0000012           0          0           5 Y N N Y
vrf1            0xe0000013           3          1          11 Y N N Y
```

VRF 「vrf1」の場合、合計ルートは「T」列 (11) になります。そのため、VRF ごとのルートの数がすべての VRF と同じでなかった場合は、「デフォルト以外の VRF のルート」の総数を計算し、VRF の数で割って、VRF ごとの平均ルート数に達するようにする必要があります。

SVD ローカルの比率：ルートの総数（x）は、SVD ローカルルートの数と特定の VRF のルートの総数を使用して検出できます。たとえば、前出の **show cef tables location 0/1/CPU0**の出力例では、L 列の数字はローカルルートの数、T 列の数字はその VRF のルートの総数を表しています。したがって、L 列と T 列の数値の比率によって、特定の VRF の比率が得られます。この比率がすべての VRF で同じでない場合は、すべての VRF で平均化する必要があります。

直接プロビジョニングされた VRF（Y）に依存する VRF の数は、ルータの設定によって異なるため、手動で計算する必要があります。たとえば、ルートインポートが、他の VRF によってエクスポートされたルートから依存型の VRF インポートをターゲットにしている場合などです。VRF は、直接プロビジョニングされる他の何らかの VRF に存在するネクストホップに依存している場合に、依存型になります。Y を自動的に計算する show コマンドはありません。これは、さまざまな VRF にルートをインポートするためのルートの設定方法に完全に依存しているためです。

BGP Accept Own

BGP Accept Own 機能を使用すると、自動送信 VPN ルート（BGP スピーカーがルートリフレクタ（RR）から受信するルート）を処理できるようになります。「自動送信」ルートは、スピーカー自体によって最初にアドバタイズされたルートです。BGP プロトコル（RFC4271）に従って、BGP スピーカーは、スピーカー自体によって送信されたアドバタイズメントを拒否します。ただし、BGP Accept Own メカニズムを使用すると、プレフィックスの特定の属性を変更するルートリフレクタから反映された場合に、ルータは自身がアドバタイズしたプレフィックスを受け入れることが可能になります。ACCEPT-OWN と呼ばれる特別なコミュニティがルートリフレクタによってプレフィックスに付加されます。これは ORIGINATOR_ID および

NEXTHOP/MP_REACH_NLRI チェックをバイパスするための受信側ルータに対する信号です。通常、BGP スピーカーは自動送信されたプレフィックスを自動送信チェック（ORIGINATOR_ID、NEXTHOP/MP_REACH_NLRI）によって検出し、受信した更新をドロップします。ただし、更新に Accept Own コミュニティがあれば、BGP スピーカーはそのルートを処理します。

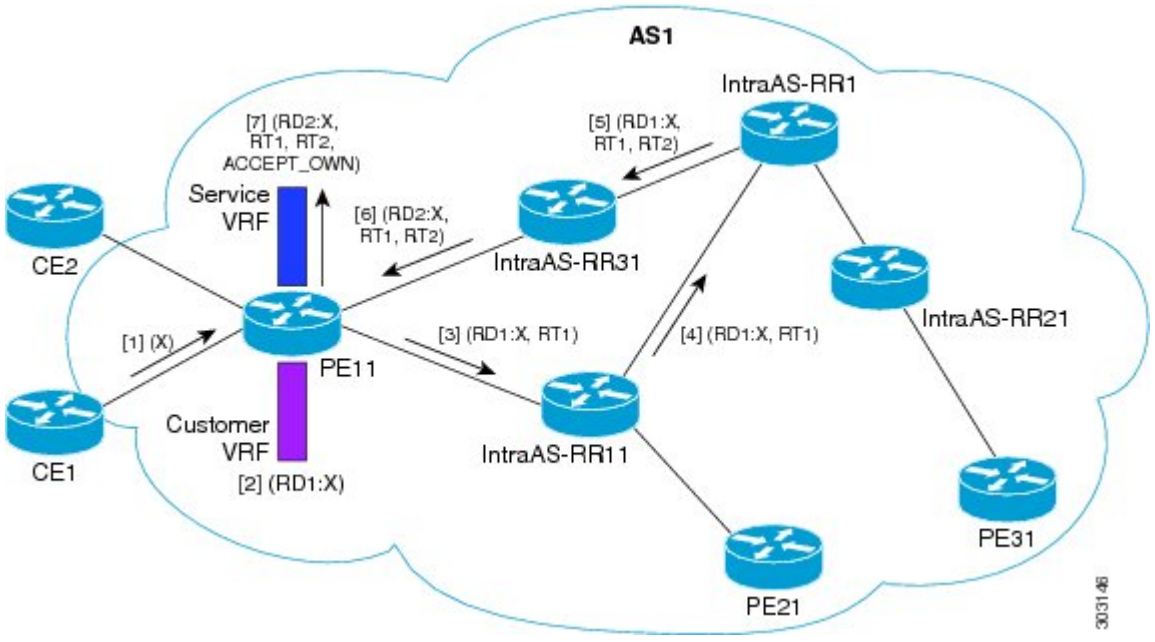
BGP Accept Own の応用例の 1 つは、MPLS VPN ネットワーク内のエクストラネットの自動設定です。エクストラネットの設定では、ある VRF にあるルートは同じ PE の別の VRF にインポートされます。通常、エクストラネットのメカニズムでは、別の VRF からのプレフィックスのインポートを制御するために、エクストラネット VRF のインポート RT またはインポートポリシーを編集する必要があります。ただし、Accept Own 機能を使用すると、ルート リフレクタは、PE で設定変更することなく、その制御をアサートできます。このように Accept Own 機能によって、異なる VRF 間でのルートのインポートの制御を集中管理できます。

BGP Accept Own 機能は、ネイバー コンフィギュレーション モードの VPNv4 および VPNv6 アドレス ファミリ向けにのみサポートされています。

Accept Own コミュニティと RT を処理するルートリフレクタ

ACCEPT_OWN コミュニティは、InterAS ルートリフレクタ（InterAS-RR）によってアウトバウンド ルート ポリシーを使用して発信されます。ACCEPT_OWN のコミュニティ属性を持つプレフィックスの伝搬を最小限に抑えるために、この属性は送信元 PE に対するアウトバウンド ルート ポリシーを使用して InterAS-RR に付加されます。InterAS-RR は、ACCEPT-OWN コミュニティを追加して RT を変更した後、仲介 RR を通じて新しい Accept Own ルートを、接続されている PE（送信元など）に送信します。ルートは、ルート ポリシーによって変更されます。

Accept Own の設定例



この設定例の内容は次のとおりです。

- PE11 にカスタマー VRF とサービス VRF が設定されています。
- OSPF は IGP として使用されます。
- VPNv4 ユニキャストおよび VPNv6 ユニキャストのアドレス ファミリが PE ネイバーと RR ネイバーとの間でイネーブルになっており、IPv4 および IPv6 が PE ネイバーと CE ネイバーとの間でイネーブルになっています。

Accept Own の設定は次のように動作します。

1. CE1 がプレフィックス X を発信します。
2. プレフィックス X は、カスタマー VRF に (RD1:X) として設定されています。
3. プレフィックス X は IntraAS-RR11 に (RD1:X, RT1) としてアドバタイズされます。
4. IntraAS-RR11 が InterAS-RR1 に X を (RD1:X, RT1) としてアドバタイズします。
5. InterAS-RR1 はインバウンドのプレフィックス X とアウトバウンドの ACCEPT_OWN コミュニティに RT2 を付加し、IntraAS-RR31 にプレフィックス X をアドバタイズします。
6. IntraAS-RR31 が PE11 に X をアドバタイズします。
7. PE11 は X をサービス VRF に (RD2:X,RT1, RT2, ACCEPT_OWN) としてインストールします。

リモート PE：Accept Own ルートの処理

リモート PE（送信元 PE 以外の PE）は、すべての同等ルート間の最適パスを計算します。この最適パス アルゴリズムは、Accept Own パスが Accept Own でないパスよりも優先されるように変更されています。最適パスの比較は IGP メトリックの比較の直前に実行されます。リモート PE がルートリフレクタ 1 から Accept Own パスを受信し、ルートリフレクタ 2 から Accept Own でないパスを受信し、これらのパスが同一であった場合は、Accept Own パスが優先されます。そのためインポートは Accept Own パスで実行されます。

不等コストの連続ロードバランシングに対する BGP DMZ リンク帯域幅

不等コストの連続ロードバランシングに対するボーダー ゲートウェイ プロトコル非武装地帯（BGP DMZ）リンク帯域幅により、BGP DMZ リンク帯域幅を使用して、ローカル ノード上で連続プレフィックスに対する不等コスト ロードバランシングをサポートできます。不均等ロードバランシングは、BGP ネイバー コンフィギュレーション モードの **dmz-link-bandwidth** コマンドと、インターフェイス コンフィギュレーション モードの **bandwidth** コマンドを使用して実行します。

BGP の BFD マルチホップ サポート

BGP では、双方向フォワーディング検出マルチホップ（BFD-MH）のサポートが有効になっています。BFD マルチホップでは複数のネットワーク ホップにまたがることのある 2 つのアドレス間に BFD セッションを確立します。Cisco IOS XR ソフトウェア BFD マルチホップは RFC

5883 に基づきます。BFD マルチホップの詳細については、 *Interface and Hardware Component Configuration Guide for Cisco ASR 9000 Series Routers*および *Interface and Hardware Component Command Reference for Cisco ASR 9000 Series Routers*を参照してください。

BGP Multi-Instance および Multi-AS

自律システム（AS）に対応するルータでは、複数の BGP インスタンスがサポートされています。各 BGP インスタンスは、同じまたは異なる RP/DRP ノードで実行される独立したプロセスです。BGP インスタンス間ではプレフィックス テーブルは共有されません。分散 BGP と同様に、共通 adj-rib-in（bRIB）は不要です。BGP インスタンスは互いに通信することはなく、また互いにピアリングを設定することもありません。個々のインスタンスは他のルータとのピアリングを独立して設定できます。

Multi-AS BGP を使用すると、Multi-Instance BGP の各インスタンスに異なる AS 番号を設定できるようになります。

Multi-Instance および Multi-AS BGP は次の機能を備えています。

- 共通ルーティング インフラストラクチャを使用して、複数のルータによって提供されるサービスを単一の IOS-XR ルータに統合するメカニズム。
- 異なる BGP インスタンスに異なる AF を設定することにより、AF の分離を実現するメカニズム。
- 複数のインスタンス間でピアリング セッション全体を分散させることによって、セッションのスケールを高めることができる手段。
- 個々のインスタンスに異なる BGP テーブルを伝送させることにより、プレフィックスのスケール（特に RR で）を高めることができるメカニズム。
- 特定の状況における BGP コンバージェンスの改善。
- NSR を含むすべての BGP 機能は、すべてのインスタンスに対応しています。
- ロードおよびコミット ルータ レベルの操作は、以前に確認または適用された構成上で実行できます。

制約事項

- ルータは最大 4 つの BGP インスタンスをサポートします。
- 各 BGP インスタンスには、固有の ルータ ID が必要です。
- 各 BGP インスタンスで設定できるアドレス ファミリは 1 つだけです（VPNv4、VPNv6 および RT 制約は複数の BGP インスタンスで設定できます）。
- IPv4/IPv6 ユニキャストは、IPv4/IPv6 ラベル付きユニキャストが設定されている同じ BGP インスタンス内にある必要があります。
- IPv4/IPv6 マルチキャストは、IPv4/IPv6 ユニキャストが設定されている同じ BGP インスタンス内にある必要があります。

- 単一の BGP インスタンスに対するすべての設定変更を同時にコミットすることができます。ただし、複数のインスタンスに対する設定変更は同時にコミットできません。
- 同じリモート ルータとのピアリング時に、BGP の update-source をすべてのインスタンスのデフォルト VRF で一意にすることが推奨されます。

RPKI に基づく BGP プレフィックスの発信元検証

BGP ルートは、BGP アナウンスメントの形で、プレフィックスが経由したドメイン間パスを識別する自律システム（AS）の設定と、アドレス プレフィックスを関連付けます。この設定は、BGP 内で AS_PATH 属性として表され、プレフィックスを発信した AS で開始されます。

誤ったプレフィックスのアナウンス、中間者攻撃など、BGP に対する既知の脅威を低減しやすくするためのセキュリティ要件の 1 つは、BGP ルートの発信元 AS を検証する能力です。アドレス プレフィックスの発信元であるとする AS 番号（BGP ルートの AS_PATH 属性から導出）は、プレフィックスの所有者によって検証および許可される必要があります。

Resource Public Key Infrastructure（RPKI）は、IP アドレスとリソースとしての AS 番号の公的で検証可能なデータベースを構築するためのアプローチです。RPKI は、BGP（インターネット）プレフィックスから許可された元の AS 番号への情報マッピングなどの情報を含む、グローバルに分散されたデータベースです。BGP を実行しているルータは、RPKI に接続して、BGP パスの元の AS を検証できます。

BGP RPKI の送信元バインド機能を使用すると、RPKI サーバ接続に使用する送信元の IP アドレスとインターフェイスを指定できます。たとえば、この機能では、ループバック インターフェイスから送信元となる RPKI セッションを設定できます。

BGP の origin-as 検証はデフォルトで有効になっています。

RPKI キャッシュ サーバの設定

リソース公開キー インフラストラクチャ（RPKI）キャッシュ サーバ パラメータを設定するには、次の作業を実行します。

RPKI サーバのコンフィギュレーション モードで RPKI キャッシュ サーバ パラメータを設定します。RPKI サーバ コンフィギュレーション モードを開始するには、ルータ BGP コンフィギュレーション モードで **rpki server** コマンドを使用します。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **rpki server** {*host-name* | *ip-address*}
4. **bind-source interface** *name*
5. 次のいずれかのコマンドを使用します。
   - **transport ssh port** *port_number*
   - **transport tcp port** *port_number*
6. （任意） **username** *user_name*

7. （任意） **password** *password*
8. **preference** *preference_value*
9. **purge-time** *time*
10. 次のいずれかのコマンドを使用します。
    - **refresh-time** *time*
    - **refresh-time off**
11. 次のいずれかのコマンドを使用します。
    - **response-time** *time*
    - **response-time off**
12. **commit**
13. （任意） **shutdown**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)#router bgp 100`	BGP AS 番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**rpki server** {*host-name* | *ip-address*} 例： `RP/0/RSP0/cpu 0: router(config-bgp)#rpki server 10.2.3.4`	RPKI サーバのコンフィギュレーション モードを開始し、RPKI のキャッシュ パラメータを設定します。
ステップ 4	**bind-source interface** *name* 例： `Router#(config-bgp)# bind-source interface Loopback2`	RPKI サーバ接続に使用する送信元インターフェイスとしてループバック インターフェイスを指定します。

ステップ 5	次のいずれかのコマンドを使用します。
	• **transport ssh port** *port_number*
	• **transport tcp port** *port_number*
	例：
	`RP/0/RSP0/cpu 0: router(config-bgp-rpki-server)#transport ssh port 22`
	または
	`RP/0/RSP0/cpu 0: router(config-bgp-rpki-server)#transport tcp port 2`
		RPKI キャッシュの転送方法を指定します。
		• **ssh**：SSH を使用して RPKI キャッシュに接続するには **ssh** を選択します。
		• **tcp**：TCP（暗号化されていない）を使用して RPKI キャッシュに接続するには **tcp** を選択します。
		• **port** *port_number*：指定した RPKI キャッシュ転送に使用するポート番号を指定します。TCP の場合、サポートされているポート番号の範囲は 1 ~ 65535 です。SSH の場合は、ポート番号 22 を使用します。

	コマンドまたはアクション	目的
		（注） SSH を介した RPKI キャッシュ転送の場合は、カスタム ポート番号を指定しないでください。SSH を介した RPKI にはポート 22 を使用する必要があります。 （注） transport には TCP と SSH のいずれかを設定できます。transport を変更すると、キャッシュ セッションがフラップします。

ステップ 6	（任意） **username** *user_name*
	例：
	`RP/0/RSP0/cpu 0:`
	`router(config-bgp-rpki-server)#username ssh_rpki_uname`
		RPKI キャッシュ サーバの（SSH）ユーザ名を指定します。

ステップ 7	（任意） **password** *password*
	例：
	`RP/0/RSP0/cpu 0:`
	`router(config-bgp-rpki-server)#password ssh_rpki_pass`
		RPKI キャッシュ サーバの（SSH）パスワードを指定します。
		（注） 「username」と「password」の設定は、SSH 転送方式がアクティブな場合にのみ適用されます。

ステップ 8	**preference** *preference_value*
	例：
	`RP/0/RSP0/cpu 0:`
	`router(config-bgp-rpki-server)#preference 1`
		RPKI キャッシュのプリファレンス値を指定します。プリファレンス値の範囲は 1 ～ 10 です。設定するプリファレンス値は低い方が適切です。

ステップ 9	**purge-time** *time*
	例：
	`RP/0/RSP0/cpu 0:`
	`router(config-bgp-rpki-server)#purge-time 30`
		キャッシュ セッションのドロップ後に、BGP がキャッシュからのルートを保持するまで待機する時間を設定します。破棄時間は秒単位で設定します。破棄時間の範囲は 30 ～ 360 秒です。

ステップ 10	次のいずれかのコマンドを使用します。
	• **refresh-time** *time*
	• **refresh-time off**
	例：
	`RP/0/RSP0/cpu 0:`
	`router(config-bgp-rpki-server)#refresh-time 20`
	または
	`RP/0/RSP0/cpu 0:`
	`router(config-bgp-rpki-server)#refresh-time off`
		キャッシュへの定期的なシリアル クエリー送信操作の間に BGP が待機する時間を設定します。リフレッシュの時間を秒単位で設定します。リフレッシュの時間の範囲は 15 ～ 3600 秒です。
		シリアルクエリーを定期的に送信しないように指定するには、**off** オプションを設定します。

	コマンドまたはアクション	目的

ステップ 11	次のいずれかのコマンドを使用します。
	• **response-time** *time*
	• **response-time off**
	例：
	`RP/0/RSP0/cpu 0:`
	`router(config-bgp-rpki-server)#response-time 30`
	または
	`RP/0/RSP0/cpu 0:`
	`router(config-bgp-rpki-server)#response-time off`
		シリアルまたはリセットのクエリーを送信した後に BGP が応答を待機する時間を設定します。応答時間を秒の単位で設定します。応答時間の範囲は 15 ～ 3600 秒です。
		応答を無期限に待機するには、**off** オプションを設定します。

ステップ 12	**commit**	

ステップ 13	（任意） **shutdown**
	例：
	`RP/0/RSP0/cpu 0:`
	`router(config-bgp-rpki-server)#shutdown`
		RPKIキャッシュのシャットダウンを設定します。

RPKI 最適パス計算の設定

RPKI 最適パス計算オプションを設定するには、次の作業を実行します。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **bgp bestpath origin-as use validity**
4. **bgp bestpath origin-as allow invalid**
5. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)#router bgp 100`	BGP AS 番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGPルーティング プロセスを設定できます。
ステップ 3	**bgp bestpath origin-as use validity** 例： `RP/0/RSP0/cpu 0: router(config-bgp)#bgp bestpath origin-as use validity`	BGPベストパス処理でのパスのプリファレンスに影響するBGPパスの有効性状態を有効にします。この設定は、ルータ BGP アドレス ファミリ サブモードでも設定できます。

	コマンドまたはアクション	目的
ステップ 4	**bgp bestpath origin-as allow invalid** 例： `RP/0/RSP0/cpu 0: router(config-bgp)#bgp bestpath origin-as allow invalid`	すべての「無効な」パスをBGP最適パス計算対象にします。 （注） この設定はグローバルアドレスファミリ、ネイバー、およびネイバー アドレス ファミリの各サブモードでも設定できます。ルータ BGP とアドレスファミリサブモードで bgp bestpath origin-as allow invalid を設定すると、すべての「無効な」パスがBGP最適パス計算対象になります。デフォルトではこのようなパスは最適パス候補になりません。ネイバーまたはネイバー アドレスファミリ サブモードで bgp bestpath origin-as を設定すると、その特定のネイバーまたはネイバーアドレスファミリのすべての「無効な」パスがベストパス候補として見なされます。このネイバーは eBGP ネイバーでなければなりません。 この設定は、**bgp bestpath origin-as use validity** 設定がイネーブルの場合にのみ有効になります。
ステップ 5	**commit**	

グローバル プレフィックス用 BGP 3107 PIC アップデート

グローバル プレフィックス用 BGP 3107 PIC アップデート機能は、MPLS VPN プロバイダー ネットワークでの、グローバル IPv4 および IPv6 プレフィックス用のプレフィックス独立コンバージェンス（PIC）アップデートをサポートします。この機能は、BGP を使用した、グローバルな IPv4 または IPv6 のプレフィックス用の MPLS ラベルの配布について記述した、RFC 3107 に基づいています。これにより、IGP の拡張性が向上され、高速コンバージェンスのための PIC アップデートも実現されます。

RFC 3107 により、ルートおよびラベルを BGP で伝送できるようになります。特定のルートへの配布に BGP を使用する場合は、このルートにマッピングされている MPLS ラベルの配布にも使用できます。特定の 1 つのルートに対するラベルマッピング情報は、ルート自体の配布に使用される、同じ BGP アップデート メッセージに同梱されます。RFC 3107 では、OSPF からネクスト ホップ ループをフィルタリングでき、LDP によってアドバタイズされるラベルを削減できます。この実装によって、OSPF および LDP データベースが大幅に削減されます。

3107 PIC 実装では、additional-path 設定を含めて、次のアドレス ファミリをサポートしています。

- address-family ipv4 unicast
- address-family ipv6 unicast

- address-family vpnv4 unicast
- address-family vpnv6 unicast

（注） address-family l2vpn vpls-vpws では、additional-path をサポートしていません。したがって、address-family l2vpn vpls-vpws を使用する L2VPN サービスでは、PIC コンバージェンス時間が保証されません。

3107 PIC 実装では、次の Cisco IOS XR 機能をサポートしています。

- 3107 の PIC エッジ
- Traffic Engineering Fast-reroute（TE FRR）：逐語的なトンネルを使用して、コア リンク障害に対する 50 ミリ秒以内のトラフィック コンバージェンスが保証されます。
- L2VPN サービス（VPWS）
- L3VPN VPNv4 サービス
- 6 PE サービス
- 6 VPE サービス
- VPLS サービス

グローバル プレフィックス用 BGP 3107 PIC アップデート実装では、Light-Weight Recursive（LW-RLDI）オブジェクトの代わりに共有 Recursive Load Info（RLDI）転送オブジェクトを使用します。RLDI は複数リーフ間で共有されますが、LW-RLDI はリーフごとにインスタンス化されます。処理がプレフィックスに依存しなくなるため、共有は PIC アップデートの処理で有用です。

RIB および FIB 用 BGP プレフィックス独立コンバージェンス

RIB および FIB 用 BGP PIC によって、PE-CE としてのスタティック再帰のサポートおよび Fast Re-Route トリガーを使用したバックアップ アクティベーションの高速化のサポートが加わります。

RIB および FIB 用 BGP PIC 機能では、次の要素をサポートしています。

- コンバージェンス時間をさらに削減する、高速な PE-CE リンク ダウン検出用の FRR に似たトリガー（PIC エッジの高速アクティベーション）。
- スタティック再帰ルートのための PIC エッジ。
- 明示的に /32 スタティック ルートを設定しない、PIC エッジのための BFD シングルホップ トリガー。
- 第 1（IGP）レベルでの失敗トリガーの際の第 3 レベル以降での再帰 PIC アクティベーション。

- BGP ネクスト ホップの解決に関して、FIB が BGP と同期していることを保証する、FIB での BGP パス再帰制約。

BGP PIC エッジが設定されている場合、**neighbor shutdown** コマンドを設定しても、バックアップパスに切り替える CEF はトリガーされません。代わりに、BGP はルーティングテーブルの最上位プレフィックスから最後まで 1 つずつ CEF のフィードを再開するため、時間遅延が発生します。

⚠

注意 この時間遅延によって、ネットワーク内でブラックホールが発生します。回避策として、**neighbor shutdown** コマンドを設定する前に、トラフィックをバックアップパスに手動でルーティングしておく必要があります。

BGP アップデート メッセージのエラー処理

BGP アップデート メッセージのエラー処理によって、セッションのリセットを避けるためにエラー アップデート メッセージの処理における BGP の動作が変わります。IETF IDR *I-D:draft-ietf-idr-error-handling* で説明されているアプローチに基づいて、Cisco IOS XR BGP アップデート メッセージのエラー処理を実装すると、重大度、更新エラーが発生する可能性、属性のタイプなどの要素に基づいて、BGP 更新エラーはさまざまなカテゴリに分類されます。各カテゴリで発生したエラーは、ドラフトに沿って処理されます。セッションのリセットは、エラーの処理プロセス中は可能な限り回避されます。一部のカテゴリのエラー処理は、デフォルトの動作を有効または無効にする設定コマンドによって制御されます。

基本の BGP 仕様に応じて、不正な属性を含むアップデート メッセージを受信した BGP スピーカは、不正な属性が受信されたセッションをリセットする必要があります。セッションのリセットは、不正な属性があるルートだけでなくセッションを介して交換される他の有効なルートにも影響するので、この動作は好ましくありません。

BGP 属性のフィルタリング

BGP 属性フィルタ機能によって、BGP アップデート メッセージ内の BGP アップデートの整合性を確認し、無効な属性を検出したときには最適な対応を行います。BGP アップデート メッセージには、必須およびオプションの属性のリストが含まれています。アップデート メッセージのこれらの属性には、MED、LOCAL_PREF、COMMUNITY などが含まれています。属性の形式が正しくない場合は、ルータの受信側でこれらの属性をフィルタ処理する必要があります。BGP 属性フィルタ機能では、着信アップデート メッセージで受信した属性をフィルタリングします。属性フィルタは、受信側ルータで好ましくない動作を引き起こす可能性のある属性を排除するためにも使用できます。

BGP アップデートの中には、ネットワーク層到達可能性情報（NLRI）またはアップデート メッセージ内の他のフィールドなどの誤った形式の属性のために、形式が不正になるものがあります。これらの不正なアップデートを受信すると、受信側ルータで好ましくない動作が発生します。このような不正な動作は、アップデート メッセージの解析時や、受信した NLRI の再

アドバタイズ時に発生することがあります。このような場合に備えて、受信側でこれらの破損した属性をフィルタ処理することが重要です。

BGP 属性のフィルタリングのアクション

属性フィルタリングを設定するには、1 つまたはある範囲の属性コードと対応するアクションを指定します。実行できるアクションには次のものがあります。

- Treat-as-withdraw：対応する IPv4 ユニキャストまたは MP_REACH NLRI があれば、ネイバーの Adj-RIB-In から取り消します。
- Discard Attribute：一致した部分の属性は廃棄され、アップデート メッセージの残りの部分は正常に処理されます。

受信したアップデート メッセージに 1 つ以上のフィルタされた属性が含まれている場合、メッセージに対して設定されたアクションが実行されます。オプションで、さらに詳細なデバッグを行うためにアップデート メッセージを保存して、コンソールに syslog メッセージを表示することもできます。

属性がフィルタと一致した場合は、属性のその後の処理は停止され、対応するアクションが実行されます。

属性フィルタ グループ コマンド モードを開始するには、**attribute-filter group** コマンドを使用します。属性を破棄または更新メッセージを「取り消し」アクションとして処理するには、属性フィルタ グループ コマンド モードで **attribute** コマンドを使用します。

BGP のエラー処理と属性フィルタリングの syslog メッセージ

不正な形式のアップデート パケットをルータが受信すると、ROUTING-BGP-3-MALFORM_UPDATE タイプの ios_msg がコンソールに出力されます。このレートは、すべてのネイバーで 1 分間に 1 つのメッセージになるよう制限されています。不正なパケットが「Discard Attribute」（A5）または「Local Repair」（A6）アクションの対象になった場合は、ネイバー 1 つおよびアクション 1 つごとに ios_msg メッセージが出力されます。これは、ネイバーが直前の「Established」状態に到達して以降に受信した不正な形式のアップデートの数とは関係ありません。

BGP エラー処理の syslog メッセージの例を次に示します。

```
%ROUTING-BGP-3-MALFORM_UPDATE : Malformed UPDATE message received from neighbor 13.0.3.50
 - message length 90 bytes,
 error flags 0x00000840, action taken "TreatAsWithdraw".
Error details: "Error 0x00000800, Field "Attr-missing", Attribute 1 (Flags 0x00, Length
 0), Data []"
```

これは「Discard Attribute」アクションに対する BGP 属性フィルタリングの syslog メッセージの例です。

```
[4843.46]RP/0/0/CPU0:Aug 21 17:06:17.919 : bgp[1037]: %ROUTING-BGP-5-UPDATE_FILTERED :
One or more attributes were filtered from UPDATE message received from neighbor 40.0.101.1
 - message length 173 bytes,
```

```
 action taken "DiscardAttr".
 Filtering details: "Attribute 16 (Flags 0xc0): Action "DiscardAttr"". NLRIs: [IPv4
Unicast] 88.2.0.0/17
```

これは「Treat-as-withdraw」アクションに対する BGP 属性フィルタリングの syslog メッセージの例です。

```
[391.01]RP/0/0/CPU0:Aug 20 19:41:29.243 : bgp[1037]: %ROUTING-BGP-5-UPDATE_FILTERED :
 One or more attributes were filtered from UPDATE message received from neighbor 40.0.101.1
 - message length 166 bytes,
 action taken "TreatAsWdr".
 Filtering details: "Attribute 4 (Flags 0xc0): Action "TreatAsWdr"". NLRIs: [IPv4 Unicast]
 88.2.0.0/17
```

BGP リンクステート

BGP リンクステート（LS）は、BGP を介して内部ゲートウェイ プロトコル（IGP）リンクステートデータベースを伝えるために定義されたアドレス ファミリ識別子（AFI）およびサブアドレス ファミリ識別子（SAFI）です。BGP LS は、ネットワーク トポロジ情報をトポロジ サーバおよびアプリケーション層トラフィック最適化（ALTO）サーバに提供します。BGP LS では、集約、情報の非表示、および抽象化に対するポリシー ベースの制御が可能です。BGP LS は、IS-IS および OSPFv2 をサポートしています。

（注） IGP は、リモート ピアからの BGP LS データを使用しません。BGP は、ルータの他のコンポーネントに受信した BGP LS データをダウンロードしません。

BGP パーマネント ネットワーク

BGP パーマネント ネットワーク機能は、BGP 経由のスタティック ルーティングをサポートしています。（ルート ポリシーで識別された）IPv4 または IPv6 宛先への BGP ルートは、管理用に作成して、BGP ピアに選択的にアドバタイズできます。これらのルートは、管理上削除されるまでルーティング テーブルに残ります。

パーマネント ネットワークは、プレフィックスのセットを永続的なものとして定義するために使用されます。つまり、プレフィックスのセットのアップストリームにおいて BGP のアドバタイズメントまたは取り消しは 1 回しかありません。プレフィックス セットの各ネットワークに対し、BGP 固定パスが作成され、優先度はそのピアから受信される他の BGP パスよりも低く扱われます。BGP 固定パスが最適パスである場合は RIB にダウンロードされます。

グローバル アドレス ファミリ コンフィギュレーション モードの **permanent-network** コマンドは、ルートポリシーを使用して固定パスが設定されるプレフィックス（ネットワーク）のセットを識別します。ネイバー アドレスファミリ コンフィギュレーション モードの **advertise permanent-network** コマンドは、固定パスをアドバタイズする必要があるピアの識別に使用されます。別の最適パスが使用可能であっても、固定パスは常にアドバタイズ パーマネント ネッ

トワーク設定を持つピアにアドバタイズされます。固定パスは、固定パスを受信するように設定されていないピアにはアドバタイズされません。

パーマネント ネットワーク機能は、デフォルトの仮想ルーティングおよび転送（VRF）下の IPv4 ユニキャストおよび IPv6 ユニキャスト アドレス ファミリ内のプレフィックスのみをサポートします。

制約事項

次の制限は、パーマネント ネットワークの設定時に適用されます。

- パーマネント ネットワーク プレフィックスは、グローバル アドレス ファミリでルート ポリシーによって指定する必要があります。
- グローバル アドレス ファミリ コンフィギュレーション モードでルート ポリシーを使用してパーマネント ネットワークを構成し、それをネイバー アドレス ファミリ コンフィギュレーション モードで設定する必要があります。
- パーマネント ネットワーク設定を削除する場合は、ネイバー アドレス ファミリ コンフィギュレーション モードの設定を削除してから、グローバル アドレス ファミリ コンフィギュレーション モードから削除します。

アップデート生成のための BGP-RIB のフィードバック メカニズム

アップデート生成機能のためのボーダー ゲートウェイ プロトコル ルーティング情報ベース（BGP-RIB）のフィードバック メカニズムによって、ネットワークで不完全なルート アドバタイズメントが行われて、それによってパケット損失が発生するのを防ぐことができます。このメカニズムによって、ルートがネイバーにアドバタイズされる前にローカルに組み込まれるようになります。

BGP は RIB からのフィードバックを待ちます。このフィードバックには、BGP によって RIB に組み込まれたルートが、BGP がネイバーにアップデートを送信する前に転送情報ベース（FIB）に組み込まれたことが示されています。RIB は BCDL のフィードバック メカニズムを使用して、そのバージョンのルートが FIB によって使用されたかを判断し、BGP をそのバージョンで更新します。BGP がアップデートを送信するのは、FIB が組み込んだバージョン以下のバージョンのルートだけです。この選択的な更新によって、BGP が不完全なアップデートを送信しないようになり、ルータのリロード、LC OIR、または代替パスが使用可能になるリンク フラップ後にデータ プレーンがプログラミングされる前であっても、トラフィックの引き込みが行われるようになります。

BGP が RIB に組み込んだルートが FIB に組み込まれたことを示す RIB からのフィードバックを BGP が待機し、その後で BGP がネイバーにアップデートを送信するように設定するには、ルータ アドレスファミリ IPv4 またはルータ アドレスファミリ VPNv4 コンフィギュレーション モードで **update wait-install** コマンドを使用します。**show bgp**、**show bgp neighbors**、および **show bgp process performance-statistics** コマンドを実行すると、update wait-install 設定の情報が表示されます。

BGP VRF ダイナミック ルートのリーク

Border Gateway Protocol（BGP）ダイナミック ルートのリーク機能では、デフォルトの VRF（グローバル VRF）とその他すべての非デフォルト VRF 間にルートをインポートできるようにし、グローバルと VPN ホスト間に接続を提供します。インポート プロセスによって VRF テーブルにインターネット ルートが組み込まれるか、またはインターネット テーブルに VRF ルートが組み込まれて、接続を提供します。

（注）

- 直接接続されたルートは、デフォルトの VRF から非デフォルトの VRF に BGP VRF ダイナミックルートリークを使用してリークできません。
- リークされたルートは、宛先 VRF 内のルートを対象にしたり、上書きしたりすることはできません。たとえば、2 台の接続されたルータ R1 （宛先 VRF 「dest-vrf」）と R2（送信元 VRF 「source-vrf」）を考えてみましょう。ルート CR-1 に接続された source-vrf が dest-vrf にリークされます。この場合、dest-vrf からのルートは、source-vrf からリークされたルート CR-1 の対象となるか、または上書きされます。

ダイナミック ルート リークは次の方法で有効になります。

- VRF アドレスファミリ コンフィギュレーション モードで **import from default-vrf route-policy** *route-policy-name* ［**advertise-as-vpn**］コマンドを使用して、デフォルト VRF から非デフォルト VRF にインポートする。

  advertise-as-vpn オプションが設定されている場合、デフォルト VRF から非デフォルト VRF にインポートしたパスは、PE と CE にアドバタイズされます。**advertise-as-vpn** オプションが設定されていない場合、デフォルト VRF から非デフォルト VRF にインポートされたパスは PE にアドバタイズされません。ただし、この場合も CE にはパスがアドバタイズされます。

- VRF アドレスファミリ コンフィギュレーション モードで **export to default-vrf route-policy** *route-policy-name* コマンドを使用して、非デフォルト VRF からデフォルト VRF にインポートする。

インポートしたルートをフィルタリングするには、ルート ポリシーが必要です。これにより、インターネット テーブルと VRF テーブル間でのルートの意図せぬインポートや対応するセキュリティ問題を低減します。

インポートできるプレフィックスの数にハードリミットはありません。インポートによりインポート先の VRF に新しいプレフィックスが作成されるため、プレフィックスとパスの総数が増加します。ただし、グルーバル ルートをインポートしている各 VRF がグローバル テーブルを受け取るネイバーと同等のワークロードを追加します。これは、ユーザが一部を除くすべてのプレフィックスをフィルタリングした場合も同様です。したがって、インポートする VRF の適切な数は 5 ～ 10 個です。

ユーザ定義の Martian チェック

Cisco IOS XR ソフトウェア リリース 5.1.0 では、IP アドレスプレフィックスが次のものである場合、Martian チェックを無効にできます。

- IPv4 アドレス プレフィックス
  - 0.0.0.0/8
  - 127.0.0.0/8
  - 224.0.0.0/4
- IPv6 アドレス プレフィックス
  - ::
  - ::0002 - ::ffff
  - ::ffff:a.b.c.d
  - fe80:xxxx
  - ffxx:xxxx

復元力のある CE 単位のラベルモード

復元力のある CE 単位のラベルは、CE 単位のラベルモードの拡張機能で、プレフィックス独立コンバージェンス（PIC）とロードバランシングをサポートします。

現時点では、プレフィックスごと、CE ごと、および VRF 単位の 3 つのラベルモードに次の制限があります。

- ASR 9000 イーサネット ライン カードと A9K-SIP-700 に対するサポートなし
- PIC に対するサポートなし
- 複数の CE にわたるロード バランシングに対するサポートなし
- PIC をサポートするローカル トラフィックの迂回時の一時的な転送ループ
- EIBGP マルチパスのロード バランシングに対するサポートなし
- 転送パフォーマンスへの影響
- ネットワーク内の別のベンダーのルータでのプレフィックス単位のラベルモードによるスケールの問題

復元力のある CE 単位のラベルスキームでは、CE パスまたはネクストホップのそれぞれの一意のセットに対して BGP が LSD に一意の書き換えラベルをインストールします。BGP テーブルにこのラベルをポイントする 1 つ以上のプレフィックスが含まれている場合があります。また、BGP は CE パス（プライマリ）と、オプションのバックアップ PE パスを RIB にインストールします。FIB は LSD からラベル書き換え情報を、RIB から IP パスを学習します。

安定状態では、弾力性のある CE ごとのラベル宛のラベル付きのトラフィックには、すべての CE ネクスト ホップにわたってロード バランシングが行われます。すべての CE パスが失敗すると、そのラベル宛のすべてのトラフィックが IP ルックアップとなり、使用可能な場合は、バックアップ PE に転送されます。このアクションはラベルをポイントする可能性があるプレフィックスの数と関係なく、ラベル上で実行されるため、プライマリ パスの障害時は PIC の動作になります。

BGP と OSPF での過剰なパントフロートラップの実装

BGP と OSPF での過剰なパントフロートラップ（EPFT） 機能は、制御パケットトラフィックの割り当てられた共有よりも多くのリモートデバイスからの制御パケットトラフィックを識別し、軽減しようとします。リモートデバイスは、送信元の MAC アドレスで識別されます。リモートデバイスがルータに対して制御パケットトラフィックを送信すると、そのルータの CPU を保護するために、制御パケットは Local Packet Transport Service（LPTS） キューによってパントされ、ポリシングされます。1 台のデバイスから過剰なレートの制御パケットトラフィックが送信される場合は、ポリサーキューがいっぱいになり、多くのパケットがドロップされます。1 台の「バッドアクター」デバイスからのレートが他のデバイスのレートを大幅に超えている場合、他のデバイスのほとんどはルータまでの制御パケットをまったく取得できません。過剰なパントフロートラップ機能は、この状況に対処します。

過剰なパントフロートラップに関する情報

過剰なパントフロートラップ（EPFT）機能は、物理インターフェイス、サブインターフェイス、バンドルインターフェイス、およびバンドル サブインターフェイスからの制御パケットトラフィックをモニタします。この機能は、OSPF と BGP でバッドアクターを特定するのに役立ちます。EPFT は、送信元 MAC 単位で OSPF と BGP のルーティングプロトコルをモニタします。バッドアクターが検出されると、特定の期間にわたって制御パケットがドロップされ、送信元 MAC が一定の期間（デフォルトでは15分間）「ペナルティボックス」に配置されます。ペナルティタイムアウトの終了時に、特定の送信元MAC の TCAM エントリがドロップから除外されます。その後もリモートデバイスからの過剰なレートのパケットトラフィックが着信する場合は、リモートデバイスは再度トラップされます。

（注） 過剰なパントフロートラップ機能が有効になっていない場合でも、「バッドアクター」が他のデバイスのサービスのみに影響を与えることがあります。ただし、それらはルータをダウンさせることはできません。

EPFT 実装の制約事項

EPFT 機能の実装には、次の制約事項が適用されます。

- EPFT はサブスクライバ インターフェイスでは有効になっていません。
- BGP および OSPF ルーティングプロトコルのみがサポートされます。

- OSPFV3 はサポートされていません。
- OSPF および BGP パケットは、バッドアクターを特定した後、特定の期間（デフォルトは 15分）は完全にドロップされます。この場合はペナルティポリシングは行われません。
- サブスクライバインターフェイスまたはインターフェイスベースのフローが設定されている場合、**routing-protocol-enable** コマンドは設定できません。また、逆も同様です。つまり、 **routing-protocol-enable** コマンドが設定されている場合、サブスクライバ インターフェイスまたはインターフェイスベースのフローは設定できません。
- サテライト ICL インターフェイスは EPFT モニタリングから除外されます。

過剰なパントフロートラップ処理の有効化

OSPF または BGP プロトコルで過剰なパントフロートラップ（EPFT） 機能を有効にし、指定されたペナルティタイムアウト期間を使用するには、このタスクを実行します。

始める前に

EPFT は、サブスクライバ以外のインターフェイスでのみ有効にできます。

手順の概要

1. **configure**
2. **lpts punt excessive-flow-trap non-subscriber-interfaces mac**
3. **lpts punt excessive-flow-trap penalty-timeout** *protocol time*
4. **lpts punt excessive-flow-trap routing-protocols-enable**
5. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**lpts punt excessive-flow-trap non-subscriber-interfaces mac** 例： `RP/0/RSP0/cpu 0: router(config)# lpts punt excessive-flow-trap non-subscriber-interfaces mac`	過剰パントフロートラップ機能をサブスクライバ以外のターフェイスに対して有効にします。
ステップ 3	**lpts punt excessive-flow-trap penalty-timeout** *protocol time* 例： `RP/0/RSP0/cpu 0: router(config)# lpts punt excessive-flow-trap penalty-timeout bgp 10`	ペナルティタイムアウト値を設定します。これは、プロトコルのペナルティボックスに送信元 MAC トラップが配置される期間です。ペナルティタイムアウト値は分単位で、範囲は 1 ～ 1000 です。デフォルトのペナリティタイムアウト値は 15 分です。

	コマンドまたはアクション	目的
ステップ 4	**lpts punt excessive-flow-trap routing-protocols-enable** 例： `RP/0/RSP0/cpu 0: router(config)# lpts punt excessive-flow-trap routing-protocols-enable`	L3 ルーティングプロトコルで EPFT を有効にします。
ステップ 5	**commit**	

過剰なパントフロートラップ処理の有効化：例

次に、サブスクライバ以外のインターフェイスに対して過剰なパントフロートラップを有効にする例を示します。

```
configure
lpts punt excessive-flow-trap
 penalty-timeout ospf 20  <<optional>>
 penalty-timeout bgp 20   <<optional>>
 non-subscriber-interfaces mac  <<This is mandatory for routing protcols to be enabled>>
routing-protocols-enable
end
!!
```

過剰なパントフロートラップ機能に関して、バッドアクター、ペナルティステータス、およびその他の詳細情報を表示するには、次のいずれかの **show** コマンドを EXEC モードで使用します。

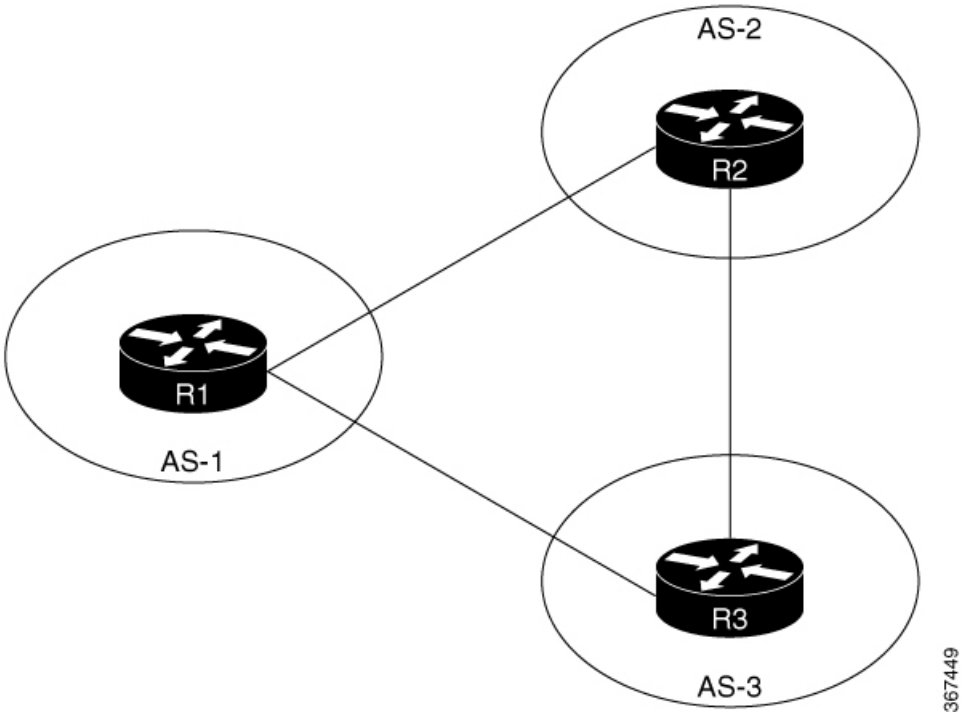
- **show lpts punt excessive-flow-trap** ［*protocol* ］
- **show lpts punt excessive-flow-trap all**

BGP マルチパスの機能拡張

- マルチパス プレフィックスのネクストホップ計算の上書きは許可されていません。**next-hop-unchanged multipath** コマンドを使用すると、マルチパス プレフィックスのネクストホップ計算の上書きが無効になります。
- マルチパスの計算時に as-path オンワードを無視する機能が追加されます。 **bgp multipath as-path ignore onwards** コマンドを使用すると、マルチパスの計算時に as-path オンワードが無視されます。

マルチパスを計算している間に複数の接続されたルータが以降の as-path を無視し始めると、ルーティングループが発生します。そのため、ループを形成する可能性があるルータでは **bgp multipath as-path ignore onwards** コマンドを設定しないでください。

図 *11* : ループの形成を示すトポロジ



異なる自律システム（AS-1、AS-2、および AS-3）内の 3 台のルータ、R1、R2、および R3 を想定します。これらのルータは相互に接続されています。R1 が R2 と R3 にプレフィックスをアナウンスします。R2 と R3 の両方がマルチパスを使用して設定されており、また、bgp multipath as-path で以降のコマンドが無視されます。R3 はマルチパスとして設定されているため、R2 はそのトラフィックの一部を R3 に送信します。同様に、R3 はそのトラフィックの一部を R2 に送信します。これにより、R3 と R2 の間に転送ループが発生します。そのため、このような転送ループを回避するには、接続されたルータで **bgp multipath as-path ignore onwards** コマンドを設定しないでください。

BGP SAFI-2 および SAFI-129 を使用した MVPN

BGP は、マルチキャスト VPN（MVPN）の後続のアドレスファミリ識別子（SAFI）-2 および SAFI-129 をサポートしています。

SAFI-129 は、コア IPv4 ネットワークでマルチキャストルーティングをサポートする機能を提供します。SAFI-129 は、BGP ベースの MVPN をサポートしています。SAFI-129 の追加により、マルチキャストで、ユニキャスト トポロジに依存しないこともあるアップストリーム マルチキャスト ホップを選択できるようになります。カスタマーエッジ（CE）ルータから学習したマルチキャストルートまたはリモートプロバイダーエッジ（PE）ルータから学習したマルチキャスト VPN ルートは、マルチキャストルーティング情報ベース（MuRIB）にインストールされます。この MuRIB には、マルチキャストに固有のルートが入力され、ユニキャスト転送では使用されません。PE-CE BGP プレフィックスは SAFI-2 を使用してアドバタイズされ、PE-PE のルートは SAFI-129 を使用してアドバタイズされます。

BGP Monitoring Protocol の概要

BGP Monitoring Protocol（BMP）機能により、BGP スピーカー（BMP クライアントという）をモニタできるようになります。BMP サーバとして機能するようにデバイスを設定して、複数のアクティブ ピア セッションが確立された 1 つまたは複数の BMP クライアントをモニタできます。また、1 つ以上の BMP サーバに接続するように BMP クライアントを設定することもできます。BMP 機能では、複数の BMP サーバ（プライマリ サーバとして設定）を、アクティブな状態で相互に独立して機能しながら BMP クライアントをモニタするように設定できます。

BMP プロトコルは、隣接するルーティング情報ベースやピアの着信 （Adj-RIB-In）テーブルへの継続的なアクセス、およびモニタリングステーションが詳細な分析のために使用できると特定の統計情報の定期的なダンプを提供します。BMP は、ピアの Adj-RIB-In テーブルをポリシーごとに表示します。

すべての BGP インスタンスに対して、グローバルに設定された複数の BMP サーバが存在する場合があります。設定された BMP サーバは複数のスピーカーインスタンス間で共通であり、インスタンス内の各 BGP ピアは、BMP サーバのすべてまたは一部によるモニタリング用に設定でき、BGP スピーカーの観点からは BGP ピアと BMP サーバ間の「Any-to-Any」マップを提供します。いずれかの BGP ピアが起動する前に BMP サーバが設定されている場合は、BGP ピアが起動するとすぐにモニタリングが開始されます。BMP サーバの設定は、その特定の BMP サーバによってモニタされるように設定されている BGP ピアがない場合にのみ削除できます。

BMP クライアントと BMP サーバ間のセッションは、プレーン TCP（暗号化/カプセル化なし）で動作します。BMP サーバとの TCP セッションが確立されていない場合、クライアントは 7 秒ごとに接続を再試行します。

BMP サーバは、そのクライアント（BGP スピーカー）にメッセージを送信しません。メッセージフローは一方向（BGP スピーカから BMP サーバへ）のみです。

Cisco NCS 5500 シリーズ ルータでは、最大 8 台の BMP サーバを設定できます。各 BMP サーバはサーバ ID で指定され、IP アドレス、ポート番号などの特定のパラメータを設定できます。ホストとポートの詳細を使用して BMP サーバを正常に設定すると、BGP スピーカーは BMP サーバへの接続を試行します。TCP 接続が設定されると、最初のメッセージとして開始メッセージが送信されます。

bmp server により、ユーザは複数（独立かつ非同期）の BMP サーバ接続の設定が可能になります。

BGP スピーカーのすべてのネイバーは、必ずしも BMP クライアントである必要はありません。BMP クライアントは、BMP サーバとの直接 TCP 接続を持っているクライアントです。これらの BGP スピーカーはそれぞれ、多数の BGP ネイバーまたはピアを持つことができます。BGP スピーカーの下で、そのネイバーのいずれかが BMP モニタリング用に設定されている場合、その特定のピアルータのメッセージのみが BMP サーバに送信されます。

BMP サーバへのセッション接続は、BMP クライアントでの初期遅延後に試行されます。この初期遅延は設定できます。初期遅延が設定されていない場合は、7 秒のデフォルトの接続遅延が使用されます。複数の BMP サーバの状態が厳密に切り替わり、リフレッシュ遅延が小さい特定の状況下で、初期遅延を設定することが重要になります。これにより、冗長なルートリフ

レッシュが生成される可能性があります。これにより、大量のネットワークトラフィックが発生し、デバイスに負荷がかかります。初期遅延が異なると、ネットワークとルータの負荷スパイクが低減される可能性があります。

初期遅延後、BMP サーバへの TCP 接続が試行されます。サーバ接続がアップ状態になると、モニタリングが有効になっているピアがあるかどうかが確認されます。すでにモニタされている BGP ピアが「ESTAB」状態になると、スピーカーはそのピアの「peer-up」メッセージを BMP サーバに送信します。BGP ピアがルートリフレッシュ要求を受信すると、ネイバーが更新を送信します。このルートリフレッシュは、各 BMP サーバに設定された遅延に基づいて開始されます。これをルートリフレッシュ遅延といいます。モニタするネイバが複数ある場合、それらが有効になっている BMP サーバに基づいて、各ネイバにはリフレッシ遅延が設定されます。すべての BGP ネイバーがリフレッシュ要求に応えて更新を送信すると、BMP サーバ内のテーブルが最新の状態になります。BMP モニタリングの開始後にネイバーが接続を確立する場合は、ルートリフレッシュ要求は必要ありません。そのネイバーから受信したすべてのルートが BMP サーバに送信されます。

（注） BMP プレインバウンドポリシーのルートモニタリングの場合、新しい BMP サーバが起動すると、BGP スピーカーによってルートリフレッシュ要求がピアルータに送信されます。ただし、BMP ポストインバウンドポリシーのルートモニタリングの場合、ルートリフレッシュ要求は、新しい BMP サーバが起動したときにピアルータに送信されません。これは、BMP テーブルが更新生成に使用されるためです。

複数の BMP サーバが立て続けにアクティブ化される場合は、BGP ピアにリフレッシュ要求をバッチ化すると便利です。**bmp server initial-refresh-delay** コマンドを使用して、最初の BMP サーバが起動したときにリフレッシュメカニズムをトリガーする際の遅延を設定できます。このタイムフレーム内に他の BMP サーバがオンラインになった場合は、1 セットのリフレッシュ要求のみが BGP ピアに送信されます。また、BGP スピーカーからのすべてのリフレッシュ要求をスキップし、ピアからのすべての着信メッセージだけをモニタするように、**bmp server initial-refresh-delay skip** コマンドを設定することもできます。

クライアントとサーバの設定では、デバイスのリソース負荷を最小限に抑え、過度なネットワーク トラフィックが発生しないようにすることが推奨されます。BMP 設定では、サーバとクライアントの間の接続でフラッピングが発生しないように、BMP サーバ上でさまざまな遅延タイマーを設定できます。

BGP の実装方法

BGP ルーティングのイネーブル化

BGP ルーティングをイネーブルにし、BGP ルーティング プロセスを設定するには、次の作業を実行します。BGP ネイバーの設定は、BGP ルーティングのイネーブル化の一部として含まれています。

（注） BGP ルーティングをイネーブルにするには、1 つ以上のネイバーおよび 1 つ以上のアドレス ファミリを設定する必要があります。**address family** コマンドおよび **remote as** コマンドを使用して、リモート AS とアドレス ファミリの両方を持つ 1 つ以上のネイバーをグローバルに設定する必要があります。

始める前に

BGP はルータ ID（設定済みループバック アドレスなど）を取得できなければなりません。1 つ以上のアドレス ファミリを BGP ルータ コンフィギュレーションに設定する必要があり、同じアドレス ファミリをネイバーの下にも設定する必要があります。

（注） ネイバーが外部 BGP（eBGP）ピアとして設定されている場合は、**route-policy** コマンドを使用して、インバウンドおよびアウトバウンドのルート ポリシーをネイバー上に設定する必要があります。

（注） 2 つのピア間で eBGP ネイバーシップを確立している間、BGP は 2 つのピアが直接接続されているかどうかをチェックします。ピアが直接接続されていない場合、デフォルトでは、BGP は関係を確立しようとしません。2 つの BGP ピアが直接接続されておらず、ルータのループバック間でピアリングが必要な場合は、**ignore-connected-check** コマンドを使用できます。このコマンドは、BGP 制御パケットの送信元 IP が宛先と同じネットワーク内にあるかどうかを確認するために BGP が実行するデフォルトのチェックを無効にします。このシナリオでは、TTL 値が 1 の場合 、**ignore-connected-check** が使用されていれば十分です。

egp-multihop *ttl* の設定は、ピアが直接接続されておらず、その間に多くのルータが存在する場合に必要です。**egp-multihop** *ttl* コマンドが設定されていない場合、デフォルトでは、eBGP は BGP メッセージを伝送するパケットの TTL を 1 に設定します。eBGP を複数ホップ離れているルータ間で設定する必要がある場合は、TTL 値を設定する必要があります。この TTL 値は、それらの間のホップ数以上にする必要があります。たとえば、2 つの BGP ピアリング ルータ R1 と R4 の間にホップが 2 つ（R2、R3）がある場合は、TTL 値を 3 に設定する必要があります。

手順の概要

1. **configure**
2. **route-policy** *route-policy-name*
3. **end-policy**
4. **commit**
5. **configure**
6. **router bgp** *as-number*
7. **bgp router-id** *ip-address*

8. **address-family** { **ipv4** | **ipv6** } **unicast**
9. **exit**
10. **neighbor** *ip-address*
11. **remote-as** *as-number*
12. **address-family** { **ipv4** | **ipv6** } **unicast**
13. **route-policy** *route-policy-name* { **in** | **out** }
14. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	

ステップ 2	**route-policy** *route-policy-name*
	例：
	`RP/0/RSP0/cpu 0: router(config)# route-policy drop-as-1234`
	`RP/0/RSP0/cpu 0: router(config-rpl)# if as-path passes-through '1234' then`
	`RP/0/RSP0/cpu 0: router(config-rpl)# apply check-communities`
	`RP/0/RSP0/cpu 0: router(config-rpl)# else`
	`RP/0/RSP0/cpu 0: router(config-rpl)# pass`
	`RP/0/RSP0/cpu 0: router(config-rpl)# endif`
		（任意）ルートポリシーを作成し、ルートポリシー コンフィギュレーション モードを開始します。このモードではルート ポリシーを定義できます。

ステップ 3	**end-policy** 例： `RP/0/RSP0/cpu 0: router(config-rpl)# end-policy`	（任意）ルート ポリシーの定義を終了し、ルート ポリシー コンフィギュレーション モードを終了します。
ステップ 4	**commit**	
ステップ 5	**configure**	
ステップ 6	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 120`	BGP AS 番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 7	**bgp router-id** *ip-address* 例： `RP/0/RSP0/cpu 0: router(config-bgp)# bgp router-id 192.168.70.24`	指定したルータ ID で、ローカル ルータを設定します。

	コマンドまたはアクション	目的
ステップ 8	**address-family** { **ipv4** | **ipv6** } **unicast** 例： `RP/0/RSP0/cpu 0: router(config-bgp)# address-family ipv4 unicast`	IPv4 または IPv6 のいずれかのアドレス ファミリを指定し、アドレスファミリのコンフィギュレーション サブモードを開始します。 このコマンドのすべてのキーワードと引数のリストを参照するには、CLI ヘルプ（?）を使用します。
ステップ 9	**exit** 例： `RP/0/RSP0/cpu 0: router(config-bgp-af)# exit`	現在のコンフィギュレーション モードを終了します。
ステップ 10	**neighbor** *ip-address* 例： `RP/0/RSP0/cpu 0: router(config-bgp)# neighbor 172.168.40.24`	BGP ルーティングのためにルータをネイバー コンフィギュレーション モードにして、ネイバーの IP アドレスを BGP ピアとして設定します。
ステップ 11	**remote-as** *as-number* 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# remote-as 2002`	ネイバーを作成し、リモート自律システム番号を割り当てます。
ステップ 12	**address-family** { **ipv4** | **ipv6** } **unicast** 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# address-family ipv4 unicast`	IPv4 または IPv6 のいずれかのアドレス ファミリを指定し、アドレスファミリのコンフィギュレーション サブモードを開始します。 このコマンドのすべてのキーワードと引数のリストを参照するには、CLI ヘルプ（?）を使用します。
ステップ 13	**route-policy** *route-policy-name* { **in** | **out** } 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr-af)# route-policy drop-as-1234 in`	（任意）指定したポリシーを着信 IPv4 ユニキャスト ルートに適用します。
ステップ 14	**commit**	

特定の自律システムに対する複数の BGP インスタンスの設定

特定の自律システムに複数の BGP インスタンスを設定するには、次のタスクを実行します。

単一の BGP インスタンスに対するすべての設定変更を同時にコミットすることができます。ただし、複数のインスタンスに対する設定変更は同時にコミットできません。

手順の概要

1. **configure**
2. **router bgp** *as-number* [**instance** *instance name*]
3. **bgp router-id** *ip-address*
4. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* [**instance** *instance name*] 例： `RP/0/RSP0/CPU0:router(config)# router bgp 100 instance inst1`	ユーザが指定した BGP インスタンスに対し BGP コンフィギュレーション モードを開始します。
ステップ 3	**bgp router-id** *ip-address* 例： `RP/0/RSP0/CPU0:router(config-bgp)# bgp router-id 10.0.0.0`	BGP スピーキング ルータの固定ルータ ID（BGP インスタンス）を設定します。 （注） 各 BGP インスタンスに一意のルータ ID を手動で設定する必要があります。
ステップ 4	**commit**	

BGP のルーティング ドメイン コンフェデレーションの設定

BGP のルーティング ドメイン コンフェデレーションを設定するには、次の作業を実行します。これには、コンフェデレーション ID の指定と、コンフェデレーションに属す自律システムの指定を含みます。

ルーティング ドメイン コンフェデレーションを設定すると、自律システムを複数の自律システムに分割して、これを 1 つのコンフェデレーションにグループ化することによって、内部 BGP（iBGP）メッシュを削減することができます。それぞれの自律システムは、そのシステム自身内で完全にメッシュ化されていて、同じコンフェデレーションの別の自律システムとの接続を数個持ちます。このコンフェデレーションによりネクストホップおよびローカルプリファレンス情報が維持され、これにより、すべての自律システムに対して Interior Gateway Protocol（IGP）を 1 つ維持できるようになります。外部からは、このコンフェデレーションは単一の自律システムであるかのように見えます。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **bgp confederation identifier** *as-number*
4. **bgp confederation peers** *as-number*
5. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router# router bgp 120`	自律システム番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**bgp confederation identifier** *as-number* 例： `RP/0/RSP0/cpu 0: router(config-bgp)# bgp confederation identifier 5`	BGP コンフェデレーション ID を指定します。

ステップ 4	**bgp confederation peers** *as-number*
	例：
	`RP/0/RSP0/cpu 0: router(config-bgp)# bgp confederation peers 1091`
	`RP/0/RSP0/cpu 0: router(config-bgp)# bgp confederation peers 1092`
	`RP/0/RSP0/cpu 0: router(config-bgp)# bgp confederation peers 1093`
	`RP/0/RSP0/cpu 0: router(config-bgp)# bgp confederation peers 1094`
	`RP/0/RSP0/cpu 0: router(config-bgp)# bgp confederation peers 1095`
	`RP/0/RSP0/cpu 0: router(config-bgp)# bgp confederation peers 1096`
		BGP 自律システムが指定された BGP コンフェデレーション ID に属することを指定します。例に示すように、複数の AS 番号を同じコンフェデレーション ID に関連付けることができます。

ステップ 5	**commit**	

リンク障害後の eBGP セッションの即時リセット

デフォルトでは、リンクがダウンすると、直接隣接する外部ピアの BGP セッションはすべて即時にリセットされます。自動リセットをディセーブルにするには **bgp fast-external-fallover disable** コマンドを使用します。自動リセットをイネーブルにするには **no bgp fast-external-fallover disable** コマンドを使用します。

BGP タイマー値が 10 および 30 に設定されているノードで eBGP セッションの数が 3500 に達すると、eBGP セッションはフラップします。3500 を超える数の eBGP セッションに対応するには、**lpts pifib hardware police location** *location-id* コマンドを使用してパケット レートを大きくします。eBGP セッションを増加する設定の例を次に示します。

```
RP/0/RSP0/cpu 0: router#configure
RP/0/RSP0/cpu 0: router(config)#lpts pifib hardware police location 0/2/CPU0
RP/0/RSP0/cpu 0: router(config-pifib-policer-per-node)#flow bgp configured rate 4000
RP/0/RSP0/cpu 0: router(config-pifib-policer-per-node)#flow bgp known rate 4000
RP/0/RSP0/cpu 0: router(config-pifib-policer-per-node)#flow bgp default rate 4000
RP/0/RSP0/cpu 0: router(config-pifib-policer-per-node)#commit
```

ネイバー変更のロギング

ネイバー変更のロギングはデフォルトでイネーブルになっています。ロギングをオフにするには、**log neighbor changes disable** コマンドを使用します。ロギングがディセーブルにされている場合にロギングを再びイネーブルにするには、**no log neighbor changes disable** コマンドを使用します。

BGP タイマーの調整

BGP ネイバーにタイマーを設定するには、次のタスクを実行します。

BGP は、定期実行アクティビティ（キープアライブ メッセージの送信、ネイバーがダウンしたと判断する条件となるそのネイバーからメッセージを受信しなかった期間など）を制御するために、特定のタイマーを使用します。ルータ コンフィギュレーション モードで timers bgp コマンドを使用して設定した値は、特定のネイバーでネイバー コンフィギュレーション モードで timers コマンドを使用すると上書きできます。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **timers bgp** *keepalive hold-time*
4. **neighbor** *ip-address*
5. **timers** *keepalive hold-time*
6. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 123`	自律システム番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**timers bgp** *keepalive hold-time* 例： `RP/0/RSP0/cpu 0: router(config-bgp)# timers bgp 30 90`	すべてのネイバーのデフォルトのキープアライブ時間とデフォルトの保留時間を設定します。
ステップ 4	**neighbor** *ip-address* 例： `RP/0/RSP0/cpu 0: router(config-bgp)# neighbor 172.168.40.24`	BGP ルーティングのためにルータをネイバー コンフィギュレーション モードにして、ネイバーの IP アドレスを BGP ピアとして設定します。

	コマンドまたはアクション	目的
ステップ 5	**timers** *keepalive hold-time* 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# timers 60 220`	（任意）BGP ネイバーのキープアライブ タイマーと保持時間タイマーを設定します。
ステップ 6	**commit**	

BGP のデフォルト ローカル プリファレンス値の変更

BGP パスのデフォルト ローカル プリファレンス値を設定するには、次の作業を実行します。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **bgp default local-preference** *value*
4. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 120`	自律システム番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**bgp default local-preference** *value* 例： `RP/0/RSP0/cpu 0: router(config-bgp)# bgp default local-preference 200`	デフォルト値 100 以外のデフォルト ローカル プリファレンス値を設定します。100 より大きい値を設定して推奨度を上げるか、または 100 未満の値を設定して推奨度を低くすることができます。
ステップ 4	**commit**	

BGP の MED メトリックの設定

メトリックがまだ設定されていないルート（MED 属性が設定されていない、受信されたルート）をピアにアドバタイズするように Multi Exit Discriminator（MED）を設定するには、次の作業を実行します。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **default-metric** *value*
4. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 120`	自律システム番号を指定し、BGP コンフィギュレーションモードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**default-metric** *value* 例： `RP/0/RSP0/cpu 0: router(config-bgp)# default metric 10`	まだメトリックが設定されていないルート（MED 属性を持たない、受信されたルート）をピアにアドバタイズするように MED を設定する場合に使用されるデフォルトのメトリックを設定します。
ステップ 4	**commit**	

BGP の重みの設定

ネイバーから受信したルートに重みを割り当てるには、次のタスクを実行します。重みとは、ベストパス選択プロセスを制御するためにパスに割り当てる数値です。ほとんどのトラフィックで特定のネイバーを優先する場合、**weight** コマンドを使用して、そのネイバーから学習したすべてのルートに大きい重みを割り当てることができます。

始める前に

（注） 新たに設定した重みを反映するには、**clear bgp** コマンドを使用する必要があります。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **neighbor** *ip-address*
4. **remote-as** *as-number*
5. **address-family** { **ipv4** | **ipv6** } **unicast**
6. **weight** *weight-value*
7. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 120`	自律システム番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**neighbor** *ip-address* 例： `RP/0/RSP0/cpu 0: router(config-bgp)# neighbor 172.168.40.24`	BGP ルーティングのためにルータをネイバー コンフィギュレーション モードにして、ネイバーの IP アドレスを BGP ピアとして設定します。
ステップ 4	**remote-as** *as-number* 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# remote-as 2002`	ネイバーを作成し、リモート自律システム番号を割り当てます。
ステップ 5	**address-family** { **ipv4** | **ipv6** } **unicast** 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# address-family ipv4 unicast`	IPv4 または IPv6 のいずれかのアドレス ファミリを指定し、アドレス ファミリのコンフィギュレーション サブモードを開始します。 このコマンドのすべてのキーワードと引数のリストを参照するには、CLI ヘルプ（?）を使用します。
ステップ 6	**weight** *weight-value* 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr-af)# weight 41150`	ネイバーから学習したすべてのルートに重みを割り当てます。
ステップ 7	**commit**	

BGP 最適パス計算の調整

デフォルトの BGP 最適パスの計算の動作を変更するには、次の作業を実行します。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **bgp bestpath med missing-as-worst**
4. **bgp bestpath med always**
5. **bgp bestpath med confed**

6. **bgp bestpath as-path ignore**
7. **bgp bestpath compare-routerid**
8. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 126`	自律システム番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**bgp bestpath med missing-as-worst** 例： `RP/0/RSP0/cpu 0: router(config-bgp)# bgp bestpath med missing-as-worst`	このパスを最も必要のないパスにするために、このパス内の不明 MED 属性の値は無限であると見なすように、BGP ソフトウェアに指示します。
ステップ 4	**bgp bestpath med always** 例： `RP/0/RSP0/cpu 0: router(config-bgp)# bgp bestpath med always`	パスがどの自律システムから受信されたかに関係なく、すべてのパスの間でプレフィックスについて MED を比較するように、指定した自律システムの BGP スピーカーを設定します。
ステップ 5	**bgp bestpath med confed** 例： `RP/0/RSP0/cpu 0: router(config-bgp)# bgp bestpath med confed`	コンフェデレーション ピアから学習したパスについて MED 値を BGP ソフトウェアで比較できるようにします。
ステップ 6	**bgp bestpath as-path ignore** 例： `RP/0/RSP0/cpu 0: router(config-bgp)# bgp bestpath as-path ignore`	最適パスを選択するときに、自律システム パスの長さが無視されるように BGP ソフトウェアを設定します。
ステップ 7	**bgp bestpath compare-routerid** 例： `RP/0/RSP0/cpu 0: router(config-bgp)# bgp bestpath compare-routerid`	類似パスのルータ ID を比較するように自律システムの BGP スピーカーを設定します。
ステップ 8	**commit**	

BGP バックドア ルートの指定

外部ボーダーゲートウェイ プロトコル（eBGP）のアドミニストレーティブディスタンスに、ローカルにソースされたBGPルートのアドミニストレーティブディスタンスを設定し、Interior Gateway Protocol（IGP）ルートよりも推奨度を低くするには、次の作業を実行します。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **address-family** { **ipv4** | **ipv6** } **unicast**
4. **network** { *ip-address* / *prefix-length* | *ip-address mask* } **backdoor**
5. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 120`	自律システム番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**address-family** { **ipv4**	**ipv6** } **unicast** 例： `RP/0/RSP0/cpu 0: router(config-bgp)# address-family ipv4 unicast`	IPv4 または IPv6 のいずれかのアドレス ファミリを指定し、アドレスファミリのコンフィギュレーション サブモードを開始します。 このコマンドのすべてのキーワードと引数のリストを参照するには、CLI ヘルプ（?）を使用します。
ステップ 4	**network** { *ip-address* / *prefix-length*	*ip-address mask* } **backdoor** 例： `RP/0/RSP0/cpu 0: router(config-bgp-af)# network 172.20.0.0/16`	指定されたネットワークを作成してアドバタイズするようにローカル ルータを設定します。
ステップ 5	**commit**	

集約アドレスの設定

BGP ルーティング テーブルに集約エントリを作成するには、次の作業を実行します。

手順の概要

1. **configure**
2. **router bgp** *as-number*

3. **address-family** { **ipv4** | **ipv6** } **unicast**
4. **aggregate-address** *address/mask-length* [**as-set**] [**as-confed-set**] [**summary-only**] [**route-policy** *route-policy-name*]
5. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 120`	自律システム番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**address-family** { **ipv4**	**ipv6** } **unicast** 例： `RP/0/RSP0/cpu 0: router(config-bgp)# address-family ipv4 unicast`	IPv4 または IPv6 のいずれかのアドレス ファミリを指定し、アドレス ファミリのコンフィギュレーション サブモードを開始します。 このコマンドのすべてのキーワードと引数のリストを参照するには、CLI ヘルプ（?）を使用します。
ステップ 4	**aggregate-address** *address/mask-length* [**as-set**] [**as-confed-set**] [**summary-only**] [**route-policy** *route-policy-name*] 例： `RP/0/RSP0/cpu 0: router(config-bgp-af)# aggregate-address 10.0.0.0/8 as-set`	集約アドレスを作成します。このルートにアドバタイズされたパスは、集約されるすべてのパスに含まれるすべての要素で構成された自律システム セットです。 • **as-set** キーワードは、関係するパスから自律システム セット パス情報およびコミュニティ情報を生成します。 • **as-confed-set** キーワードは、関係するパスから自律システム コンフェデレーション セット パス情報を生成します。 • **summary-only** キーワードは、アップデートから具体的なルートをすべてフィルタリングします。 • **route-policy** *route-policy-name* キーワードおよび引数は、集約ルートの属性の設定に使用されるルート ポリシーを指定します。
ステップ 5	**commit**	

IGP への iBGP ルートの再配布

Intermediate System-to-Intermediate System（IS-IS）や Open Shortest Path First（OSPF）など、内部ゲートウェイ プロトコル（IGP）に iBGP ルートを再配布するには、次の作業を実行します。

（注） **bgp redistribute-internal** コマンドを使用するには、すべての BGP ルートを IP ルーティング テーブルに再インストールするために、**clear route *** コマンドを発行する必要があります。

注意 IGP への iBGP ルートの再配布は、自律システム内にルーティング ループが作成される原因となる可能性があります。このコマンドの使用には注意が必要です。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **bgp redistribute-internal**
4. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 120`	自律システム番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**bgp redistribute-internal** 例： `RP/0/RSP0/cpu 0: router(config-bgp)# bgp redistribute-internal`	IGP（IS-IS や OSPF など）への iBGP ルートの再配布を許可します。
ステップ 4	**commit**	

過剰パスの破棄の設定

BGP 最大プレフィックス過剰パスの破棄を設定するには、次のタスクを実行します。

手順の概要

1. **configure**

2. **router bgp** *as-number*
3. **neighbor** *ip-address*
4. **address-family** { **ipv4** | **ipv6** } **unicast**
5. **maximum-prefix** *maximum* **discard-extra-paths**
6. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure** 例： `RP/0/RSP0/cpu 0: router# configure`	グローバル コンフィギュレーション モードを開始します。
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 10`	自律システム番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**neighbor** *ip-address* 例： `RP/0/RSP0/cpu 0: router(config-bgp)# neighbor 10.0.0.1`	BGP ルーティングのためにルータをネイバー コンフィギュレーション モードにして、ネイバーの IP アドレスを BGP ピアとして設定します。
ステップ 4	**address-family** { **ipv4** | **ipv6** } **unicast** 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# address-family ipv4 unicast`	IPv4 または IPv6 のいずれかのアドレス ファミリを指定し、アドレス ファミリのコンフィギュレーション サブモードを開始します。
ステップ 5	**maximum-prefix** *maximum* **discard-extra-paths** 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr-af)# maximum-prefix 1000 discard-extra-paths`	許可されるプレフィックス数の制限を設定します。 最大プレフィックスの制限を超えると過剰パスを破棄するように過剰パスの破棄を設定します。
ステップ 6	**commit**	

ネイバー単位の TCP MSS の設定

ネイバーによって継承されるネイバーグループに TCP MSS を設定するには、次のタスクを実行します。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **address-family ipv4 unicast**
4. **exit**
5. **neighbor-group** *name*

6. **tcp mss** *segment-size*
7. **address-family ipv4 unicast**
8. **exit**
9. **exit**
10. **neighbor** *ip-address*
11. **remote-as** *as-number*
12. **use neighbor-group** *group-name*
13. **address-family ipv4 unicast**
14. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure** 例： `RP/0/RSP0/cpu 0: router# configure`	グローバル コンフィギュレーション モードを開始します。
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 10`	自律システム番号を指定し、BGP コンフィギュレーションモードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**address-family ipv4 unicast** 例： `RP/0/RSP0/cpu 0: router(config-bgp)# address-family ipv4 unicast`	IPv4 アドレス ファミリ ユニキャストを指定し、アドレス ファミリ コンフィギュレーション モードを開始します。
ステップ 4	**exit** 例： `RP/0/RSP0/cpu 0: router(config-bgp-af)# exit`	ルータ アドレス ファミリ コンフィギュレーション モードを終了し、BGP コンフィギュレーション モードに戻ります。
ステップ 5	**neighbor-group** *name* 例： `RP/0/RSP0/cpu 0: router(config-bgp)# neighbor-group n1`	ネイバー グループ コンフィギュレーション モードを開始します。
ステップ 6	**tcp mss** *segment-size* 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp)# tcp mss 500`	TCP 最大セグメントサイズを設定します。範囲は 68 ～ 10000 です。
ステップ 7	**address-family ipv4 unicast** 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp)# address-family ipv4 unicast`	IPv4 アドレス ファミリ ユニキャストを指定し、アドレス ファミリ コンフィギュレーション モードを開始します。

	コマンドまたはアクション	目的
ステップ 8	**exit** 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp-af)# exit`	ルータ アドレス ファミリ コンフィギュレーション モードを終了します。
ステップ 9	**exit** 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp)# exit`	ネイバー グループ コンフィギュレーション モードを終了します。
ステップ 10	**neighbor** *ip-address* 例： `RP/0/RSP0/cpu 0: router(config-bgp)# neighbor 10.0.0.2`	BGP ルーティングのためにルータをネイバー コンフィギュレーション モードにして、ネイバーの IP アドレスを BGP ピアとして設定します。
ステップ 11	**remote-as** *as-number* 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# remote-as 1`	ネイバーを作成し、リモート自律（AS）システム番号を割り当てます。 • 2 バイト自律システム番号（ASN）の範囲は 1 ～ 65535 です。 • asplain 形式の 4 バイト自律システム番号（ASN）の範囲は、1 ～ 4294967295 です。 • asdot 形式の 4 バイト自律システム番号（ASN）の範囲は、1.0 ～ 65535.65535 です。
ステップ 12	**use neighbor-group** *group-name* 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# use neighbor-group n1`	BGP ネイバーが指定されたネイバー グループから設定を継承することを指定します。

ステップ 13	**address-family ipv4 unicast**
	例：
	`RP/0/RSP0/cpu 0: router(config-bgp-nbr)# address-family ipv4 unicast`
	`RP/0/RSP0/cpu 0: router(config-bgp-nbr-af)#`
		IPv4 アドレス ファミリ ユニキャストを指定し、アドレス ファミリ コンフィギュレーション モードを開始します。

ステップ 14	**commit**	

ネイバー単位の TCP MSS の無効化

ネイバーグループの特定のネイバーに対する TCP MSS を無効にするには、このタスクを実行します。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **address-family ipv4 unicast**
4. **exit**
5. **neighbor-group** *name*
6. **tcp mss** *segment-size*
7. **address-family ipv4 unicast**
8. **exit**
9. **exit**
10. **neighbor** *ip-address*
11. **remote-as** *as-number*
12. **use neighbor-group** *group-name*
13. **tcp mss inheritance-disable**
14. **address-family ipv4 unicast**
15. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure** 例： `RP/0/RSP0/cpu 0: router# configure`	グローバル コンフィギュレーション モードを開始します。
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 10`	自律システム番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**address-family ipv4 unicast** 例： `RP/0/RSP0/cpu 0: router(config-bgp)# address-family ipv4 unicast`	IPv4 アドレス ファミリ ユニキャストを指定し、アドレス ファミリ コンフィギュレーション モードを開始します。
ステップ 4	**exit** 例： `RP/0/RSP0/cpu 0: router(config-bgp-af)# exit`	ルータ アドレス ファミリ コンフィギュレーション モードを終了し、BGP コンフィギュレーション モードに戻ります。
ステップ 5	**neighbor-group** *name* 例： `RP/0/RSP0/cpu 0: router(config-bgp)# neighbor-group n1`	ネイバー グループ コンフィギュレーション モードを開始します。
ステップ 6	**tcp mss** *segment-size* 例：	TCP 最大セグメントサイズを設定します。範囲は 68 ～ 10000 です。

	コマンドまたはアクション	目的
	`RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp)# tcp mss 500`	
ステップ 7	**address-family ipv4 unicast** 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp)# address-family ipv4 unicast`	IPv4 アドレス ファミリ ユニキャストを指定し、アドレス ファミリ コンフィギュレーション モードを開始します。
ステップ 8	**exit** 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp-af)# exit`	ルータ アドレス ファミリ コンフィギュレーション モードを終了します。
ステップ 9	**exit** 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp)# exit`	ネイバー グループ コンフィギュレーション モードを終了します。
ステップ 10	**neighbor** *ip-address* 例： `RP/0/RSP0/cpu 0: router(config-bgp)# neighbor 10.0.0.2`	BGP ルーティングのためにルータをネイバー コンフィギュレーション モードにして、ネイバーの IP アドレスを BGP ピアとして設定します。
ステップ 11	**remote-as** *as-number* 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# remote-as 1`	ネイバーを作成し、リモート自律（AS）システム番号を割り当てます。 • 2 バイト自律システム番号（ASN）の範囲は 1 ～ 65535 です。 • asplain 形式の 4 バイト自律システム番号（ASN）の範囲は、1 ～ 4294967295 です。 • asdot 形式の 4 バイト自律システム番号（ASN）の範囲は、1.0 ～ 65535.65535 です。
ステップ 12	**use neighbor-group** *group-name* 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# use neighbor-group n1`	BGP ネイバーが指定されたネイバー グループから設定を継承することを指定します。
ステップ 13	**tcp mss inheritance-disable** 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# tcp mss inheritance-disable`	ネイバーに対する TCP MSS を無効にします。

	コマンドまたはアクション	目的

ステップ 14	**address-family ipv4 unicast**
	例：
	`RP/0/RSP0/cpu 0: router(config-bgp-nbr)# address-family ipv4 unicast`
	`RP/0/RSP0/cpu 0: router(config-bgp-nbr-af)#`
		IPv4 アドレス ファミリ ユニキャストを指定し、アドレス ファミリ コンフィギュレーション モードを開始します。

ステップ 15	**commit**	

マルチプロトコル BGP へのプレフィックスの再配布

別のプロトコルからマルチプロトコル BGP へプレフィックスを再配布するには、次のタスクを実行します。

再配布とは、あるルーティング プロトコルから別のルーティング プロトコルへプレフィックスを挿入するプロセスです。ここでは、別のルーティングプロトコルのプレフィックスをマルチプロトコル BGP に挿入する方法について説明します。具体的には、**redistribute** コマンドを使用してマルチプロトコル BGP に再配布されるプレフィックスは、ユニキャストデータベースまたはマルチキャストデータベース、あるいはその両方に挿入されます。

（注） BGP は、VRF での ISIS ルートの再配布をサポートしていません。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **address-family** { **ipv4** | **ipv6** } **unicast**
4. 次のいずれかを実行します。
   - **redistribute connected** [**metric** *metric-value*] [**route-policy** *route-policy-name*]
   - **redistribute eigrp** *process-id* [**match** { **external** | **internal** }] [**metric** *metric-value*] [**route-policy** *route-policy-name*]
   - **redistribute ospf** *process-id* [**match** { **external** [**1** | **2**] | **internal** | **nssa-external** [**1** | **2**]]} [**metric** *metric-value*] [**route-policy** *route-policy-name*]
   - **redistribute ospfv3** *process-id* [**match** { **external** [**1** | **2**] | **internal** | **nssa-external** [**1** | **2**]]} [**metric** *metric-value*] [**route-policy** *route-policy-name*]
   - **redistribute rip** [**metric** *metric-value*] [**route-policy** *route-policy-name*]
   - **redistribute static** [**metric** *metric-value*] [**route-policy** *route-policy-name*]
5. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 120`	自律システム番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**address-family** { **ipv4** | **ipv6** } **unicast** 例： `RP/0/RSP0/cpu 0: router(config-bgp)# address-family ipv4 unicast`	IPv4 または IPv6 のいずれかのアドレス ファミリを指定し、アドレス ファミリのコンフィギュレーション サブモードを開始します。 このコマンドのすべてのキーワードと引数のリストを参照するには、CLI ヘルプ（?）を使用します。
ステップ 4	次のいずれかを実行します。 • **redistribute connected** [**metric** *metric-value*] [**route-policy** *route-policy-name*] • **redistribute eigrp** *process-id* [**match** { **external** | **internal** }] [**metric** *metric-value*] [**route-policy** *route-policy-name*] • **redistribute ospf** *process-id* [**match** { **external** [**1** | **2**] | **internal** | **nssa-external** [**1** | **2**]]} [**metric** *metric-value*] [**route-policy** *route-policy-name*] • **redistribute ospfv3** *process-id* [**match** { **external** [**1** | **2**] | **internal** | **nssa-external** [**1** | **2**]]} [**metric** *metric-value*] [**route-policy** *route-policy-name*] • **redistribute rip** [**metric** *metric-value*] [**route-policy** *route-policy-name*] • **redistribute static** [**metric** *metric-value*] [**route-policy** *route-policy-name*] 例： `RP/0/RSP0/cpu 0: router(config-bgp-af)# redistribute ospf 110`	指定したインスタンスからのルートがBGPに再配布されるようにします。
ステップ 5	**commit**	

BGP ルート ダンプニングの設定

BGP ルート ダンプニングを設定してモニタするには、次の作業を実行します。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **address-family** { **ipv4** | **ipv6** } **unicast**
4. **bgp dampening** [*half-life* [*reuse suppress max-suppress-time*] | **route-policy** *route-policy-name*]
5. **commit**
6. **show bgp** [**ipv4** { **unicast** | **multicast** | **labeled-unicast** | **all** } | **ipv6 unicast** | **all** { **unicast** | **multicast** | **all** | **labeled-unicast** } | **vpnv4 unicast** [**rd** *rd-address*] | **vrf** { *vrf-name* | **all** } [**ipv4** { **unicast** | **labeled-unicast** } | **ipv6 unicast**]] **flap-statistics**
7. **show bgp** [**ipv4** { **unicast** | **multicast** | **labeled-unicast** | **all** } | **ipv6 unicast** | **all** { **unicast** | **multicast** | **all** | **labeled-unicast** } | **vpnv4 unicast** [**rd** *rd-address*] | **vrf** { *vrf-name* | **all** } [**ipv4** { **unicast** | **labeled-unicast** } | **ipv6 unicast**]] **flap-statistics regexp** *regular-expression*
8. **show bgp** [**ipv4** { **unicast** | **multicast** | **labeled-unicast** | **all** } | **ipv6 unicast** | **all** { **unicast** | **multicast** | **all** | **labeled-unicast** } | **vpnv4 unicast** [**rd** *rd-address*] | **vrf** { *vrf-name* | **all** } [**ipv4** { **unicast** | **labeled-unicast** } | **ipv6 unicast**]] **route-policy** *route-policy-name*
9. **show bgp** [**ipv4** { **unicast** | **multicast** | **labeled-unicast** | **all** } | **ipv6 unicast** | **all** { **unicast** | **multicast** | **all** | **labeled-unicast** } | **vpnv4 unicast** [**rd** *rd-address*] | **vrf** { *vrf-name* | **all** } [**ipv4** { **unicast** | **labeled-unicast** } | **ipv6 unicast**]] { *mask* | */prefix-length* }}
10. **show bgp** [**ipv4** { **unicast** | **multicast** | **labeled-unicast** | **all** } | **ipv6 unicast** | **all** { **unicast** | **multicast** | **all** | **labeled-unicast** } | **vpnv4 unicast** [**rd** *rd-address*] | **vrf** { *vrf-name* | **all** } [**ipv4** { **unicast** | **labeled-unicast** } | **ipv6 unicast**]] **flap-statistics** { *ip-address* [{ *mask* | */prefix-length* } [**longer-prefixes**
11. **clear bgp** [**ipv4** { **unicast** | **multicast** | **labeled-unicast** | **all** } | **ipv6 unicast** | **all** { **unicast** | **multicast** | **all** | **labeled-unicast** } | **vpnv4 unicast** [**rd** *rd-address*] | **vrf** { *vrf-name* | **all** } [**ipv4** { **unicast** | **labeled-unicast** } | **ipv6 unicast**]] **flap-statistics**
12. **clear bgp** [**ipv4** { **unicast** | **multicast** | **labeled-unicast** | **all** } | **ipv6 unicast** | **all** { **unicast** | **multicast** | **all** | **labeled-unicast** } | **vpnv4 unicast** [**rd** *rd-address*] | **vrf** { *vrf-name* | **all** } [**ipv4** { **unicast** | **labeled-unicast** } | **ipv6 unicast**]] **flap-statistics regexp** *regular-expression*
13. **clear bgp** [**ipv4** { **unicast** | **multicast** | **labeled-unicast** | **all** } | **ipv6 unicast** | **all** { **unicast** | **multicast** | **all** | **labeled-unicast** } | **vpnv4 unicast** [**rd** *rd-address*] | **vrf** { *vrf-name* | **all** } [**ipv4** { **unicast** | **labeled-unicast** } | **ipv6 unicast**]] **route-policy** *route-policy-name*
14. **clear bgp** [**ipv4** { **unicast** | **multicast** | **labeled-unicast** | **all** } | **ipv6 unicast** | **all** { **unicast** | **multicast** | **all** | **labeled-unicast** } | **vpnv4 unicast** [**rd** *rd-address*] | **vrf** { *vrf-name* | **all** } [**ipv4** { **unicast** | **labeled-unicast** } | **ipv6 unicast**]] **flap-statistics** *network* / *mask-length*
15. **clear bgp** [**ipv4** { **unicast** | **multicast** | **labeled-unicast** | **all** } | **ipv6 unicast** | **all** { **unicast** | **multicast** | **all** | **labeled-unicast** } | **vpnv4 unicast** [**rd** *rd-address*] | **vrf** { *vrf-name* | **all** } [**ipv4** { **unicast** | **labeled-unicast** } | **ipv6 unicast**]] **flap-statistics** *ip-address* / *mask-length*
16. **show bgp** [**ipv4** { **unicast** | **multicast** | **labeled-unicast** | **all** } | **ipv6 unicast** | **all** { **unicast** | **multicast** | **all** | **labeled-unicast** } | **vpnv4 unicast** [**rd** *rd-address*] | **vrf** { *vrf-name* | **all** } [**ipv4** { **unicast** | **labeled-unicast** } | **ipv6 unicast**]] **dampened-paths**
17. **clear bgp** [**ipv4** { **unicast** | **multicast** | **labeled-unicast** | **all** } | **ipv6 unicast** | **all** { **unicast** | **multicast** | **all** | **labeled-unicast** } | **vpnv4 unicast** [**rd** *rd-address*] | **vrf** { *vrf-name* | **all** } [**ipv4** { **unicast** | **labeled-unicast** } | **ipv6 unicast**]] **dampening** *ip-address* / *mask-length*

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 120`	自律システム番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**address-family** { **ipv4** | **ipv6** } **unicast** 例： `RP/0/RSP0/cpu 0: router(config-bgp)# address-family ipv4 unicast`	IPv4 または IPv6 のいずれかのアドレス ファミリを指定し、アドレス ファミリのコンフィギュレーション サブモードを開始します。 このコマンドのすべてのキーワードと引数のリストを参照するには、CLI ヘルプ（?）を使用します。
ステップ 4	**bgp dampening** [*half-life* [*reuse suppress max-suppress-time*] | **route-policy** *route-policy-name*] 例： `RP/0/RSP0/cpu 0: router(config-bgp-af)# bgp dampening 30 1500 10000 120`	指定したアドレス ファミリに対して BGP ダンプニングを設定します。
ステップ 5	**commit**	
ステップ 6	**show bgp** [**ipv4** { **unicast** | **multicast** | **labeled-unicast** | **all** } | **ipv6** **unicast** | **all** { **unicast** | **multicast** | **all** | **labeled-unicast** } | **vpnv4 unicast** [**rd** *rd-address*] | **vrf** { *vrf-name* | **all** } [**ipv4** { **unicast** | **labeled-unicast** } | **ipv6 unicast**]] **flap-statistics** 例： `RP/0/RSP0/cpu 0: router# show bgp flap statistics`	BGP フラップ統計情報を表示します。
ステップ 7	**show bgp** [**ipv4** { **unicast** | **multicast** | **labeled-unicast** | **all** } | **ipv6** **unicast** | **all** { **unicast** | **multicast** | **all** | **labeled-unicast** } | **vpnv4 unicast** [**rd** *rd-address*] | **vrf** { *vrf-name* | **all** } [**ipv4** { **unicast** | **labeled-unicast** } | **ipv6 unicast**]] **flap-statistics regexp** *regular-expression* 例： `RP/0/RSP0/cpu 0: router# show bgp flap-statistics regexp _1$`	正規表現に一致するすべてのパスの BGP フラップ統計情報を表示します。

	コマンドまたはアクション	目的
ステップ 8	**show bgp** [**ipv4** { **unicast** | **multicast** | **labeled-unicast** | **all** } | **ipv6** **unicast** | **all** { **unicast** | **multicast** | **all** | **labeled-unicast** } | **vpnv4 unicast** [**rd** *rd-address*] | **vrf** { *vrf-name* | **all** } [**ipv4** { **unicast** | **labeled-unicast** } | **ipv6 unicast**]] **route-policy** *route-policy-name* 例： `RP/0/RSP0/cpu 0: router(config)# show bgp flap-statistics route-policy policy_A`	指定されたルート ポリシーの BGP フラップ統計情報を表示します。
ステップ 9	**show bgp** [**ipv4** { **unicast** | **multicast** | **labeled-unicast** | **all** } | **ipv6** **unicast** | **all** { **unicast** | **multicast** | **all** | **labeled-unicast** } | **vpnv4 unicast** [**rd** *rd-address*] | **vrf** { *vrf-name* | **all** } [**ipv4** { **unicast** | **labeled-unicast** } | **ipv6 unicast**]] { *mask* | */prefix-length* }} 例： `RP/0/RSP0/cpu 0: router# show bgp flap-statistics 172.20.1.1`	指定されたプレフィックスの BGP フラップを表示します。
ステップ 10	**show bgp** [**ipv4** { **unicast** | **multicast** | **labeled-unicast** | **all** } | **ipv6** **unicast** | **all** { **unicast** | **multicast** | **all** | **labeled-unicast** } | **vpnv4 unicast** [**rd** *rd-address*] | **vrf** { *vrf-name* | **all** } [**ipv4** { **unicast** | **labeled-unicast** } | **ipv6 unicast**]] **flap-statistics** { *ip-address* [{ *mask* | */prefix-length* } [**longer-prefixes** 例： `RP/0/RSP0/cpu 0: router# show bgp flap-statistics 172.20.1.1 longer-prefixes`	指定された IP アドレスのより具体的なエントリの BGP フラップ統計情報を表示します。
ステップ 11	**clear bgp** [**ipv4** { **unicast** | **multicast** | **labeled-unicast** | **all** } | **ipv6** **unicast** | **all** { **unicast** | **multicast** | **all** | **labeled-unicast** } | **vpnv4 unicast** [**rd** *rd-address*] | **vrf** { *vrf-name* | **all** } [**ipv4** { **unicast** | **labeled-unicast** } | **ipv6 unicast**]] **flap-statistics** 例： `RP/0/RSP0/cpu 0: router# clear bgp all all flap-statistics`	すべてのルートの BGP フラップ統計情報をクリアします。
ステップ 12	**clear bgp** [**ipv4** { **unicast** | **multicast** | **labeled-unicast** | **all** } | **ipv6** **unicast** | **all** { **unicast** | **multicast** | **all** | **labeled-unicast** } | **vpnv4 unicast** [**rd** *rd-address*] | **vrf** { *vrf-name* | **all** } [**ipv4** {	指定された正規表現に一致するすべてのパスの BGP フラップ統計情報をクリアします。

	コマンドまたはアクション	目的
	unicast | **labeled-unicast** } | **ipv6 unicast**]] **flap-statistics regexp** *regular-expression* 例： `RP/0/RSP0/cpu 0: router# clear bgp ipv4 unicast flap-statistics regexp _1$`	
ステップ 13	**clear bgp** [**ipv4** { **unicast** | **multicast** | **labeled-unicast** | **all** } | **ipv6 unicast** | **all** { **unicast** | **multicast** | **all** | **labeled-unicast** } | **vpnv4 unicast** [**rd** *rd-address*] | **vrf** { *vrf-name* | **all** } [**ipv4** { **unicast** | **labeled-unicast** } | **ipv6 unicast**]] **route-policy** *route-policy-name* 例： `RP/0/RSP0/cpu 0: router# clear bgp ipv4 unicast flap-statistics route-policy policy_A`	指定されたルート ポリシーの BGP フラップ統計情報をクリアします。
ステップ 14	**clear bgp** [**ipv4** { **unicast** | **multicast** | **labeled-unicast** | **all** } | **ipv6 unicast** | **all** { **unicast** | **multicast** | **all** | **labeled-unicast** } | **vpnv4 unicast** [**rd** *rd-address*] | **vrf** { *vrf-name* | **all** } [**ipv4** { **unicast** | **labeled-unicast** } | **ipv6 unicast**]] **flap-statistics** *network* / *mask-length* 例： `RP/0/RSP0/cpu 0: router# clear bgp ipv4 unicast flap-statistics 192.168.40.0/24`	指定されたネットワークの BGP フラップ統計情報をクリアします。
ステップ 15	**clear bgp** [**ipv4** { **unicast** | **multicast** | **labeled-unicast** | **all** } | **ipv6 unicast** | **all** { **unicast** | **multicast** | **all** | **labeled-unicast** } | **vpnv4 unicast** [**rd** *rd-address*] | **vrf** { *vrf-name* | **all** } [**ipv4** { **unicast** | **labeled-unicast** } | **ipv6 unicast**]] **flap-statistics** *ip-address* / *mask-length* 例： `RP/0/RSP0/cpu 0: router# clear bgp ipv4 unicast flap-statistics 172.20.1.1`	指定されたネイバーから受信したルートの BGP フラップ統計情報をクリアします。
ステップ 16	**show bgp** [**ipv4** { **unicast** | **multicast** | **labeled-unicast** | **all** } | **ipv6 unicast** | **all** { **unicast** | **multicast** | **all** | **labeled-unicast** } | **vpnv4 unicast** [**rd** *rd-address*] | **vrf** { *vrf-name* | **all** } [**ipv4** { **unicast** | **labeled-unicast** } | **ipv6 unicast**]] **dampened-paths** 例：	抑制が解除されるまでの時間を含む、ダンプニングされたルートを表示します。

	コマンドまたはアクション	目的
	`RP/0/RSP0/cpu 0: router# show bgp dampened-paths`	
ステップ 17	**clear bgp** [**ipv4** { **unicast** | **multicast** | **labeled-unicast** | **all** } | **ipv6** **unicast** | **all** { **unicast** | **multicast** | **all** | **labeled-unicast** } | **vpnv4 unicast** [**rd** *rd-address*] | **vrf** { *vrf-name* | **all** } [**ipv4** { **unicast** | **labeled-unicast** } | **ipv6 unicast**]] **dampening** *ip-address* / *mask-length* 例： `RP/0/RSP0/cpu 0: router# clear bgp dampening`	ルート ダンプニング情報をクリアし、抑制されたルートを抑制解除します。 注意 **clear bgp dampening** コマンドは常に個々のアドレスファミリに対して使用してください。システムの通常動作中は、**clear bgp dampening** でアドレスファミリに対して **all** オプションを絶対に使用しないでください。たとえば `clear bgp ipv4 unicast dampening prefix x.x.x./y` を使用してください。

ルーティングテーブル更新時のポリシー適用

ルーティングテーブルにインストールされるルートにルーティングポリシーを適用するには、次の作業を実行します。

始める前に

テーブルポリシーのフィルタリングに使用可能なサポートされている属性と操作のリストについては、*Routing Configuration Guide for Cisco ASR 9000 Series Routers*（本書）の「でのルーティングポリシーの実装 Cisco ASR 9000 シリーズ ルータ 」のモジュールを参照してください。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **address-family** { **ipv4** \| **ipv6** } **unicast**
4. **table-policy** *policy-name*
5. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 120.6`	自律システム番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。

	コマンドまたはアクション	目的
ステップ 3	**address-family** { **ipv4** | **ipv6** } **unicast** 例： `RP/0/RSP0/cpu 0: router(config-bgp)# address-family ipv4 unicast`	IPv4 または IPv6 のいずれかのアドレス ファミリを指定し、アドレス ファミリのコンフィギュレーション サブモードを開始します。 このコマンドのすべてのキーワードと引数のリストを参照するには、CLI ヘルプ（?）を使用します。
ステップ 4	**table-policy** *policy-name* 例： `RP/0/RSP0/cpu 0: router(config-bgp-af)# table-policy tbl-plcy-A`	ルーティング テーブルにインストールされるルートに、指定されたポリシーを適用します。
ステップ 5	**commit**	

BGP アドミニストレーティブ ディスタンスの設定

あるルートのクラスよりも別のルートのクラスを優先するために使用できるアドミニストレーティブ ディスタンスを使用することを指定するには、次の作業を実行します。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **address-family** { **ipv4** | **ipv6** } **unicast**
4. **distance bgp** *external-distance internal-distance local-distance*
5. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 120`	自律システム番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**address-family** { **ipv4** | **ipv6** } **unicast** 例： `RP/0/RSP0/cpu 0: router(config-bgp)# address-family ipv4 unicast`	IPv4 または IPv6 のいずれかのアドレス ファミリ ユニキャストを指定し、アドレス ファミリのコンフィギュレーション サブモードを開始します。 このコマンドのすべてのキーワードと引数のリストを参照するには、CLI ヘルプ（?）を使用します。

	コマンドまたはアクション	目的
ステップ 4	**distance bgp** *external-distance internal-distance local-distance* 例： `RP/0/RSP0/cpu 0: router(config-bgp-af)# distance bgp 20 20 200`	あるルートのクラスよりも別のルートのクラスを優先するために外部、内部、およびローカルのアドミニストレーティブディスタンスを設定します。値が高いほど、信頼性のランクは低くなります。
ステップ 5	**commit**	

BGP ネイバー グループおよびネイバーの設定

BGP ネイバー グループを設定し、ネイバーにネイバー グループの設定を適用するには、次の作業を実行します。ネイバー グループは、ネイバーに関連するアドレス ファミリから独立した設定とアドレス ファミリ固有の設定を持つテンプレートです。

ネイバー グループを設定すると、各ネイバーは、**use** コマンド経由で設定を継承できるようになります。ネイバー グループを使用するように設定されているネイバーは、デフォルトでネイバー グループの設定すべて（アドレス ファミリに依存しない設定とアドレス ファミリ固有の設定を含む）を継承します。継承された設定を上書きするには、ネイバーに対して直接コマンドを設定するか、または **use** コマンドを使用して、セッション グループ、またはアドレス ファミリ グループを設定します。

ネイバー グループではアドレス ファミリに依存しない設定を行うことができます。アドレス ファミリ固有の設定では、アドレス ファミリ サブモードを開始するようにネイバー グループのアドレス ファミリを設定する必要があります。

ネイバー グループ コンフィギュレーション モードでは、ネイバー グループについて、アドレス ファミリに依存しないパラメータを設定できます。ネイバー グループ コンフィギュレーション モードで **address-family** コマンドを使用します。

neighbor group コマンドを使用してネイバー グループ名を指定した後で、オプションをそのネイバー グループに割り当てることができます。

（注） 指定されたネイバー グループで設定できるコマンドはすべて、ネイバーでも設定できます。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **address-family** { **ipv4** | **ipv6** } **unicast**
4. exit
5. **neighbor-group** *name*
6. **remote-as** *as-number*
7. **address-family** { **ipv4** | **ipv6** } **unicast**

8. **route-policy** *route-policy-name* { **in** | **out** }
9. **exit**
10. **exit**
11. **neighbor** *ip-address*
12. **use neighbor-group** *group-name*
13. **remote-as** *as-number*
14. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 120`	自律システム番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**address-family** { **ipv4** | **ipv6** } **unicast** 例： `RP/0/RSP0/cpu 0: router(config-bgp)# address-family ipv4 unicast`	IPv4 または IPv6 のいずれかのアドレス ファミリ ユニキャストを指定し、アドレス ファミリのコンフィギュレーション サブモードを開始します。 このコマンドのすべてのキーワードと引数のリストを参照するには、CLI ヘルプ（?）を使用します。
ステップ 4	exit 例： `RP/0/RSP0/cpu 0: router(config-bgp-af)# exit`	現在のコンフィギュレーション モードを終了します。
ステップ 5	**neighbor-group** *name* 例： `RP/0/RSP0/cpu 0: router(config-bgp)# neighbor-group nbr-grp-A`	ルータをネイバー グループ コンフィギュレーション モードにします。
ステップ 6	**remote-as** *as-number* 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp)# remote-as 2002`	ネイバーを作成し、リモート自律システム番号を割り当てます。
ステップ 7	**address-family** { **ipv4** | **ipv6** } **unicast** 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp)# address-family ipv4 unicast`	IPv4 または IPv6 のいずれかのアドレス ファミリ ユニキャストを指定し、アドレス ファミリのコンフィギュレーション サブモードを開始します。 このコマンドのすべてのキーワードと引数のリストを参照するには、CLI ヘルプ（?）を使用します。

	コマンドまたはアクション	目的
ステップ 8	**route-policy** *route-policy-name* { **in** | **out** } 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp-af)# route-policy drop-as-1234 in`	（任意）指定したポリシーを着信 IPv4 ユニキャスト ルートに適用します。
ステップ 9	**exit** 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp-af)# exit`	現在のコンフィギュレーション モードを終了します。
ステップ 10	**exit** 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp)# exit`	現在のコンフィギュレーション モードを終了します。
ステップ 11	**neighbor** *ip-address* 例： `RP/0/RSP0/cpu 0: router(config-bgp)# neighbor 172.168.40.24`	BGP ルーティングのためにルータをネイバー コンフィギュレーション モードにして、ネイバーの IP アドレスを BGP ピアとして設定します。
ステップ 12	**use neighbor-group** *group-name* 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# use neighbor-group nbr-grp-A`	（任意）BGP ネイバーが指定されたネイバー グループから設定を継承することを指定します。
ステップ 13	**remote-as** *as-number* 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# remote-as 2002`	ネイバーを作成し、リモート自律システム番号を割り当てます。
ステップ 14	**commit**	

BGP のルートリフレクタの設定

BGP のルート リフレクタを設定するには、次の作業を実行します。

route-reflector-client コマンドで設定されるネイバーはすべてクライアント グループのメンバーであり、その他の iBGP ピアはローカル ルータ リフレクタの非クライアント グループのメンバーです。

ルート リフレクタは、そのクライアントとあわせてクラスタを形成します。クライアントからなるクラスタには通常、ルート リフレクタが 1 つ存在します。このようなインスタンスでは、

クラスタはソフトウェアにより、ルート リフレクタのルータ ID と認識されます。冗長性を高め、ネットワークでのシングルポイント障害を回避するために、クラスタに複数のリフレクタが含まれていることもあります。この場合、このクラスタのルート リフレクタはすべて、同じ 4 バイトのクラスタ ID を使って設定する必要があります。これはルート リフレクタが、同じクラスタに属する別のルート リフレクタからのアップデートを認識できるようにするためです。クラスタに複数のルータ リフレクタがある場合にクラスタ ID を設定するには、**bgp cluster-id** コマンドを使用します。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **bgp cluster-id** *cluster-id*
4. **neighbor** *ip-address*
5. **remote-as** *as-number*
6. **address-family** { **ipv4** | **ipv6** } **unicast**
7. **route-reflector-client**
8. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 120`	自律システム番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**bgp cluster-id** *cluster-id* 例： `RP/0/RSP0/cpu 0: router(config-bgp)# bgp cluster-id 192.168.70.1`	クラスタに対応するルート リフレクタの 1 つとして、ローカル ルータを設定します。クラスタを識別するために、指定したクラスタ ID を設定します。
ステップ 4	**neighbor** *ip-address* 例： `RP/0/RSP0/cpu 0: router(config-bgp)# neighbor 172.168.40.24`	BGP ルーティングのためにルータをネイバー コンフィギュレーション モードにして、ネイバーの IP アドレスを BGP ピアとして設定します。
ステップ 5	**remote-as** *as-number* 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# remote-as 2003`	ネイバーを作成し、リモート自律システム番号を割り当てます。

	コマンドまたはアクション	目的
ステップ 6	**address-family** { **ipv4** | **ipv6** } **unicast** 例： `RP/0/RSP0/cpu 0: router(config-nbr)# address-family ipv4 unicast`	IPv4 または IPv6 のいずれかのアドレス ファミリ ユニキャストを指定し、アドレス ファミリのコンフィギュレーション サブモードを開始します。 このコマンドのすべてのキーワードと引数のリストを参照するには、CLI ヘルプ（?）を使用します。
ステップ 7	**route-reflector-client** 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr-af)# route-reflector-client`	BGP ルート リフレクタとしてルータを設定し、そのクライアントとしてネイバーを設定します。
ステップ 8	**commit**	

ルート ポリシーによる BGP ルート フィルタリングの設定

ルート ポリシーによる BGP ルーティング フィルタリングを設定するには、次の作業を実行します。

始める前に

インバウンドおよびアウトバウンドのネイバー ポリシー フィルタリングで使用可能なサポートされている属性と操作のリストについては、『*Cisco ASR 9000* シリーズ アグリゲーション サービス ルータ ルーティング設定ガイド』（本書）の「*Cisco IOS XR* ソフトウェア Cisco ASR 9000 シリーズ ルータ」のモジュールを参照してください。

手順の概要

1. **configure**
2. **route-policy** *name*
3. **end-policy**
4. **router bgp** *as-number*
5. **neighbor** *ip-address*
6. **address-family** { **ipv4** \| **ipv6** } **unicast**
7. **route-policy** *route-policy-name* { **in** \| **out** }
8. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	

	コマンドまたはアクション	目的

ステップ 2	**route-policy** *name*
	例：
	`RP/0/RSP0/cpu 0: router(config)# route-policy drop-as-1234`
	`RP/0/RSP0/cpu 0: router(config-rpl)# if as-path passes-through '1234' then`
	`RP/0/RSP0/cpu 0: router(config-rpl)# apply check-communities`
	`RP/0/RSP0/cpu 0: router(config-rpl)# else`
	`RP/0/RSP0/cpu 0: router(config-rpl)# pass`
	`RP/0/RSP0/cpu 0: router(config-rpl)# endif`
		（任意）ルートポリシーを作成し、ルートポリシーコンフィギュレーションモードを開始します。このモードではルート ポリシーを定義できます。

ステップ 3	**end-policy** 例： `RP/0/RSP0/cpu 0: router(config-rpl)# end-policy`	（任意）ルート ポリシーの定義を終了し、ルート ポリシー コンフィギュレーション モードを終了します。
ステップ 4	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 120`	自律システム番号を指定し、BGP コンフィギュレーションモードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 5	**neighbor** *ip-address* 例： `RP/0/RSP0/cpu 0: router(config-bgp)# neighbor 172.168.40.24`	BGP ルーティングのためにルータをネイバー コンフィギュレーション モードにして、ネイバーの IP アドレスを BGP ピアとして設定します。
ステップ 6	**address-family** { **ipv4** | **ipv6** } **unicast** 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# address-family ipv4 unicast`	IPv4 または IPv6 のいずれかのアドレス ファミリ ユニキャストを指定し、アドレスファミリのコンフィギュレーション サブモードを開始します。 このコマンドのすべてのキーワードと引数のリストを参照するには、CLI ヘルプ（?）を使用します。
ステップ 7	**route-policy** *route-policy-name* { **in** | **out** } 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr-af)# route-policy drop-as-1234 in`	指定されたポリシーをインバウンドルートに適用します。
ステップ 8	**commit**	

BGP 属性フィルタリングの設定

BGP 属性フィルタリングを設定するには、次のタスクを実行します。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **attribute-filter group** *attribute-filter group name*
4. **attribute** *attribute code* { **discard** | **treat-as-withdraw** }

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 100`	自律システム番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**attribute-filter group** *attribute-filter group name* 例： `RP/0/RSP0/cpu 0: router(config-bgp)# attribute-filter group ag_discard_med`	属性フィルタ グループ名を指定し、属性フィルタ グループ コンフィギュレーション モードを開始することで、BGP ネイバーに特定の属性フィルタ グループを設定できます。
ステップ 4	**attribute** *attribute code* { **discard** | **treat-as-withdraw** } 例： `RP/0/RSP0/cpu 0: router(config-bgp-attrfg)# attribute 24 discard`	単一またはある範囲の属性コードと関連するアクションを指定します。実行できるアクションには次のものがあります。 • Treat-as-withdraw：アップデート メッセージを取り消すかを検討します。対応する IPv4 ユニキャストまたは MP_REACH NLRI があれば、ネイバーの Adj-RIB-In から取り消します。 • Discard Attribute：この属性を廃棄します。一致した部分の属性は廃棄され、アップデート メッセージの残りの部分は正常に処理されます。

BGP ネクスト ホップ トリガー遅延の設定

BGP ネクスト ホップ トリガー遅延を設定するには、次の作業を実行します。ルーティング情報ベース（RIB）では変更の重大度に基づいてダンプニング通知が分類されます。イベント通知はクリティカルおよび非クリティカルとして分類されます。この作業では、クリティカル イベントと非クリティカル イベントの最小バッチ間隔を指定できます。

手順の概要

1. **configure**
2. **router bgp** *as-number*

3. **address-family** { **ipv4** | **ipv6** } **unicast**
4. **nexthop trigger-delay** { **critical** *delay* | **non-critical** *delay* }
5. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 120`	自律システム番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**address-family** { **ipv4**	**ipv6** } **unicast** 例： `RP/0/RSP0/cpu 0: router(config-bgp)# address-family ipv4 unicast`	IPv4 または IPv6 のいずれかのアドレス ファミリ ユニキャストを指定し、アドレス ファミリのコンフィギュレーション サブモードを開始します。 このコマンドのすべてのキーワードと引数のリストを参照するには、CLI ヘルプ（?）を使用します。
ステップ 4	**nexthop trigger-delay** { **critical** *delay*	**non-critical** *delay* } 例： `RP/0/RSP0/cpu 0: router(config-bgp-af)# nexthop trigger-delay critical 15000`	重要なネクスト ホップ トリガー遅延を設定します。
ステップ 5	**commit**	

BGP 更新でのネクストホップ処理の無効化

ネイバーに対するネクスト ホップの計算をディセーブルにし、BGP アップデートのネクスト ホップ フィールドにユーザ自身のアドレスを挿入するには、次の作業を実行します。ルートをアドバタイズするときに使用する最適なネクスト ホップの計算をディセーブルにすると、すべてのルートがネットワーク デバイスによってネクスト ホップとしてアドバタイズされます。

（注） ネクスト ホップ処理は、アドレス ファミリ グループ、ネイバー グループ、またはネイバー アドレス ファミリに対して無効にすることができます。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **neighbor** *ip-address*

4. **remote-as** *as-number*
5. **address-family** { **ipv4** | **ipv6** } **unicast**
6. **next-hop-self**
7. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 120`	自律システム番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**neighbor** *ip-address* 例： `RP/0/RSP0/cpu 0: router(config-bgp)# neighbor 172.168.40.24`	BGP ルーティングのためにルータをネイバー コンフィギュレーション モードにして、ネイバーの IP アドレスを BGP ピアとして設定します。
ステップ 4	**remote-as** *as-number* 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# remote-as 206`	ネイバーを作成し、リモート自律システム番号を割り当てます。
ステップ 5	**address-family** { **ipv4** | **ipv6** } **unicast** 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# address-family ipv4 unicast`	IPv4 または IPv6 のいずれかのアドレス ファミリ ユニキャストを指定し、アドレス ファミリのコンフィギュレーション サブモードを開始します。 このコマンドのすべてのキーワードと引数のリストを参照するには、CLI ヘルプ（?）を使用します。
ステップ 6	**next-hop-self** 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr-af)# next-hop-self`	指定されたネイバーにアドバタイズされるすべてのルートのネクスト ホップ属性をローカル ルータのアドレスに設定します。ルートをアドバタイズするときに使用する最適なネクストホップの計算をディセーブルにすると、すべてのルートがローカル ネットワーク デバイスによってネクスト ホップとしてアドバタイズされます。
ステップ 7	**commit**	

BGPコミュニティおよび拡張コミュニティアドバタイズメントの設定

コミュニティ属性および拡張コミュニティ属性を eBGP ネイバーに送信することを指定するには、次の作業を実行します。これらの属性は、デフォルトでは eBGP ネイバーに送信されません。これに対して、iBGP ネイバーには常に送信されます。ここでは、コミュニティ属性を送信できるようにする方法の例を示します。拡張コミュニティを送信できるようにするには、**send-community-ebgp** キーワードを **send-extended-community-ebgp** キーワードで置き換えます。

send-community-ebgp コマンドをネイバー グループまたはアドレス ファミリ グループに対して設定すると、このグループを使用するすべてのネイバーが設定を継承します。あるネイバーに対して特別にこのコマンドを設定すると、継承された値が上書きされます。

（注） BGP コミュニティと拡張コミュニティ フィルタリングは、iBGP ネイバーには設定できません。コミュニティと拡張コミュニティは、VPNv4、MDT、IPv4、および IPv6 アドレス ファミリでは常に iBGP ネイバーに送信されます。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **neighbor** *ip-address*
4. **remote-as** *as-number*
5. **address-family**{**ipv4** {**labeled-unicast** | **unicast** | **mdt** | **multicast** | **mvpn** | **tunnel**} | **ipv6** {**labeled-unicast** | **mvpn** | **unicast**}}
6. 次のいずれかのコマンドを使用します。
   - **send-community-ebgp**
   - **send-extended-community-ebgp**
7. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 120`	自律システム番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**neighbor** *ip-address* 例： `RP/0/RSP0/cpu 0: router(config-bgp)# neighbor 172.168.40.24`	BGP ルーティングのためにルータをネイバー コンフィギュレーション モードにして、ネイバーの IP アドレスを BGP ピアとして設定します。

	コマンドまたはアクション	目的
ステップ 4	**remote-as** *as-number* 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# remote-as 2002`	ネイバーを作成し、リモート自律システム番号を割り当てます。
ステップ 5	**address-family**{**ipv4** {**labeled-unicast** | **unicast** | **mdt** | **multicast** | **mvpn** | **tunnel**} | **ipv6** {**labeled-unicast** | **mvpn** | **unicast**}} 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# address-family ipv6 unicast`	指定のアドレス ファミリに対応しネイバー アドレス ファミリ コンフィギュレーション モードを開始します。**ipv4** または **ipv6** アドレス ファミリ キーワードと、指定したアドレス ファミリ サブ モード ID の 1 つを使用します。 IPv6 アドレス ファミリ モードでは、次のサブモードをサポートしています。 • **labeled-unicast** • **mvpn** • **unicast** IPv4 アドレス ファミリ モードでは、次のサブモードをサポートしています。 • **labeled-unicast** • **mdt** • **multicast** • **mvpn** • **rt-filter** • **tunnel** • **unicast** アドレス ファミリ サブモードのサポートの詳細については、 *Routing Command Reference for Cisco ASR 9000 Series Routers*の「*BGP Commands*」のモジュールの **address-family (BGP)** コマンドを参照してください。
ステップ 6	次のいずれかのコマンドを使用します。 • **send-community-ebgp** • **send-extended-community-ebgp** 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr-af)# send-community-ebgp` または	ルータが（デフォルトでは eBGP ネイバーでディセーブルにされている）コミュニティ属性と拡張コミュニティ属性を指定された eBGP ネイバーに送信することを指定します。

	コマンドまたはアクション	目的
	`RP/0/RSP0/cpu 0: router(config-bgp-nbr-af)# send-extended-community-ebgp`	
ステップ 7	**commit**	

BGP コスト コミュニティの設定

BGP コスト コミュニティを設定するには、次のタスクを実行します。

BGP は同一宛先への複数のパスを受信し、最適パス アルゴリズムを使用して RIB にインストールする最適なパスを決定します。ユーザが部分比較後に出力点を決定できるようにするため、最適パス選択処理で同等パスのタイブレークのためにコスト コミュニティが定義されます。

手順の概要

1. **configure**
2. **route-policy** *name*
3. **set extcommunity cost** { *cost-extcommunity-set-name* | *cost-inline-extcommunity-set* } [**additive**]
4. **end-policy**
5. **router bgp** *as-number*
6. 次のいずれかを実行します。
   - **default-information originate**
   - **aggregate-address** *address/mask-length* [**as-set**] [**as-confed-set**] [**summary-only**] [**route-policy** *route-policy-name*]
   - **address-family** { **ipv4 unicast** | **ipv4 multicast** | **ipv4 tunnel** | **ipv6 unicast** | **vpnv4 unicast** } **redistribute connected** [**metric** *metric-value*] [**route-policy** *route-policy-name*]
   - **address-family** { **ipv4 unicast** | **ipv4 multicast** | **ipv4 tunnel** | **ipv6 unicast** | **vpnv4 unicast** } **redistribute eigrp** *process-id* [**match** { **external** | **internal** }] [**metric** *metric-value*] [**route-policy** *route-policy-name*]
   - **address-family** { **ipv4 unicast** | **ipv4 multicast** | **ipv4 tunnel** | **ipv6 unicast** | **vpnv4 unicast** } **redistribute isis** *process-id* [**level** { **1** | **1-inter-area** | **2** }] [**metric** *metric-value*] [**route-policy** *route-policy-name*]
   - **address-family** { **ipv4 unicast** | **ipv4 multicast** | **ipv4 tunnel** | **ipv6 unicast** | **vpnv4 unicast** } **redistribute ospf** *process-id* [**match** { **external** [**1** | **2**] | **internal** | **nssa-external** [**1** | **2**]}] [**metric** *metric-value*] [**route-policy** *route-policy-name*]
7. 次のいずれかを実行します。
   - **address-family** { **ipv4 unicast** | **ipv4 multicast** | **ipv4 tunnel** | **ipv4 mdt** | **ipv6 unicast** | **ipv6 multicast** | **vpnv4 unicast** | **vpnv6 unicast** } **redistribute ospfv3** *process-id* [**match** { **external** [**1** | **2**] | **internal** | **nssa-external** [**1** | **2**]}] [**metric** *metric-value*] [**route-policy** *route-policy-name*]
   - **address-family** { **ipv4 unicast** | **ipv4 multicast** | **ipv4 tunnel** | **ipv4 mdt** | **ipv6 unicast** | **ipv6 multicast** | **vpnv4 unicast** | **vpnv6 unicast** } **redistribute rip** [**metric** *metric-value*] [**route-policy** *route-policy-name*]

- **address-family** { **ipv4 unicast** | **ipv4 multicast** | **ipv4 tunnel** | **ipv4 mdt** | **ipv6 unicast** | **ipv6 multicast** | **vpnv4 unicast** | **vpnv6 unicast** } **redistribute static** [**metric** *metric-value*] [**route-policy** *route-policy-name*]
- **address-family** { **ipv4 unicast** | **ipv4 multicast** | **ipv4 tunnel** | **ipv4 mdt** | **ipv6 unicast** | **ipv6 multicast** | **vpnv4 unicast** | **vpnv6 unicast** } **network** { *ip-address/prefix-length* | *ip-address mask* } [**route-policy** *route-policy-name*]
- **neighbor** *ip-address* **remote-as** *as-number* **address-family** { **ipv4 unicast** | **ipv4 multicast** | **ipv4 tunnel** | **ipv4 mdt** | **ipv6 unicast** | **ipv6 multicast** | **vpnv4 unicast** | **vpnv6 unicast** }
- **route-policy** *route-policy-name* { **in** | **out** }

8. **commit**
9. **show bgp** [**vrf** *vrf-name*] *ip-address*

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**route-policy** *name* 例： `RP/0/RSP0/cpu 0: router(config)# route-policy costA`	ルート ポリシー コンフィギュレーション モードに切り替え、設定するルート ポリシーの名前を指定します。
ステップ 3	**set extcommunity cost** { *cost-extcommunity-set-name* | *cost-inline-extcommunity-set* } [**additive**] 例： `RP/0/RSP0/cpu 0: router(config)# set extcommunity cost cost_A`	コストの BGP 拡張コミュニティ属性を指定します。
ステップ 4	**end-policy** 例： `RP/0/RSP0/cpu 0: router(config)# end-policy`	ルート ポリシーの定義を終了して、ルート ポリシー コンフィギュレーション モードを終了します。
ステップ 5	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 120`	BGP コンフィギュレーション モードを開始します。このモードでは BGP ルーティング プロセスを設定できます。
ステップ 6	次のいずれかを実行します。 • **default-information originate** • **aggregate-address** *address/mask-length* [**as-set**] [**as-confed-set**] [**summary-only**] [**route-policy** *route-policy-name*] • **address-family** { **ipv4 unicast** | **ipv4 multicast** | **ipv4 tunnel** | **ipv6 unicast** | **vpnv4 unicast** }	コスト コミュニティを付加ポイント（ルート ポリシー）に適用します。

	コマンドまたはアクション	目的
	redistribute connected [**metric** *metric-value*] [**route-policy** *route-policy-name*] • **address-family** { **ipv4 unicast** | **ipv4 multicast** | **ipv4 tunnel** | **ipv6 unicast** | **vpnv4 unicast** } **redistribute eigrp** *process-id* [**match** { **external** | **internal** }] [**metric** *metric-value*] [**route-policy** *route-policy-name*] • **address-family** { **ipv4 unicast** | **ipv4 multicast** | **ipv4 tunnel** | **ipv6 unicast** | **vpnv4 unicast** } **redistribute isis** *process-id* [**level** { **1** | **1-inter-area** | **2** }] [**metric** *metric-value*] [**route-policy** *route-policy-name*] • **address-family** { **ipv4 unicast** | **ipv4 multicast** | **ipv4 tunnel** | **ipv6 unicast** | **vpnv4 unicast** } **redistribute ospf** *process-id* [**match** { **external** [**1** | **2**] | **internal** | **nssa-external** [**1** | **2**]}] [**metric** *metric-value*] [**route-policy** *route-policy-name*]	
ステップ 7	次のいずれかを実行します。 • **address-family** { **ipv4 unicast** | **ipv4 multicast** | **ipv4 tunnel** | **ipv4 mdt** | **ipv6 unicast** | **ipv6 multicast** | **vpnv4 unicast** | **vpnv6 unicast** } **redistribute ospfv3** *process-id* [**match** { **external** [**1** | **2**] | **internal** | **nssa-external** [**1** | **2**]}] [**metric** *metric-value*] [**route-policy** *route-policy-name*] • **address-family** { **ipv4 unicast** | **ipv4 multicast** | **ipv4 tunnel** | **ipv4 mdt** | **ipv6 unicast** | **ipv6 multicast** | **vpnv4 unicast** | **vpnv6 unicast** } **redistribute rip** [**metric** *metric-value*] [**route-policy** *route-policy-name*] • **address-family** { **ipv4 unicast** | **ipv4 multicast** | **ipv4 tunnel** | **ipv4 mdt** | **ipv6 unicast** | **ipv6 multicast** | **vpnv4 unicast** | **vpnv6 unicast** } **redistribute static** [**metric** *metric-value*] [**route-policy** *route-policy-name*] • **address-family** { **ipv4 unicast** | **ipv4 multicast** | **ipv4 tunnel** | **ipv4 mdt** | **ipv6 unicast** | **ipv6 multicast** | **vpnv4 unicast** | **vpnv6 unicast** } **network** { *ip-address/prefix-length* | *ip-address mask* } [**route-policy** *route-policy-name*] • **neighbor** *ip-address* **remote-as** *as-number* **address-family** { **ipv4 unicast** | **ipv4 multicast** | **ipv4 tunnel** | **ipv4 mdt** | **ipv6 unicast** | **ipv6 multicast** | **vpnv4 unicast** | **vpnv6 unicast** } • **route-policy** *route-policy-name* { **in** | **out** }	

	コマンドまたはアクション	目的
ステップ 8	**commit**	
ステップ 9	**show bgp** [**vrf** *vrf-name*] *ip-address* 例： `RP/0/RSP0/cpu 0: router# show bgp 172.168.40.24`	コスト コミュニティを次の形式で表示します。 Cost: *POI* : *cost-community-ID* : *cost-number*

ネイバーからのソフトウェアツーストア更新の設定

ネイバーからソフトウェアツーストア更新を受信するように設定するには、次の作業を実行します。

ネイバーがルート リフレッシュに対応している場合は、soft-reconfiguration inbound コマンドによって、ルート リフレッシュ要求がネイバーに送信されるようになります。ネイバーがルート リフレッシュに対応していない場合は、ネイバーが受信ルートを再学習するようにするため、**clear bgp soft** コマンドを使用してネイバーをリセットする必要があります。BGP インバウンド ソフト リセットを使用したネイバーのリセット （170 ページ）を参照してください。

（注） ネイバーからのアップデートの保存は、ネイバーがルート リフレッシュに対応しているか、soft-reconfiguration inbound コマンドが設定されている場合にだけ機能します。ネイバーがルート リフレッシュに対応しており、**soft-reconfiguration inbound** コマンドが設定されていても、このコマンドで **always** オプションが使用されていない場合は元のルートは格納されません。元のルートはルート リフレッシュ要求によって容易に復元できます。ルート リフレッシュは、ルーティング情報を再送信するためにピアに要求を送信します。soft-reconfiguration inbound コマンドは、変更されていない形式でピアから受信したすべてのパスを保存し、クリアする際にこれらの保存されたパスを参照します。ソフト再設定はメモリに負荷がかかる処理です。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **neighbor** *ip-address*
4. **address-family** { **ipv4** | **ipv6** } **unicast**
5. **soft-reconfiguration inbound** [**always**]
6. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	

	コマンドまたはアクション	目的
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 120`	自律システム番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**neighbor** *ip-address* 例： `RP/0/RSP0/cpu 0: router(config-bgp)# neighbor 172.168.40.24`	BGP ルーティングのためにルータをネイバー コンフィギュレーション モードにして、ネイバーの IP アドレスを BGP ピアとして設定します。
ステップ 4	**address-family** { **ipv4** | **ipv6** } **unicast** 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# address-family ipv4 unicast`	IPv4 または IPv6 のいずれかのアドレス ファミリ ユニキャストを指定し、アドレスファミリのコンフィギュレーション サブモードを開始します。 このコマンドのすべてのキーワードと引数のリストを参照するには、CLI ヘルプ（?）を使用します。
ステップ 5	**soft-reconfiguration inbound** [**always]** 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr-af)# soft-reconfiguration inbound always`	指定したネイバーから受信したアップデートを格納するようにソフトウェアを設定します。ソフト再設定インバウンドを設定すると、ソフトウェアは変更またはフィルタ処理されたルートのほかに、元の変更されていないルートを格納することになります。これにより、インバウンドポリシーの変更後に「ソフト クリア」を実行できるようになります。 ソフト再設定により、ピアがルート フレッシュに対応していない場合、ソフトウェアはポリシー適用前に受信した更新を格納できます（対応している場合は更新のコピーが格納されます）。 **always** キーワードを使用すると、ルート リフレッシュがピアでサポートされている場合でも、ソフトウェアにコピーが格納されます。
ステップ 6	**commit**	

BGP パーシステンス

BGPパーシステンスにより、ローカルルータは、ネイバーセッションがダウンした後でも、設定されたネイバーから学習したルートを保持できます。BGPパーシステンスは、長期的グレースフル リスタート（LLGR）とも呼ばれます。LLGRはグレースフルリスタート（GR）が終了した後、またはGR が有効になっていない場合はただちに有効になります。LLGR は、LLGR の失効タイマーが期限切れになったとき、またはネイバーがルートを改訂した後にEnd-of-RIB マーカーを送信したときに終了します。ネイバーのLLGRが終了すると、そのネイバーからのルートのうち、失効状態のままであるルートはすべて削除されます。LLGR機能は、ネイバー

に設定されている場合、BGP OPEN メッセージでそのネイバーに通知されます。LLGR は、グレースフルリスタートとは次のように異なります。

- GR よりも長時間有効にできます。
- LLGR 失効ルートはルート選択（ベストパス計算）時の優先順位が最も低くなります。
- LLGR 失効ルートは、ベストパスとして選択されている場合に、接続されている LLGR_STALE コミュニティを使用してアドバタイズされます。LLGR に対応していないルータには、まったくアドバタイズされません。
- ネイバーへの転送パスがダウンしていることが検出された場合、LLGR 失効ルートは削除されません。
- ネイバーがルートを再度アドバタイズしない場合でも、ネイバーへの BGP セッションが複数回ダウンしても LLGR 失効ルートは削除されません。
- NO_LLGR コミュニティを持つルートは保持されません。

BGPは、ネイバーがBGPパーシステンス機能をネゴシエートするまで、コミュニティ65535:6、65535:7 を含む更新をネイバーに渡しません。コミュニティ 65535:6 と 65535:7 はそれぞれ LLGR_STALE と NO_LLGR 用に予約されていますが、リリース5.2.2 より前にこれらのコミュニティを設定している場合は、BGP の動作が予測できない場合があります。コミュニティ 65535:6 と 65535:7 は設定しないことをお勧めします。

BGP パーシステンス機能は次の AFI でのみサポートされています。

- VPNv4 と VPNv6
- RT 制約
- フロー スペック（IPv4、IPv6、VPNv4、VPNv6）
- プライベート IPv4 および IPv6（IPv4/v6 アドレスファミリ内部 VRF）

BGP 永続化設定：例

次に、BGP ネイバー3.3.3.3 で長時間グレースフルリスタート（LLGR）の失効時間を16777215 に設定する例を示します。

```
router bgp 100
 neighbor 3.3.3.3
  remote-as 30813
  update-source Loopback0
  graceful-restart stalepath-time 150
  address-family vpnv4 unicast
   long-lived-graceful-restart capable
   long-lived-graceful-restart stale-time send 16777215 accept 16777215
  !
  address-family vpnv6 unicast
   long-lived-graceful-restart capable
   long-lived-graceful-restart stale-time send 16777215 accept 16777215
```

BGP グレースフルメンテナンス

BGP リンクまたはルータがダウンすると、ネットワーク内の他のルータは、障害が発生したルータを通過していたトラフィックに代替パスがある場合は、そのパスを検索します。関係するすべてのルータが代替パスに関して一致するまでに必要な時間をコンバージェンス時間といいます。コンバージェンス時間の間に、ダウンしているルータまたはリンクに送信されるトラフィックがドロップされます。BGP グレースフルメンテナンス機能により、ルータまたはリンクが動作を停止する前に、ネットワークでコンバージェンスを実行できます。ネットワークが代替パスにトラフィックを再ルーティングする間、ルータまたはリンクはサービス状態に維持されます。影響を受けるルータまたはリンクにまだ到達していないトラフィックは、以前と同様に引き続き配信されます。すべてのトラフィックが再ルーティングされた後は、ルータまたはリンクを安全に動作を停止させることができます。

グレースフルメンテナンス機能は、代替パスが存在し、プライマリパスが取り消された時点でこれらの代替パスがルータにとって不明である場合に役立ちます。この機能は、プライマリパスが取り消される前に、これらの代替パスを提供します。この機能は、コンバージェンス時間が長いネットワークに最適です。大規模なルーティングテーブルやルートリフレクタの存在などのいくつかの要因によって、コンバージェンス時間が長くなる可能性があります。

BGP ルータまたはリンクがサービスに組み込まれると、コンバージェンス中にトラフィックが失われる可能性もありますが、ルータまたはリンクが動作を停止した場合よりも低くなります。BGP グレースフルメンテナンス機能はこのシナリオでも使用できます。

BGP グレースフルメンテナンスの制約事項

BGP グレースフルメンテナンスには、次の制約事項が適用されます。

- 影響を受けるルータが GSHUT コミュニティ属性を送信するように設定されている場合、そのルータを受信するネットワーク内の他のルータは、それを解釈するように設定する必要があります。コミュニティとルーティングポリシーを一致させ、より低い優先順位を設定する必要があります。
- LOCAL_PREF 属性は別の AS には送信されません。そのため、eBGP リンクでは LOCAL_PREF オプションを使用できません。

（注） この制約事項は、AS コンフェデレーションのメンバと AS 間の eBGP リンクには適用されません。

- 代替ルートがネットワーク内に存在している必要があります。存在していない場合は、低い優先順位をアドバタイズしても効果はありません。たとえば、代替ルートを持たないシングルホーム接続のカスタマールータのグレースフルメンテナンスを設定する利点はありません。
- 送信側ルータの出力または受信側ルータの入力のいずれかで、時間を消費するポリシーが存在する場合は、グレースフルメンテナンス動作は時間がかかることがあります。

- eBGP ASBR ネイバーを設定すると、BGP を介して直接接続されたルートに対して暗黙的ヌルラベルがアドバタイズされます。ユーザが eBGP ネイバーをシャットダウンした場合、システムの取り消しがネイバー状態の変更を書き換えるため、ラベルは再プログラミングされません。暗黙的ヌルラベル機能のサポートにより、ネイバーフラップの上書きの追加または削除の観点から、チャーンを回避するのに役立ちます。

グレースフルメンテナンスの動作

グレースフルメンテナンスがアクティブになると、影響を受けるルートは優先順位を下げて再度アドバタイズされます。そのため、隣接するルータが代替ルートを選択することになります。次のいずれかの方法を使用して、ルート優先順位の低下を通知します。

- **GSHUT コミュニティの追加**：リモートルータが優先順位を自由に設定できるようにするには、この方法を使用します。受信側ルータは、ポリシー内のこのコミュニティと一致しており、それ自体の優先順位を設定する必要があります。

- **LOCAL_PREF 値の低減**：内部 BGP ネイバーに対して機能します。リモートルータが GSHUT コミュニティと一致しない場合は、この方法を使用します。

- **AS パスを前に付加**：内部および外部の両方の BGP ネイバーに対して機能します。リモートルータが GSHUT コミュニティと一致しない場合は、この方法を使用します。

グレースフルメンテナンスが BGP 接続でアクティブになると、次の 2 つの動作が発生します。

1. 接続から受信したすべてのルートが優先順位の低い他のネイバーに再度アドバタイズされます。これは、実際に他のネイバーにアドバタイズされたルートに対してのみ実行されます。受信したルートがベストパスとして選択されていないためアドバタイズされていなかった可能性があります。この場合、再アドバタイズされません。
2. 接続にアドバタイズされたすべてのルートが優先順位の低いものから再アドバタイズされます。

最初の動作が実行されるようにするために、接続から受信したすべてのルートが graceful-shut という内部属性でタグ付けされます。この属性は、ルータにのみ内部的に分類され、BGP はアドバタイズしません。この属性は、**show bgp** コマンドを使用してルートを表示した場合に表示されます。これは GSHUT コミュニティとは異なります。GSHUT コミュニティは BGP によってアドバタイズされ、**show bgp** コマンドを使用してルートを表示したときにコミュニティリストに表示されます。

graceful-shut 属性を持つすべてのルートには、ルート選択時に最低の優先順位が与えられます。グレースフルメンテナンスでの BGP セッションで送信または受信した新しいルート更新も、前述のように処理されます。

相互自律システム

パブリックインターネット内の別の AS に低い優先順位をアドバタイズすると、遠くのネットワークで不要なルーティングアドバタイズメントが発生する場合がありますが、これは望まし

くありません。ルータが GSHUT コミニティから eBGP ネイバーへ発信するには、ネイバーアドレスファミリへの追加設定（**send-community-gshut-ebgp**）が必要です。

（注） これは、受信した時点でこのコミュニティをすでに備えているルータの GSHUT コミニュティには影響しません。このルータが GSHUT を追加したときにのみ、そのコミニュティに影響を与えます。

自動シャットダウンなし

グレースフルメンテナンス機能は、シャットダウンを実行しません。グレースフルメンテナンスが設定されている場合は、システムの再起動によっても設定されたままになります。これは、ルータまたはBGPネイバーのシャットダウンとともに使用することを目的としています。オペレータは、必要な場合は常に明示的にシャットダウンする必要があります。グレースフルメンテナンスが不要になったら、オペレータが明示的にグレースフルメンテナンスを非アクティブ化する必要があります。グレースフルメンテナンスは、シャットダウンが完了した後か、または非アクティブ化されたファシリティが再び起動した後に、非アクティブにできます。起動操作によってグレースフルメンテナンスを有効にしたままにするかどうかは、起動操作時に一時的なルーティングが問題であるかどうかによって異なります。

グレースフルメンテナンス後のシャットダウンのタイミング

グレースフルメンテナンスのアクティブ化の結果として、ネットワークが収束した後にルータまたはリンクをシャットダウンできます。コンバージェンスに 1 秒未満しかかからない場合と、1 時間以上かかる場合があります。残念ながら、単一のルータは、ネットワーク全体がいつ収束したかを認識できません。グレースフルメンテナンスのアクティブ化の後、更新の送信を開始するまでに数秒かかることがあります。また、**show bgp <vrf> <afi> <safi> summary** コマンドの出力のネイバーの「InQ」と「OutQ」は、BGP メッセージングのレベルを示しています。コンバージェンス後は、InQ と OutQ の両方が 0 になる必要があります。ネイバーはトラフィックの送信を停止させる必要があります。ただし、代替パスがない場合、トラフィックの送信が停止されることはありません。この場合、トラフィック損失を防ぐことはできません。

BGP ルータ（すべてのネイバー）でのグレースフルメンテナンスのアクティブ化

グレースフルメンテナンスを BGP ルータでアクティブにすることで、すべてのネイバーに対して **graceful-maintenance** に **activate** が設定されることになります。この 1 つの設定で、**graceful-maintenance** が設定されているすべてのネイバーに移動し、そこに **activate** を追加するのと同じ結果が得られます。キーワードの **all-neighbors** を追加し、それにより **graceful-maintenance activate all-neighbors** となった場合、ルータは、すべてのネイバーに **graceful-maintenance activate** を設定したかのように動作します。

（注） すべてのネイバーのすべてのルートに GSHUT コミュニティの送信ができる場合にのみ、BGP ルータインスタンスでグレースフルメンテナンスをアクティブにすることをお勧めします。すべてのネイバーへのすべてのルートを再送信すると、大規模なルータでは著しい時間がかかることがあります。代替ルートを持たないネイバーへの GSHUT の送信は無意味です。ルータにこのようなネイバーが多数ある場合は、それらのネイバーでグレースフルメンテナンスをアクティブにしないことによって、多くの時間を節約できます。

BGP グレースフルメンテナンス機能を使用すると、単一のネイバー、BGP セッション全体のネイバーグループ、またはすべてのネイバーで、グレースフルメンテナンスを有効にすることができます。ネイバーサブモードでグレースフルメンテナンスを有効にするには、次の 2 つの点を考慮します。

1. グレースフルシャットダウン属性を持つこのネイバーにアドバタイズされたすべてのルートは、GSHUT コミュニティを使用してそのネイバーにアドバタイズされます。
2. グレースフル メンテナンス コンフィギュレーション モードを開始して、さらに設定ができるようにします。

グレースフルメンテナンスで **activate** キーワードを使用すると、次のようになります。

1. このネイバーから受信したすべてのルートがグレースフルシャットダウン属性を取得します。
2. このネイバーにアドバタイズされたすべてのルートは、GSHUT コミュニティを使用してそのネイバーに再アドバタイズされます。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **graceful-maintenance activate** ［ **all-neighbors** | **retain-routes** ］
4. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 120`	BGP AS 番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**graceful-maintenance activate** ［ **all-neighbors**	**retain-routes** ］ 例：	ネイバーで設定されているように、g-shut コミュニティとその他の属性を持つルートをアナウンスします。これにより、ネイバーはこのルータからのルートを拒否し、代替を選択します。これにより、ルー

	コマンドまたはアクション	目的
	`RP/0/RSP0/cpu 0: router(config-bgp)# graceful-maintenance activate all-neighhbors`	タをグレースフルにするか、または非稼働状態にすることができます。 **all-neighbors** キーワードを使用した場合、グレースフルメンテナンスはアクティブ化されていないネイバーに対してもアクティブになります。**retain-routes** を選択すると、BGP プロセスが停止したときに、RIB が BGP ルートを保持するようになります。 ルータ全体ではなくBGPのみをダウンさせる必要がある場合や、ローカルBGPのメンテナンス時に隣接するルータが動作し続けることがわかっている場合は、**retain-routes** オプションを使用します。別のプロトコルまたはデフォルトルートによって提供される代替ルートがRIBにある場合は、BGPプロセスが停止した後にBGPルートを保持しないことを推奨します。
ステップ 4	**commit**	

次のタスク

グレースフルメンテナンスをアクティブにした後は、すべてのルートが送信されるのを待ってから、隣接しているネイバーが、ルータまたはメンテナンス中のリンクからトラフィックをリダイレクトするようにする必要があります。トラフィックがリダイレクトされた後は、ルータまたはリンクをサービスに復帰させても差し支えありません。すべてのルートがいつ送信されたのかを明確に知る方法はありませんが、 **show bgp summary** コマンドを使用してネイバーの OutQ を確認できます。OutQ が値 0 に達すれば、送信されるアップデートはありません。

単一のネイバーでのグレースフルメンテナンスのアクティブ化

単一のネイバーに対してグレースフルメンテナンスをアクティブにするには、次の手順を実行します。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **neighbor** *ip-address*
4. **graceful-maintenance activate**
5. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	

	コマンドまたはアクション	目的
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 120`	BGP AS 番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**neighbor** *ip-address* 例： `RP/0/RSP0/cpu 0: router(config-bgp)# neighbor 172.168.40.24`	BGP ルーティングのためにルータをネイバー コンフィギュレーション モードにして、ネイバーの IP アドレスを BGP ピアとして設定します。
ステップ 4	**graceful-maintenance activate** 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# graceful-maintenance activate`	グレースフルメンテナンス属性を持つルートをアナウンスします。
ステップ 5	**commit**	

ネイバーグループのグレースフルメンテナンスのアクティブ化

ネイバーのグループでグレースフルメンテナンスをアクティブにするには、次の手順を実行します。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **neighbor-group** *Neighbor-group name*
4. **graceful-maintenance activate**
5. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 120`	BGP AS 番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**neighbor-group** *Neighbor-group name* 例：	ルータをネイバー グループ コンフィギュレーション モードにします。

	コマンドまたはアクション	目的
	`RP/0/RSP0/cpu 0: router(config-bgp)# neighbor-group AS_1`	
ステップ 4	**graceful-maintenance activate** 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbrgrp)# graceful-maintenance activate`	グレースフルメンテナンス属性を持つルートをアナウンスします。
ステップ 5	**commit**	

次のタスク

GSHUT コミュニティを追加するには、このルータの eBGP ネイバーのネイバーアドレスファミリに、 **send-community-gshut-ebgp** コマンドを設定する必要があります。

（注） GSHUT コミュニティの送信は、eBGP ネイバーのすべてのアドレスファミリでも望ましいとは限りません。特定のアドレスファミリセットを GSHUT コミュニティの対象にするには、 **send community-gshut-ebgp** コマンドを使用します。

ルートの優先順位を下げるためのルータへの指示

BGP グレースフルメンテナンス機能は、代替パスの可用性がある場合にのみ動作します。代替ルートがリンクまたはルータを停止する前に引き継ぐことができるように、より低い優先順位のルートをアドバタイズする必要があります。ルートの優先順位を変更するには、次の手順を実行します。

（注） グレースフルメンテナンスの属性は、アウトバウンドポリシーが適用された後に、ルート更新メッセージに追加されます。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **neighbor** *ip-address*
4. **remote-as** *as-number*
5. **graceful-maintenance** **as-prepends** *value*| **local-preference** *value*

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 120`	BGP AS 番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**neighbor** *ip-address* 例： `RP/0/RSP0/cpu 0: router(config-bgp)# neighbor 172.168.40.24`	BGP ルーティングのためにルータをネイバー コンフィギュレーション モードにして、ネイバーの IP アドレスを BGP ピアとして設定します。
ステップ 4	**remote-as** *as-number* 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# remote-as 2002`	ネイバーを作成し、リモート自律システム番号を割り当てます。
ステップ 5	**graceful-maintenance as-prepends** *value*| **local-preference** *value* 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# graceful-maintenance local-preference 4`	ローカル AS 番号がルートの AS パスの先頭に追加され、ルートに指定されたローカルの優先順位の値を使用して GSHUT コミュニティをアドバタイズする回数を指定します。ルータが GSHUT コミュニティをアドバタイズするときにルートに追加するときに、LOCAL_PREF 属性も変更して、コマンドに指定されているローカル AS 番号を先頭に追加します。GSHUT を送信することで、ネイバールータが低い優先順位を処理する方法に柔軟性がもたらされます。そのため、ルートポリシーで照合したうえで、最適な処理を行うことができます。一方、単純なネットワークでは、他の場所でルートポリシーを作成するよりも、ローカルの優先順位を 0 に設定する方が簡単です。 （注） LOCAL_PREF は実際の eBGP ネイバーには送信されませんが、コンフェデレーション AS eBGP ネイバーに送信されます。eBGP ネイバーの優先順位を下げるには、as-prepends の値を入力する必要があります。

例：ルートポリシーと一致する GSHUT コミュニティを設定してルートの優先順位を下げる

```
route-policy gshut
  if community matches-any gshut then
    set local-preference 0
  endif
  pass
end-policy


neighbor 666.0.0.3
    address-family ipv4 unicast
      route-policy gshut in
```

（注） GSHUT ネイバーから受信したルートは、GSHUT 属性でマークされ、GSHUT コミュニティを使用して受信したルートと区別されます。ネイバーがメンテナンスから除外されると、そのパスの属性は削除されますが、コミュニティでは削除されません。この属性は内部的なものであり、BGP メッセージでは送信されません。パス選択時にルートを拒否するために使用されます。

ルータまたはリンクの動作の再開

ルータまたはリンクの動作を再開させる前に、グレースフルメンテナンスを最初にアクティブにしてから、**activate** 設定を削除する必要があります。

BGP グレースフルメンテナンスを確認するための show コマンドの出力

この項では、BGP グレースフルメンテナンスがアクティブになっていることを確認し、関連する属性を確認するために使用できる show コマンドを示します。

BGP グレースフルメンテナンスがアクティブになっている場合にグレースフルシャットダウンコミュニティとグレースフルシャットパスの属性を表示するには、**show bgp <IP address>** コマンドを使用します。

```
RP/0/0/CPU0:R4#show bgp 5.5.5.5
...
10.10.10.1 from 10.10.10.1 (192.168.0.5)
Received Label 24000
Origin incomplete, metric 0, localpref 100, valid, internal, best, group-best,
import-candidate
Received Path ID 0, Local Path ID 1, version 4
Community: graceful-shutdown
Originator: 192.168.0.5, Cluster list: 192.168.0.1
```

次の **show bgp community graceful-shutdown** コマンドの出力例には、グレースフル メンテナンス機能が表示されています。

```
RP/0/0/CPU0:R4#show bgp community graceful-shutdown
BGP router identifier 192.168.0.4, local AS number 4
BGP generic scan interval 60 secs
BGP table state: Active
```

```
Table ID: 0xe0000000 RD version: 18
BGP main routing table version 18
BGP scan interval 60 secs
Status codes: s suppressed, d damped, h history, * valid, > best
i - internal, r RIB-failure, S stale, N Nexthop-discard
Origin codes: i - IGP, e - EGP, ? - incomplete
Network Next Hop Metric LocPrf Weight Path
* 5.5.5.5/32 10.10.10.1 88 0 1 ?
Processed 1 prefixes, 1 paths
```

次に、グレースフルメンテナンス機能の属性を表示するために、IPアドレスと設定引数およびキーワードを指定した **show bgp neighbors** コマンドの出力例を示します

```
RP/0/0/CPU0:R1#show bgp neighbor 12.12.12.5
...
Graceful Maintenance locally active, Local Pref=45, AS prepends=3
...
For Address Family: IPv4 Unicast
...
GSHUT Community attribute sent to this neighbor
...
********************************************************************************
RP/0/0/CPU0:R1#show bgp neighbor 12.12.12.5 configuration
neighbor 12.12.12.5
remote-as 1 []
graceful-maintenance 1 []
gr-maint local-preference 45 []
gr-maint as-prepends 3 []
gr-maint activate []
```

次に、グレースフルメンテナンス機能の属性が表示される **show rpl community-set** コマンドの出力例を示します。

```
RP/0/0/CPU0:R5#show rpl community-set
Listing for all Community Set objects
community-set gshut
graceful-shutdown
end-set
```

次に、グレースフルメンテナンスがアクティブになっている BGP ネイバーが起動したときに発行されるsyslogの例を示します。これは、コンバージェンス後にグレースフルメンテナンスを非アクティブ化するように通知する警告テキストです。

```
RP/0/0/CPU0:Jan 28 22:01:36.356 : bgp[1056]: %ROUTING-BGP-5-ADJCHANGE : neighbor 10.10.10.4
 Up (VRF: default) (AS: 4)
WARNING: Graceful Maintenance is Active
```

L3VPN iBGP PE-CE

L3VPN iBGP PE-CE 機能は、プロバイダー エッジ（PE）デバイスとカスタマー エッジ（CE）デバイス間でBGPルーティング情報を交換するiBGP（内部Border Gateway Protocol）セッションの確立に役立ちます。2 つの BGP ピア間の BGP セッションは、それらの BGP ピアが同じ自律システム内に存在する場合に、iBGP セッションと呼ばれます。

L3VPN iBGP PE-CE の概要

プロバイダーエッジ（PE）またはカスタマーエッジ（CE）のルーティングプロトコルとしてBGPを使用すると、VPNプロバイダー自律システム（AS）とカスタマー ネットワーク自律シ

ステム間の外部ピアリングとしてピエアリングセッションが設定されます。L3VPN iBGP PE-CE 機能では、PE デバイスと CE デバイスが、PE と CE 間で広く使用されている外部 BGP ピアリングの代わりに内部 ボーダー ゲートウェイ プロトコル（iBGP）としてピアリングを行って Border Gateway Protocol（BGP）ルーティング情報を交換できます。このメカニズムは、VRF ベースの CE が iBGP として設定されている各 PE デバイスで適用されます。これにより、サービス プロバイダー（SP）は、CE に自律システムのオーバーライドを設定する必要がなくなります。この機能を有効にした場合は、異なる自律システムを使用した仮想プライベート ネットワーク（VPN）サイトの設定は不要です。

neighbor internal-vpn-client コマンドを使用すると、PE デバイスが VPN クラウド全体を CE デバイスに対して内部 VPN クライアントとして動作させることができます。これらの CE デバイスは、VRF 内部の iBGP PE-CE 接続を通じて VPN クラウドに内部的に接続されます。この接続が確立されると、PE デバイスは CE-learned パスを ATTR_SET という属性内にカプセル化し、それを VPN コアからリモートの PE デバイスまで iBGP-sourced パスで伝送します。リモートの PE デバイスでは、この属性に個別の属性が割り当てられ、送信元 CE パスが抽出されてリモート CE デバイスに送信されます。

ATTR_SET はオプションの遷移属性で、受け取った CE パス属性を伝送します。ATTR_SET 属性は、次のように BGP 更新メッセージ内にエンコードされます。

```
+-------------------------------+
| Attr Flags (O|T) Code = 128  |
+-------------------------------+
| Attr. Length (1 or 2 octets) |
+-------------------------------+
| Origin AS (4 octets)         |
+-------------------------------+
| Path attributes (variable)   |
+-------------------------------+
```

Origin AS は、ATTR_SET が生成される VPN カスタマーの AS です。ATTR_SET の最小長は 4 バイト、最大長は BGP 更新メッセージの必須フィールドと属性を考慮した後のパス属性でサポートされる最大値です。最大長は 3,500 バイトまでにすることをお勧めします。ATTR_SET には、属性の MP_REACH、MP_UNREACH、NEW_AS_PATH、NEW_AGGR、NEXT_HOP、および ATTR_SET 自体を含めること（ATTR_SET 内に ATTR_SET）はできません。ATTR_SET の中にこれらの属性が見つかった場合、ATTR_SET は無効と見なされ、対応するエラー処理メカニズムが呼び出されます。

L3VPN iBGP PE-CE の制限

次に、L3VPN iBGP PE-CE の設定に適用される制限を示します。

- IBGP PE CE 機能を切り替えてネイバーが route-refresh または soft-reconfiguration inbound をサポートしなくなった場合は、手動のセッション フラップを実行して変更を確認する必要があります。これが発生した場合は、次のメッセージが表示されます。

```
RP/0/0/CPU0: %ROUTING-BGP-5-CFG_CHG_RESET: Internal VPN client configuration change
 on neighbor 10.10.10.1 requires HARD reset
(clear bgp 10.10.10.1) to take effect.
```

- iBGP PE CE CLI 設定は、ネイバー/セッション グループを除き、デフォルト VRF のピアには使用できません。

- この機能は、通常の VPN クライアント（eBGP VPN クライアント）上では動作しません。
- ATTR_SET 内にパックされた属性は、iBGP CE 上の inbound route-policy で加えられた変更を反映し、指定した VRF の export route-policy で加えられた変更は反映しません。
- iBGP PE-CE ピアリング セッションで設定された同じ VPN の異なる VRF（つまり、異なる PE ルータ内）は、それぞれの VRF で異なるルート識別子（RD）を使用する必要があります。iBGP PE CE 機能は、RD 値が入力 VRF と出力 VRFで同じである場合は機能しません。

L3VPN iBGP PE-CE の設定

L3VPN iBGP PE-CE は、ネイバー、ネイバー グループ、またはセッション グループで有効にすることができます。L3VPN iBGP PE-CE を設定するには、次のステップを実行します。

始める前に

CE は、内部 BGP ピアである必要があります。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **vrf** *vrf-name*
4. **neighbor** *ip-address* **internal-vpn-client**
5. **commit**
6. **show bgp vrf** *vrf-name* **neighbors** *ip-address*
7. **show bgp** {**vpnv4** | **vpnv6** } **unicast rd**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 120`	自律システム番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**vrf** *vrf-name* 例： `RP/0/RSP0/cpu 0: router(config-bgp)# vrf blue`	VRF インスタンスを設定します。
ステップ 4	**neighbor** *ip-address* **internal-vpn-client** 例：	ルーティング情報を交換する相手の CE ネイバー デバイスを設定します。**neighbor internal-vpn-client** コ

	コマンドまたはアクション	目的
	`RP/0/RSP0/cpu 0: router(config-bgp-vrf)# neighbor 10.0.0.0 internal-vpn-client`	マンドは VPN 属性セット内の iBGP-CE ネイバー パスをスタックします。
ステップ 5	**commit**	
ステップ 6	**show bgp vrf** *vrf-name* **neighbors** *ip-address*	VRF CE ピアの iBGP PE-CE 機能が有効かどうかが表示されます。
ステップ 7	**show bgp** {**vpnv4** | **vpnv6** } **unicast rd**	L3VPN iBGP PE-CE が CE 上で有効になっている場合は、コマンドの出力に ATTR_SET 属性が表示されます。

例

例：L3VPN iBGP PE-CE の設定

次の例は、L3VPN iBGP PE-CE の設定方法を示しています。

```
R1(config-bgp-vrf-nbr)#neighbor 10.10.10.1 ?
. . .
 internal-vpn-client     Preserve iBGP CE neighbor path in ATTR_SET across VPN core
. . .
R1(config-bgp-vrf-nbr)#neighbor 10.10.10.1 internal-vpn-client
router bgp 65001
 bgp router-id 100.100.100.2
 address-family ipv4 unicast
address-family vpnv4 unicast
 !
 vrf ce-ibgp
  rd 65001:100
  address-family ipv4 unicast
  !
  neighbor 10.10.10.1
   remote-as 65001
   internal-vpn-client
```

次に、L3VPN iBGP PE-CE が CE ピアで有効になっている場合の **show bgp vrf** *vrf-name* **neighbors** *ip-address* コマンドの出力例を示します。

```
R1#show bgp vrf ce-ibgp neighbors 10.10.10.1
BGP neighbor is 10.10.10.1, vrf ce-ibgp
 Remote AS 65001, local AS 65001, internal link
 Remote router ID 100.100.100.1
  BGP state = Established, up for 00:00:19
  . . .
 Multi-protocol capability received
  Neighbor capabilities:
    Route refresh: advertised (old + new) and received (old + new)
    4-byte AS: advertised and received
    Address family IPv4 Unicast: advertised and received
CE attributes will be preserved across the core
  Received 2 messages, 0 notifications, 0 in queue
  Sent 2 messages, 0 notifications, 0 in queue
  . . .
```

次に、L3VPN iBGP PE-CE が CE ピアで有効になっている場合の **show bgp vpn4/vpn6 unicast rd** コマンドの出力例を示します。

```
BGP routing table entry for 1.1.1.0/24, Route Distinguisher: 200:300
Versions:
  Process           bRIB/RIB  SendTblVer
  Speaker                 10          10
Last Modified: Aug 28 13:11:17.000 for 00:01:00
Paths: (1 available, best #1)
  Advertised to update-groups (with more than one peer):
    0.2
Path #1: Received by speaker 0
  Advertised to update-groups (with more than one peer):
    0.2
  Local, (Received from a RR-client)
    20.20.20.2 from 20.20.20.2 (100.100.100.2)
      Received Label 24000
      Origin IGP, localpref 100, valid, internal, best, group-best, import-candidate,
      not-in-vrf Received Path ID 0, Local Path ID 1, version 10
      Extended community: RT:228:237
     ATTR-SET [
         Origin-AS: 200
         AS-Path: 51320 52325 59744 12947 21969 50346 18204 36304 41213
 23906 33646
         Origin: incomplete
         Metric: 204
         Local-Pref: 234
         Aggregator: 304 34.3.3.3
         Atomic Aggregator
        Community: 1:60042 2:41661 3:47008 4:9280 5:39778 6:1069 7:15918
 8:8994 9:52701
10:10268 11:26276 12:8506 13:7131 14:65464 15:14304 16:33615 17:54991
18:40149 19:19401
         Extended community: RT:100:1 RT:1.1.1.1:1]
```

フロー タグの伝達

フロー タグ伝達機能では、ルート ポリシーとユーザ ポリシー間に相関関係を構築できます。BGP を使用したフロー タグ伝達では、AS 番号、プレフィックス リスト、コミュニティ文字列、および拡張コミュニティなどのルーティング属性に基づいてユーザ側でトラフィックをステアリングできます。フロー タグは論理数値識別子で、FIB ルックアップ テーブル内の FIB エントリのルーティング属性の 1 つとして RIB を通じて配布されます。フロー タグは、RPL からの「set」操作を使用してインスタンス化され、フロー タグ値に対してアクション（ポリシールール）が関連付けられている C3PL PBR ポリシーで参照されます。

フロー タグの伝達は次の場合に使用できます。

- 宛先 IP アドレス（コミュニティ番号を使用）またはプレフィックス（コミュニティ番号または AS 番号を使用）に基づいてトラフィックを分類する。
- カスタマー サイトのサービス レベル契約（SLA）に基づくサービス エッジに到達するパスのコストに合致する TE グループを選択する。

- SLA とそのクライアントに基づいて、特定のカスタマーにトラフィック ポリシー（TE グループの選択）を適用する。
- アプリケーション サーバまたはキャッシュ サーバにトラフィックを迂回させる。

フロータグ伝達のコマンドの詳細については、 *Routing Command Reference for Cisco ASR 9000 Series Routers*の「BGP Commands」のモジュールを参照してください。

フロー タグ伝達の制限

Border Gateway Protocol を使用した QoS ポリシー伝達（QPPB）とフロー タグ機能の ASR9K プラットフォームでの併用については、いくつかの制約事項があります。次の作業を行います。

- ルート ポリシーには、「set qos-group」または「set flow-tag」のいずれかを使用できますが、prefix-set に両方は使用できません。
- qos-group と route policy flow-tag のルート ポリシーに重複するルートは使用できません。QPPB とフロー タグの機能は、それらが使用するルート ポリシーに重複するルートがない場合に限り、（同じインターフェイス上でも、異なるインターフェイス上でも）共存できます。
- ルート ポリシーとポリシー マップに qos-group と flow-tag を混在させて使用することはお勧めしません。

ソース ベースと宛先ベースのフロー タグ

ソースベースのフローのタグ機能では、着信パケットの発信元アドレスに割り当てられているフロータグに基づいてパケットを照合できます。一致した場合は、このポリシーでサポートされている PBR アクションを適用できます。

送信元と送信先ベースのフロー タグの設定

指定したインターフェイスにフロータグを適用するには、このタスクを実行します。パケットは、着信パケットの発信元アドレスに割り当てられているフロー タグに基づいて照合されます。

（注） インターフェイスでQPPBとフロータグ機能の両方を同時にイネーブルにすることはできません。

手順の概要

1. **configure**
2. **interface** *type interface-path-id*
3. **ipv4 | ipv6 bgp policy propagation input flow-tag**{**destination** | **source**}
4. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**interface** *type interface-path-id* 例： `RP/0/RSP0/cpu 0: router(config-if)# interface GigabitEthernet 0/0/0/0`	インターフェイス コンフィギュレーション モードを開始して、1 つ以上のインターフェイスを VRF に関連付けます。
ステップ 3	**ipv4 | ipv6 bgp policy propagation input flow-tag{destination | source}** 例： `RP/0/RSP0/cpu 0: router(config-if)# ipv4 bgp policy propagation input flow-tag source`	送信元または送信先の IP アドレスのフロー タグ ポリシーの伝達をインターフェイスで有効にします。
ステップ 4	**commit**	

例

次の show コマンドは、ルータに適用された RBP ポリシーを使用して出力を表示します。

```
show running-config interface gigabitEthernet 0/0/0/12
Thu Feb 12 01:51:37.820 UTC
interface GigabitEthernet0/0/0/12
 service-policy type pbr input flowMatchPolicy
 ipv4 bgp policy propagation input flow-tag source
 ipv4 address 192.5.1.2 255.255.255.0
!

RP/0/RSP0/CPU0:ASR9K-0#show running-config policy-map type pbr flowMatchPolicy
Thu Feb 12 01:51:45.776 UTC
policy-map type pbr flowMatchPolicy
 class type traffic flowMatch36
  transmit
 !
 class type traffic flowMatch38
  transmit
 !
 class type traffic class-default
 !
 end-policy-map
!

RP/0/RSP0/CPU0:ASR9K-0#show running-config class-map type traffic flowMatch36
Thu Feb 12 01:52:04.838 UTC
class-map type traffic match-any flowMatch36
 match flow-tag 36
 end-class-map
!
```

BGP での VPN ルーティングおよび転送インスタンスの設定

機能を設定するライン カード スロットに使用可能なレイヤ 3 VPN ライセンスがある場合に限り、レイヤ 3（仮想プライベート ネットワーク）を設定できます。拡張 IP ライセンスが有効になっている場合、インターフェイスで 4096 レイヤ 3 VPN ルーティングおよび転送インスタンス（VRF）を設定できます。インフラストラクチャ VRF のライセンスが有効な場合は、8 つのレイヤ 3 VRF をラインカードに設定できます。

拡張 IP ライセンスの詳細については、*System Management Configuration Guide for Cisco ASR 9000 Series Routers*の「Software Entitlement on Cisco IOS XR Software」のモジュールを参照してください。

適切なライセンスが有効になっていないと、次のエラー メッセージが表示されます。

```
RP/0/RSP0/cpu 0: router#LC/0/0/CPU0:Dec 15 17:57:53.653 : rsi_agent[247]:
%LICENSE-ASR9K_LICENSE-2-INFRA_VRF_NEEDED : 5 VRF(s) are configured without license
A9K-iVRF-LIC in violation of the Software Right To Use Agreement.
This feature may be disabled by the system without the appropriate license.
Contact Cisco to purchase the license immediately to avoid potential service interruption.
```

（注） L2VPN サービスの設定に AIP ライセンスは必要ありません。

次の作業は、BGP に VPN ルーティングおよび転送（VRF）インスタンスを設定する場合に実行します。

プロバイダーエッジ ルータでの仮想ルーティングおよび転送テーブルの定義

プロバイダー エッジ（PE）ルータに VPN ルーティングおよび転送（VRF）テーブルを定義するには、次の作業を実行します。

手順の概要

1. **configure**
2. **vrf** *vrf-name*
3. **address-family** { **ipv4** | **ipv6** } **unicast**
4. **maximum prefix** *maximum* [*threshold*]
5. **import route-policy** *policy-name*
6. **import route-target** [*as-number* **:** *nn* | *ip-address* **:** *nn*]
7. **export route-policy** *policy-name*
8. **export route-target** [*as-number* **:** *nn* | *ip-address* **:** *nn*]
9. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	

	コマンドまたはアクション	目的
ステップ 2	**vrf** *vrf-name* 例： `RP/0/RSP0/cpu 0: router(config)# vrf vrf_pe`	VRF インスタンスを設定します。
ステップ 3	**address-family** { **ipv4** | **ipv6** } **unicast** 例： `RP/0/RSP0/cpu 0: router(config-vrf)# address-family ipv4 unicast`	IPv4 または IPv6 のいずれかのアドレス ファミリを指定し、アドレスファミリのコンフィギュレーション サブモードを開始します。 このコマンドのすべてのキーワードと引数のリストを参照するには、CLI ヘルプ（?）を使用します。
ステップ 4	**maximum prefix** *maximum* [*threshold*] 例： `RP/0/RSP0/cpu 0: router(config-vrf-af)# maximum prefix 2300`	VRF テーブルで許可するプレフィックスの数に制限を設定します。 ルートの最大数はダイナミック ルーティング プロトコルと、スタティックまたは接続されたルートに適用されます。 *mid-threshold* 引数を使用して、プレフィックスを制限するしきい値のパーセンテージを指定できます。
ステップ 5	**import route-policy** *policy-name* 例： `RP/0/RSP0/cpu 0: router(config-vrf-af)# import route-policy policy_a`	（任意）VRF にインポートする内容をより細かく制御します。このインポート フィルタでは、指定された *policy-name* 引数に一致しないプレフィックスは破棄されます。
ステップ 6	**import route-target** [*as-number* **:** *nn* | *ip-address* **:** *nn*] 例： `RP/0/RSP0/cpu 0: router(config-vrf-af)# import route-target 234:222`	ルート ターゲット（RT）拡張コミュニティのリストを指定します。指定されたインポート ルート ターゲット拡張コミュニティと関連付けられているプレフィックスだけが VRF にインポートされます。
ステップ 7	**export route-policy** *policy-name* 例： `RP/0/RSP0/cpu 0: router(config-vrf-af)# export route-policy policy_b`	（任意）VRF にエクスポートする内容をより細かく制御します。このエクスポート フィルタでは、指定された *policy-name* 引数に一致しないプレフィックスは破棄されます。
ステップ 8	**export route-target** [*as-number* **:** *nn* | *ip-address* **:** *nn*] 例： `RP/0/RSP0/cpu 0: routerr(config-vrf-af)# export route-target 123;234`	ルート ターゲット拡張コミュニティのリストを指定します。エクスポート ルート ターゲット コミュニティは、リモート PE にアドバタイズされる際にプレフィックスと関連付けられます。リモート PE は、これらのエクスポート ルート ターゲット コミュニティと一致するインポート RT を持つ VRF に、これらのプレフィックスをインポートします。

	コマンドまたはアクション	目的
ステップ 9	**commit**	

ルート識別子の設定

ルート識別子（RD）により、複数の VPN ルーティングおよび転送（VRF）インスタンスにおいてプレフィックスが固有になります。

L3VPN マルチパス同一ルート識別子（RD）環境では、プレフィックスを RIB にインストールするかどうかは、プレフィックスの最適パスに基づいて決まります。稀に設定が誤っている場合（最適パスが RIB にインストールできる有効なパスではない場合）、BGP はプレフィックスをドロップし、その他のパスを考慮しません。この動作は RD のセットアップによって異なります。最適マルチパスが RIB にインストールするパスとして無効な場合には、非最適マルチパスがインストールされます。

RD を設定するには、次の作業を実行します。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **bgp router-id** *ip-address*
4. **vrf** *vrf-name*
5. **rd** { *as-number* **:** *nn* | *ip-address* **:** *nn* | **auto** }
6. 次のいずれかを実行します。
   - **end**
   - **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 120`	BGP コンフィギュレーション モードを開始します。このモードでは BGP ルーティング プロセスを設定できます。
ステップ 3	**bgp router-id** *ip-address* 例： `RP/0/RSP0/cpu 0: router(config-bgp)# bgp router-id 10.0.0.0`	BGP スピーキング ルータの固定ルータ ID を設定します。
ステップ 4	**vrf** *vrf-name* 例：	VRF インスタンスを設定します。

	コマンドまたはアクション	目的
	`RP/0/RSP0/cpu 0: router(config-bgp)# vrf vrf_pe`	
ステップ 5	**rd** { *as-number* **:** *nn* | *ip-address* **:** *nn* | **auto** } **例**： `RP/0/RSP0/cpu 0: router(config-bgp-vrf)# rd 345:567`	ルート識別子を設定します。 ルータが自動的に一意の RD を VRF に割り当てるようにする場合は、 **auto** キーワードを使用します。 ルータ コンフィギュレーション モードで **bgp router-id** コマンドを使用してルータ ID が設定されている場合にのみ、RD を自動で割り当てることができます。これにより、自動 RD 生成に使用できるグローバルで固有のルータ ID を設定できます。VRF のルータ ID はグローバルで固有である必要はありません。また、自動 RD 生成で VRF ルータ ID を使用することは正しくありません。ルータ ID を 1 つにすると、いつ再起動してもルータ ID が固定であるため、BGP グレースフル リスタートで RD 情報のチェックポイントも行いやすくなります。

ステップ 6	次のいずれかを実行します。
	• **end**
	• **commit**
	例：
	`RP/0/RSP0/cpu 0: router(config-bgp-vrf)# end`
	または
	`RP/0/RSP0/cpu 0: router(config-bgp-vrf)# commit`
		設定変更を保存します。
		• **end** コマンドを実行すると、変更をコミットするように要求されます。
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)?[cancel]:`
		• **yes** を入力すると、実行コンフィギュレーション ファイルに変更が保存され、コンフィギュレーション セッションが終了して、ルータが EXEC コンフィギュレーション モードに戻ります。
		• **no** を入力すると、コンフィギュレーション セッションが終了して、ルータが EXEC コンフィギュレーション モードに戻ります。変更はコミットされません。
		• **cancel** と入力すると、現在のコンフィギュレーションセッションが継続します。コンフィギュレーション セッションは終了せず、設定変更もコミットされません。
		• 実行コンフィギュレーションファイルに変更を保存し、コンフィギュレーションセッションを継続するには、**commit** コマンドを使用します。

PE-PE または PE-RR 内部 BGP セッションの設定

BGP がプロバイダー エッジ（PE）ルータ間で VPN 到着達可能性情報を送信できるようにするには、PE-PE 内部 BGP（iBGP）セッションを設定する必要があります。PE はリモート PE ルータから送信される VPN 情報を使用して VPN 接続と使用するラベル値を判別します。これにより、リモート（出力）ルータはパケット転送で正しい VPN へのパケットを逆多重化できます。

PE ルータで設定されている VPN に接続するすべての PE および RR ルータに対して PE-PE、PE ルート リフレクタ（RR）iBGP セッションが定義されます。

PE-PE iBGP セッションを設定し、PE でグローバル VPN オプションを設定するには、次の作業を実行します。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **address-family vpnv4 unicast**
4. **exit**
5. **neighbor** *ip-address*
6. **remote-as** *as-number*
7. **description** *text*
8. **password** { **clear** | **encrypted** } *password*
9. **shutdown**
10. **timers** *keepalive hold-time*
11. **update-source** *type interface-id*
12. **address-family vpnv4 unicast**
13. **route-policy** *route-policy-name* **in**
14. **route-policy** *route-policy-name* **out**
15. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 120`	自律システム番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**address-family vpnv4 unicast** 例： `RP/0/RSP0/cpu 0: router(config-bgp)# address-family vpvn4 unicast`	VPN アドレス ファミリ コンフィギュレーション モードを開始します。

	コマンドまたはアクション	目的
ステップ 4	**exit** 例： `RP/0/RSP0/cpu 0: router(config-bgp-af)# exit`	現在のコンフィギュレーション モードを終了します。
ステップ 5	**neighbor** *ip-address* 例： `RP/0/RSP0/cpu 0: router(config-bgp)# neighbor 172.16.1.1`	PE の iBGP ネイバーを設定します。
ステップ 6	**remote-as** *as-number* 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# remote-as 1`	ネイバーをリモート自律システム番号に割り当てます。
ステップ 7	**description** *text* 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# description neighbor 172.16.1.1`	（任意）ネイバーの説明を指定します。description は、コメントを保存するために使用されます。ソフトウェアの機能には影響しません。
ステップ 8	**password** { **clear** | **encrypted** } *password* 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# password encrypted 123abc`	2 つの BGP ネイバーの間の TCP 接続上で Message Digest 5（MD5）認証をイネーブルにします。
ステップ 9	**shutdown** 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# shutdown`	指定されたネイバーのあらゆるアクティブセッションを終了し、すべての関連するルーティング情報を削除します。
ステップ 10	**timers** *keepalive hold-time* 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# timers 12000 200`	BGP ネイバーのタイマーを設定します。
ステップ 11	**update-source** *type interface-id* 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# update-source gigabitEthernet 0/1/5/0`	ネイバーとの iBGP セッションを形成するときに、iBGP セッションが特定のインターフェイスのプライマリ IP アドレスをローカル アドレスとして使用できるようにします。

	コマンドまたはアクション	目的
ステップ 12	**address-family vpnv4 unicast** 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# address-family vpvn4 unicast`	VPN ネイバー アドレス ファミリ設定モードを開始します。
ステップ 13	**route-policy** *route-policy-name* **in** 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr-af)# route-policy pe-pe-vpn-in in`	着信ルートのルーティング ポリシーを指定します。ポリシーを使用すると、ルートのフィルタリングやルート属性の変更ができます。
ステップ 14	**route-policy** *route-policy-name* **out** 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr-af)# route-policy pe-pe-vpn-out out`	発信ルートのルーティング ポリシーを指定します。ポリシーを使用すると、ルートのフィルタリングやルート属性の変更ができます。
ステップ 15	**commit**	

RTコミュニティの定義済みセットがあるルートを保持するためのルートリフレクタの設定

プロバイダーエッジ（PE）は、設定されている VPN のインポート ルート ターゲット（RT）に一致するルートを保持している必要があります。PE ルータは、他の VPNv4ルートをすべて破棄できます。ただし、ルートリフレクタ（RR）はすべての VPNv4 ルートを維持する必要があります。これは、RR は PE ルータとピアになる可能性があり、別の PE が別の RT タグ付き VPNv4 ルートを要求する（RR をスケーラブルにしない）場合があるためです。RR は RT コミュニティの定義済みのセットを持つルートだけを保持するように設定できます。また、一部の RR は、別の VPN セットを提供するように設定することもできます（これによりスケーラビリティが高まります）。PE で設定された VRF にサービスを提供するすべての RR とピアになるように PE を設定します。PE がまだルートを保持していない RT を使用して、新しい VRF を設定すると、この PE は RR に対してルート リフレッシュ要求を発行し、関連する VPN ルートを取得します。

（注） PE-RR のセッションで拡張コミュニティの Outbound Route Filter（ORF）をサポートしている場合には、このプロセスの効率が高まる場合があることに注意してください。

特定の RT でタグ付けされたルートを保持するようにリフレクタを設定するには、次のタスクを実行します。

手順の概要

1. **configure**

2. **router bgp** *as-number*
3. **address-family vpnv4 unicast**
4. **retain route-target** { **all** | **route-policy** *route-policy-name* }
5. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 120`	自律システム番号を指定し、BGP コンフィギュレーションモードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**address-family vpnv4 unicast** 例： `RP/0/RSP0/cpu 0: router(config-bgp)# address-family vpvn4 unicast`	VPN アドレス ファミリ コンフィギュレーション モードを開始します。
ステップ 4	**retain route-target** { **all** | **route-policy** *route-policy-name* } 例： `RP/0/RSP0/cpu 0: router(config-bgp-af)# retain route-target route-policy rr_ext-comm`	特定の RT でタグ付けされたルートを保持するようにリフレクタを設定します。 *route-policy-name* 引数には、RR がパスを保持するためにそのパスに含まれている必要がある拡張コミュニティをリストするポリシー名を指定します。 （注） これがルートリフレクタのデフォルトの動作であるため、 **all** キーワードは不要です。
ステップ 5	**commit**	

PE-CE プロトコルとしての BGP の設定

PE で BGP を設定し、BGP を使用した PE-CE 通信を確立するには、次のタスクを実行します。このタスクは、VRF と VRF 以外の両方の設定で実行できます。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **vrf** *vrf-name*
4. **bgp router-id** *ip-address*
5. **label mode per-ce**
6. **address-family** { **ipv4** | **ipv6** } **unicast**
7. **network** { *ip-address* / *prefix-length* | *ip-address mask* }

8. **aggregate-address** *address* / *mask-length*
9. **exit**
10. **neighbor** *ip-address*
11. **remote-as** *as-number*
12. **password** { **clear** | **encrypted** } *password*
13. **ebgp-multihop** [*ttl-value*]
14. 次のいずれかを実行します。
    - **address-family** { **ipv4** | **ipv6** } **unicast**
    - **address-family {ipv4 {unicast | labeled-unicast} | ipv6 unicast}**
15. **site-of-origin** [*as-number* **:** *nn* | *ip-address* **:** *nn*]
16. **as-override**
17. **allowas-in** [*as-occurrence-number*]
18. **route-policy** *route-policy-name* **in**
19. **route-policy** *route-policy-name* **out**
20. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 120`	自律システム番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**vrf** *vrf-name* 例： `RP/0/RSP0/cpu 0: router(config-bgp)# vrf vrf_pe_2`	PE ルータで特定の VRF の BGP ルーティングをイネーブルにします。
ステップ 4	**bgp router-id** *ip-address* 例： `RP/0/RSP0/cpu 0: router(config-bgp-vrf)# bgp router-id 172.16.9.9`	BGP スピーキング ルータの固定ルータ ID を設定します。
ステップ 5	**label mode per-ce** 例： `RP/0/RSP0/cpu 0: router(config-bgp-vrf)# label mode per-ce`	• CE 単位のラベルモードを設定して PE ルータでの追加ルックアップを回避し、ラベル スペースを節約します（デフォルトのラベルモードはプレフィックス単位です）。このモードでは、PE ルータは、すべての即時ネクスト ホップ（ほとんどの場合、これは CE ルータ）に 1 個のラベルを割り当てます。このラベルはネクスト ホップに直接マップされるため、データ転送中に VRF ルート ルックアップが実行される

	コマンドまたはアクション	目的
		ことはありません。ただし、割り当てられるラベルの数は、各 VRF に 1 つではなく、各 CE に 1 個です。BGP はすべてのネクスト ホップを認識するため、各ネクスト ホップにラベルを割り当てます（各 PE-CE インターフェイスではありません）。発信インターフェイスがマルチアクセス インターフェイスで、ネイバーのメディア アクセス コントロール（MAC）アドレスが不明な場合は、アドレス解決プロトコル（ARP）がパケット転送の間にトリガーされます。 • **per-vrf** キーワードは、一意の VRF からアドバタイズされたすべてのルートに同じラベルを使用するように設定します。
ステップ 6	**address-family** { **ipv4** | **ipv6** } **unicast** 例： `RP/0/RSP0/cpu 0: router(config-vrf)# address-family ipv4 unicast`	IPv4 または IPv6 のいずれかのアドレス ファミリ ユニキャストを指定し、アドレス ファミリのコンフィギュレーション サブモードを開始します。 このコマンドのすべてのキーワードと引数のリストを参照するには、CLI ヘルプ（?）を使用します。
ステップ 7	**network** { *ip-address* / *prefix-length* | *ip-address mask* } 例： `RP/0/RSP0/cpu 0: router(config-bgp-vrf-af)# network 172.16.5.5/24`	VRF のコンテキストでアドレス ファミリのテーブルのネットワーク プレフィックスを発信します。
ステップ 8	**aggregate-address** *address* / *mask-length* 例： `RP/0/RSP0/cpu 0: router(config-bgp-vrf-af)# aggregate-address 10.0.0.0/24`	コアに保持されている状態を削減するためルーティング情報を集約するように VRF アドレス ファミリ コンテキストで集約を設定します。この集約により、PE エッジでの効率がいくらか低下します。これはパケットの最終ネクスト ホップを決定するために、さらにルックアップが必要になるためです。設定すると、一連のコンポーネント プレフィックスの代わりにサマリー プレフィックスがアドバタイズされます。これはより詳細な集約です。PE は集約のラベルを 1 つだけアドバタイズします。コンポーネント プレフィックスでは CE へのネクスト ホップが異なることがあるため、データ転送時に追加のルックアップを実行する必要があります。

	コマンドまたはアクション	目的
ステップ 9	**exit** 例： `RP/0/RSP0/cpu 0: router(config-bgp-vrf-af)# exit`	現在のコンフィギュレーション モードを終了します。
ステップ 10	**neighbor** *ip-address* 例： `RP/0/RSP0/cpu 0: router(config-bgp-vrf)# neighbor 10.0.0.0`	CE ネイバーを設定します。 *ip-address* 引数は、プライベート アドレスにする必要があります。
ステップ 11	**remote-as** *as-number* 例： `RP/0/RSP0/cpu 0: router(config-bgp-vrf-nbr)# remote-as 2`	CE ネイバーのリモート AS を設定します。
ステップ 12	**password** { **clear** | **encrypted** } *password* 例： `RP/0/RSP0/cpu 0: router(config-bgp-vrf-nbr)# password encrypted 234xyz`	2 つの BGP ネイバー間の TCP 接続で Message Digest 5（MD5）認証をイネーブルにします。
ステップ 13	**ebgp-multihop** [*ttl-value*] 例： `RP/0/RSP0/cpu 0: router(config-bgp-vrf-nbr)# ebgp-multihop 55`	直接接続していないネットワーク上の外部ピアへの BGP 接続を受け入れて試行するように CE ネイバーを設定します。
ステップ 14	次のいずれかを実行します。 • **address-family** { **ipv4** | **ipv6** } **unicast** • **address-family {ipv4 {unicast | labeled-unicast} | ipv6 unicast}** 例： `RP/0/RSP0/cpu 0: router(config-vrf)# address-family ipv4 unicast`	IPv4 または IPv6 のいずれかのアドレス ファミリ ユニキャストを指定し、アドレス ファミリのコンフィギュレーション サブモードを開始します。 このコマンドのすべてのキーワードと引数のリストを参照するには、CLI ヘルプ（?）を使用します。
ステップ 15	**site-of-origin** [*as-number* **:** *nn* | *ip-address* **:** *nn*] 例： `RP/0/RSP0/cpu 0: router(config-bgp-vrf-nbr-af)# site-of-origin 234:111`	site-of-origin（SoO）拡張コミュニティを設定します。この CE ネイバーから学習されたルートは、その他の PE にアドバタイズされる前に SoO 拡張コミュニティのタグが付けられます。PE ルータで as-override が設定されている場合にループを検出する目的で SoO が使用されることがよくあります。プレフィックスが同じサイトにループする場合、

	コマンドまたはアクション	目的
		PE はこのことを検出して CE に更新を送信しません。
ステップ 16	**as-override** 例： `RP/0/RSP0/cpu 0: router(config-bgp-vrf-nbr-af)# as-override`	PE ルータで AS オーバーライドを設定します。これにより PE ルータは CE の ASN をそれ自体の（PE）ASN に置き換えます。 （注） この情報が失われることが原因でルーティング ループが発生することがあります。as-override によって引き起こされるループを防ぐには、as-override と site-of-origin を組み合わせて使用します。
ステップ 17	**allowas-in** [*as-occurrence-number*] 例： `RP/0/RSP0/cpu 0: router(config-bgp-vrf-nbr-af)# allowas-in 5`	PE 自律システム番号（ASN）を持つ AS パスを指定された回数だけ許可します。 ハブアンドスポーク型 VPN ネットワークは、HUB CE を通じて、HUB PE へのルーティング情報のループバックを必要とします。この場合、PE ASN が存在するために HUB PE によってループバック情報がドロップされます。これを回避するため、PE ASN が指定された回数に達していても **allowas-in** コマンドを使用してプレフィックスを許可します。
ステップ 18	**route-policy** *route-policy-name* **in** 例： `RP/0/RSP0/cpu 0: router(config-bgp-vrf-nbr-af)# route-policy pe_ce_in_policy in`	着信ルートのルーティング ポリシーを指定します。ポリシーを使用すると、ルートのフィルタリングやルート属性の変更ができます。
ステップ 19	**route-policy** *route-policy-name* **out** 例： `RP/0/RSP0/cpu 0: router(config-bgp-vrf-nbr-af)# route-policy pe_ce_out_policy out`	発信ルートのルーティング ポリシーを指定します。ポリシーを使用すると、ルートのフィルタリングやルート属性の変更ができます。
ステップ 20	**commit**	

IGP の BGP への再配布

VRF アドレス ファミリへのプロトコルの再配布を設定するには、次の作業を実行します。

内部ゲートウェイ プロトコル（IGP）が PE-CE プロトコルとして使用されている場合でも、インポート ロジックは BGP を経由して実行されます。したがって、すべての IGP ルートを BGP VRF テーブルにインポートする必要があります。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **vrf** *vrf-name*
4. **address-family** { **ipv4** | **ipv6** } **unicast**
5. 次のいずれかを実行します。
   - **redistribute connected** [**metric** *metric-value*] [**route-policy** *route-policy-name*]
   - **redistribute eigrp** *process-id* [**match** { **external** | **internal** }] [**metric** *metric-value*] [**route-policy** *route-policy-name*]
   - **redistribute isis** *process-id* [**level** { **1** | **1-inter-area** | **2** }] [**metric** *metric-value*] [**route-policy** *route-policy-name*]
   - **redistribute ospf** *process-id* [**match** { **external** [**1** | **2**] | **internal** | **nssa-external** [**1** | **2**] }] [**metric** *metric-value*] [**route-policy** *route-policy-name*]
   - **redistribute ospfv3** *process-id* [**match** { **external** [**1** | **2**] | **internal** | **nssa-external** [**1** | **2**] }] [**metric** *metric-value*] [**route-policy** *route-policy-name*]
   - **redistribute rip** [**metric** *metric-value*] [**route-policy** *route-policy-name*]
   - **redistribute static** [**metric** *metric-value*] [**route-policy** *route-policy-name*]
6. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 120`	自律システム番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**vrf** *vrf-name* 例： `RP/0/RSP0/cpu 0: router(config-bgp)# vrf vrf_a`	PE ルータで特定の VRF の BGP ルーティングをイネーブルにします。
ステップ 4	**address-family** { **ipv4**	**ipv6** } **unicast** 例： `RP/0/RSP0/cpu 0: router(config-vrf)# address-family ipv4 unicast`	IPv4 または IPv6 のいずれかのアドレス ファミリ ユニキャストを指定し、アドレスファミリのコンフィギュレーション サブモードを開始します。 このコマンドのすべてのキーワードと引数のリストを参照するには、CLI ヘルプ（?）を使用します。
ステップ 5	次のいずれかを実行します。 • **redistribute connected** [**metric** *metric-value*] [**route-policy** *route-policy-name*]	VRF アドレス ファミリ コンテキストでプロトコルの再配布を設定します。 redistribute コマンドは、PE-CE ルータ間で BGP が使用されていない場合に使用します。PE-CE ルータ間

	コマンドまたはアクション	目的
	• **redistribute eigrp** *process-id* [**match** { **external** | **internal** }] [**metric** *metric-value*] [**route-policy** *route-policy-name*] • **redistribute isis** *process-id* [**level** { **1** | **1-inter-area** | **2** }] [**metric** *metric-value*] [**route-policy** *route-policy-name*] • **redistribute ospf** *process-id* [**match** { **external** [**1** | **2**] | **internal** | **nssa-external** [**1** | **2**]}] [**metric** *metric-value*] [**route-policy** *route-policy-name*] • **redistribute ospfv3** *process-id* [**match** { **external** [**1** | **2**] | **internal** | **nssa-external** [**1** | **2**]}] [**metric** *metric-value*] [**route-policy** *route-policy-name*] • **redistribute rip** [**metric** *metric-value*] [**route-policy** *route-policy-name*] • **redistribute static** [**metric** *metric-value*] [**route-policy** *route-policy-name*] 例： `RP/0/RSP0/cpu 0: router(config-bgp-vrf-af)# redistribute eigrp 23`	で BGP が使用されている場合は、使用されている IGP を BGP に再配布して、他方の PE サイトとの VPN接続を確立する必要があります。テーブル間でのインポートおよびエクスポートにも再配布が必要です。
ステップ 6	**commit**	

BGP のキーチェーンの設定

キーチェーンは、さまざまな MAC 認証アルゴリズムをサポートして安全な認証を実現し、円滑なキー ロールオーバーを実装します。BGP のキーチェーンを設定するには、次の作業を実行します。このタスクはオプションです。

（注） ネイバー グループまたはセッション グループのキーチェーンが設定されている場合、そのグループを使用するネイバーはキーチェーンを継承します。あるネイバーのために特別に設定されたコマンドの値は、継承された値を上書きします。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **neighbor** *ip-address*
4. **remote-as** *as-number*
5. **keychain** *name*
6. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 120`	自律システム番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**neighbor** *ip-address* 例： `RP/0/RSP0/cpu 0: router(config-bgp)# neighbor 172.168.40.24`	BGP ルーティングのためにルータをネイバー コンフィギュレーション モードにして、ネイバーの IP アドレスを BGP ピアとして設定します。
ステップ 4	**remote-as** *as-number* 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# remote-as 2002`	ネイバーを作成し、リモート自律システム番号を割り当てます。
ステップ 5	**keychain** *name* 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# keychain kych_a`	キーチェーンに基づく認証を設定します。
ステップ 6	**commit**	

BGP ネイバーの無効化

設定を削除せずにネイバーを管理シャットダウンするには、次の作業を実行します。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **neighbor** *ip-address*
4. **shutdown**
5. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	

	コマンドまたはアクション	目的
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 127`	自律システム番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**neighbor** *ip-address* 例： `RP/0/RSP0/cpu 0: router(config-bgp)# neighbor 172.168.40.24`	BGP ルーティングのためにルータをネイバー コンフィギュレーション モードにして、ネイバーの IP アドレスを BGP ピアとして設定します。
ステップ 4	**shutdown** 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# shutdown`	指定されたネイバーのすべてのアクティブ セッションをディセーブルにします。
ステップ 5	**commit**	

BGP インバウンド ソフト リセットを使用したネイバーのリセット

指定されたグループまたはネイバーの指定アドレス ファミリに対してインバウンド ソフト リセットをトリガーするには、次の作業を実行します。グループは、***** 、 *ip-address* 、 *as-number* 、または **external** キーワードおよび引数によって指定されます。

ネイバーのインバウンドポリシーまたはアウトバウンドポリシーを変更する場合、またはルーティングアップデートの送信または受信に影響を与えるその他の設定を変更する場合には、ネイバーのリセットが便利です。インバウンド ソフト リセットがトリガーされた場合、ネイバーが ROUTE_REFRESH 機能をアドバタイズしていれば、BGP はデフォルトでこのネイバーに REFRESH 要求を送信します。ネイバーが ROUTE_REFRESH 機能をアドバタイズしているかどうかを判別するには、**show bgp neighbors** コマンドを使用します。

手順の概要

1. **show bgp neighbors**
2. **clear bgp** { **ipv4** { **unicast** | **multicast** | **all** | **tunnel** } | **ipv6** **unicast** | **all** { **unicast** | **multicast** | **all** | **tunnel** } | **vpnv4 unicast** | **vrf** { *vrf-name* | **all** } { **ipv4** **unicast** | **ipv6** **unicast** } { ***** | *ip-address* | **as** *as-number* | **external** } **soft** [**in** [**prefix-filter**] | **out**]

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**show bgp neighbors** 例： `RP/0/RSP0/cpu 0: router# show bgp neighbors`	ネイバーから受信したルート リフレッシュ機能がイネーブルであることを確認します。

	コマンドまたはアクション	目的
ステップ 2	**clear bgp** { **ipv4** { **unicast** | **multicast** | **all** | **tunnel** } | **ipv6 unicast** | **all** { **unicast** | **multicast** | **all** | **tunnel** } | **vpnv4 unicast** | **vrf** { *vrf-name* | **all** } { **ipv4 unicast** | **ipv6 unicast** } { ***** | *ip-address* | **as** *as-number* | **external** } **soft** [**in** [**prefix-filter**] | **out**] 例： `RP/0/RSP0/cpu 0: router# clear bgp ipv4 unicast 10.0.0.1 soft in`	BGP ネイバーをソフト リセットします。 • ***** キーワードを指定すると、すべての BGP ネイバーがリセットされます。 • *ip-address* 引数では、リセットするネイバーのアドレスを指定します。 • *as-number* 引数では、自律システム番号に一致するすべてのネイバーがリセットされることを指定します。 • **external** キーワードは、すべての外部ネイバーがリセットされることを指定します。

BGP アウトバウンド ソフト リセットを使用したネイバーのリセット

指定されたグループまたはネイバーの指定アドレス ファミリに対してアウトバウンド ソフト リセットをトリガーするには、次の作業を実行します。グループは、*****、*ip-address*、*as-number*、または **external** キーワードおよび引数によって指定されます。

ネイバーのアウトバウンド ポリシーまたはアウトバウンド ポリシーを変更する場合、またはルーティング アップデートの送信または受信に影響を与えるその他の設定を変更する場合には、ネイバーのリセットが便利です。

アウトバウンド ソフト リセットがトリガーされると、BGP は、このアドレス ファミリに対するルートをすべて、指定されたネイバーに再送信します。

ネイバーが ROUTE_REFRESH 機能をアドバタイズしているかどうかを判別するには、**show bgp neighbors** コマンドを使用します。

手順の概要

1. **show bgp neighbors**
2. **clear bgp** { **ipv4** { **unicast** \| **multicast** \| **all** \| **tunnel** } \| **ipv6 unicast** \| **all** { **unicast** \| **multicast** \| **all** \| **tunnel** } \| **vpnv4 unicast** \| **vrf** { *vrf-name* \| **all** } { **ipv4 unicast** \| **ipv6 unicast** } { ***** \| *ip-address* \| **as** *as-number* \| **external** } **clear bgp** { **ipv4** \| **ipv6**} { **unicast** \| **labeled-unicast** } **soft** [**in** [**prefix-filter**] \|]

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**show bgp neighbors** 例： `RP/0/RSP0/cpu 0: router# show bgp neighbors`	ネイバーから受信したルート リフレッシュ機能がイネーブルであることを確認します。

	コマンドまたはアクション	目的
ステップ 2	**clear bgp** { **ipv4** { **unicast** | **multicast** | **all** | **tunnel** } | **ipv6 unicast** | **all** { **unicast** | **multicast** | **all** | **tunnel** } | **vpnv4 unicast** | **vrf** { *vrf-name* | **all** } { **ipv4 unicast** | **ipv6 unicast** } { ***** | *ip-address* | **as** *as-number* | **external** } **clear bgp** { **ipv4 | ipv6**} { **unicast** | **labeled-unicast** } **soft** [**in** [**prefix-filter**] |] 例： `RP/0/RSP0/cpu 0: router# clear bgp ipv4 unicast 10.0.0.2 soft out`	BGP ネイバーをソフト リセットします。 • ***** キーワードを指定すると、すべての BGP ネイバーがリセットされます。 • *ip-address* 引数では、リセットするネイバーのアドレスを指定します。 • *as-number* 引数では、自律システム番号に一致するすべてのネイバーがリセットされることを指定します。 • **external** キーワードは、すべての外部ネイバーがリセットされることを指定します。

BGP ハード リセットを使用したネイバーのリセット

ハード リセットを使用してネイバーをリセットするには、次の作業を実行します。ハード リセットにより、ネイバーへの TCP 接続が削除され、ネイバーから受信したすべてのルートが BGP テーブルから削除され、その後このネイバーとのセッションが再確立されます。**graceful** キーワードを指定すると、ネイバーからのルートは BGP テーブルから即座に削除されず、古い（stale）ルートとしてマークされます。セッションの再確立後、ネイバーから再受信されなかった古いルートはすべて削除されます。

手順の概要

1. **clear bgp** { **ipv4** { **unicast** \| **multicast** \| **all** \| **tunnel** } \| **ipv6 unicast** \| **all** { **unicast** \| **multicast** \| **all** \| **tunnel** } \| **vpnv4 unicast** \| **vrf** { *vrf-name* \| **all** } { **ipv4 unicast** \| **ipv6 unicast** } \| { ***** \| *ip-address* \| **as** *as-number* \| **external** } [**graceful**] **soft** [**in** [**prefix-filter**] \| **out**] **clear bgp** { **ipv4 \| ipv6**} { **unicast** \| **labeled-unicast** }

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**clear bgp** { **ipv4** { **unicast** | **multicast** | **all** | **tunnel** } | **ipv6 unicast** | **all** { **unicast** | **multicast** | **all** | **tunnel** } | **vpnv4 unicast** | **vrf** { *vrf-name* | **all** } { **ipv4 unicast** | **ipv6 unicast** } | { ***** | *ip-address* | **as** *as-number* | **external** } [**graceful**] **soft** [**in** [**prefix-filter**] | **out**] **clear bgp** { **ipv4 | ipv6**} { **unicast** | **labeled-unicast** } 例： `RP/0/RSP0/cpu 0: router# clear bgp ipv4 unicast 10.0.0.3 graceful soft out`	BGP ネイバーをクリアします。 • ***** キーワードを指定すると、すべての BGP ネイバーがリセットされます。 • *ip-address* 引数では、リセットするネイバーのアドレスを指定します。 • *as-number* 引数では、自律システム番号に一致するすべてのネイバーがリセットされることを指定します。

	コマンドまたはアクション	目的
		• **external** キーワードは、すべての外部ネイバーがリセットされることを指定します。 **graceful** キーワードはグレースフル リスタートを指定します。

キャッシュ、テーブル、およびデータベースのクリア

特定のキャッシュ、テーブル、またはデータベースのすべての内容を削除するには、次のタスクを実行します。**clear bgp** コマンドは、指定されたネイバーグループのセッションをリセット（ハードリセット）します。これにより、ネイバーへの TCP 接続が削除され、ネイバーから受信したすべてのルートが BGP テーブルから削除され、その後このネイバーとのセッションが再確立されます。キャッシュ、テーブル、またはデータベースは、特定の構造が無効になったり、無効になるおそれのあるときに、クリアすることが必要になります。

手順の概要

1. **clear bgp** { **ipv4** { **unicast** | **multicast** | **all** | **tunnel** } | **ipv6 unicast** | **all** { **unicast** | **multicast** | **all** | **tunnel** } | **vpnv4 unicast** | **vrf** { *vrf-name* | **all** } { **ipv4 unicast** | **ipv6 unicast** } *ip-address*
2. **clear bgp external**
3. **clear bgp ***

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**clear bgp** { **ipv4** { **unicast**	**multicast**	**all**	**tunnel** }	**ipv6 unicast**	**all** { **unicast**	**multicast**	**all**	**tunnel** }	**vpnv4 unicast**	**vrf** { *vrf-name*	**all** } { **ipv4 unicast**	**ipv6 unicast** } *ip-address* 例： `RP/0/RSP0/cpu 0: router# clear bgp ipv4 172.20.1.1`	指定されたネイバーをクリアします。
ステップ 2	**clear bgp external** 例： `RP/0/RSP0/cpu 0: router# clear bgp external`	すべての外部ピアをクリアします。
ステップ 3	**clear bgp *** 例： `RP/0/RSP0/cpu 0: router# clear bgp *`	すべての BGP ネイバーをクリアします。

システムおよびネットワーク統計情報の表示

特定の統計情報（BGPルーティングテーブル、キャッシュ、およびデータベースの内容など）を表示するには、次のタスクを実行します。提供される情報は、リソースの使用状況を判定してネットワークの問題を解決するために使用されます。さらに、ノードの到達可能性に関する情報を表示し、そのパケットが経由するネットワーク内のルーティングパスを検出することもできます。

手順の概要

1. **show bgp cidr-only**
2. **show bgp community** *community-list* [**exact-match**]
3. **show bgp regexp** *regular-expression*
4. **show bgp**
5. **show bgp neighbors** *ip-address* [**advertised-routes** | **dampened-routes** | **flap-statistics** | **performance-statistics** | **received** *prefix-filter* | **routes**]
6. **show bgp paths**
7. **show bgp neighbor-group** *group-name* **configuration**
8. **show bgp summary**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**show bgp cidr-only** 例： `RP/0/RSP0/cpu 0: router# show bgp cidr-only`	不自然なネットワーク マスク（クラスレス ドメイン間ルーティング（CIDR））を持つルートを表示します。
ステップ 2	**show bgp community** *community-list* [**exact-match**] 例： `RP/0/RSP0/cpu 0: router# show bgp community 1081:5 exact-match`	指定されたBGPコミュニティに一致するルートを表示します。
ステップ 3	**show bgp regexp** *regular-expression* 例： `RP/0/RSP0/cpu 0: router# show bgp regexp "^3 "`	指定した自律システム パスの正規表現と一致するルートを表示します。
ステップ 4	**show bgp** 例： `RP/0/RSP0/cpu 0: router# show bgp`	BGPルーティングテーブル内のエントリを表示します。

	コマンドまたはアクション	目的
ステップ 5	**show bgp neighbors** *ip-address* [**advertised-routes** | **dampened-routes** | **flap-statistics** | **performance-statistics** | **received** *prefix-filter* | **routes**] 例： `RP/0/RSP0/cpu 0: router# show bgp neighbors 10.0.101.1`	指定したネイバーへのBGP接続に関する情報を表示します。 • **advertised-routes** キーワードを指定すると、ルータがネイバーにアドバタイズするすべてのルートが表示されます。 • **dampened-routes** キーワードを指定すると、ネイバーから学習したダンプ済みのルートが表示されます。 • **flap-statistics** キーワードを指定すると、ネイバーから学習したルートのフラップ統計情報が表示されます。 • **performance-statistics** キーワードを指定すると、このネイバーのBGPプロセスによって実行された作業に関連するパフォーマンス統計情報が表示されます。 • **received** *prefix-filter* キーワードと引数を指定すると、プレフィックス リスト フィルタが表示されます。 • **routes** キーワードを指定すると、ネイバーから学習したルートが表示されます。
ステップ 6	**show bgp paths** 例： `RP/0/RSP0/cpu 0: router# show bgp paths`	データベース内のすべてのBGPパスを表示します。
ステップ 7	**show bgp neighbor-group** *group-name* **configuration** 例： `RP/0/RSP0/cpu 0: router# show bgp neighbor-group group_1 configuration`	指定したネイバーグループによって継承された設定を含む、ネイバーグループの有効な設定を表示します。
ステップ 8	**show bgp summary** 例： `RP/0/RSP0/cpu 0: router# show bgp summary`	BGP 接続すべての状況を表示します。

BGP プロセス情報の表示

特定の BGP プロセス情報を表示するには、次のタスクを実行します。

手順の概要

1. **show bgp process**
2. **show bgp ipv4 unicast summary**
3. **show bgp vpnv4 unicast summary**
4. **show bgp vrf** (*vrf-name* | **all** }
5. **show bgp process detail**
6. **show bgp summary**
7. **show placement program bgp**
8. **show placement program brib**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**show bgp process** 例： `RP/0/RSP0/cpu 0: router# show bgp process`	BGP プロセスのステータスと要約情報を表示します。出力には、さまざまなグローバルおよびアドレス ファミリ固有の BGP 設定が表示されます。プロセスによって送受信されたネイバー、アップデート メッセージ、および通知メッセージの数の要約も表示されます。
ステップ 2	**show bgp ipv4 unicast summary** 例： `RP/0/RSP0/cpu 0: router# show bgp ipv4 unicast summary`	IPv4 ユニキャスト アドレス ファミリのネイバーの要約を表示します。
ステップ 3	**show bgp vpnv4 unicast summary** 例： `RP/0/RSP0/cpu 0: router# show bgp vpnv4 unicast summary`	VPNv4 ユニキャスト アドレス ファミリのネイバーの要約を表示します。
ステップ 4	**show bgp vrf** (*vrf-name* | **all** } 例： `RP/0/RSP0/cpu 0: router# show bgp vrf vrf_A`	BGP VPN 仮想ルーティングおよび転送（VRF）情報を表示します。
ステップ 5	**show bgp process detail** 例： `RP/0/RSP0/cpu 0: router# show bgp processes detail`	さまざまな内部構造タイプによって使用されているメモリなど、詳細なプロセス情報を表示します。
ステップ 6	**show bgp summary** 例： `RP/0/RSP0/cpu 0: router# show bgp summary`	BGP 接続すべての状況を表示します。

	コマンドまたはアクション	目的
ステップ 7	**show placement program bgp** 例： `RP/0/RSP0/cpu 0: router# show placement program bgp`	BGP プログラムの情報を表示します。 • 「拒否された場所」としてプログラムが表示される場合（プログラムの場所を特定できないなど）、**show placement program bgp** コマンドを使用して、その場所を表示できます。 • プログラムが配置されても起動されない場合、プログラムが配置されてから経過した時間の長さが [Waiting to start] 列に表示されます。
ステップ 8	**show placement program brib** 例： `RP/0/RSP0/cpu 0: router# show placement program brib`	bRIB プログラムの情報を表示します。 • 「拒否された場所」としてプログラムが表示される場合（プログラムの場所を特定できないなど）、**show placement program bgp** コマンドを使用して、その場所を表示できます。 • プログラムが配置されても起動されない場合、プログラムが配置されてから経過した時間の長さが [Waiting to start] 列に表示されます。

BGP アップデート グループのモニタリング

この作業では、BGP アップデート グループの処理に関する情報を表示します。

手順の概要

1. **show bgp** [**ipv4** { **unicast** | **multicast** | **all** | **tunnel** } | **ipv6** { **unicast** | **all** } | **all** { **unicast** | **multicast** | **all** | **tunnel** } | **vpnv4 unicast** | **vrf** { *vrf-name* | **all** } [**ipv4 unicast**] **update-group** [**neighbor** *ip-address* | *process-id.index* [**summary** | **performance-statistics**]]

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**show bgp** [**ipv4** { **unicast**	**multicast**	**all**	**tunnel** }	**ipv6** { **unicast**	**all** }	**all** { **unicast**	**multicast**	**all**	**tunnel** }	**vpnv4 unicast**	**vrf** { *vrf-name*	**all** } [**ipv4 unicast**] **update-group** [**neighbor** *ip-address*	*process-id.index* [**summary**	**performance-statistics**]] 例：	BGP アップデート グループの情報を表示します。 • *ip-address* 引数を指定すると、そのネイバーが属するアップデート グループが表示されます。 • *process-id.index* 引数では、表示する特定のアップデート グループを選択します。この引数は「プロセスID（ドット）インデックス」の形式で指定します。プロセス ID の範囲は 0 ～ 254

	コマンドまたはアクション	目的
	`RP/0/RSP0/cpu 0: router# show bgp update-group 0.0`	です。インデックスの範囲は 0 ～ 4294967295 です。 • **summary** キーワードを指定すると、特定のアップデート グループに含まれているネイバーに関する要約情報が表示されます。 • このコマンドに引数を指定しないと、（指定したアドレス ファミリの）すべてのアップデート グループの情報が表示されます。 • **performance-statistics** キーワードを指定すると、アップデート グループのパフォーマンス統計情報が表示されます。

BGP ノンストップルーティングの設定

BGP ノンストップルーティング（BGP NSR）はデフォルトで有効になっています。また、無効になっている BGP NSR を有効に戻すには、**no nsr disable** コマンドを使用します。

BGP ノンストップ ルーティングの無効化

BGP ノンストップ ルーティング（NSR）を無効にするには、次のタスクを実行します。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **nsr disable**
4. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 120`	BGP ルーティング プロセスを設定するため、BGP AS 番号を指定して BGP コンフィギュレーション モードを開始します。
ステップ 3	**nsr disable** 例：	BGP ノンストップ ルーティングを無効にします。

	コマンドまたはアクション	目的
	`RP/0/RSP0/cpu 0: router(config-bgp)# nsr disable`	
ステップ 4	**commit**	

BGP ノンストップ ルーティングの再有効化

BGP ノンストップ ルーティング（NSR）が無効になっている場合、次のステップを使用して BGP NSR を有効にします。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **no nsr disable**
4. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 120`	BGP ルーティング プロセスを設定するため、BGP AS 番号を指定して BGP コンフィギュレーション モードを開始します。
ステップ 3	**no nsr disable** 例： `RP/0/RSP0/cpu 0: router(config-bgp)# nsr disable`	BGP ノンストップ ルーティングを有効にします。
ステップ 4	**commit**	

Prefix Independent Convergence（PIC）のプライマリバックアップパスのインストール

転送テーブルにバックアップパスをインストールし、PE-CE リンク障害が発生した場合に Prefix Independent Convergence（PIC）を提供するには、次のタスクを実行します。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. 次のいずれかを実行します。

- **address-family** {**vpnv4 unicast** | **vpnv6 unicast**}
- **vrf** *vrf-name* {**ipv4 unicast** | **ipv6 unicast**}

4. **additional-paths selection route-policy** *route-policy-name*
5. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 100`	自律システム番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	次のいずれかを実行します。 • **address-family** {**vpnv4 unicast**	**vpnv6 unicast**} • **vrf** *vrf-name* {**ipv4 unicast**	**ipv6 unicast**} 例： `RP/0/RSP0/cpu 0: router(config-bgp)# address-family vpnv4 unicast`	アドレスファミリまたはVRFアドレスファミリを指定して、アドレスファミリまたはVRFアドレスファミリのコンフィギュレーション サブモードを開始します。

ステップ 4	**additional-paths selection route-policy** *route-policy-name*
	例：
	`RP/0/RSP0/cpu 0: router(config-bgp-af)# additional-paths selection route-policy ap1`
		プレフィックスの追加パス選択モードを設定します。
		（注） **additional-paths selection** コマンドを適切なルートポリシーとともに使用して、バックアップパスを計算し、プレフィックス独立コンバージェンス（PIC）機能を有効にします。
		ルートポリシーの設定は、プレフィックスの追加パス選択モードを設定するための前提条件です。追加選択コマンドで使用するルートポリシー設定の例を次に示します。
		`route-policy ap1`
		`  set path-selection backup 1 install`
		`end-policy`

ステップ 5	**commit**	

プライマリ パスのローカル ラベル割り当ての保持

プライマリ PE で以前にプライマリ パスに割り当てられたローカル ラベルを、再コンバージェンス後に設定期間にわたって保持するには、次の作業を実行します。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **address-family { vpnv4 unicast | vpnv6 unicast }**
4. **retain local-label** *minutes*
5. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 100`	自律システム番号を指定し、BGP コンフィギュレーションモードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**address-family { vpnv4 unicast | vpnv6 unicast }** 例： `RP/0/RSP0/cpu 0: router(config-bgp)# address-family vpnv4 unicast`	アドレス ファミリを指定し、アドレス ファミリのコンフィギュレーション サブモードを開始します。
ステップ 4	**retain local-label** *minutes* 例： `RP/0/RSP0/cpu 0: router(config-bgp-af)# retain local-label 10`	プライマリ PE で以前にプライマリ パスに割り当てられたローカル ラベルを、再コンバージェンス後 10 分間保持します。
ステップ 5	**commit**	

BGP 追加パスの設定

BGP 追加パス機能を設定するには、次の作業を行います。

手順の概要

1. **configure**
2. **route-policy** *route-policy-name*
3. **if** *conditional-expression* **then** *action-statement* **else**
4. **pass endif**
5. **end-policy**
6. **router bgp** *as-number*
7. **address-family {ipv4 {unicast | multicast} | ipv6 {unicast | multicast | l2vpn vpls-vpws | vpnv4 unicast | vpnv6 unicast }**
8. **additional-paths receive**
9. **additional-paths send**

10. **additional-paths selection route-policy** *route-policy-name*
11. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**route-policy** *route-policy-name* 例： `RP/0/RSP0/cpu 0: router (config)#route-policy add_path_policy`	ルート ポリシーを定義して、ルート ポリシー コンフィギュレーション モードを開始します。

ステップ 3	**if** *conditional-expression* **then** *action-statement* **else**
	例：
	`RP/0/RSP0/cpu 0: router (config-rpl)#if community matches-any (*) then`
	`set path-selection all advertise`
	`else`
		特定のルートのアクションとディスポジションを決定します。

ステップ 4	**pass endif**
	例：
	`RP/0/RSP0/cpu 0: router(config-rpl-else)#pass`
	`RP/0/RSP0/cpu 0: router(config-rpl-else)#endif`
		処理のためにルートを渡し、ifステートメントを終了します。

ステップ 5	**end-policy** 例： `RP/0/RSP0/cpu 0: router(config-rpl)#end-policy`	ルートポリシーの定義を終了して、ルートポリシー コンフィギュレーション モードを終了します。
ステップ 6	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)#router bgp 100`	自律システム番号を指定し、BGP コンフィギュレーションモードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 7	**address-family** {**ipv4** {**unicast** | **multicast**} | **ipv6** {**unicast** | **multicast** | **l2vpn vpls-vpws** | **vpnv4 unicast** | **vpnv6 unicast** } 例： `RP/0/RSP0/cpu 0: router(config-bgp)#address-family ipv4 unicast`	アドレス ファミリを指定し、アドレス ファミリのコンフィギュレーションサブモードを開始します。
ステップ 8	**additional-paths receive** 例： `RP/0/RSP0/cpu 0: router(config-bgp-af)#additional-paths receive`	対応ピアのプレフィックスのマルチパス受信機能を設定します。
ステップ 9	**additional-paths send** 例：	対応ピアのプレフィックスのマルチパス送信機能を設定します。

	コマンドまたはアクション	目的
	RP/0/RSP0/cpu 0: router(config-bgp-af)#additional-paths send	
ステップ 10	**additional-paths selection route-policy** *route-policy-name* 例： RP/0/RSP0/cpu 0: router(config-bgp-af)#additional-paths selection route-policy add_path_policy	プレフィックスの追加パス選択機能を設定します。
ステップ 11	**commit**	

iBGP マルチパス ロード シェアリングの設定

iBGP マルチパス ロード シェアリングを設定するには、次の作業を実行します。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **address-family** {**ipv4**|**ipv6**} {**unicast**|**multicast**}
4. **maximum-paths ibgp** *number*
5. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： RP/0/RSP0/cpu 0: router(config)# router bgp 100	自律システム番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**address-family** {**ipv4**|**ipv6**} {**unicast**|**multicast**} 例： RP/0/RSP0/cpu 0: router(config-bgp)# address-family ipv4 multicast	IPv4 または IPv6 のいずれかのアドレス ファミリを指定し、アドレス ファミリのコンフィギュレーション サブモードを開始します。
ステップ 4	**maximum-paths ibgp** *number* 例： RP/0/RSP0/cpu 0: router(config-bgp-af)# maximum-paths ibgp 30	ロード シェアリング用の iBGP パスの最大数を設定します。
ステップ 5	**commit**	

AiGP によるプレフィックスの生成

AiGP メトリックを使用したルートの生成を設定するには、次の作業を実行します。

始める前に

Accumulated Interior Gateway Protocol（AiGP）メトリックを使用したルートの生成は設定により制御されます。次の条件を満たす再配布ルートに AiGP 属性が付加されます。

- AiGP でルートを再配布するプロトコルがイネーブルに設定されている。
- このルートは、ボーダー ゲートウェイ プロトコル（BGP）に再配布された Interior Gateway Protocol（iGP）ルートです。AiGP 属性に割り当てられた値はルートの iGP ネクスト ホップの値か、または route-policy によって設定された値です。
- このルートは BGP に再配布されたスタティック ルートです。割り当てられた値はルートのネクスト ホップの値か、route-policy によって設定された値です。
- このルートはネットワーク ステートメントによって BGP にインポートされます。割り当てられた値はルートのネクスト ホップの値か、route-policy によって設定された値です。

手順の概要

1. **configure**
2. **route-policy** *aigp_policy*
3. **set aigp-metricigp-cost**
4. **exit**
5. **router bgp** *as-number*
6. **address-family** {**ipv4** | **ipv6**} **unicast**
7. **redistribute ospf** *osp* **route-policy** *plcy_name***metric** *value*
8. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**route-policy** *aigp_policy* 例： `RP/0/RSP0/cpu 0: router(config)# route-policy aip_policy`	ルート ポリシー コンフィギュレーション モードを開始してルート ポリシーを設定します。
ステップ 3	**set aigp-metricigp-cost** 例： `RP/0/RSP0/cpu 0: router(config-rpl)# set aigp-metric igp-cost`	内部ルーティング プロトコル コストを aigp メトリックとして設定します。

	コマンドまたはアクション	目的
ステップ 4	**exit** 例： `RP/0/RSP0/cpu 0: router(config-rpl)# exit`	ルートポリシー コンフィギュレーション モードを終了します。
ステップ 5	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 100`	BGP AS 番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。

ステップ 6	**address-family** {**ipv4** | **ipv6**} **unicast**
	例：
	`RP/0/RSP0/cpu 0:`
	`router(config-bgp)# address-family ipv4 unicast`
		IPv4 または IPv6 のいずれかのアドレス ファミリを指定し、アドレス ファミリのコンフィギュレーション サブモードを開始します。

ステップ 7	**redistribute ospf** *osp* **route-policy** *plcy_name***metric** *value*
	例：
	`RP/0/RSP0/cpu 0:`
	`router(config-bgp-af)#redistribute ospf osp`
	`route-policy aigp_policy metric 1`
		OSPF への AiBGP メトリックの再配布を許可します。

ステップ 8	**commit**	

BGP Accept Own の設定

BGP Accept Own を設定するには、次の作業を実行します。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **neighbor** *ip-address*
4. **remote-as** *as-number*
5. **update-source** *type interface-path-id*
6. **address-family** {*vpnv4 unicast* | *vpnv6 unicast*}
7. **accept-own** [**inheritance-disable**]

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `Router(config)#router bgp 100`	自律システム番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。

	コマンドまたはアクション	目的
ステップ 3	**neighbor** *ip-address* 例： `Router(config-bgp)#neighbor 10.1.2.3`	BGP ルーティングのためにルータをネイバー コンフィギュレーション モードにして、ネイバーの IP アドレスを BGP ピアとして設定します。
ステップ 4	**remote-as** *as-number* 例： `Router(config-bgp-nbr)#remote-as 100`	ネイバーにリモート自律システム番号を割り当てます。
ステップ 5	**update-source** *type interface-path-id* 例： `Router(config-bgp-nbr)#update-source Loopback0`	ネイバーでセッションを形成するとき、特定のインターフェイスからのプライマリ IP アドレスをローカル アドレスとしてセッションで使用できます。
ステップ 6	**address-family** {*vpnv4 unicast* | *vpnv6 unicast*} 例： `Router(config-bgp-nbr)#address-family vpnv6 unicast`	アドレス ファミリを VPNv4 または IPv6 として指定し、ネイバーアドレスファミリのコンフィギュレーション モードを開始します。
ステップ 7	**accept-own** [**inheritance-disable**] 例： `Router(config-bgp-nbr-af)#accept-own`	Accept_Own コミュニティが含まれる自動送信 VPN ルートの処理をイネーブルにします。 「Accept Own」設定をディセーブルにし、親コンフィギュレーションから「Accept Own」が継承されないようにするには、 **inheritance-disable** キーワードを使用します。

BGP リンク状態の設定

BGP リンク状態の設定

BGP リンクステート（LS）情報を BGP ネイバーと交換するには、次のステップを実行します。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **neighbor** *ip-address*
4. **remote-as** *as-number*
5. **address-family link-state link-state**
6. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 100`	BGP AS 番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**neighbor** *ip-address* 例： `RP/0/RSP0/cpu 0: router(config-bgp)# neighbor 10.0.0.2`	CE ネイバーを設定します。ip-address 引数は、プライベート アドレスである必要があります。
ステップ 4	**remote-as** *as-number* 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# remote-as 1`	CE ネイバーのリモート AS を設定します。
ステップ 5	**address-family link-state link-state** 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# address-family link-state link-state`	BGP リンクステート情報を指定されたネイバーに配布します。
ステップ 6	**commit**	

ドメイン識別子の設定

固有識別子 4 オクテット ASN を設定するには、次のステップを実行します。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **address-family link-state link-state**
4. **domain-distinguisher** *unique-id*
5. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 100`	BGP AS 番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**address-family link-state link-state** 例： `RP/0/RSP0/cpu 0: router(config-bgp)# address-family link-state link-state`	アドレスファミリ リンクステート コンフィギュレーション モードを開始します。
ステップ 4	**domain-distinguisher** *unique-id* 例： `RP/0/RSP0/cpu 0: router(config-bgp-af)# domain-distinguisher 1234`	固有識別子 4 オクテット ASN を設定します。範囲は 1 ～ 4294967295 です。
ステップ 5	**commit**	

BGP パーマネントネットワークの設定

BGP パーマネントネットワークの設定

BGP パーマネント ネットワークを設定するには、次のタスクを実行します。パーマネント ネットワーク（パス）が設定されるプレフィックス（ネットワーク）のセットを識別するには、少なくとも 1 つのルート ポリシーを設定する必要があります。

手順の概要

1. **configure**
2. **prefix-set** *prefix-set-name*
3. **exit**
4. **route-policy** *route-policy-name*
5. **end-policy**
6. **router bgp** *as-number*
7. **address-family { ipv4 | ipv6 } unicast**
8. **permanent-network route-policy** *route-policy-name*
9. **commit**
10. **show bgp {ipv4 | ipv6} unicast** *prefix-set*

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	

ステップ 2	**prefix-set** *prefix-set-name*
	例：
	`RP/0/RSP0/cpu 0: router(config)# prefix-set PERMANENT-NETWORK-IPv4`
	`RP/0/RSP0/cpu 0: router(config-pfx)# 1.1.1.1/32,`
	`RP/0/RSP0/cpu 0: router(config-pfx)# 2.2.2.2/32,`
	`RP/0/RSP0/cpu 0: router(config-pfx)# 3.3.3.3/32`
	`RP/0/RSP0/cpu 0: router(config-pfx)# end-set`
		プレフィックス セット コンフィギュレーション モードを開始し、連続したビット セットと非連続のビット セットに対しプレフィックス セットを定義します。

ステップ 3	**exit** 例： `RP/0/RSP0/cpu 0: router(config-pfx)# exit`	プレフィックス セット コンフィギュレーション モードを終了し、グローバル コンフィギュレーション モードを開始します。

ステップ 4	**route-policy** *route-policy-name*
	例：
	`RP/0/RSP0/cpu 0: router(config)# route-policy POLICY-PERMANENT-NETWORK-IPv4`
	`RP/0/RSP0/cpu 0: router(config-rpl)# if destination in PERMANENT-NETWORK-IPv4 then`
	`RP/0/RSP0/cpu 0: router(config-rpl)# pass`
	`RP/0/RSP0/cpu 0: router(config-rpl)# endif`
		ルート ポリシーを作成し、ルート ポリシー コンフィギュレーション モードを開始します。このモードではルート ポリシーを定義できます。

ステップ 5	**end-policy** 例： `RP/0/RSP0/cpu 0: router(config-rpl)# end-policy`	ルート ポリシーの定義を終了して、ルート ポリシー コンフィギュレーション モードを終了します。
ステップ 6	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 100`	自律システム番号を指定して、BGP コンフィギュレーション モードを開始します。
ステップ 7	**address-family { ipv4 | ipv6 } unicast** 例： `RP/0/RSP0/cpu 0: router(config-bgp)# address-family ipv4 unicast`	IPv4 または IPv6 のいずれかのアドレス ファミリ ユニキャストを指定し、アドレス ファミリのコンフィギュレーション サブモードを開始します。

	コマンドまたはアクション	目的
ステップ 8	**permanent-network route-policy** *route-policy-name* 例： `RP/0/RSP0/cpu 0: router(config-bgp-af)# permanent-network route-policy POLICY-PERMANENT-NETWORK-IPv4`	ルート ポリシーで定義されているプレフィックスのセットに対しパーマネントネットワーク（パス）を設定します。
ステップ 9	**commit**	
ステップ 10	**show bgp {ipv4 | ipv6} unicast** *prefix-set* 例： `RP/0/RSP0/cpu 0: routershow bgp ipv4 unicast`	（オプション）プレフィックス セットが BGP でパーマネント ネットワークであるかどうかを表示します。

パーマネントネットワークのアドバタイズ方法

固定パスがアドバタイズされる必要があるピアを識別するには、このタスクを実行します。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **neighbor** *ip-address*
4. **remote-as** *as-number*
5. **address-family { ipv4 | ipv6 } unicast**
6. **advertise permanent-network**
7. **commit**
8. **show bgp {ipv4 | ipv6} unicast neighbor** *ip-address*

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 100`	自律システム番号を指定して、BGP コンフィギュレーション モードを開始します。
ステップ 3	**neighbor** *ip-address* 例： `RP/0/RSP0/cpu 0: router(config-bgp)# neighbor`	BGP ルーティングのためにルータをネイバー コンフィギュレーション モードにして、ネイバーの IP アドレスを BGP ピアとして設定します。

	コマンドまたはアクション	目的
	`10.255.255.254`	
ステップ 4	**remote-as** *as-number* 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# remote-as 4713`	ネイバーをリモート自律システム番号に割り当てます。
ステップ 5	**address-family { ipv4 | ipv6 } unicast** 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# address-family ipv4 unicast`	IPv4 または IPv6 のいずれかのアドレス ファミリ ユニキャストを指定し、アドレスファミリのコンフィギュレーション サブモードを開始します。
ステップ 6	**advertise permanent-network** 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr-af)# advertise permanent-network`	パーマネント ネットワーク（パス）がアドバタイズされるピアを指定します。
ステップ 7	**commit**	
ステップ 8	**show bgp {ipv4 | ipv6} unicast neighbor** *ip-address* 例： `RP/0/RSP0/cpu 0: routershow bgp ipv4 unicast neighbor 10.255.255.254`	（オプション）ネイバーが BGP パーマネント ネットワークを受信できるかどうかを表示します。

BGP 不等コストの連続ロードバランシングの有効化

外部 BGP（eBGP）、内部 BGP（iBGP）、および eiBGP の不等コストの連続ロードバランシングを有効にし、BGP が非武装地帯（DMZ）リンクのリンク帯域幅属性を送信できるようにするには、次のタスクを実行します。

マルチプロトコル内部 BGP（MP-iBGP）セッション（IPv4 または VPNv4）を介した、リモート PE への PE ルータのアップデートにリンク帯域幅拡張コミュニティが含まれている場合、**maximum-paths** コマンドが有効になっていれば、リモート PE が自動的にロードバランシングを実行します。

不等コストの連続ロード バランシングは、最大で 8 つのパスに対してのみ行われます。

（注） BGP不等コスト連続ロードバランシング機能の有効化は、CPPベースのカードではサポートされていません。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **address-family** { **ipv4** | **ipv6** } **unicast**
4. **maximum-paths** { **ebgp** | **ibgp** | **eibgp** } *maximum* [**unequal-cost**]
5. **exit**
6. **neighbor** *ip-address*
7. **dmz-link-bandwidth**
8. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 120`	自律システム番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。
ステップ 3	**address-family** { **ipv4**	**ipv6** } **unicast** 例： `RP/0/RSP0/cpu 0: router(config-bgp)# address-family ipv4 unicast`	IPv4 または IPv6 のいずれかのアドレス ファミリ ユニキャストを指定し、アドレス ファミリのコンフィギュレーション サブモードを開始します。 このコマンドのすべてのキーワードと引数のリストを参照するには、CLI ヘルプ（?）を使用します。
ステップ 4	**maximum-paths** { **ebgp**	**ibgp**	**eibgp** } *maximum* [**unequal-cost**] 例： `RP/0/RSP0/cpu 0: router(config-bgp-af)# maximum-paths ebgp 3`	BGPによりルーティングテーブルにインストールされるパラレル ルートの最大数を設定します。 （注） • 最大パスの有効な値は、ASR 9000イーサネットラインカードの場合は 8、ASR 9000拡張イーサネットラインカードの場合は 32 です。 • ASR 9000 イーサネットラインカードは、設定されている最大パス値が 8 を超える場合でも、転送ハードウェアにインストールするルートの数を 8 に制限します。

	コマンドまたはアクション	目的
		• **ebgp** *maximum* ：マルチパスに eBGP パスのみを考慮します。 • **ibgp** *maximum* [**unequal-cost**] ：iBGP 学習パス間でのロード バランシングを考慮します。 • **eibgp** *maximum* ：eBGP および iBGP 学習パスの両方のロードバランシングを考慮します。eiBGP は常に不等コスト ロード バランシングを実行します。 eiBGP が適用されると eBGP ロード バランシングまたは iBGP ロード バランシングは設定できませんが、eBGP ロード バランシングと iBGP ロード バランシングは共存できます。
ステップ 5	**exit** 例： `RP/0/RSP0/cpu 0: router(config-bgp-af)# exit`	現在のコンフィギュレーション モードを終了します。
ステップ 6	**neighbor** *ip-address* 例： `RP/0/RSP0/cpu 0: router(config-bgp)# neighbor 10.0.0.0`	CE ネイバーを設定します。 *ip-address* 引数は、プライベート アドレスにする必要があります。
ステップ 7	**dmz-link-bandwidth** 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# dmz-link-bandwidth`	eBGP および iBGP ネイバーへのリンクのために、非武装地帯（DMZ）リンク帯域幅拡張コミュニティを開始します。
ステップ 8	**commit**	

VRF ダイナミックルートのリークの設定

次のステップを実行して、デフォルト VRF から非デフォルト VRF にルートをインポートするか、または非デフォルト VRF からデフォルト VRF にルートをインポートします。

始める前に

ダイナミック ルート リークを設定するには、ルート ポリシーが必要です。ルート ポリシーを設定するには、グローバル コンフィギュレーション モードで**route-policy** *route-policy-name* コマンドを使用します。

手順の概要

1. **configure**
2. **vrf** *vrf_name*
3. **address-family** {**ipv4** | **ipv6**} **unicast**
4. 次のいずれかのオプションを使用します。
   - **import from default-vrf route-policy** *route-policy-name* [**advertise-as-vpn**]
   - **export to default-vrf route-policy** *route-policy-name*
5. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**vrf** *vrf_name* 例： `RP/0/RSP0/CPU0:PE51_ASR-9010(config)#vrf vrf_1`	VRF コンフィギュレーション モードを開始します。
ステップ 3	**address-family** {**ipv4**	**ipv6**} **unicast** 例： `RP/0/RSP0/cpu 0: router(config-vrf)#address-family ipv6 unicast`	VRF アドレス ファミリ コンフィギュレーション モードを開始します。

ステップ 4	次のいずれかのオプションを使用します。
	• **import from default-vrf route-policy** *route-policy-name* [**advertise-as-vpn**]
	• **export to default-vrf route-policy** *route-policy-name*
	例：
	`RP/0/RSP0/cpu 0: router(config-vrf-af)#import from default-vrf route-policy rpl_dynamic_route_import`
	または
	`RP/0/RSP0/cpu 0: router(config-vrf-af)#export to default-vrf route-policy rpl_dynamic_route_export`
		デフォルト VRF から非デフォルト VRF にルートをインポートするか、または非デフォルト VRF からデフォルト VRF にルートをインポートします。
		• **import from default-vrf**：デフォルト VRF から非デフォルト VRF へのインポートを設定します。
		advertise-as-vpn オプションが設定されている場合、デフォルト VRF から非デフォルト VRF にインポートしたパスは、PE と CE にアドバタイズされます。**advertise-as-vpn** オプションが設定されていない場合、デフォルト VRF から非デフォルト VRF にインポートされたパスは PE にアドバタイズされません。ただし、この場合も CE にはパスがアドバタイズされます。
		• **export to default-vrf**：非デフォルト VRF からデフォルト VRF へのインポートを設定します。デフォルト VRF からインポートされたパスが他の PE にアドバタイズされます。

ステップ 5	**commit**	

次のタスク

次の **show bgp** コマンドの出力には、ダイナミック ルート リーク設定の情報が表示されます。

- **show bgp prefix** コマンドを使用すると、インポートしたパスの送信元 RD と送信元 VRF が表示されます。これには、IPv4 または IPv6 ユニキャスト プレフィックスにインポートしたパスがある場合も含まれます。
- **show bgp imported-routes** コマンドを使用すると、デフォルト VRF の IPv4 ユニキャストおよび IPv6 ユニキャストのアドレスファミリが表示されます。

選択的 VRF ダウンロードの有効化

選択的 VRF ダウンロードを有効にするには、**svd platform enable** コマンドの後にルータのリロードを設定します。

（注） デフォルトでは、選択的 VRF ダウンロードは無効になっています。

手順の概要

1. **admin**
2. **configure**
3. **svd platform enable**
4. **commit**
5. **show svd state**
6. **admin**
7. **reload location all**
8. **exit**
9. **show svd role**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**admin** 例： `RP/0/RSP0/cpu 0: router# admin`	管理 EXEC モードを開始します。
ステップ 2	**configure** 例： `RP/0/RSP0/cpu 0: router(admin)#configure`	管理コンフィギュレーション モードを開始します。
ステップ 3	**svd platform enable** 例：	選択的 VRF ダウンロードを有効にします。

	コマンドまたはアクション	目的
	RP/0/RSP0/cpu 0: router(admin-config)#svd platform enable	
ステップ 4	**commit**	
ステップ 5	**show svd state** 例： RP/0/RSP0/cpu 0: router#show svd state Selective VRF Download (SVD) Feature State: SVD Configuration State Enabled SVD Operational State Enabled	選択的VRFダウンロード機能の状態情報を表示します。
ステップ 6	**admin** 例： RP/0/RSP0/cpu 0: router#admin	管理者モードを開始します。
ステップ 7	**reload location all** 例： RP/0/RSP0/cpu 0: router(admin)#reload loc all Tue Feb 12 07:51:25.279 UTC Preparing system for backup. This may take a few minutes especially for large configurations. Status report: node0_RSP0_CPU0: START TO BACKUP Status report: node0_RSP0_CPU0: BACKUP HAS COMPLETED SUCCESSFULLY [Done] Proceed with reload? [confirm]RP/0/RSP0/CPU0::This node received reload	シャーシをリロードします。
ステップ 8	**exit** 例： RP/0/RSP0/cpu 0: router(admin)#exit	管理者 EXEC モードを終了し、EXEC モードを開始します。
ステップ 9	**show svd role** 例： RP/0/RSP0/cpu 0: router#show svd role Tue Feb 12 07:50:26.908 UTC Codes: (C) : user Configured role Node Name IPv4 Role IPv6 Role ——————————————————————— 0/RSP0/CPU0 Standard Standard 0/0/CPU0 Customer Facing Not Interested 0/1/CPU0 Customer Facing Not Interested	VRF インターフェイスがあるラインカードに SVD ロールが「カスタマー向け」であることを確認することで、選択的 VRF ダウンロードがアクティブになっているかどうかを確認します。

次のタスク

svd platform enable コマンドを使用して SVD を有効にした後に **selective-vrf-download disable** を使用して SVD をオフにしないでください。

選択的 VRF ダウンロードの無効化

デフォルトでは、選択的VRFダウンロードは無効になっています。ただし、SVDが有効になっている場合は、次のタスクを実行して機能を無効にします。

手順の概要

1. **admin**
2. **configure**
3. **no svd platform enable**
4. **commit**
5. **show svd state**
6. **admin**
7. **reload location all**
8. **exit**
9. **show svd role**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**admin** 例： `RP/0/RSP0/cpu 0: router# admin`	管理 EXEC モードを開始します。
ステップ 2	**configure** 例： `RP/0/RSP0/cpu 0: router(admin)#configure`	管理コンフィギュレーション モードを開始します。
ステップ 3	**no svd platform enable** 例： `RP/0/RSP0/CPU0:PE51_ASR-9010(admin-config)#no svd platform enable`	選択的 VRF ダウンロードを無効にします。
ステップ 4	**commit**	

ステップ 5	**show svd state**
	例：
	`RP/0/RSP0/cpu 0: router#show svd state`
	`Selective VRF Download (SVD) Feature State:`
	`SVD Configuration State Unsupported`
	`SVD Operational State Unsupported`
		選択的VRFダウンロード機能の状態情報を表示します。

	コマンドまたはアクション	目的
ステップ 6	**admin** 例： `RP/0/RSP0/cpu 0: router#admin`	管理者モードを開始します。

ステップ 7	**reload location all**
	例：
	`RP/0/RSP0/cpu 0: router(admin)#reload loc all`
	`Tue Feb 12 07:51:25.279 UTC`
	
	`Preparing system for backup. This may take a few minutes especially for large configurations.`
	`Status report: node0_RSP0_CPU0: START TO BACKUP`
	`Status report: node0_RSP0_CPU0: BACKUP HAS COMPLETED SUCCESSFULLY`
	`[Done]`
	`Proceed with reload? [confirm]RP/0/RSP0/CPU0::This node received reload`
		シャーシをリロードします。

ステップ 8	**exit** 例： `RP/0/RSP0/cpu 0: router(admin)#exit`	管理者 EXEC モードを終了し、EXEC モードを開始します。

ステップ 9	**show svd role**
	例：
	`RP/0/RSP0/cpu 0: router#show svd role`
	
	`Codes: (C) : user Configured role`
	`Node Name IPv4 Role IPv6 Role`
	`0/RSP0/CPU0 Standard Standard`
	`0/0/CPU0 Standard Standard`
	`0/1/CPU0 Standard Standard`
		VRF インターフェイスがあるラインカードに SVD ロールが「標準」であることを確認することで、選択的VRF ダウンロードが非アクティブになっているかどうかを確認します。

復元力のある CE 単位のラベルモードの設定

VRF アドレスファミリでの復元力のある CE 単位のラベルモードの設定

VRF アドレスファミリに復元力のある CE 単位のラベルモードを設定するには、次のタスクを実行します。

（注） 復元力のある CE 単位の 6PE ラベル割り当ては、CRS-1 ルータと CRS-3 ルータではサポートされていません。ASR 9000 ルータでのみサポートされています。

手順の概要

1. **configure**
2. **router bgp***as-number*
3. **vrf***vrf-instance*
4. **address-family {ipv4 | ipv6} unicast**
5. **label mode per-ce**
6. 次のいずれかを実行します。
   - **end**
   - **commit**

手順の詳細

ステップ 1 **configure**

例：

```
RP/0/RSP0/cpu 0: router# configure
RP/0/RSP0/cpu 0: router(config)#
```

グローバル コンフィギュレーション モードを開始します。

ステップ 2 **router bgp***as-number*

例：

```
RP/0/RSP0/cpu 0: router(config)# router bgp 666
RP/0/RSP0/cpu 0: router(config-bgp)#
```

自律システム番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。

ステップ 3 **vrf***vrf-instance*

例：

```
RP/0/RSP0/cpu 0: router(config-bgp)# vrf vrf-pe
RP/0/RSP0/cpu 0: router(config-bgp-vrf)#
```

VRF インスタンスを設定します。

ステップ 4 **address-family {ipv4 | ipv6} unicast**

例：

```
RP/0/RSP0/cpu 0: router(config-bgp-vrf)# address-family ipv4 unicast
RP/0/RSP0/cpu 0: router(config-bgp-vrf-af)#
```

IPv4 または IPv6 のいずれかのアドレス ファミリ ユニキャストを指定し、アドレス ファミリのコンフィギュレーション サブモードを開始します。

ステップ 5 **label mode per-ce**

例：

```
RP/0/RSP0/cpu 0: router(config-bgp-vrf-af)# label mode per-ce
RP/0/RSP0/cpu 0: router(config-bgp-vrf-af)#
```

復元力のある CE 単位のラベルモードを設定します。

ステップ 6 次のいずれかを実行します。

- **end**
- **commit**

例：

```
RP/0/RSP0/cpu 0: router(config-bgp-vrf-af)# end
```

または

```
RP/0/RSP0/cpu 0: router(config-bgp-vrf-af)# commit
```

設定変更を保存します。

- **end** コマンドを実行すると、変更をコミットするように要求されます。

```
Uncommitted changes found, commit them before exiting(yes/no/cancel)?[cancel]:
```

  - **yes** と入力すると、実行コンフィギュレーション ファイルに変更が保存され、コンフィギュレーション セッションが終了して、ルータが EXEC モードに戻ります。
  - **no** と入力すると、コンフィギュレーションセッションが終了して、ルータが EXEC モードに戻ります。変更はコミットされません。
  - **cancel** と入力すると、現在のコンフィギュレーションセッションが継続します。コンフィギュレーション セッションは終了せず、設定変更もコミットされません。

- 実行コンフィギュレーションファイルに変更を保存し、コンフィギュレーションセッションを継続するには、**commit** コマンドを使用します。

ルートポリシーを使用した復元力のある CE 単位のラベルモードの設定

ルートポリシーを使用して復元力のある CE 単位のラベルモードを設定するには、次のタスクを実行します。

（注） 復元力のある CE 単位の 6PE ラベル割り当ては、CRS-1 ルータと CRS-3 ルータではサポートされていません。ASR 9000 ルータでのみサポートされています。

手順の概要

1. **configure**
2. **route-policy** *policy-name*
3. **set label mode per-ce**
4. 次のいずれかを実行します。
   - **end**
   - **commit**

手順の詳細

ステップ 1 **configure**

例：

```
RP/0/RSP0/cpu 0: router# configure
RP/0/RSP0/cpu 0: router(config)#
```

グローバル コンフィギュレーション モードを開始します。

ステップ 2 **route-policy** *policy-name*

例：

```
RP/0/RSP0/cpu 0: router(config)# route-policy route1
RP/0/RSP0/cpu 0: router(config-rpl)#
```

ルート ポリシーを作成し、ルート ポリシー コンフィギュレーション モードを開始します。

ステップ 3 **set label mode per-ce**

例：

```
RP/0/RSP0/cpu 0: router(config-rpl)# set label mode per-ce
RP/0/RSP0/cpu 0: router(config-rpl)#
```

復元力のある CE 単位のラベルモードを設定します。

ステップ 4 次のいずれかを実行します。

- **end**
- **commit**

例：

```
RP/0/RSP0/cpu 0: router(config-rpl)# end
```

または

```
RP/0/RSP0/cpu 0: router(config-rpl)# commit
```

設定変更を保存します。

- **end** コマンドを実行すると、変更をコミットするように要求されます。

```
Uncommitted changes found, commit them before exiting(yes/no/cancel)?[cancel]:
```

- **yes** と入力すると、実行コンフィギュレーション ファイルに変更が保存され、コンフィギュレーション セッションが終了して、ルータが EXEC モードに戻ります。
- **no** と入力すると、コンフィギュレーションセッションが終了して、ルータが EXEC モードに戻ります。変更はコミットされません。
- **cancel** と入力すると、現在のコンフィギュレーションセッションが継続します。コンフィギュレーション セッションは終了せず、設定変更もコミットされません。

- 実行コンフィギュレーションファイルに変更を保存し、コンフィギュレーションセッションを継続するには、**commit** コマンドを使用します。

BGP の実装の設定例

ここでは、次の設定例について説明します。

BGP のイネーブル化：例

次に、BGP をイネーブルにする例を示します。

```
prefix-set static
   2020::/64,
   2012::/64,
   10.10.0.0/16,
   10.2.0.0/24
end-set

route-policy pass-all
  pass
end-policy
route-policy set_next_hop_agg_v4
  set next-hop 10.0.0.1
end-policy

route-policy set_next_hop_static_v4
  if (destination in static) then
    set next-hop 10.1.0.1
  else
    drop
  endif
end-policy

route-policy set_next_hop_agg_v6
  set next-hop 2003::121
end-policy

route-policy set_next_hop_static_v6
  if (destination in static) then
     set next-hop 2011::121
```

```
  else
     drop
  endif
end-policy

router bgp 65000
  bgp fast-external-fallover disable
  bgp confederation peers
    65001
    65002
  bgp confederation identifier 1
  bgp router-id 1.1.1.1
  address-family ipv4 unicast
    aggregate-address 10.2.0.0/24 route-policy set_next_hop_agg_v4
    aggregate-address 10.3.0.0/24
    redistribute static route-policy set_next_hop_static_v4
  address-family ipv4 multicast
    aggregate-address 10.2.0.0/24 route-policy set_next_hop_agg_v4
    aggregate-address 10.3.0.0/24
    redistribute static route-policy set_next_hop_static_v4
  address-family ipv6 unicast
    aggregate-address 2012::/64 route-policy set_next_hop_agg_v6
    aggregate-address 2013::/64
    redistribute static route-policy set_next_hop_static_v6
  address-family ipv6 multicast
    aggregate-address 2012::/64 route-policy set_next_hop_agg_v6
    aggregate-address 2013::/64
    redistribute static route-policy set_next_hop_static_v6
  neighbor 10.0.101.60
    remote-as 65000
    address-family ipv4 unicast
    address-family ipv4 multicast
  neighbor 10.0.101.61
    remote-as 65000
    address-family ipv4 unicast
    address-family ipv4 multicast
  neighbor 10.0.101.62
    remote-as 3
    address-family ipv4 unicast
      route-policy pass-all in
      route-policy pass-all out
    address-family ipv4 multicast
      route-policy pass-all in
      route-policy pass-all out
  neighbor 10.0.101.64
    remote-as 5
    update-source Loopback0
    address-family ipv4 unicast
      route-policy pass-all in
      route-policy pass-all out
    address-family ipv4 multicast
      route-policy pass-all in
      route-policy pass-all out
```

BGP アップデート グループの表示：例

次に、EXEC コンフィギュレーション モードで実行された **show bgp update-group** コマンドの出力例を示します。

```
show bgp update-group

 Update group for IPv4 Unicast, index 0.1:
   Attributes:
     Outbound Route map:rm
     Minimum advertisement interval:30
   Messages formatted:2, replicated:2
   Neighbors in this update group:
     10.0.101.92

 Update group for IPv4 Unicast, index 0.2:
   Attributes:
     Minimum advertisement interval:30
   Messages formatted:2, replicated:2
   Neighbors in this update group:
     10.0.101.91
```

BGP ネイバー設定：例

情報を共有するように自律システムの BGP ネイバーを設定する例を次に示します。この例ではBGP ルータを自律システム 109 に割り当て、自律システムの送信元として 2 つのネットワークのリストが表示される例を示します。3 つのリモートルータ（とその自律システム）のアドレスのリストが表示されます。設定したルータは、ネットワーク 172 .16 .0.0 と 192.168 .7.0 と隣接ルータに関する情報を共有します。リストの 1 番目のルータは別の自律システムにあり、2 番目の **neighbor** および **remote-as** コマンドによってアドレス 172 .26 .234.2; の内部ネイバーが（同じ自律システム番号を使用して）指定され、3 番目の **neighbor** および **remote-as** コマンドによって別の自律システムのネイバーが指定されます。

```
route-policy pass-all
 pass
end-policy
router bgp 109
 address-family ipv4 unicast
  network 172.16.0.0 255.255.0.0
  network 192.168.7.0 255.255.0.0
  neighbor 172.16.200.1
   remote-as 167
   exit
 address-family ipv4 unicast
  route-policy pass-all in
  route-policy pass-out out
  neighbor 172.26.234.2
   remote-as 109
   exit
 address-family ipv4 unicast
  neighbor 172.26.64.19
   remote-as 99
   exit
 address-family ipv4 unicast
  route-policy pass-all in
  route-policy pass-all out
```

BGP コンフェデレーション：例

次に、コンフェデレーションのいくつかのピアを表示する設定の例を示します。このコンフェデレーションは、自律システム番号 6001、6002、および 6003 の 3 つの内部自律システムから構成されています。コンフェデレーション外の BGP スピーカーには、このコンフェデレーションは（**bgp confederation identifier** コマンドによって指定される）自律システム番号 666 を持つ通常の自律システムのように見えます。

自律システム 6001 の BGP スピーカーで、**bgp confederation peers** コマンドは、自律システム 6002 および 6003 からのピアを特別な eBGP ピアとしてマークします。したがって、ピア 171.16 .232.55 および 171。16 .232.56 は、この更新でローカルプリファレンス、ネクスト ホップ、および未変更の MED を取得します。171 のルータ。19 .69.1 のルータは通常の eBGP スピーカーであり、このピアからの更新は、自律システム 666 のピアから受け取る通常の eBGP 更新とまったく同じです。

```
router bgp 6001
 bgp confederation identifier 666
 bgp confederation peers
  6002
  6003
   exit
 address-family ipv4 unicast
  neighbor 171.16.232.55
  remote-as 6002
   exit
 address-family ipv4 unicast
  neighbor 171.16.232.56
  remote-as 6003
   exit
 address-family ipv4 unicast
  neighbor 171.19.69.1
  remote-as 777
```

自律システム 6002 の BGP スピーカーでは、自律システム 6001 および 6003 からのピアは特別な eBGP ピアとして設定されます。ピア 171 。17 .70.1 のピアは通常の iBGP ピアであり、ピア 199.99.99.2 は自律システム 700 の通常の eBGP ピアです。

```
router bgp 6002
 bgp confederation identifier 666
 bgp confederation peers
  6001
  6003
   exit
 address-family ipv4 unicast
  neighbor 171.17.70.1
   remote-as 6002
   exit
 address-family ipv4 unicast
  neighbor 171.19.232.57
   remote-as 6001
   exit
 address-family ipv4 unicast
  neighbor 171.19.232.56
   remote-as 6003
   exit
```

```
 address-family ipv4 unicast
  neighbor 171.19.99.2
   remote-as 700
   exit
 address-family ipv4 unicast
  route-policy pass-all in
  route-policy pass-all out
```

自律システム 6003 の BGP スピーカーでは、自律システム 6001 および 6002 からのピアは特別な eBGP ピアとして設定されます。ピア 192 。168 .200.200 のピアは、自律システム 701 の通常の eBGP ピアです。

```
router bgp 6003
 bgp confederation identifier 666
 bgp confederation peers
  6001
  6002
   exit
 address-family ipv4 unicast
  neighbor 171.19.232.57
   remote-as 6001
   exit
 address-family ipv4 unicast
  neighbor 171.19.232.55
   remote-as 6002
   exit
 address-family ipv4 unicast
  neighbor 192.168.200.200
   remote-as 701
   exit
 address-family ipv4 unicast
  route-policy pass-all in
  route-policy pass-all out
```

下記は、同じ例の自律システム 701 の BGP スピーカー 192 .168 .200.205 から受信する設定の一部を示します。ネイバー 171.16 .232.56 は自律システム 666 の通常の eBGP スピーカーとして設定されます。コンフェデレーション外部のピアは、この自律システムが複数の自律システムに内部分割されることを認識しません。

```
router bgp 701
 address-family ipv4 unicast
  neighbor 172.16.232.56
   remote-as 666
   exit
 address-family ipv4 unicast
  route-policy pass-all in
  route-policy pass-all out
   exit
 address-family ipv4 unicast
  neighbor 192.168.200.205
   remote-as 701
```

BGP ルートリフレクタ：例

次に、アドレスファミリを使用して、内部BGP ピア 10.1.1.1 をユニキャストプレフィックスとマルチキャスト プレフィックスの両方のルート リフレクタ クライアントとして設定する例を示します。

```
router bgp 140
 address-family ipv4 unicast
  neighbor 10.1.1.1
   remote-as 140
   address-family ipv4 unicast
    route-reflector-client
    exit
   address-family ipv4 multicast
    route-reflector-client
```

BGP ノンストップルーティング設定：例

次に、BGP NSR を有効にする例を示します。

```
configure
router bgp 120
nsr
end
```

次に、BGP NSR をディセーブルにする例を示します。

```
configure
router bgp 120
no nsr
end
```

プライマリバックアップパスのインストール：例

次に、プライマリバックアップパスのインストールを有効にする例を示します。

```
 router bgp 120
address-family ipv4 unicast
 additional-paths receive
 additional-paths send
 additional-paths selection route-policy bgp_add_path
!
!
end
```

ローカル ラベル割り当ての保持：例

次に、プライマリ PE のプライマリ パスに以前に割り当てたローカル ラベルを再コンバージェンス後 10 分にわたって維持する例を示します。

```
router bgp 100
address-family l2vpn vpls-vpws
  retain local-label 10
end
```

iBGP マルチパス負荷共有設定：例

次に、負荷共有に 30 のパスが使用されている設定の例を示します。

```
router bgp 100
 address-family ipv4 multicast
  maximum-paths ibgp 30
 !
!
end
```

過剰パスの破棄の設定：例

次に、IPv4 アドレス ファミリに対する過剰パスの破棄機能を設定する例を示します。

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# router bgp 10
RP/0/RSP0/CPU0:router(config-bgp)# neighbor 10.0.0.1
RP/0/RSP0/CPU0:router(config-bgp-nbr)# address-family ipv4 unicast
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# maximum-prefix 1000 discard-extra-paths
RP/0/RSP0/CPU0:router(config-bgp-vrf-af)# commit
```

過剰パス情報の破棄の表示：例

次の画面出力では、過剰パスの破棄オプションの詳細を示しています。

```
RP/0/0/CPU0:ios# show bgp neighbor 10.0.0.1

BGP neighbor is 10.0.0.1
Remote AS 10, local AS 10, internal link
Remote router ID 0.0.0.0
BGP state = Idle (No best local address found)
Last read 00:00:00, Last read before reset 00:00:00
Hold time is 180, keepalive interval is 60 seconds
Configured hold time: 180, keepalive: 60, min acceptable hold time: 3
Last write 00:00:00, attempted 0, written 0
Second last write 00:00:00, attempted 0, written 0
Last write before reset 00:00:00, attempted 0, written 0
Second last write before reset 00:00:00, attempted 0, written 0
Last write pulse rcvd not set last full not set pulse count 0
```

```
Last write pulse rcvd before reset 00:00:00
Socket not armed for io, not armed for read, not armed for write
Last write thread event before reset 00:00:00, second last 00:00:00
Last KA expiry before reset 00:00:00, second last 00:00:00
Last KA error before reset 00:00:00, KA not sent 00:00:00
Last KA start before reset 00:00:00, second last 00:00:00
Precedence: internet
Multi-protocol capability not received
Received 0 messages, 0 notifications, 0 in queue
Sent 0 messages, 0 notifications, 0 in queue
Minimum time between advertisement runs is 0 secs

For Address Family: IPv4 Unicast
BGP neighbor version 0
Update group: 0.1 Filter-group: 0.0 No Refresh request being processed
Route refresh request: received 0, sent 0
0 accepted prefixes, 0 are bestpaths
Cumulative no. of prefixes denied: 0.
Prefix advertised 0, suppressed 0, withdrawn 0
Maximum prefixes allowed 10 (discard-extra-paths) <<<<<<<<<<<<<<<<<<<<<
Threshold for warning message 75%, restart interval 0 min
AIGP is enabled
An EoR was not received during read-only mode
Last ack version 1, Last synced ack version 0
Outstanding version objects: current 0, max 0
Additional-paths operation: None
Send Multicast Attributes

Connections established 0; dropped 0
Local host: 0.0.0.0, Local port: 0, IF Handle: 0x00000000
Foreign host: 10.0.0.1, Foreign port: 0
Last reset 00:00:00
```

ネイバー単位の TCP MSS の設定：例

次に、ネイバー単位の TCP MSS を設定する方法、ネイバー単位の TCP MSS を無効にする方法、および TCP MSS の設定を解除する方法の例を示します。

トポロジのシナリオ

次の図に、ネイバー単位の TCP MSS 設定の基本的なシナリオを示します。

R1 の設定は次のようになります。

```
router bgp 1
  bgp router-id 10.0.0.1
  address-family ipv4 unicast
  !
  neighbor-group n1
    tcp mss 100
    address-family ipv4 unicast
  !
 !
```

```
  neighbor 10.0.0.2
    remote-as 1
    use neighbor-group n1
    address-family ipv4 unicast
  !
 !
!
```

R2 の設定は次のようになります。

```
router bgp 1
  bgp router-id 10.0.0.2
  address-family ipv4 unicast
  !
  neighbor 10.0.0.1
    remote-as 1
    address-family ipv4 unicast
  !
 !
!
```

ネイバー単位の TCP MSS の設定: 例

次に、ネイバーグループにネイバー単位の TCP MSS を設定する例を示します。

```
router bgp 1
bgp router-id 10.0.0.1
address-family ipv4 unicast
!
neighbor-group n1
tcp mss 500
address-family ipv4 unicast
!
!
neighbor 10.0.0.2
remote-as 1
use neighbor-group n1
address-family ipv4 unicast
!
!
!
!
end
```

ネイバー単位の TCP MSS の無効化：例

次に、ネイバーグループに TCP MSS を設定し、TCP MSS 値を継承するネイバーのいずれかで継承の無効化を設定する例を示します。

```
router bgp 1
bgp router-id 10.0.0.1
address-family ipv4 unicast
!
neighbor-group n1
tcp mss 500
address-family ipv4 unicast
!
```

```
!
neighbor 10.0.0.2
remote-as 1
use neighbor-group n1
tcp mss inheritance-disable
address-family ipv4 unicast
!
!
!
!
end
```

TCP MSS の設定解除：例

次に、TCP MSS の設定を解除する例を示します。

```
RP/0/0/CPU0:ios(config)#router bgp 1
RP/0/0/CPU0:ios(config-bgp)#neighbor-group n1
RP/0/0/CPU0:ios(config-bgp-nbrgrp)#no tcp mss 500
RP/0/0/CPU0:ios(config-bgp-nbrgrp)#commit
```

ネイバー単位の TCP MSS の確認：例

次に、ルータのネイバー単位の TCP MSS 機能を確認する例を示します。

```
RP/0/0/CPU0:ios#show bgp neighbor 10.0.0.2

BGP neighbor is 10.0.0.2
Remote AS 1, local AS 1, internal link
Remote router ID 10.0.0.2
BGP state = Established, up for 00:09:17
Last read 00:00:16, Last read before reset 00:00:00
Hold time is 180, keepalive interval is 60 seconds
Configured hold time: 180, keepalive: 60, min acceptable hold time: 3
Last write 00:00:16, attempted 19, written 19
Second last write 00:01:16, attempted 19, written 19
Last write before reset 00:00:00, attempted 0, written 0
Second last write before reset 00:00:00, attempted 0, written 0
Last write pulse rcvd Dec 7 11:58:42.411 last full not set pulse count 23
Last write pulse rcvd before reset 00:00:00
Socket not armed for io, armed for read, armed for write
Last write thread event before reset 00:00:00, second last 00:00:00
Last KA expiry before reset 00:00:00, second last 00:00:00
Last KA error before reset 00:00:00, KA not sent 00:00:00
Last KA start before reset 00:00:00, second last 00:00:00
Precedence: internet
Multi-protocol capability received
Neighbor capabilities:
Route refresh: advertised (old + new) and received (old + new)
Graceful Restart (GR Awareness): advertised and received
4-byte AS: advertised and received
Address family IPv4 Unicast: advertised and received
Received 12 messages, 0 notifications, 0 in queue
Sent 12 messages, 0 notifications, 0 in queue
Minimum time between advertisement runs is 0 secs
TCP Maximum Segment Size 500

For Address Family: IPv4 Unicast
```

```
BGP neighbor version 4
Update group: 0.2 Filter-group: 0.1 No Refresh request being processed
Route refresh request: received 0, sent 0
0 accepted prefixes, 0 are bestpaths
Cumulative no. of prefixes denied: 0.
Prefix advertised 0, suppressed 0, withdrawn 0
Maximum prefixes allowed 1048576
Threshold for warning message 75%, restart interval 0 min
AIGP is enabled
An EoR was received during read-only mode
Last ack version 4, Last synced ack version 0
Outstanding version objects: current 0, max 0
Additional-paths operation: None
Send Multicast Attributes
```

次に、TCP MSS の設定を確認する例を示します。

```
RP/0/0/CPU0:ios#show bgp neighbor 10.0.0.2 configuration

neighbor 10.0.0.2
remote-as 1 []
tcp-mss 400 [n:n1]
address-family IPv4 Unicast []
```

次に、TCP 接続エンドポイントの情報を表示する例を示します。

```
RP/0/0/CPU0:ios#show tcp brief

  PCB      VRF-ID     Recv-Q Send-Q Local Address          Foreign Address        State
0x08789b28 0x60000000      0      0  :::179                  :::0
LISTEN
0x08786160 0x00000000      0      0  :::179                  :::0
LISTEN
0xecb0c9f8 0x60000000      0      0  10.0.0.1:12404          10.0.0.2:179           ESTAB
0x0878b168 0x60000000      0      0  11.0.0.1:179            11.0.0.2:61177         ESTAB
0xecb0c6b8 0x60000000      0      0  0.0.0.0:179             0.0.0.0:0
LISTEN
0x08781590 0x00000000      0      0  0.0.0.0:179             0.0.0.0:0
LISTEN
```

次に、特定の PCB 値について TCP 接続情報を表示する例を示します。

```
RP/0/0/CPU0:ios#show tcp pcb 0xecb0c9f8

Connection state is ESTAB, I/O status: 0, socket status: 0
Established at Sun Dec 7 11:49:39 2014

PCB 0xecb0c9f8, SO 0xecb01b68, TCPCB 0xecb01d78, vrfid 0x60000000,
Pak Prio: Medium, TOS: 192, TTL: 255, Hash index: 1322
Local host: 10.0.0.1, Local port: 12404 (Local App PID: 19840)
Foreign host: 10.0.0.2, Foreign port: 179

Current send queue size in bytes: 0 (max 24576)
Current receive queue size in bytes: 0 (max 32768) mis-ordered: 0 bytes
Current receive queue size in packets: 0 (max 0)

Timer Starts Wakeups Next(msec)
Retrans 17 2 0
SendWnd 0 0 0
```

```
TimeWait 0 0 0
AckHold 13 5 0
KeepAlive 1 0 0
PmtuAger 0 0 0
GiveUp 0 0 0
Throttle 0 0 0

iss: 1728179225 snduna: 1728179536 sndnxt: 1728179536
sndmax: 1728179536 sndwnd: 32517 sndcwnd: 1000
irs: 2055835995 rcvnxt: 2055836306 rcvwnd: 32536 rcvadv: 2055868842

SRTT: 206 ms, RTTO: 300 ms, RTV: 59 ms, KRTT: 0 ms
minRTT: 10 ms, maxRTT: 230 ms

ACK hold time: 200 ms, Keepalive time: 0 sec, SYN waittime: 30 sec
Giveup time: 0 ms, Retransmission retries: 0, Retransmit forever: FALSE
Connect retries remaining: 30, connect retry interval: 30 secs

State flags: none
Feature flags: Win Scale, Nagle
Request flags: Win Scale

Datagrams (in bytes): MSS 500, peer MSS 1460, min MSS 500, max MSS 1460

Window scales: rcv 0, snd 0, request rcv 0, request snd 0
Timestamp option: recent 0, recent age 0, last ACK sent 0
Sack blocks {start, end}: none
Sack holes {start, end, dups, rxmit}: none

Socket options: SO_REUSEADDR, SO_REUSEPORT, SO_NBIO
Socket states: SS_ISCONNECTED, SS_PRIV
Socket receive buffer states: SB_DEL_WAKEUP
Socket send buffer states: SB_DEL_WAKEUP
Socket receive buffer: Low/High watermark 1/32768
Socket send buffer : Low/High watermark 2048/24576, Notify threshold 0

PDU information:
#PDU's in buffer: 0
FIB Lookup Cache: IFH: 0x200 PD ctx: size: 0 data:
Num Labels: 0 Label Stack:
```

AiGP によるプレフィックスの生成：例

次に、AiGP メトリック属性を使用してプレフィックスを生成するための設定例を示します。

```
route-policy aigp-policy
  set aigp-metric 4
  set aigp-metric igp-cost
end-policy
!
router bgp 100
 address-family ipv4 unicast
  network 10.2.3.4/24 route-policy aigp-policy
  redistribute ospf osp1 metric 4 route-policy aigp-policy
 !
!
end
```

BGP Accept Own の設定：例

次に、BGP Accept Own を PE ルータに設定する例を示します。

```
router bgp 100
 neighbor 45.1.1.1
   remote-as 100
   update-source Loopback0
   address-family vpnv4 unicast
    route-policy pass-all in
    accept-own
    route-policy drop_111.x.x.x out
   !
   address-family vpnv6 unicast
    route-policy pass-all in
    accept-own
    route-policy drop_111.x.x.x out
   !
  !
```

次の例は、BGP Accept Own のための InterAS-RR の設定を示しています。

```
router bgp 100
 neighbor 45.1.1.1
  remote-as 100
  update-source Loopback0
  address-family vpnv4 unicast
   route-policy rt_stitch1 in
   route-reflector-client
   route-policy add_bgp_ao out
  !
  address-family vpnv6 unicast
   route-policy rt_stitch1 in
   route-reflector-client
   route-policy add_bgp_ao out
  !
 !
extcommunity-set rt cs_100:1
  100:1
end-set
!
extcommunity-set rt cs_1001:1
  1001:1
end-set
!
route-policy rt_stitch1
  if extcommunity rt matches-any cs_100:1 then
    set extcommunity rt cs_1000:1 additive
 endif
end-policy
!
route-policy add_bgp_ao
  set community (accept-own) additive
end-policy
!
```

BGP 不等コストの連続ロード バランシング：例

次に、不等コストの連続ロード バランシングの設定例を示します。

```
interface Loopback0
 ipv4 address 20.20.20.20 255.255.255.255
!
interface MgmtEth0/RSP0/CPU0/0
 ipv4 address 8.43.0.10 255.255.255.0
!
interface TenGigE0/3/0/0
 bandwidth 8000000
 ipv4 address 11.11.11.11 255.255.255.0
 ipv6 address 11:11:0:1::11/64
!
interface TenGigE0/3/0/1
 bandwidth 7000000
 ipv4 address 11.11.12.11 255.255.255.0
 ipv6 address 11:11:0:2::11/64
!
interface TenGigE0/3/0/2
 bandwidth 6000000
 ipv4 address 11.11.13.11 255.255.255.0
 ipv6 address 11:11:0:3::11/64
!
interface TenGigE0/3/0/3
 bandwidth 5000000
 ipv4 address 11.11.14.11 255.255.255.0
 ipv6 address 11:11:0:4::11/64
!
interface TenGigE0/3/0/4
 bandwidth 4000000
 ipv4 address 11.11.15.11 255.255.255.0
 ipv6 address 11:11:0:5::11/64
!
interface TenGigE0/3/0/5
 bandwidth 3000000
 ipv4 address 11.11.16.11 255.255.255.0
 ipv6 address 11:11:0:6::11/64
!
interface TenGigE0/3/0/6
 bandwidth 2000000
 ipv4 address 11.11.17.11 255.255.255.0
 ipv6 address 11:11:0:7::11/64
!
interface TenGigE0/3/0/7
 bandwidth 1000000
 ipv4 address 11.11.18.11 255.255.255.0
 ipv6 address 11:11:0:8::11/64
!
interface TenGigE0/4/0/0
 description CONNECTED TO IXIA 1/3
 transceiver permit pid all
!
interface TenGigE0/4/0/2
 ipv4 address 9.9.9.9 255.255.0.0
 ipv6 address 9:9::9/64
 ipv6 enable
!
route-policy pass-all
  pass
end-policy
!
router static
 address-family ipv4 unicast
  202.153.144.0/24 8.43.0.1
 !
!
```

```
router bgp 100
 bgp router-id 20.20.20.20
 address-family ipv4 unicast
  maximum-paths eibgp 8
  redistribute connected
 !
 neighbor 11.11.11.12
  remote-as 200
  dmz-link-bandwidth
  address-family ipv4 unicast
   route-policy pass-all in
   route-policy pass-all out
  !
 !
 neighbor 11.11.12.12
  remote-as 200
  dmz-link-bandwidth
  address-family ipv4 unicast
   route-policy pass-all in
   route-policy pass-all out
  !
 !
 neighbor 11.11.13.12
  remote-as 200
  dmz-link-bandwidth
  address-family ipv4 unicast
   route-policy pass-all in
   route-policy pass-all out
  !
 !
 neighbor 11.11.14.12
  remote-as 200
  dmz-link-bandwidth
  address-family ipv4 unicast
   route-policy pass-all in
   route-policy pass-all out
  !
 !
 neighbor 11.11.15.12
  remote-as 200
  dmz-link-bandwidth
  address-family ipv4 unicast
   route-policy pass-all in
   route-policy pass-all out
  !
 !
 neighbor 11.11.16.12
  remote-as 200
  dmz-link-bandwidth
  address-family ipv4 unicast
   route-policy pass-all in
   route-policy pass-all out
  !
 !
 neighbor 11.11.17.12
  remote-as 200
  dmz-link-bandwidth
  address-family ipv4 unicast
   route-policy pass-all in
   route-policy pass-all out
  !
 !
 neighbor 11.11.18.12
  remote-as 200
```

```
  dmz-link-bandwidth
  address-family ipv4 unicast
   route-policy pass-all in
   route-policy pass-all out
  !
 !
!
end
```

VRF ダイナミック ルートの設定：例

次に、VRF ダイナミックルートリークを設定する例を示します。

デフォルトの VRF からデフォルト以外の VRF へのルートのインポート

```
vrf vrf_1
 address-family ipv6 unicast
  import from default-vrf route-policy rpl_dynamic_route_import
 !
end
```

非デフォルト VRF からデフォルト VRF へのルートのインポート：

```
vrf vrf_1
 address-family ipv6 unicast
    export to default-vrf route-policy rpl_dynamic_route_export
 !
end
```

復元力のある CE 単位のラベルモードの設定：例

VRF アドレスファミリでの復元力のある CE 単位のラベルモードの設定：例

次に、VRF アドレス ファミリに復元力のある CE 単位のラベルモードを設定する例を示します。

```
RP/0/RSP0/cpu 0: router# configure
RP/0/RSP0/cpu 0: router(config)# router bgp 666
RP/0/RSP0/cpu 0: router(config-bgp)# vrf vrf-pe
RP/0/RSP0/cpu 0: router(config-bgp-vrf)# address-family ipv4 unicast
RP/0/RSP0/cpu 0: router(config-bgp-vrf-af)# label mode per-ce
RP/0/RSP0/cpu 0: router(config-bgp-vrf-af)# end
```

ルートポリシーを使用した復元力のある CE 単位のラベルモードの設定：例

次に、ルートポリシーを使用してふぐ元力のある CE 単位のラベルモードを設定する例を示します。

```
RP/0/RSP0/cpu 0: router# configure
RP/0/RSP0/cpu 0: router(config)# route-policy route1
RP/0/RSP0/cpu 0: router(config-rpl)# set label mode per-ce
RP/0/RSP0/cpu 0: router(config-rpl)# end
```

フロー タグの伝達

フロー タグ伝達機能では、ルート ポリシーとユーザ ポリシー間に相関関係を構築できます。BGP を使用したフロー タグ伝達では、AS 番号、プレフィックス リスト、コミュニティ文字列、および拡張コミュニティなどのルーティング属性に基づいてユーザ側でトラフィックをステアリングできます。フロー タグは論理数値識別子で、FIB ルックアップ テーブル内の FIB エントリのルーティング属性の 1 つとして RIB を通じて配布されます。フロー タグは、RPL からの「set」操作を使用してインスタンス化され、フロー タグ値に対してアクション（ポリシールール）が関連付けられている C3PL PBR ポリシーで参照されます。

フロー タグの伝達は次の場合に使用できます。

- 宛先 IP アドレス（コミュニティ番号を使用）またはプレフィックス（コミュニティ番号または AS 番号を使用）に基づいてトラフィックを分類する。
- カスタマー サイトのサービス レベル契約（SLA）に基づくサービス エッジに到達するパスのコストに合致する TE グループを選択する。
- SLA とそのクライアントに基づいて、特定のカスタマーにトラフィック ポリシー（TE グループの選択）を適用する。
- アプリケーション サーバまたはキャッシュ サーバにトラフィックを迂回させる。

フロー タグ伝達の制限

Border Gateway Protocol を使用した QoS ポリシー伝達（QPPB）とフロー タグ機能の併用については、いくつかの制限があります。次の作業を行います。

- ルート ポリシーには、「set qos-group」または「set flow-tag」のいずれかを使用できますが、prefix-set に両方は使用できません。
- qos-group と route policy flow-tag のルート ポリシーに重複するルートは使用できません。QPPB とフロー タグの機能は、それらが使用するルート ポリシーに重複するルートがない場合に限り、（同じインターフェイス上でも、異なるインターフェイス上でも）共存できます。
- ルート ポリシーとポリシー マップに qos-group と flow-tag を混在させて使用することはお勧めしません。

次の作業

BGP コマンドの詳細については、 *Routing Command Reference for Cisco ASR 9000 Series Routers* を参照してください。

その他の参考資料

ここでは、BGP の実装に関する関連資料について説明します。

関連資料

関連項目	マニュアル タイトル
BGP コマンド：コマンド構文の詳細、コマンド モード、コマンド履歴、デフォルト設定、使用上のガイドライン、および例	*Routing Command Reference for Cisco ASR 9000 Series Routers*
シスコ エクスプレス フォワーディング（CEF）コマンド：詳細なコマンド構文、コマンド モード、コマンド履歴、デフォルト、使用上のガイドライン、および例	*IP Addresses and Services Command Reference for Cisco ASR 9000 Series Routers*
MPLS VPN 設定情報。	*MPLS Configuration Guide for Cisco ASR 9000 Series RoutersMPLS Configuration Guide for Cisco NCS 560 Series Routers*
双方向フォワーディング検出（BFD）	*Interface and Hardware Component Configuration Guide for Cisco ASR 9000 Series Routers* および *Interface and Hardware Component Command Reference for Cisco ASR 9000 Series Routers*
タスク ID 情報。	*System Security Configuration Guide for Cisco ASR 9000 Series Routers*の「Configuring AAA Services on Cisco ASR 9000 Series Router」のモジュール

標準

標準	タイトル
draft-bonica-tcp-auth-05.txt	『*Authentication for TCP-based Routing and Management Protocols*』（R. Bonica、B. Weis、S. Viswanathan、A. Lange、O. Wheeler）
draft-ietf-idr-bgp4-26.txt	『*A Border Gateway Protocol 4*』（Y. Rekhter、T.Li、S. Hares）
draft-ietf-idr-bgp4-mib-15.txt	『*Definitions of Managed Objects for the Fourth Version of Border Gateway Protocol (BGP-4)*』（J. Hass、S. Hares）
draft-ietf-idr-cease-subcode-05.txt	『*Subcodes for BGP Cease Notification Message*』（Enke Chen、V. Gillet）
draft-ietf-idr-avoid-transition-00.txt	『*Avoid BGP Best Path Transitions from One External to Another*』（Enke Chen、Srihari Sangli）
draft-ietf-idr-as4bytes-12.txt	『*BGP Support for Four-octet AS Number Space*』（Quaizar Vohra、Enke Chen）

MIB

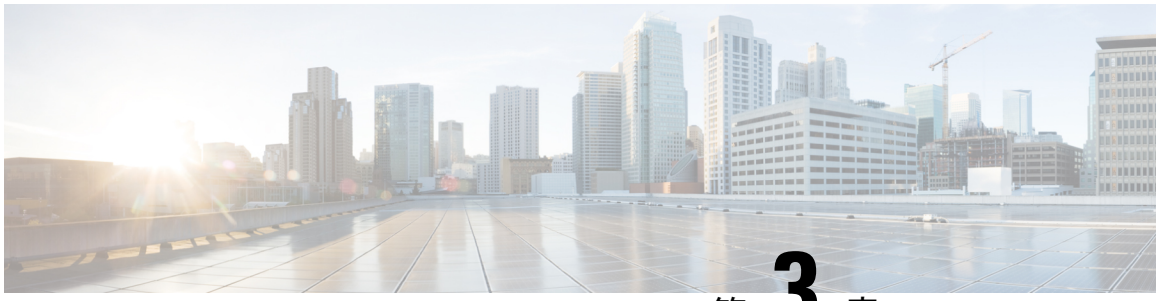
MIB	MIB のリンク
—	Cisco IOS XR ソフトウェアを使用して MIB の場所を特定してダウンロードするには、次の URL にある Cisco MIB Locator を使用して、[Cisco Access Products] メニューからプラットフォームを選択します。 https://mibs.cloudapps.cisco.com/ITDIT/MIBS/servlet/index

RFC

RFC	タイトル
RFC 1700	『Assigned Numbers』
RFC 1997	『BGP Communities Attribute』
RFC 2385	『Protection of BGP Sessions via the TCP MD5 Signature Option』
RFC 2439	『BGP Route Flap Damping』
RFC 2545	『Use of BGP-4 Multiprotocol Extensions for IPv6 Inter-Domain Routing』
RFC 2796	『BGP Route Reflection - An Alternative to Full Mesh IBGP』
RFC 2858	『Multiprotocol Extensions for BGP-4』
RFC 2918	『Route Refresh Capability for BGP-4』
RFC 3065	『Autonomous System Confederations for BGP』
RFC 3392	『Capabilities Advertisement with BGP-4』
RFC 4271	『A Border Gateway Protocol 4 (BGP-4)』
RFC 4364	『BGP/MPLS IP Virtual Private Networks (VPNs)』
RFC 4724	『*Graceful Restart Mechanism for BGP*』

シスコのテクニカル サポート

説明	リンク
シスコのテクニカル サポート Web サイトでは、製品、テクノロジー、ソリューション、技術的なヒント、およびツールへのリンクなどの、数千ページに及ぶ技術情報が検索可能です。Cisco.com に登録済みのユーザは、このページから詳細情報にアクセスできます。	http://www.cisco.com/techsupport



第 3 章

BGP フロースペックの実装

フロースペックは、任意のアプリケーションで使用できるボーダー ゲートウェイ プロトコルのネットワーク層到達可能性情報（BGP NLRI）として BGP を介してフロー仕様ルールを配布する手順を指定し、フロー仕様ルールを符号化するための手順を定義します。また、（分散型）サービス妨害攻撃を軽減するためのパケットフィルタリングを目的とするアプリケーションも定義します。

（注） このモジュールに示されている BGP フロースペックの詳細と BGP フロースペックのコマンドに関する詳細な説明については、*Routing Command Reference for Cisco ASR 9000 Series Routers* の「*BGP Flowspec Commands*」の章を参照してください。

BGP フロースペックの実装の機能履歴

リリース 5.2.0	この機能が導入されました。
リリース 5.3.2	BGP フロースペックでの NLRI ポリシーのサポート

• BGP フロー仕様 （223 ページ）

BGP フロー仕様

BGP フロー仕様（フロースペック）機能を使用すると、多数の BGP ピアルータ間でフィルタリングおよびポリシング機能を迅速に展開および伝達して、ネットワーク上で分散型サービス妨害（DDoS）攻撃の影響を軽減できます。

RTBH（リモートでトリガーされたブラックホール）などの DDoS 攻撃の軽減に対する従来の方法では、特別なコミュニティを使用して攻撃を受けている Web サイトのアドレスのアドバタイズに BGP ルートが挿入されます。境界ルータ上のこの特別なコミュニティは、ネクストホップを破棄またはヌルにするための特殊なネクストホップに設定します。これにより、疑わ

しい送信元からネットワークへのトラフィックが防止されます。これによって優れた保護がもたらされますが、サーバは完全に到達不能になります。

一方、BGP フロースペックでは、よりきめ細やかなアプローチを可能にし、送信元、宛先、L4 パラメータ、および長さ、フラグメントなどのパケットの詳細と特定のフローを照合するための手順を効果的に構築できるようにします。フロースペックでは、境界ルータで次のいずれかのアクションを動的にインストールできます。

- トラフィックをドロップする
- 分析のために別の VRF にトラフィックを挿入する、または
- トラフィックを許可する一方で、定義された特定のレートでポリシングする

そのため、ルートポリシー言語でドロップするために境界ルータをネクストホップに関連付ける必要がある特別なコミュニティを使用してルートを送信する代わりに、BGP フロースペックは特定のフロー形式を境界ルータに送信し、クラスマップとポリシーマップを使用してある種の ACL を作成するように指示して、アドバタイズするルールを実装します。これを実現するため、BGP フロースペックは BGP プロトコルに新しい NLRI（ネットワーク層到達可能性情報）を追加します。BGP フロースペックの実装に関する情報 （226 ページ）に、フロー仕様、サポートされている一致基準、およびトラフィックのフィルタリングアクションの詳細を示します。

フロースペックは、RPL を使用したフロースペック NLRI の送信元と宛先に基づいてフィルタリングでき、また、ネイバーの接続点に適用できます。

制限事項

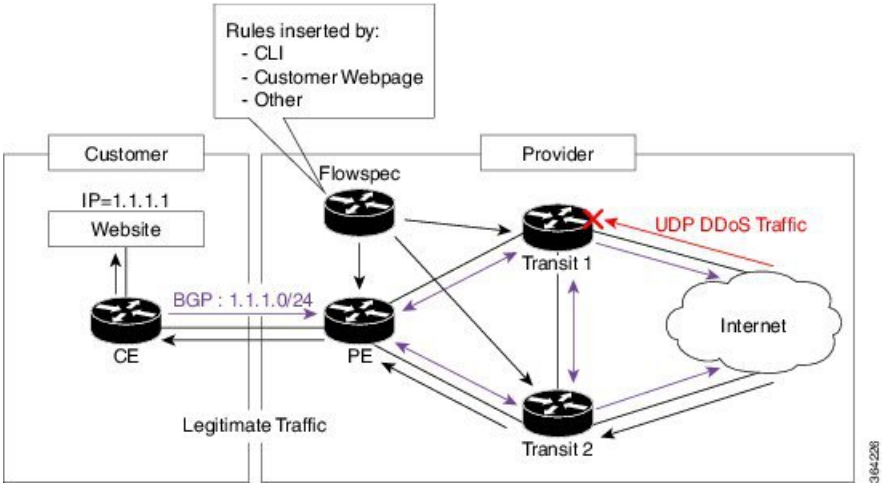
BGP フロースペックには、次の制限が適用されます。

- フロースペックは、次の Cisco ASR 9000 第 1 世代イーサネットラインカードではサポートされていません。
  - A9K-40G（40 ポート 10/100/1000）
  - A9K-4T（4 ポート 10GE）
  - A9K-2T20G（コンボカード）
  - A9K-8T/4
  - A9K-8T
  - A9K-16T/8（16 ポート 10GE）
- フロースペックは、サブスクライバとサテライトインターフェイスではサポートされていません。
- フロースペックルールでは、最大 5 つの複数値の範囲を指定できます。
- フロースペックルールでは、アドレスファミリを混在させることはできません。

- 複数の一致シナリオでは、最初に一致したフロースペックルールのみが適用されます。
-
- QoS は BGP フロースペックよりも優先されます。
- BGP フロースペックは、マルチキャストまたは MPLS トラフィックをサポートしていません。

BGP フロースペックの概念アーキテクチャ

次の図では、フロースペックルータ（コントローラ）はフロー（一致基準とアクション）を使用してプロバイダーエッジ上に設定されています。フロースペックルータは、これらのフローを他のエッジルータと AS（つまり、Transit 1、Transit 2、および PE）にアドバタイズします。これらの中継ルータは、フローをハードウェアにインストールします。フローがハードウェアにインストールされると、中継ルータはルックアップを実行して、着信トラフィックが定義されたフローと一致するかどうかを確認し、適切なアクションを実行することができます。このシナリオでのアクションは、ネットワーク自体のエッジにある DDoS トラフィックを「ドロップ」し、カスタマーエッジにクリーンで正当なトラフィックのみを配信することです。



この項では、フロースペックの機能に関する CLI 設定の例を示します。最初に、フロースペックルータで、着信トラフィックに対して実行する一致アクションの基準を定義します。これには、設定の PBR の部分が含まれています。**service-policy type** で、実際の PBR ポリシーを定義し、フロースペックに追加する必要がある一致とアクション基準の組み合わせを含んでいます。この例では、ポリシーはアドレスファミリ IPv4 の下に追加されるため、IPv4 フロースペックルールとして伝達されます。

```
Flowspec router CLI example:

class-map type traffic match-all cm1
     match source-address ipv4 100.0.0.0/24

policy-map type pbr pm1
    class type traffic cm1
      drop
```

```
flowspec
    address-family ipv4
       service-policy type pbr pm0

Transient router CLI:

flowspec
    address-family ipv4
       service-policy type pbr pm1
```

フロースペックの設定に使用する手順の詳細とコマンドについては、ePBR を使用した BGP フロースペックの設定 （235 ページ）を参照してください。

BGP フロースペックの実装に関する情報

BGP フロースペックを実装するには、次の概念を理解する必要があります。

フロー仕様

フロー仕様は、IP トラフィックに適用可能な複数の一致基準から成る n タプルです。特定の IP パケットは、指定されたすべての基準に一致する場合に、定義されたフローと一致していると見なされます。特定のフローは、特定のアプリケーションに応じて、属性のセットに関連付けることができます。このような属性には、到達可能性情報（つまり、NEXT_HOP）が含まれている場合と含まれていない場合があります。

どのフロー固有ルートも、実質的にはルールであり、一致部分（NLRI フィールドで符号化）とアクション部分（BGP 拡張コミュニティとして符号化）から構成されています。BGP フロースペックルールは、一致パラメータとアクションパラメータを表す同等の C3PL ポリシーに内部的に変換されます。一致とアクションのサポートは、基盤となるプラットフォーム ハードウェア機能によって異なります。サポートされている一致基準とアクション （226 ページ）およびトラフィック フィルタリング アクション （231 ページ）で、サポートされている一致（タプル定義）パラメータとアクションパラメータについて説明します。

サポートされている一致基準とアクション

フロー仕様 NLRI タイプには、宛先プレフィックス、送信元プレフィックス、プロトコル、ポートなどの複数のコンポーネントが含まれている場合があります。この NLRI は、BGP によって不透明ビット文字列プレフィックスとして処理されます。各ビット文字列は、一連の属性を関連付けることができるデータベースエントリのキーを識別します。この NLRI 情報は、MP_REACH_NLRI 属性と MP_UNREACH_NLRI 属性を使用してエンコードされます。対応するアプリケーションがネクストホップ情報を必要としない場合は常に、MP_REACH_NLRI 属性で 0 オクテット長のネクストホップとしてエンコードされ、受信時に無視されます。MP_REACH_NLRI と MP_UNREACH_NLRI の NLRI フィールドは、その後に可変長 NLRI 値が続く 1 オクテットまたは 2 オクテットの NLRI 長フィールドとしてエンコードされます。NLRI の長さはオクテットで表されます。

フロー仕様 NLRI タイプはオプションの複数のサブコンポーネントで構成されます。特定のパケットがフロースペックと一致すると見なされるのは、そのスペック内に存在するすべてのコ

ンポーネントの共通点（AND）に合致する場合です。定義できるサポート対象コンポーネントのタイプまたはタプルを次に示します。

タプル定義の可能性

BGP フロースペック NLRI タイプ	QoS 一致フィールド	説明とシンタックスの構築	値の入力方法
タイプ 1	IPv4 または IPv6 の宛先アドレス	照合する宛先プレフィックスを定義します。プレフィックスは、BGP UPDATE メッセージで、プレフィックス情報を格納するための十分なオクテットが続く長さでエンコードされます。 エンコーディング：<type (1 octet), prefix length (1 octet), prefix> **構文**： **match destination-address** {**ipv4** | **ipv6**} *address/mask length*	プレフィックス長
タイプ 2	IPv4 または IPv6 の送信元アドレス	照合する送信元プレフィックスを定義します。 エンコーディング：<type (1 octet), prefix-length (1 octet), prefix> **構文**： **match source-address** {**ipv4** | **ipv6**} *address/mask length*	プレフィックス長
タイプ 3	IPv4 最終ネクストヘッダーまたは IPv6 プロトコル	IP パケットの IP プロトコル値バイトとの照合に使用する {operator, value} ペアのセットが含まれています。 エンコーディング：<type (1 octet), [op, value]+> **構文**： タイプ 3： **match protocol** {*protocol-value* | *min-value* **-** *max-value*}	複数値の範囲

タイプ 4	IPv4 または IPv6 の送信元ポートまたは宛先ポート	送信元または宛先の TCP/UDP ポートと照合する {operation, value} ペアのリストを定義します。値は、1 バイトまたは 2 バイトの数量としてエンコードされます。パケットの IP プロトコル フィールドに TCP または UDP 以外の値がある場合、パケットがフラグメント化されていて最初のフラグメントではない場合、またはシステムがトランスポートヘッダーを見つけることができない場合は、ポート、送信元ポート、および宛先ポートの各コンポーネントは、FALSE と評価されます。 1 つの一致文字列でサポートされるポート番号は最大 5 つです。 エンコーディング：<type (1 octet), [op, value]+> **構文**： **match source-port** {*source-port-value* | *min-value* *-max-value*} **match destination-port** {*destination-port-value* | *min-value* *-max-value*}	複数値の範囲
タイプ 5	IPv4 または IPv6 の宛先ポート	TCP パケットまたは UDP パケットの宛先ポートの照合に使用する {operation, value} ペアのリストを定義します。値は、1 バイトまたは 2 バイトの数量としてエンコードされます。 エンコーディング：<type (1 octet), [op, value]+> **構文**： **match destination-port** {*destination-port-value* | [*min-value* *-max-value*]}	複数値の範囲

タイプ 6	IPv4 または IPv6 の送信元ポート	TCP パケットまたは UDP パケットの送信元ポートの照合に使用する {operation, value} ペアのリストを定義します。値は、1 バイトまたは 2 バイトの数量としてエンコードされます。 1 つの一致文字列でサポートされるポート番号は最大 5 つです。 エンコーディング：<type (1 octet), [op, value]+> **構文**： **match source-port** {*source-port-value* | [*min-value* **-** *max-value*]}	複数値の範囲
タイプ 7	IPv4 または IPv6 の ICMP タイプ	ICMP パケットのタイプフィールドの照合に使用する {operation, value} ペアのリストを定義します。値は、1 バイトを使用してエンコードされます。ICMP タイプとコード指定子は、プロトコル値が ICMP ではない場合は常に FALSE と評価されます。 エンコーディング：<type (1 octet), [op, value]+> **構文**： **match** {**ipv4** | **ipv6**} **icmp-type** {*value* | *min-value* *-max-value*}	単一の値 （注） 複数の値の範囲はサポートされていません。
タイプ 8	IPv4 または IPv6 の ICMP コード	ICMP パケットのコードフィールドの照合に使用する {operation, value} ペアのリストを定義します。値は、1 バイトを使用してエンコードされます。 エンコーディング：<type (1 octet), [op, value]+> **構文**： **match** {**ipv4** | **ipv6**} **icmp-code** {*value* | *min-value* *-max-value*}	単一の値 （注） 複数の値の範囲はサポートされていません。

タイプ 9	IPv4 または IPv6 の TCP フラグ（2 バイトに予約ビットを含む） （注） 予約済みビットと NS ビットはサポートされていません	ビットマスク値は、1 バイトまたは 2 バイトのビットマスクとしてエンコードできます。1 バイトが指定されている場合は、TCP ヘッダーのバイト 13 に一致します。これには、4 番目の 32 ビットワードのビット 8 ～ 15 が含まれています。2 バイトエンコーディングが使用されている場合、TCP ヘッダーのバイト 12 と 13 は、「don't care」値を持つデータオフセットフィールドと一致します。ポート指定子と同様に、このコンポーネントは、TCP パケットではないパケットについては FALSE と評価します。このタイプは、ビットマスクオペランド形式を使用します。これは、下位ニブルの数値演算子形式とは異なります。 エンコーディング：<type (1 octet), [op, bitmask]+> **構文**： **match tcp-flag** *value* **bit-mask** *mask_value*	ビット マスク
タイプ 10	IPv4 または IPv6 のパケット長	IP パケットの合計長（レイヤ 2 を除くが、IP ヘッダーを含む）に一致します。値は、1 バイトまたは 2 バイトの数量を使用してエンコードされます。 エンコーディング：<type (1 octet), [op, value]+> **構文**： **matchpacket length** {*packet-length-value* | *min-value* *−max-value*}	複数値の範囲

タイプ 11	IPv4 または IPv6 の DSCP	6 ビット DSCP フィールドの照合に使用する {operation, value} ペアのリストを定義します。値は 1 バイトを使用してエンコードされます。この場合、最上位 2 ビットがゼロで、最下位 6 ビットに DSCP 値が含まれています。 エンコーディング：<type (1 octet), [op, value]+> **構文**： **match dscp** {*dscp-value* | *min-value* **-** *max-value*}	複数値の範囲
タイプ 12	IPv4 または IPv6 のフラグメンテーションビット	クラスマップの一致基準としてフラグメントタイプを識別します。 エンコーディング：<type (1 octet), [op, bitmask]+> **構文**： **match fragment type** [**is-fragment**]	ビット マスク

特定のフロースペックルールでは、制約事項なしに複数のアクションの組み合わせを指定できます。ただし、一致基準とアクション間でのアドレスファミリの混在は許可されていません。たとえば、IPv4 のマッチングを IPv6 のアクションと組み合わせることはできず、その逆も同様です。

（注） リダイレクト IP ネクストホップは、デフォルトの VRF の場合にのみサポートされています。

トラフィック フィルタリング アクション（231 ページ）に、フローに関連付けることができるアクションに関する情報を示します。ePBR を使用した BGP フロースペックの設定（235 ページ）では、必要なタプル定義とアクションシーケンスを使用して BGP フロースペックを設定する手順について説明します。

トラフィック フィルタリング アクション

トラフィック フィルタリング フロー仕様のデフォルトアクションでは、その特定のルールに一致する IP トラフィックを受け入れます。次の拡張コミュニティ値を使用して、特定のアクションを指定できます。

タイプ	Extended Community	PBR アクション	説明

0x8006	traffic-rate 0 traffic-rate <rate>	Drop ポリシング	トラフィックレート拡張コミュニティは、自律システム境界を越えた非過渡的な拡張コミュニティであり、次の拡張コミュニティのエンコーディングを使用します。 最初の 2 つのオクテットで 2 オクテットの ID を伝送します。この ID は 2 バイトの AS 番号から割り当てることができます。4 バイトの AS 番号がローカルに存在する場合は、このような AS 番号の最下位 2 バイトを使用できます。この値は情報にすぎません。残りの 4 オクテットで、IEEE 浮動小数点 [IEEE. 754.1985] 形式のレート情報（バイト/秒単位）を伝送します。トラフィックレートが0の場合は、特定のフローのすべてのトラフィックが破棄されることになります。 **コマンド構文** police rate < > | drop
0x8008	redirect-vrf	VRF のリダイレクト	リダイレクト拡張コミュニティを使用すると、そのインポートポリシー内の指定されたルートターゲットをリストする VRF ルーティングインスタンスにトラフィックをリダイレクトできます。複数のローカルインスタンスがこの基準に一致する場合、それらの間でローカルな選択が行われます（たとえば、ルート識別子値が最も小さいインスタンスを選択できます）。この拡張コミュニティは、ルートターゲット拡張コミュニティ [RFC4360] と同じエンコーディングを使用します。 **ルートターゲットに基づくコマンドシンタックス** redirect {ipv6} extcommunity rt <route_target_string>
0x8009	traffic-marking	DSCP の設定	トラフィックマーキング拡張コミュニティは、通過する IP パケットの DiffServ コードポイント（DSCP ）ビットを対応する値に変更するようにシステムに指示します。この拡張コミュニティは 5 つのゼロバイトのシーケンスとしてエンコードされ、その後に 6 番目のバイトの最下位 6 ビットでエンコードされた DSCP 値が続きます。 **コマンド構文** set dscp <6 bit value> set ipv6 traffic-class <8 bit value>

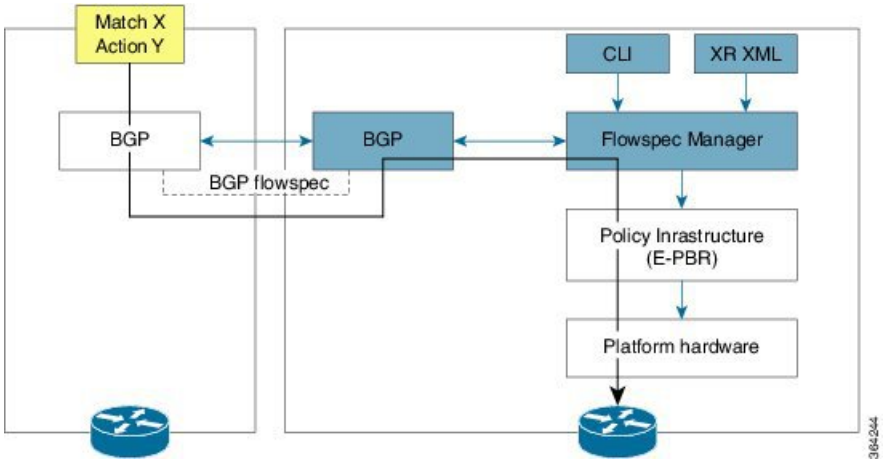
0x0800	IP NH のリダイレクト	リダイレクト IPv4 または IPv6 ネクストホップ	1 つ以上のフロースペック NLRI の到達可能性をアナウンスします。BGP スピーカーが redirect-to-IP 拡張コミュニティを使用して UPDATE メッセージを受信した場合、このパスをベストパスとして持つメッセージ内のすべてのフロースペック NLRI にトラフィック フィルタリング ルールを作成することが予想されます。フィルタエントリは、NLRI フィールドに記述されている IP パケットに一致し、それらをリダイレクトするか、または関連付けられた MP_REACH_NLRI の「Network Address of Next- Hop」フィールドに指定されている IPv4 または IPv6 アドレスにコピーします。 （注） redirect-to-IP 拡張コミュニティは、そのセットに redirect-to-VRF 拡張コミュニティ（タイプ 0x8008）が含まれており、redirect-to-IP 拡張コミュニティを無視する必要がある場合を除き、他のすべてのフロースペック拡張コミュニティのセットで有効です。 **コマンド構文** redirect {ipv6} next-hop <ipv4/v6 address> {ipv4/v6 address}

クラスマップの作成（237 ページ） では、クラスマップの特定の一致基準を設定する方法について説明します。

BGP フロースペッククライアント/サーバ（コントローラ）モデルと ePBR を使用した設定

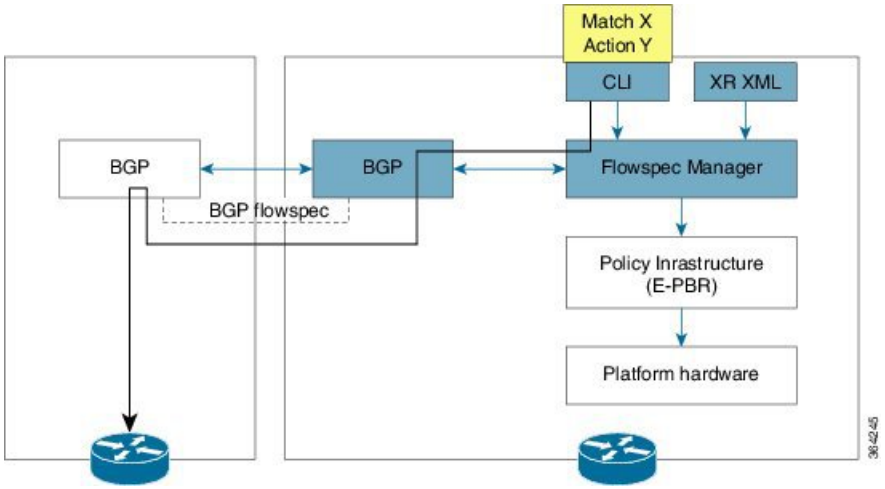
BGP フロースペックモデルは、クライアントとサーバ（コントローラ）で構成されます。コントローラは、フロースペック NRLI エントリの送信または挿入を実行します。クライアント（BGP スピーカーとして機能）はその NRLI を受信し、コントローラからの指示に従って動作するようにハードウェア転送をプログラムします。このモデルの図を下に示します。

BGP フロースペッククライアント



ここでは、左側のコントローラがフロースペック NRLI を挿入し、右側のクライアントが情報を受信し、フロースペックマネージャに送信し、ePBR（拡張ポリシーベースルーティング）インフラストラクチャを設定します。これにより、使用中の基盤プラットフォームからハードウェアがプログラムされます。

BGP フロースペックコントローラ



コントローラは、CLI を使用して NRLI インジェクション用のそのエントリを提供するように設定されます。

BGP フロースペックの設定

- **BGP 側**：アドバタイズメント用に新しいアドレスファミリを有効にする必要があります。この手順は、クライアントとコントローラの両方に適用されます。BGP フロースペックの有効化 （235 ページ）でその手順を説明します。

  クライアント側：フロースペック対応ピアの可用性を除き、固有の設定はありません。

- **コントローラ側**：これにはポリシーマップ定義が含まれており、ePBR 設定への関連付けは、クラス定義とアクションを定義するための ePBR でのそのクラスの使用という 2 つの手順で構成されます。以降のトピックで手順を説明します。

• ポリシー マップの設定 （239 ページ）
• クラスマップの作成 （237 ページ）
• ePBR ポリシーへの BGP フロースペックのリンク （240 ページ）

ePBR を使用した BGP フロースペックの設定

以降の項では、ePBR を使用して BGP フロースペックを設定する手順について説明します。

BGP フロースペック機能を有効にして設定するには、次の手順を実行します。

• BGP フロースペックの有効化 （235 ページ）
• クラスマップの作成 （237 ページ）
• ポリシー マップの設定 （239 ページ）
• ePBR ポリシーへの BGP フロースペックのリンク （240 ページ）

（注） 設定の変更を保存するには、システムでプロンプトが表示されたら、変更を確定する必要があります。

BGP フロースペックの有効化

次の手順を実行して、クライアントとサーバの両方に BGP フロースペックポリシーを伝達するためのアドレスファミリを有効にする必要があります。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **address-family** { **ipv4** | **ipv6** | **vpnv4** | **vpnv6** } **flowspec**
4. **exit**
5. **neighbor** *ip-address*
6. **remote-as** *as-number*
7. **address-family** { **ipv4** | **ipv6** } **flowspec**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 100`	自律システム番号を指定し、BGP コンフィギュレーションモードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。

	コマンドまたはアクション	目的
ステップ 3	**address-family** { **ipv4** | **ipv6** | **vpnv4** | **vpnv6** } **flowspec** 例： `RP/0/RSP0/cpu 0: router(config-bgp)# address-family ipv4 flowspec`	IPv4、IPv6、vpn4 または vpn6 アドレスファミリのいずれかを指定し、アドレス ファミリ コンフィギュレーション サブモードを開始して、フロースペックポリシーのマッピングを行うグローバルアドレスファミリを初期化します。
ステップ 4	**exit** 例： `RP/0/RSP0/cpu 0: router(config-bgp-af)# exit`	ルータを BGP コンフィギュレーション モードに戻します。
ステップ 5	**neighbor** *ip-address* 例： `RP/0/RSP0/cpu 0: router(config-bgp)#neighbor 1.1.1.1`	BGP ルーティングのためにルータをネイバー コンフィギュレーション モードにして、ネイバーの IP アドレスを BGP ピアとして設定します。
ステップ 6	**remote-as** *as-number* 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)#remote-as 100`	ネイバーにリモート自律システム番号を割り当てます。
ステップ 7	**address-family** { **ipv4** | **ipv6** } **flowspec** 例： `RP/0/RSP0/cpu 0: router(config-bgp)# address-family ipv4 flowspec`	アドレスファミリを指定し、アドレス ファミリ コンフィギュレーション サブモードを開始して、フロースペックポリシーのマッピングを行うグローバルアドレスファミリを初期化します。

フロースペックポリシーマッピング用のアドレスファミリの設定：例

```
router bgp 100
 address-family ipv4 flowspec
 ! Initializes the global address family
 address-family ipv6 flowspec
 !
 neighbor 1.1.1.1
  remote-as 100
  address-family ipv4 flowspec
  ! Ties it to a neighbor configuration
  address-family ipv6 flowspec
```

!

クラスマップの作成

ePBR 設定を BGP フロースペックに関連付けるには、クラスを定義し、そのクラスを ePBR に使用してアクションを定義するためのサブステップを実行する必要があります。クラスを定義するには、次の手順を実行します。

手順の概要

1. **configure**
2. **class-map [type traffic] [match-all]** *class-map-name*
3. **match** *match-statement*
4. **end-class-map**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**class-map [type traffic] [match-all]** *class-map-name* 例： `RP/0/RSP0/cpu 0: router(config)# class-map type traffic match all classc1`	名前を指定したクラスとパケットの照合に使用するクラスマップを作成し、クラスマップ コンフィギュレーション モードを開始します。**match-any** を指定した場合、トラフィッククラスで受信したトラフィックがトラフィッククラスの一部と分類されるには、一致基準の 1 つを満たす必要があります。これはデフォルトです。**match-all** を指定した場合は、トラフィックがすべての一致基準を満たす必要があります。
ステップ 3	**match** *match-statement* 例： `RP/0/RSP0/cpu 0: router(config-cmap)# match protocol ipv4 1 60`	指定したステートメントに基づいてクラスマップに対して一致基準を設定します。タプル 1 ～ 13 の一致ステートメントの任意の組み合わせをここで指定できます。次に、タプル定義の可能性を示します。 • タイプ 1：**match destination-address** {**ipv4** | **ipv6**} *address/mask length* • タイプ 2：**match source-address** {**ipv4** | **ipv6**} *address/mask length* • タイプ 3： **match protocol** {*protocol-value* | *min-value* *−max-value*} （注） IPv6 の場合は、最後のネクストヘッダーにマップされます。

	コマンドまたはアクション	目的
		• タイプ 4：1 つは送信元ポート、もう 1 つは宛先ポートを持つ 2 つのクラスマップを作成します。 • **match source-port** {*source-port-value* | *min-value* *-max-value*} （注） 1 つの一致文字列でサポートされるポート番号は最大 5 つです。 • **match destination-port** {*destination-port-value* | *min-value* *-max-value*} （注） これらは、TCP プロトコルと UDP プロトコルにのみ適用されます。 • タイプ 5：**match destination-port** {*destination-port-value* |[*min-value* - *max-value*]} • タイプ 6：**match source-port** {*source-port-value* |[*min-value* - *max-value*]} • タイプ 7： **match** {**ipv4** | **ipv6**}**icmp-code** {*value* | *min-value* *-max-value*} • タイプ 8： **match**{**ipv4** | **ipv6**}**icmp-type** {*value* | *min-value* *-max-value*} • タイプ 9：**match tcp-flag** *value* **bit-mask** *mask_value* • タイプ 10： **matchpacket length** {*packet-length-value* | *min-value* *-max-value*}} • タイプ 11： **match dscp** {*dscp-value* | *min-value* *-max-value*} • タイプ 12： **match fragment-type** {**dont-fragment is-fragment first-fragment last-fragment**} • タイプ 13： **match ipv6 flow-label ipv4 flow-label** {*value* | *min-value* *-max-value*} BGP フロースペック コンフィギュレーションで使用されるさまざまなコマンドの詳細については、*Routing Command Reference for Cisco ASR 9000 Series Routers*ガイドの「*BGP Flowspec Commands*」を参照してください。

	コマンドまたはアクション	目的
ステップ 4	**end-class-map** 例： `RP/0/RSP0/cpu 0: router(config-cmap)# end-class-map`	クラス マップ コンフィギュレーション モードを終了して、ルータをグローバル コンフィギュレーション モードに戻します。

次のタスク

この手順で定義されたるクラスを、ポリシー マップの設定 （239 ページ）の説明に従って、PBR ポリシーに関連付けます。

ポリシー マップの設定

この手順では、ポリシーマップを定義し、これまでにクラスマップの作成 （237 ページ）で設定したトラフィックと関連付けることができます。

手順の概要

1. **configure**
2. **policy-map type pbr** *policy-map*
3. **class** *class-name*
4. **class type traffic** *class-name*
5. アクション
6. **exit**
7. **end-policy-map**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**policy-map type pbr** *policy-map* 例： `RP/0/RSP0/cpu 0: router(config)# policy-map type pbr policyp1`	サービスポリシーを指定するために 1 つ以上のインターフェイスに付加できるポリシーマップを作成または修正し、ポリシーマップ コンフィギュレーション モードを開始します。
ステップ 3	**class** *class-name* 例： `RP/0/RSP0/cpu 0: router(config-pmap)# class class1`	ポリシーを作成または変更するクラスの名前を指定します。

	コマンドまたはアクション	目的
ステップ 4	**class type traffic** *class-name* 例： `RP/0/RSP0/cpu 0: router(config-pmap)# class type traffic classc1`	これまでに設定したトラフィッククラスをポリシーマップに関連付け、制御ポリシーマップ トラフィック クラス コンフィギュレーション モードを開始します。
ステップ 5	アクション 例： `RP/0/RSP0/cpu 0: router(config-pmap-c)# set dscp 5`	要件に従って拡張コミュニティのアクションを定義します。オプションは次のとおりです。 • トラフィックレート： **police rate** *rate* • リダイレクト VRF：**redirect** { **ipv4ipv6** }**extcommunity rt** *route_target_string* • トラフィックマーキング： **set** {**dscp** *rate* | **destination-address** {**ipv4** | **ipv6**} *8-bit value*} • リダイレクト IP NH：**redirect** { **ipv4ipv6** } **nexthop** *ipv4 addressipv6 address* { *ipv4 addressipv6 address*}
ステップ 6	**exit** 例： `RP/0/RSP0/cpu 0: router(config-pmap-c)# exit`	ルータをポリシー マップ コンフィギュレーション モードに戻します。
ステップ 7	**end-policy-map** 例： `RP/0/RSP0/cpu 0: router(config-cmap)# end-policy-map`	ポリシー マップ コンフィギュレーション モードを終了して、ルータをグローバル コンフィギュレーション モードに戻します。

次のタスク

で説明されている手順でVRFとフロースペックのポリシーマッピングを実行し、フロースペックルールを配布します。 ePBR ポリシーへの BGP フロースペックのリンク （240 ページ）

ePBR ポリシーへの BGP フロースペックのリンク

BGP フロースペックの場合、ePBR ポリシーは VRF ごとに適用され、VRF に含まれているすべてのインターフェイスに適用されます。インターフェイス上に ePBR ポリシーがすでに設定されている場合、そのポリシーは BGP フロースペックポリシーによって上書きされません。インターフェイスからポリシーを削除すると、ePBR インフラストラクチャは、VRF レベルでアクティブだった場合は、BGP フロースペックポリシーを自動的に適用します。

（注） 一度に 1 つのインターフェイスでアクティブにできる ePBR ポリシーは 1 つのみです。

手順の概要

1. **configure**
2. **flowspec**
3. **local-install interface-all**
4. **address-family ipv4**
5. **local-install interface-all**
6. **service-policy type pbr** *policy-name*
7. **exit**
8. **address-family ipv6**
9. **local-install interface-all**
10. **service-policy type pbr** *policy-name*
11. **vrf** *vrf-name*
12. **address-family ipv4**
13. **local-install interface-all**
14. **service-policy type pbr** *policy-name*
15. **exit**
16. **address-family ipv6**
17. **local-install interface-all**
18. **service-policy type pbr** *policy-name*
19. **commit**
20. **exit**
21. **show flowspec** { **afi-all** | **client** | **ipv4** | **ipv6** | **summary** | **vrf**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**flowspec** 例： `RP/0/RSP0/cpu 0: router(config)# flowspec`	フロースペック コンフィギュレーション モードを開始します。
ステップ 3	**local-install interface-all** 例： `RP/0/RSP0/cpu 0: router(config-flowspec)# local-install interface-all`	（任意）フロースペックポリシーをすべてのインターフェイスにインストールします。

	コマンドまたはアクション	目的
ステップ 4	**address-family ipv4** 例： `RP/0/RSP0/cpu 0: router(config-flowspec)# address-family ipv4`	IPv4 アドレス ファミリを指定し、アドレス ファミリ コンフィギュレーション サブモードを開始します。
ステップ 5	**local-install interface-all** 例： `RP/0/RSP0/cpu 0: router(config-flowspec-af)# local-install interface-all`	（任意）フロースペックポリシーをサブアドレス ファミリのすべてのインターフェイスにインストールします。
ステップ 6	**service-policy type pbr** *policy-name* 例： `RP/0/RSP0/cpu 0: router(config-flowspec-af)# service-policy type pbr policys1`	インターフェイスのサービスポリシーとして使用されるポリシーマップを、IPv4 インターフェイスに付加します。
ステップ 7	**exit** 例： `RP/0/RSP0/cpu 0: router(config-flowspec-af)# exit`	ルータをフロースペック コンフィギュレーション モードに戻します。
ステップ 8	**address-family ipv6** 例： `RP/0/RSP0/cpu 0: router(config-flowspec)# address-family ipv6`	IPv6 アドレスファミリを指定し、アドレス ファミリ コンフィギュレーション サブモードを開始します。
ステップ 9	**local-install interface-all** 例： `RP/0/RSP0/cpu 0: router(config-flowspec-af)# local-install interface-all`	（任意）フロースペックポリシーをサブアドレス ファミリのすべてのインターフェイスにインストールします。
ステップ 10	**service-policy type pbr** *policy-name* 例： `RP/0/RSP0/cpu 0: router(config-flowspec-af)# service-policy type pbr policys1`	インターフェイスのサービスポリシーとして使用されるポリシーマップを、IPv6 インターフェイスに付加します。

	コマンドまたはアクション	目的
ステップ 11	**vrf** *vrf-name* 例： `RP/0/RSP0/cpu 0: router(config-flowspec)# vrf vrf1`	VRF インスタンスを設定し、VRF フロースペック コンフィギュレーション サブモードを開始します。
ステップ 12	**address-family ipv4** 例： `RP/0/RSP0/cpu 0: router(config-flowspec-vrf)# address-family ipv4`	IPv4 アドレスファミリを指定し、アドレス ファミリ コンフィギュレーション サブモードを開始します。
ステップ 13	**local-install interface-all** 例： `RP/0/RSP0/cpu 0: router(config-flowspec-vrf-af)# local-install interface-all`	（任意）フロースペックポリシーをサブアドレス ファミリのすべてのインターフェイスにインストールします。
ステップ 14	**service-policy type pbr** *policy-name* 例： `RP/0/RSP0/cpu 0: router(config-flowspec-vrf-af)# service-policy type pbr policys1`	インターフェイスのサービスポリシーとして使用されるポリシーマップを、IPv4 インターフェイスに付加します。
ステップ 15	**exit** 例： `RP/0/RSP0/cpu 0: router(config-flowspec-vrf-af)# exit`	ルータを VRF フロースペック コンフィギュレーション サブモードに戻します。
ステップ 16	**address-family ipv6** 例： `RP/0/RSP0/cpu 0: router(config-flowspec-vrf)# address-family ipv6`	IPv6 アドレス ファミリを指定し、アドレス ファミリ コンフィギュレーション サブモードを開始します。
ステップ 17	**local-install interface-all** 例： `RP/0/RSP0/cpu 0: router(config-flowspec-vrf-af)# local-install interface-all`	（任意）フロースペックポリシーをサブアドレス ファミリのすべてのインターフェイスにインストールします。

	コマンドまたはアクション	目的
ステップ 18	**service-policy type pbr** *policy-name* 例： `RP/0/RSP0/cpu 0: router(config-flowspec-vrf-af)# service-policy type pbr policys1`	インターフェイスのサービスポリシーとして使用されるポリシーマップを、IPv6 インターフェイスに付加します。
ステップ 19	**commit**	
ステップ 20	**exit** 例： `RP/0/RSP0/cpu 0: router(config-flowspec-vrf-af)# exit`	ルータをフロースペック コンフィギュレーション モードに戻します。
ステップ 21	**show flowspec** { **afi-all** | **client** | **ipv4** | **ipv6** | **summary** | **vrf** 例： `RP/0/RSP0/cpu 0: routershow flowspec vrf vrf1 ipv4 summary`	（任意）インターフェイスに適用されているフロースペックポリシーを表示します。

BGP フロースペックの確認

次のさまざまな **show** コマンドを使用して、フロースペックの設定を確認します。たとえば、関連付けられた flowspec コマンドと BGP show コマンドを使用して、テーブル内のフロースペックルールの有無、存在するルールの数、定義したフロー仕様に基づいてトラフィックに対して実行されたアクションなどを確認できます。

手順の概要

1. **show processes flowspec_mgr location all**
2. **show flowspec summary**
3. **show flowspec vrf** *vrf_name* \| **all** { **afli-all** \| **ipv4** \| **ipv6** }
4. **show bgp ipv4 flowspec**

手順の詳細

	コマンドまたはアクション	目的

ステップ 1	**show processes flowspec_mgr location all**
	例：
	`# show processes flowspec_mgr location all`
	`node:  node0_3_CPU0`
		フロースペックプロセスがシステムで実行しているかどうかを指定します。フロースペックマネージャは、ハードウェアでのフロースペックルールの作成、配布、およびインストールを実行します。

	コマンドまたはアクション	目的
	Job Id: 10 PID: 43643169 Executable path: /disk0/iosxr-fwding-5.2.CSC33695-015.i/bin/flowspec_mgr Instance #: 1 Version ID: 00.00.0000 Respawn: ON Respawn count: 331 Max. spawns per minute: 12 Last started: Wed Apr 9 10:42:13 2014 Started on config: cfg/gl/flowspec/ Process group: central-services core: MAINMEM startup_path: /pkg/startup/flowspec_mgr.startup Ready: 1.113s Process cpu time: 0.225 user, 0.023 kernel, 0.248 total JID TID CPU Stack pri state TimeInState HR:MM:SS:MSEC NAME 1082 1 0 112K 10 Receive 2:50:23:0508 0:00:00:0241 flowspec_mgr 1082 2 1 112K 10 Sigwaitinfo 2:52:42:0583 0:00:00:0000 flowspec_mgr	
ステップ 2	**show flowspec summary** 例： # show flowspec summary FlowSpec Manager Summary: Tables: 2 Flows: 1 RP/0/3/CPU0:RA01_R4#	ノード全体に存在するフロースペックルールの概要を表示します。この例では、2 つのテーブルが IPv4 と IPv6 が有効になっており、テーブル全体で 1 つのフローが定義されていることを示します。
ステップ 3	**show flowspec vrf** *vrf_name* | **all** { **afli-all** | **ipv4** | **ipv6** } 例： # show flowspec vrf default ipv4 summary Flowspec VRF+AFI table summary: VRF: default AFI: IPv4 Total Flows: 1 Total Service Policies: 1 RP/0/3/CPU0:RA01_R4# --- # show flowspec vrf default ipv6 summary Flowspec VRF+AFI table summary: VRF: default AFI: IPv6 Total Flows: 0 Total Service Policies: 0 RP/0/3/CPU0:RA01_R4# --- # show flowspec vrf all afi-all summary Flowspec VRF+AFI table summary: VRF: default	フロースペックでより詳細な情報を取得するために、特定のアドレスファミリまたは特定の VRF 名に基づいて show コマンドをフィルタリングできます。この例では、「vrf default」は、フロースペックがデフォルトテーブルで定義されていることを示します。「IPv4 summary」には、そのデフォルトテーブルに存在する IPv4 フロースペックルールが表示されます。IPv6 が設定されていないため、値は、ipv6 サマリーの「Table Flows」と「Policies」パラメータに「zero」が表示されます。「VRF all」は、テーブルに設定されているすべての VRF に関する情報を表示し、afli-all はすべてのアドレスファミリ（IPv4 と IPv6）の情報を表示します。 **detail** オプションは、「Matched」フィールド、「Transmitted」フィールド、および「Dropped」フィールドを表示します。これらを使用して、定義したフロースペックルールが動作中かどうかを確認できます。この一致条件を満たすトラフィックがある場合は、何らかのアクションが実行されたかどう

	コマンドまたはアクション	目的
	AFI: IPv4 Total Flows: 1 Total Service Policies: 1 VRF: default AFI: IPv6 Total Flows: 0 Total Service Policies: 0 --- # show flowspec vrf default ipv4 Dest:110.1.1.0/24, Source:10.1.1.0/24,DPort:>=120&<=130, SPort:>=25&<=30,DSCP:=30 detail AFI: IPv4 Flow :Dest:110.1.1.0/24,Source:10.1.1.0/24, DPort:>=120&<=130,SPort:>=25&<=30,DSCP:=30 Actions :Traffic-rate: 0 bps (bgp.1) Statistics (packets/bytes) Matched : 0/0 Transmitted : 0/0 Dropped : 0/0	か（つまり、一致したパケットの数と、それらのパケットが送信されたか、ドロップされたか）を示します。
ステップ 4	**show bgp ipv4 flowspec** 例： # show bgp ipv4 flowspec Dest:110.1.1.0/24,Source:10.1.1.0/24, DPort:>=120&<=130,SPort:>=25&<=30,DSCP:=30/208 BGP routing table entry for Dest:110.1.1.0/24, Source:10.1.1.0/24,Proto:=47,DPort:>=120&<=130,SPort:>=25&<=30,DSCP:=30/208 <snip> Paths: (1 available, best #1) Advertised to update-groups (with more than one peer): 0.3 Path #1: Received by speaker 0 Advertised to update-groups (with more than one peer): 0.3 Local 0.0.0.0 from 0.0.0.0 (3.3.3.3) Origin IGP, localpref 100, valid, redistributed, best, group-best Received Path ID 0, Local Path ID 1, version 42 Extended community: FLOWSPEC Traffic-rate:100,0	コントローラルータ上に設定されているフロースペックルールがBGP側で使用できるかどうかを確認するには、このコマンドを使用します。この例では、「redistributed」は、フロースペックルールが内部では発信されていないが、フロースペックプロセスから BGP に再配布されていることを示しています。設定されている拡張コミュニティ（一致およびアクション基準をピアルータに送信するために使用されるBGP属性）もここに表示されます。この例では、定義されたアクションはトラフィックのレート制限です。

リダイレクトネクストホップの保持

ルート指定の一部としてリダイレクトネクストホップを明示的に設定できます。リダイレクトネクストホップは、関連する拡張コミュニティとともに、BGP フロースペック NLRI で MP_REACH ネクストホップとしてエンコードされます。このような flowspec ルートが受信されると、リダイレクトネクストホップのFIBルックアップによりトラフィックはリダイレクトされ、ネクストホップは、場合により IP または MPLS トンネルを介して解決できます。ネク

ストホップ接続が複数のASに及ぶ場合、eBGP境界でMP_REACHネクストホップを上書きできるため、未変更のノブを使用することでネクストホップを保持できます。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **neighbor** *ip-address*
4. **address-family** { **ipv4** | **ipv6** }
5. **flowspec next-hop unchanged**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 100`	自律システム番号を指定し、BGPコンフィギュレーションモードを開始します。このモードでは、BGPルーティング プロセスを設定できます。
ステップ 3	**neighbor** *ip-address* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 100 neighbor 1.1.1.1`	BGPルーティングのためにルータをネイバー コンフィギュレーション モードにして、ネイバーのIPアドレスをBGPピアとして設定します。
ステップ 4	**address-family** { **ipv4**	**ipv6** } 例： `RP/0/RSP0/cpu 0: router(config-bgp)# router bgp 100 neighbor 1.1.1.1 address-family ipv4`	IPv4またはIPv6のいずれかのアドレスファミリを指定し、アドレス ファミリ コンフィギュレーション サブモードを開始してグローバルアドレスファミリを初期化します。
ステップ 5	**flowspec next-hop unchanged** 例： `RP/0/RSP0/cpu 0: router(config-bgp)# router bgp 100 neighbor 1.1.1.1 address-family ipv4 flowspec next-hop unchanged`	フロースペックのネクストホップを変更せずに保持します。

BGP フロースペックの検証

BGP フロースペック検証は、IPv4またはIPv6のフロースペック SAFI ルートに対してデフォルトで有効になっています。VPNルートは、フロー検証の対象にはなりません。機能を動作させるため、次の条件のいずれかをtrueの状態に保持することを確認するためにフロー仕様NLRIが検証されます。

- フロー仕様の発信元は、フロー仕様に組み込まれている宛先プレフィックスのベストマッチのユニキャストルートの発信元と一致します。
- 異なる隣接 AS から受信したフローの宛先プレフィックスを比較した場合、前の条件で決定されたベストマッチのユニキャストルート以外に特定のユニキャストルートはありません。
- フロー仕様の AS_PATH と AS4_PATH 属性は空です。
- フロー仕様の AS_PATH および AS4_PATH 属性には、AS_SET セグメントと AS_SEQUENCE セグメントは含まれていません。

これらの条件を満たさないパスは、BGP によって適切にマーキングされ、フロースペックマネージャにはインストールされません。さらに、BGP では、EBGP で学習したフロー仕様 NLRI の AS_PATH と AS4_PATH 属性内に追加された最後の AS は、フロー仕様に組み込まれた宛先プレフィックスのベストマッチのユニキャストルートの AS_PATH と AS4_PATH 属性内に追加された最後の AS と一致する必要があります。また、redirect-to-IP 拡張コミュニティが存在する場合、デフォルトでは、BGP は eBGP ピアからフロースペックルートを受信すると、次のチェックを強制的に実行します。

フロースペックルートに IP ネクストホップ X があり、redirect-to-IP 拡張コミュニティが含まれている場合、BGP スピーカは、最も長いプレフィックス一致の AS_PATH または AS4_PATH 属性の最後の AS が eBGP ピアの AS と一致しない場合、redirect-to-ip 拡張コミュニティを破棄します（また、フロースペックルートを使用してそれ以降は伝達しません）。

フロースペックリダイレクトと検証の無効化（249ページ） では、BGP フロースペック検証を無効にする手順について説明します。

BGP フロースペックの無効化

この手順では、インターフェイス上の BGP フロースペックポリシーを無効にします。

手順の概要

1. **configure**
2. **interface** *type interface-path-id*
3. **{ ipv4 | ipv6 } flowspec disable**
4. **commit**

手順の詳細

ステップ 1 **configure**

ステップ 2 **interface** *type interface-path-id*

例：

```
RP/0/RSP0/cpu 0: router(config)# interface GigabitEthernet 0/1/1/1
```

インターフェイスを設定して、インターフェイス コンフィギュレーション モードを開始します。

ステップ 3 **{ ipv4 | ipv6 } flowspec disable**

例：

```
RP/0/RSP0/cpu 0: router(config-if)# ipv4 flowspec disable
```

選択したインターフェイス上のフロースペックポリシーを無効にします。

ステップ 4 **commit**

インターフェイスでのフロースペックの無効化

次に、インターフェイス上の BGP フロースペックを無効にし、別の PBR ポリシーを適用する例を示します。

```
Interface GigabitEthernet 0/0/0/0
 flowspec [ipv4/ipv6] disable
int g0/0/0/1
service policy type pbr test_policy
 !
!
```

フロースペックリダイレクトと検証の無効化

明示的なノブを設定することによって、フロースペック検証を eBGP セッションの全体で無効にすることができます。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **neighbor** *ip-address*
4. **address-family** { **ipv4** | **ipv6** }
5. **flowspec validation** { **disable** | **redirect disable** }

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 100`	自律システム番号を指定し、BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスを設定できます。

	コマンドまたはアクション	目的
ステップ 3	**neighbor** *ip-address* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 100 neighbor 1.1.1.1`	BGP ルーティングのためにルータをネイバー コンフィギュレーション モードにして、ネイバーの IP アドレスを BGP ピアとして設定します。
ステップ 4	**address-family** { **ipv4** | **ipv6** } 例： `RP/0/RSP0/cpu 0: router(config-bgp)# router bgp 100 neighbor 1.1.1.1 address-family ipv4`	IPv4 または IPv6 のいずれかのアドレスファミリを指定し、アドレス ファミリ コンフィギュレーション サブモードを開始してグローバルアドレスファミリを初期化します。
ステップ 5	**flowspec validation** { **disable** | **redirect disable** } 例： `RP/0/RSP0/cpu 0: router(config-bgp)# router bgp 100 neighbor 1.1.1.1 address-family ipv4 flowspec validation disable`	すべての eBGP セッションの全体に対してフロースペック検証を無効にするか、またはリダイレクトネクストホップの検証を無効にするかを選択できます。

BGP フロースペックの設定例

フロースペックルールの設定

フロースペックルールの設定例

この例では、2 つの異なる VRF に対して 2 つのフロースペックルールが作成されています。これは、192/8 から 10.0.1/24 へと宛先ポート（範囲 137、139）または 8080 へのすべてのパケットと対象とし、レート制限を blue vrf で 500 bps とし、デフォルトの VRF でドロップすることを目標としています。これは、gig 0/0/0/0 で有効になっているフロースペックを無効にすることも目標としています。

```
class-map type traffic match-all fs_tuple
 match destination-address ipv4 10.0.1.0/24
 match source-address ipv4 192.0.0.0/8
 match destination-port 137-139 8080
 end-class-map
 !
!
policy-map type pbr fs_table_blue
 class type traffic fs_tuple
  police rate 500 bps
```

```
  !
 !
 class class-default
 !
 end-policy-map
policy-map type pbr fs_table_default
 class type traffic fs_tuple
  drop
  !
 !
 class class-default
 !
 end-policy-map
flowspec
 local-install interface-all
 address-family ipv4
  service-policy type pbr fs_table_default
  !
 !
 vrf blue
  address-family ipv4
   service-policy type pbr fs_table_blue local
   !
  !
 !
!
Interface GigabitEthernet 0/0/0/0
 vrf blue
 ipv4 flowspec disable
```

ドロップパケット長

次に、ドロップパケット長アクションの設定例を示します。

```
class-map type traffic match-all match-pkt-len
 match packet length 100-150
 end-class-map
!
policy-map type pbr test2
 class type traffic match-pkt-len
  drop
 !
 class type traffic class-default
 !
 end-policy-map
!
```

特定のクラスに属するパケットを破棄するようにトラフィッククラスを設定するには、ポリシー マップ コンフィギュレーション モードで drop コマンドを使用します。この例では、100 ～ 150 の複数範囲のパケット長値が定義されています。着信トラフィックのパケット長がこの条件に一致した場合、このパケットを「ドロップ」するようにアクションが定義されています。

リダイレクトトラフィックとレート制限：例

```
class-map type traffic match-all match-src-ipv6-addr
 match source-address ipv6 3110:1::/48
 end-class-map
!
policy-map type pbr test5
 class type traffic match-src-ipv6-addr
  redirect nexthop 3010:10:11::
  police rate 20 mbps
  !
 !
 class type traffic class-default
 !
 end-policy-map
!
```

この例では、特定の送信元 IP アドレス（3110:1::/48）からのすべてのトラフィックをネクストホップアドレスにリダイレクトするために、フロースペックルールにアクションが定義されています。また、この送信元アドレスで着信するすべてのトラフィックについて、送信元アドレスのレート制限は 20 メガビット/秒になります。

グローバルからの VRF（vrf1）へのトラフィックのリダイレクト

次に、トラフィックをグローバル トラフィック リンクから個々の VRF インターフェイスへリダイレクトするための設定例を示します。

```
class-map type traffic match-all match-src-ipv6-addr
 match source-address ipv6 3110:1::/48
 end-class-map
!
policy-map type pbr test4
 class type traffic match-src-ipv6-addr
  redirect nexthop route-target 100:1
 !
 class type traffic class-default
 !
 end-policy-map
```

DSCP のリマーク

次に、set dscp アクションの設定例を示します。

```
class-map type traffic match-all match-dscp-af11
 match dscp 10
 end-class-map
!
policy-map type pbr test6
 class type traffic match-dscp-af11
  set dscp af23
 !
 class type traffic class-default
 !
 end-policy-map
!
```

この例では、トラフィックマーキング拡張コミュニティ（**match dscp**）によって、dscp 10 から dscp af23 へ通過する IP パケットの DSCP ビットを変更または設定するようにシステムに指示します。

BGP フロースペックの追加資料

以降の項では、BGP フロースペックの実装に関する関連資料について説明します。

関連資料

関連項目	マニュアル タイトル
BGP フロースペックコマンド：コマンドシンタックスの詳細、コマンドモード、コマンド履歴、デフォルト設定、使用上の注意事項、および例	*Routing Command Reference for Cisco ASR 9000 Series Routers*

標準

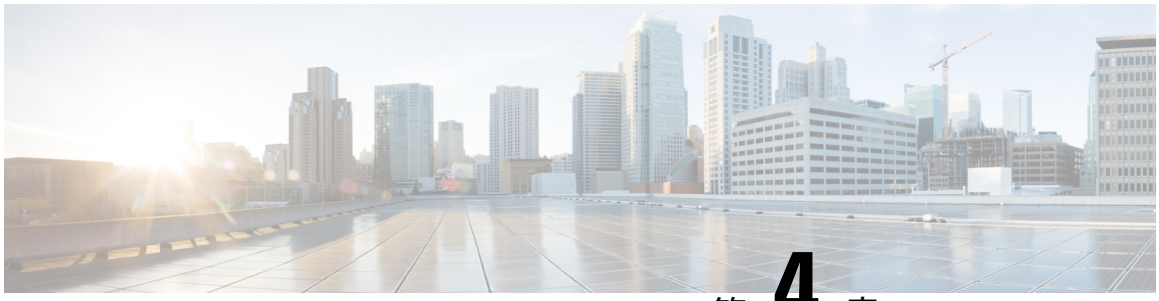
標準	タイトル
draft-ietf-idr-flow-spec-v6-05 draft-ietf-idr-flowspec-redirect-ip-01 draft-simpson-idr-flowspec-redirect-02 draft-ietf-idr-bgp-flowspec-oid-02	『*Dissemination of Flow Specification Rules for IPv6*』、 『BGP Flow-Spec Redirect to IP Action』 『BGP Flow-Spec Extended Community for Traffic Redirect to IP Next Hop』 『Revised Validation Procedure for BGP Flow Specifications』

RFC

RFC	タイトル
RFC 5575	『Dissemination of Flow Specification Rules』

シスコのテクニカル サポート

説明	リンク
シスコのテクニカル サポート Web サイトでは、製品、テクノロジー、ソリューション、技術的なヒント、およびツールへのリンクなどの、数千ページに及ぶ技術情報が検索可能です。Cisco.com に登録済みのユーザは、このページから詳細情報にアクセスできます。	http://www.cisco.com/techsupport



第 4 章

BFD の実装

このモジュールでは、Cisco ASR 9000 シリーズ ルータでの双方向フォワーディング検出（BFD）の設定について説明します。

双方向フォワーディング検出（BFD）では、隣接する転送エンジン間のパスにおける障害を低オーバーヘッド、短期間で検出できます。BFD では、あらゆるメディアおよびあらゆるプロトコル レイヤでの障害検出に単一のメカニズムを使用でき、広範な検出時間とオーバーヘッドに対応できます。障害の迅速な検出が可能なため、リンクやネイバーの障害発生時にもただちに障害に対応することができます。

双方向フォワーディング検出の実装の機能履歴

リリース	変更内容
リリース 3.7.2	BFD が導入されました。

リリース 3.9.0	• BFD での次のアプリケーションのサポートが追加されました。
	  • Hot Standby Router Protocol（HSRP）
	  • Virtual Router Redundancy Protocol（VRRP）
	• BFDセッションのフラッピングとセッションの開始遅延を最小化するために、**dampening** コマンドが追加されました。
	• 送信元 IP アドレスを指定してデフォルトを上書きするために、**echo ipv4 source** コマンドが追加されました。
	• IPv6 UDP チェックサムの計算および BFD インターフェイス コンフィギュレーション モードを有効および無効にするように、**ipv6 checksum** コマンドが追加されました。

リリース 4.0.0	次の BFD 機能のサポートが追加されました。 • OSPFv3 用 BFD • IPv6 用 BFD BFD のサポートは、次の SPA に追加されました。 • 1 ポート OC-192c/STM-64 POS/RPR XFP SPA • 2 ポート OC-48c/STM-16 POS/RPR SPA • 8 ポート OC-12c/STM-4 POS SPA
リリース 4.0.1	次の BFD 機能のサポートが追加されました。 • リンクバンドルのメンバリンク単位の BFD のサポートが追加されました。 • 非バンドルインターフェイスの BFD エコーパケットに対する遅延検出を有効にするために、**echo latency detect** コマンドが追加されました。 • 非バンドルインターフェイスの BFD セッションを開始する前にエコーパスを検証するために、**echo startup validate** コマンドが追加されました。
リリース 4.2.0	次の BFD 機能のサポートが追加されました。 • BFD マルチホップ グローバル TTL チェック。 • BGP の BFD マルチホップ サポート、および • IPv4 トラフィックの BFD マルチホップ サポート。 • マルチホップセッション用のパケットのドロップを開始するための TTL 値を指定するために、**multihop ttl-drop-threshold** コマンドが追加されました。
リリース 4.2.1	ASR9K-SIP-700 ラインカードでの BFD マルチホップ機能のサポートが追加されました。

リリース 4.2.3	論理バンドル上の BFD 機能のサポートが追加されました。
リリース 4.3.0	次の機能のサポートが追加されました。 • GRE を介した BFD • BFD IPv6 マルチホップ • 論理バンドル上の BFD
リリース 4.3.1	次の機能のサポートが追加されました。 • MPLS トラフィック エンジニアリング LSP を介した BFD • 疑似回線ヘッドエンドを介した BFD • サテライトインターフェイスを介した BFD
リリース 5.2.4	バンドル CISCO/IETF モードを介した BFD のバンドル単位のサポートが追加されました。

BFD の実装の前提条件

適切なタスク ID を含むタスク グループに関連付けられているユーザ グループに属している必要があります。このコマンド リファレンスには、各コマンドに必要なタスク ID が含まれます。ユーザ グループの割り当てが原因でコマンドを使用できないと考えられる場合、AAA 管理者に連絡してください。

次に、BFD を実装するための前提条件を示します。

- マルチプロトコル ラベル スイッチング（MPLS）で BFD を有効にする場合は、MPLS パッケージを含んだインストール済みの複合 PIE ファイル、または複合パッケージ イメージが必要です。ボーダー ゲートウェイ プロトコル（BGP）、Intermediate System-to-Intermediate System（IS-IS）、スタティック、Open Shortest Path First（OSPF）の場合は、インストール済みの Cisco IOS XR IP Unicast Routing Core Bundle イメージが必要です。
- IS-IS または OSPF を使用している場合、ルータで内部ゲートウェイ プロトコル（IGP）がアクティブになっていること。

- Cisco ASR 9000 シリーズ ルータでは、BFD をサポートしている各ラインカードが次のタスクを実行できる必要があります。
  - エコー パケットを 50ms * 3 ごとに送信（通常の状態の最小値として）。
  - 制御パケットを 150ms * 3 ごとに送信（負荷のある状態の最小値として）。
  - 最大 9600 pps のユーザ データグラム プロトコル（UDP）を送受信。これにより、15 ms エコー間隔で 144 セッション（または 150 ms エコー間隔で 1440 セッション）を維持します。
- ネイバーの BFD を有効にするには、その隣接ルータが BFD をサポートしている必要があります。
- リリース 3.9.0 よりも前の Cisco IOS XR リリースでは、BFD セッションを設定する前に、グローバル コンフィギュレーション モードで **router-id** コマンドを使用して、ローカル ルータ ID を設定することを推奨していました。ローカル ルータ ID を設定しなかった場合、デフォルトで BFD エコー モードでの IP パケットの送信元アドレスが、出力インターフェイスの IP アドレスとなります。Cisco IOS XR Release 3.9.0 以降では、**echo ipv4 source** コマンドを使用して、送信元アドレスとして使用する IP アドレスを指定できます。
- バンドル メンバ リンクでの BFD をサポートするには、次の要件が満たされていることを確認してください。
  - バンドルの両端にあるルータが、間にレイヤ 2 スイッチを使用せずにバックツーバックで接続されている。
  - BFD セッションを開始する場合、次のいずれかの設定または状態がバンドル メンバに存在する。

    リンク集約制御プロトコル（LACP）分散状態に到達している、または

    EtherChannel または POS チャネルが設定されている、または

    ホット スタンバイおよび LACP 収集状態に到達している。

BFD の実装の制約事項

BFD には、次の制約事項が適用されます。

- Cisco IOS XR ソフトウェアではデマンド モードはサポートされません。
- 次の機能では、BFD エコー モードはサポートされません。
  - バンドル VLAN での IPv4 用 BFD
  - IPv6 用 BFD（グローバルおよびリンクローカル アドレッシング）
  - uRPF での BFD（IPv4 または IPv6）

- BFD バンドル インターフェイスに複数のラックにまたがるメンバ リンクがある場合の、ラック リロードおよび活性挿抜（OIR）
- マルチホップパスの BFD

- IPv6 用 BFD には、次の制約事項があります。
- - IPv6 用 BFD は、バンドル VLAN インターフェイスではサポートされません。
  - IPv6 用 BFD スタティック ルート、OSPFv3、および BGP は、クライアントでサポートされます。
  - ネクストホップとしてリンクローカル アドレスを持つ IPv6 用 BFD スタティック ルートはサポートされません。

- バンドル メンバ リンクでの BFD の場合は、IPv4 アドレッシング タイプについてのみ、バンドル メンバ リンクごとに 1 つの BFD セッションだけが作成、モニタ、および保持されます。バンドルの IPv6 および VLAN リンクに関する制約事項は、次のとおりです。
  - IPv6 の状態はバンドル メンバで明示的にモニタされず、そのメンバ インターフェイスの IPv4 BFD セッションの状態を継承します。
  - バンドル メンバの VLAN サブインターフェイスはまた、そのメンバ インターフェイスの IPv4 BFD セッションから BFD 状態を継承します。VLAN サブインターフェイスはバンドル メンバで明示的にモニタされません。

- エコー遅延検出およびエコー検証は、バンドル インターフェイスではサポートされません。
- BFD マルチホップはデフォルト以外の任意の VRF で実行できますが、選択的 VRF ダウンロードが無効になっている必要があります。選択的 VRF ダウンロードのための設定およびコマンドの詳細については、 *Routing Configuration Guide for Cisco ASR 9000 Series Routers* および *Routing Command Reference for Cisco ASR 9000 Series Routers*を参照してください。
- GRE 上の BFD 機能は、Cisco ASR 9000 シリーズ SPA インターフェイス プロセッサ 700 ではサポートされません。
- BFD IPv6 マルチホップ機能は、Cisco ASR 9000 シリーズ SPA インターフェイス プロセッサ 700 ではサポートされません。
- 論理バンドル上の BFD 機能は、Cisco ASR 9000 シリーズ SPA インターフェイス プロセッサ 700 ではサポートされません。
- イーサネットラインカードでは、BFD MH と BLB のみがサポートされています。このラインカードでは、BFDoTE、BFDoIRB、BFDoGRE などの BFD マルチパスセッションはサポートされていません。
- サテライトセッションを介した BFD は、ASR 9000 イーサネットラインカードではサポートされていません。また、Cisco ASR 9000 シリーズ SPA インターフェイスプロセッサ 700 でもサポートされていません。

- バンドルインターフェイスに明示的なバンドルハッシュが設定されている場合、バンドルマネージャは送信元または宛先の IP アドレスに基づいてハッシュを実行します。これにより、すべてのエコーパケットがメンバリンクのいずれかでのみ送信され、他のリンクはフラッピングを開始します。

  BFD エコーには送信元ポートに基づくハッシュが必要なため、IP ベースのハッシュではメンバーリンク間でエコーパケットが配信されません。

  どちらも相互運用できないため、設定されたバンドルの IP ベースのハッシュを回避するか、またはエコーモードを無効にします。

  IP ベースのハッシュを削除するには、次の手順を実行します。

```
RP/0/RSP0/CPU0:router(config)# interface bundle-ether 1
RP/0/RSP0/CPU0:router(config)# no bundle load-balancing hash dst-ip

/* or */

RP/0/RSP0/CPU0:router(config)# no bundle load-balancing hash src-ip
```

  設定されたバンドルのエコーを無効にするには、次の手順を実行します。**echo disable** コマンドは、グローバルモードまたはインターフェイス コンフィギュレーション モードのいずれかで実行されます。

```
 RP/0/RSP0/CPU0:router(config)# bfd
RP/0/RSP0/CPU0:router(config)# interface bundle-ether 1
RP/0/RSP0/CPU0:router(config-if)# echo disable

*/ or */

RP/0/RSP0/CPU0:router(config)# bfd  echo disable
```

BFD に関する情報

Cisco IOS XR ソフトウェアと Cisco IOS ソフトウェアでの BFD の違い

すでに Cisco IOS ソフトウェアでの BFD の設定に精通している場合は、Cisco IOS XR ソフトウェア実装での BFD の設定に関する次の違いについて必ず考慮します。

- Cisco IOS XR ソフトウェアでは、BFD は OSPF や BGP インスタンスなどのダイナミック ルーティング プロトコルの下で設定されたアプリケーションです。これは、BFD がインターフェイスでのみ設定される Cisco IOS ソフトウェアでの BFD には当てはまりません。
- Cisco IOS XR ソフトウェアでは、BFD ネイバーはルーティングを介して確立されます。Cisco IOS **bfd neighbor** インターフェイス コンフィギュレーション コマンドは、Cisco IOS XR ソフトウェアではサポートされていません。
- ダイナミック ルーティング プロトコルを使用して BFD ネイバーを確立する代わりに、そのパスを定義するためのスタティック ルーティングの方法を使用して、Cisco IOS XR ソフトウェアで BFD 応答のための特定の BFD ピアまたはネイバーを確立することができま

す。実際には、Cisco IOS XR ソフトウェアでダイナミック ルーティング プロトコルの下で BFD を設定しない場合は、BFD のスタティック ルートを設定する必要があります。

- Cisco IOS ソフトウェアで BFD を実行しているルータは、**bfd neighbor** コマンドを使用してピアとして Cisco IOS XR ソフトウェアで BFD を実行するルータを指定できます。Cisco IOS XR ルータは、ピア関係を確立するために Cisco IOS ルータに戻るダイナミックルーティングまたは静的ルートを使用する必要があります。Cisco IOS および Cisco IOS XR ソフトウェアを実行しているルータの BFD ピア：例を参照してください。

nV エッジシステムでの BFD マルチパスセッションのサポート

nV エッジシステムでは、次の BFD マルチパスセッションがサポートされています。

- GRE を介した BFD
- 論理バンドル上の BFD
- IRB を介した BFD
- BFD マルチホップ（5.2.2 以降でのみサポート）
- MPLS TE を介した BFD
- サテライト経由の BFD

BFD の動作モード

Cisco IOS XR ソフトウェアは、エコー パケットを使用するか否かにかかわらず、非同期動作モードだけをサポートします。エコーなしの非同期モードでは、ローカルおよびリモート システム上のパケット スイッチング パスのさまざまな部分が関与します。ただし、エコーありの非同期モードは通常、若干広いテスト カバレッジを提供すると認識されています。これは、エコー パケットがリモート システムの通常のトラフィックと同じパケット スイッチング パスを通過する自分宛てのパケットであるためです。

BFD エコーモードは、次のインターフェイスではデフォルトで有効になっています。

- BFD バンドル インターフェイスのメンバ リンク上の IPv4 の場合。
- 最小間隔が 2 秒未満である他の物理インターフェイス上の IPv4 の場合。

BFDがエコーパケットなしで非同期で実行されている場合（図35）は、次のようになります。

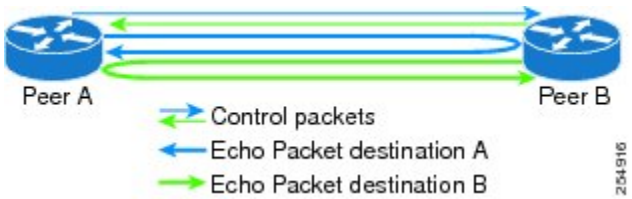
- 各システムが相互に定期的に BFD 制御パケットを送信します。BFD ルータの「ピア A」によって BFD ルータの「ピア B」に送信されたパケットは、ピア A からの送信元アドレスおよびピア B の宛先アドレスを保持します。
- 制御パケット ストリームは互いに独立していて、要求/応答モデルで動作しません。
- 連続する多数のパケットが別のシステムによって受信されない場合、セッションがダウンしたと宣言されます。

図 *12* : エコー パケットなしの *BFD* 非同期モード

BFDがエコーパケットありで非同期で実行されている場合（図36）は、次のようになります。

- BFD エコー パケットは、BFD ピアのみの転送パス経由でループ バックされ、どのプロトコル スタックでも処理されません。そのため、BFD ルータの「ピア A」によって送信されたパケットは、ピア A の送信元アドレスと宛先アドレスの両方を使用して送信できます。
- BFD 制御パケットに加えて、BFD エコー パケットが送信されます。

図 *13* : エコー パケットありの *BFD* 非同期モード



非同期モードでの制御およびエコーパケットの間隔の詳細については、BFDパケット間隔と障害検出を参照してください。

BFD パケット情報

BFD の送信元および宛先ポート

BFDペイロード制御パケットは、宛先ポート3784および送信元ポート49152を使用して、UDPパケットにカプセル化されます。イーサネットのような共有型メディアでも、BFD制御パケットは常にユニキャストパケットとしてBFD ピアに送信されます。

エコー パケットも、宛先ポート 3785 および送信元ポート 3785 を使用して、UDP パケットにカプセル化されます。

バンドルメンバを介したBFD機能は、送信ごとにエコーパケットのUDP送信元ポートの各バイトを増分します。UDP 送信元ポートの範囲は 0xC0C0 から 0xFFFF です。次に例を示します。

1 番目のエコー パケット：0xC0C0

2 番目のエコー パケット：0xC1C1

3 番目のエコー パケット：0xC2C2

UDP 送信元ポートは、連続したエコー パケットが逸脱しているバンドル メンバにハッシュされるように増分されます。

BFD パケット間隔と障害検出

BFD は、設定可能な間隔と係数を使用して、非同期モードで制御およびエコー パケットが送信される期間と、それらに対応する障害検出を指定します。

物理インターフェイスで実行されている BFD セッション、およびバンドル メンバ リンクの BFD セッションに対するこれらの間隔と障害検出時間の実装方法には違いがあります。

物理インターフェイスでの BFD パケット間隔

BFD が物理インターフェイス経由で実行されている場合、エコー モードは、設定された間隔が 2 秒未満である場合にのみ使用されます。

エコーモードが有効になっているときに物理インターフェイス経由で実行されている BFD セッションは、2 秒ごとの遅いレートで BFD 制御パケットを送信します。BFD エコー パケットはすでに高速で送信されており、エコーパケットがエコー障害検出時間内に受信されないときはリンク障害が検出されるため、高速で制御パケット障害検出を複製する必要はありません。

バンドル メンバ リンクの BFD パケット間隔

各バンドル メンバ インターフェイスでは、エコー モードが実行されている場合でも、BFD 非同期モード制御パケットはユーザ設定可能な間隔および係数値で動作します。

ただし、バンドル メンバ インターフェイスでエコーモードが有効になっているとき、BFD 非同期モードは高速レートで実行し続ける必要があります。これは、BFD エコーモードを有効にする要件の 1 つとして、バンドル メンバ インターフェイスが BFD 非同期モードで使用できることがあるためです。

バンドル メンバ リンクの BFD の最大エコー パケット間隔は、30 秒または非同期制御パケット障害検出時間のいずれかの最小値です。

エコーモードが無効になっている場合、セッションが設定されたレートで BFD 制御パケットを交換する物理インターフェイスでの BFD と動作は同じになります。

非同期モードでの制御パケット障害検出

エコーなしの非同期モードでの制御パケット障害検出は、最小間隔（非バンドルインターフェイスでは bfd minimum-interval、バンドル インターフェイスでは bfd address-family ipv4 minimum-interval）および係数（非バンドル インターフェイスでは bfd multiplier、バンドル インターフェイスでは bfd address-family ipv4 multiplier）コマンドの値を使用して実行されます。

制御パケット障害検出の場合は、ローカルの係数値がネイバーに送信されます。障害検出タイマーは、（*I* x *M*）に基づいて開始されます。ここで、*I* はネゴシエートされた間隔で、*M* はリモートエンドによって提供された係数です。

有効な制御パケットがネイバーから受信されるたびに、障害検出タイマーはリセットされます。有効な制御パケットが期間（*I* x *M*）内にネイバーから受信されない場合は、障害検出タイマーがトリガーされ、ネイバーがダウンしたと宣言されます。

非同期モードでのエコー パケット障害検出

標準のエコー障害検出方式は、非バンドルインターフェイスでは**bfd multiplier** コマンドの値、バンドルインターフェイスでは **bfd address-family ipv4 multiplier** コマンドの値に基づいたカウンタを通して実行されます。

このカウンタは、エコーパケットがエコーパケットストリームに送信された順序にかかわらず、システムがエコーパケットを送信するたびに増分され、何らかのエコーパケットが受信されるたびにゼロにリセットされます。

つまり、理想的な条件下では、BFD は一般に、バンドルインターフェイスでは期間（*I* x *M*）または（*I* x *M* x *M*）を超えるエコー障害を検出します。ここで、各値は次のとおりです。

- *I*：最小間隔の値（非バンドルインターフェイスでは bfd minimum-interval、バンドルインターフェイスでは **bfd address-family ipv4 minimum-interval**）。
- *M*：乗数（非バンドルインターフェイスでは**bfd multiplier**、バンドルインターフェイスでは **bfd address-family ipv4 multiplier**）コマンドの値。

そのため、システムがエコーパケットをまったく受信せずに係数カウントを超えて追加のエコーパケットを1つ送信した場合は、エコー障害が検出され、ネイバーがダウンしたと宣言されます（例 2）を参照）。

ただし、この標準のエコー障害検出では、BFD セッション中に（*I* x *M*）を超えて増加する可能性がある特定のエコーパケットの送信と受信の間の遅延には対処できません。この場合は、係数の期間内にいずれかのエコーパケットが引き続き受信され、カウンタがゼロにリセットされる限り、BFD はネイバーのダウンを宣言しません。Cisco IOS XR 4.0.1 以降では、非バンドルインターフェイスでこの遅延を測定するように BFD を設定できます。詳細については、例 3およびエコー パケットの遅延を参照してください。

エコー障害検出の例

ここでは、非バンドルインターフェイスでの、遅延検出の設定を行わない標準のエコーパケット処理および障害検出のいくつかのシナリオの例について説明します。これらの例では、間隔は 50 ms、また係数は 3 とします。

（注） バンドルインターフェイスにもエコー障害検出に対する同じ間隔と乗数カウンタ方式が使用されますが、これらの値は **bfd address-family ipv4 multiplier** と **bfd address-family ipv4 minimum-interval commands** によって決定され、エコーパケットの受信がないことを検出するために（(*I* x *M* x *M*）の期間を使用します。

例 1

次に、次のエコーが送信される前に各エコーパケットが返される理想的なケースの例を示します。この場合、カウンタは 1 に増分され、次のエコーが送信される前に 0 に戻されます。エコー障害は発生しません。セッション内のエコーパケットのラウンドトリップ遅延が最小間隔より短い限り、このシナリオが発生します。

```
Time (T): Echo#1 TX (count = 1)
T + 1 ms: Echo#1 RX (count = 0)
T + 50 ms: Echo#2 TX (count = 1)
T + 51 ms: Echo#2 RX (count = 0)
T + 100 ms: Echo#3 TX (count = 1)
T + 101 ms: Echo#3 RX (count = 0)
T + 150 ms: Echo#4 TX (count = 1)
T + 151 ms: Echo#4 RX (count = 0)
```

例 2

エコーパケットが一切戻らない例を次に示します。4 番目のエコーパケットの送信後、カウンタが係数値 3 を超えてエコー障害が検出されます。この場合、エコー障害検出は、150 ms（*I* x *M*）期間で発生します。

```
Time (T): Echo#1 TX (count = 1)
T + 50 ms: Echo#2 TX (count = 2)
T + 100 ms: Echo#3 TX (count = 3)
T + 150 ms: Echo#4 TX (count = 4 -> echo failure
```

例 3

次に、標準のエコー障害検出を使用しているときに、BFD セッション中に特定のエコーパケットについてラウンドトリップ遅延が（*I* x *M*）を超えて増加する場合があるが、セッション内の全体的なエコー パケットの戻りの間の遅延は（*I* x *M*）の期間を超えず、またカウンタも乗数を超えないため、ネイバーはダウンしたと宣言されることがない例を示します。

（注） Cisco IOS XR 4.0.1 以降では、**echo latency detect** コマンドを使用して、非バンドルインターフェイスのラウンドトリップ遅延を検出するように BFD を設定できます。

```
Time (T): Echo#1 TX (count = 1)
T + 1 ms: Echo#1 RX (count = 0)
T + 50 ms: Echo#2 TX (count = 1)
T + 51 ms: Echo#2 RX (count = 0)
T + 100 ms: Echo#3 TX (count = 1)
T + 150 ms: Echo#4 TX (count = 2)
T + 151 ms: Echo#3 RX (count = 0; ~50 ms roundtrip latency)
T + 200 ms: Echo#5 TX (count = 1)
T + 250 ms: Echo#6 TX (count = 2)
T + 251 ms: Echo#4 RX (count = 0; ~100 ms roundtrip latency)
T + 300 ms: Echo#7 TX (count = 1)
T + 350 ms: Echo#8 TX (count = 2)
T + 351 ms: Echo#5 RX (count = 0; ~150 ms roundtrip latency)
T + 451 ms: Echo#6 RX (count = 0; ~200 ms roundtrip latency; no failure detection)
T + 501 ms: Echo#7 RX (count = 0; ~200 ms roundtrip latency; no failure detection)
T + 551 ms: Echo#8 RX (count = 0; ~200 ms roundtrip latency; no failure detection)
```

BFD セッションでのエコー パケットの受信の間の遅延を見て、どの遅延も（I x M）の期間を超えていないことに注目してください。

```
Echo#1 RX – Echo#2 RX: 50 ms
```

```
Echo#2 RX - Echo#3 RX: 100ms
Echo#3 RX - Echo#4 RX: 100ms
Echo#4 RX - Echo#5 RX: 100ms
Echo#5 RX - Echo#6 RX: 100ms
Echo#6 RX - Echo#7 RX: 50ms
Echo#7 RX - Echo#8 RX: 50ms
```

バンドル インターフェイスでの BFD のパケット間隔と障害検出時間の概要

セッション間隔 *I* および乗数 *M* のバンドルインターフェイスの BFD の場合、次のパケット間隔および障害検出時間が BFD 非同期モードに適用されます（表 3 : バンドル インターフェイスでの BFD パケット間隔および障害検出時間の例）。

- *I* の値：BFD 制御パケットを送信する最小間隔。
- *I* x *M* の値
  - BFD 制御パケット障害検出時間。
  - BFD エコー パケットを送信する最小間隔。

BFD 制御パケット障害検出時間は、BFD セッションのダウンが宣言されるまでに BFD 制御パケットを受信せずに経過できる最大時間です。

- （*I* x *M*）x *M* の値：BFD エコーパケット障害検出時間。これは、BFD セッションのダウンが宣言されるまでに（非同期モードでのエコーパケット障害検出で説明されている標準の係数カウンタ方式を使用して）BFDエコーパケットを受信せずに経過できる最大時間です。

表 *3 :* バンドル インターフェイスでの *BFD* パケット間隔および障害検出時間の例

設定された非同期制御パケット間隔（ms）（**bfd address-family ipv4 minimum-interval**）	設定された係数（**bfd address-family ipv4 multiplier**）	非同期制御パケット障害検出時間（ms）（間隔 x 係数）	エコー パケット間隔（非同期制御パケット障害検出時間）	エコー パケット障害検出時間（エコー間隔 x 係数）
50	3	150	150	450
75	4	300	300	1200
200	2	400	400	800
2000	3	6000	6000	18000
15000	3	45000	30000[1]	90000

[1] バンドルメンバリンクの BFD の最大エコーパケット間隔は、30 秒または非同期制御パケット障害検出時間のいずれかの最小値です。

エコー パケットの遅延

Cisco IOS XR 4.0.1 よりも前の Cisco IOS XR ソフトウェア リリースでは、BFD は特定のエコー パケットの TX/RX に対する特定の遅延ではなく、エコーパケットの受信がないことだけを検出します。場合によっては、BFDエコーパケットの受信が全体として、障害検出およびパケット送信の全体的な許容値内に収まることもありますが、エコーパケットの特定のラウンドトリップについて時間とともに遅延が増加する可能性があります（例 3）。

Cisco IOS XR リリース 4.0.1 以降、ルータを設定して、非バンドルインターフェイスでエコーパケットの送受信間の実際の遅延を検出でき、また遅延がそのラウンドトリップ遅延に設定されたしきい値を超えるときにセッションをダウンできます。詳細については、エコー遅延検出に基づいた BFD セッション ティアダウンの設定を参照してください。

また、BFD セッションを開始する前に、エコー パケット パスが指定された遅延許容値内にあることも検証できます。エコー起動検証では、BFDセッションの状態変更を可能にする前に設定済みの遅延内に送信が成功するかどうかを確認するために、リンクがダウンしている間、そのリンク上でエコー パケットが定期的に送信されます。詳細については、エコー パスと遅延の検証までの BFD セッション開始の遅延を参照してください。

BFD パケットのプライオリティの設定

オーバーサブスクリプションの状態にあるすべてのインターフェイスについて、リモートBFD エコー パケットに内部プライオリティを割り当てることにより、これらの BFD パケットが他のデータ パケットによって過負荷状態にならないようにする必要があります。さらに、中間スイッチの場合に、リモート BFD エコー パケットの戻りの応答がスイッチ内のその他のすべてのパケットから保護されるように、CoS 値を適切に設定する必要があります。

イーサネット ヘッダーに設定された CoS 値はエコー メッセージ内に保持されない可能性があるため、適切な出力 QoS サービス ポリシーで CoS 値を明示的に設定する必要があります。set cos コマンドを使用して、トラフィック クラスに付加された BFD パケットの CoS 値を設定できます。クラス ベースの無条件パケット マーキングの設定の詳細については、 *Modular QoS Configuration Guide for Cisco ASR 9000 Series Routers*の「Configuring Modular QoS Packet Classification」を参照してください。

IPv4 用 BFD

Cisco IOS XR ソフトウェアは、IPv4 と IPv6 の両方での双方向フォワーディング検出（BFD）のシングルホップおよびマルチホップをサポートします。

IPv4 用 BFD のシングルホップ接続では、Cisco IOS XR ソフトウェアは、番号付けされた物理 Packet-over-SONET/SDH（POS）およびギガビットイーサネットリンクでの非同期モードとエコー モードの両方を次のようにサポートします。

- エコー モードは、BFD 制御パケットを使用してセッションが確立された後にのみ開始されます。BFD バンドル メンバ インターフェイスでは、エコーモードは常に有効になっています。物理インターフェイスの場合は、エコー パケットをサポートするために、BFD 最小間隔も 2 秒未満である必要があります。

- BFD エコー パケットは、送信元および宛先ポート 3785 を使用して、UDP/IPv4 で転送されます。IP パケットの送信元アドレスは出力インターフェイスの IP アドレス（デフォルト）か、または設定されている場合は **router-id** コマンドで指定されたアドレスか、**echo ipv4 source** コマンドで指定されたアドレスであり、宛先アドレスはローカル インターフェイス アドレスです。
- BFD 非同期パケットは、送信元ポート 49152 および宛先ポート 3784 を使用して、UDP および IPv4 で転送されます。非同期モードの場合、IP パケットの送信元アドレスはローカル インターフェイス アドレス、宛先アドレスはリモート インターフェイス アドレスとなります。

（注） BFD マルチホップは、エコー モードをサポートしません。

Cisco IOS XR ソフトウェアで BFD を設定する場合は、次の注意事項を考慮してください。

- BFD は固定長の hello プロトコルで、接続の各終端で転送パスを通じてパケットを定期的に転送します。Cisco IOS XR ソフトウェアは、BFD 適応型検出時間をサポートします。
- BFD は、次のアプリケーションと併用することができます。
  - BGP
  - IS-IS
  - EIGRP
  - OSPF

    および OSPFv3
  - MPLS トラフィック エンジニアリング（MPLS-TE）
  - 静的ルート（IPv4 および IPv6）
  - Protocol Independent Multicast（PIM）
  - Hot Standby Router Protocol（HSRP）
  - Virtual Router Redundancy Protocol（VRRP）

（注） 複数のアプリケーションが同じBFDセッションを共有するとき、最も強力なタイマーのあるアプリケーションがローカルで優先されます。その結果は、次にピア ルータとネゴシエートされます。

- BFD は、次のインターフェイス タイプでの接続でサポートされます。
  - ギガビット イーサネット（GigE）
  - 10 ギガビット イーサネット（TenGigE）

- Packet-over-SONET/SDH（POS）
- シリアル（Serial）
- 仮想 LAN（VLAN）
- Bridge Group Virtual Interface（BVI）
- サテライトインターフェイス
- バンドル、GRE、PWHE などの論理インターフェイス

（注） BFDは上のインターフェイスタイプでサポートされ、特に説明されていない限り、論理インターフェイスではサポートされません。

- Cisco IOS XR ソフトウェアは、BFD バージョン 0 およびバージョン 1 をサポートします。BFD セッションは、ネイバーに応じていずれかのバージョンを使用して確立されます。BFD バージョン 1 はデフォルト バージョンであり、セッション確立において最初に試行されます。

IPv6 用 BFD

Cisco IOS XR ソフトウェアは、IPv4 と IPv6 の両方での双方向フォワーディング検出（BFD）をサポートします。IPv6 での双方向フォワーディング検出（BFD）では、IPv6 アドレスを使用するインターフェイスでの稼働中の接続の確認をサポートします。

稼働中の接続の確認は、IPv4 と IPv6 の両方のインターフェイスについて、同じサービスとプロセスによって実行されます。同一ラインカード上で、IPv4 と IPv6 の両方の BFD セッションを同時に実行することができます。

IPv4 用 BFD でサポートされるものと同じ機能と設定が IPv6 用 BFD でもサポートされます。

バンドル VLAN での BFD

バンドル VLAN 上の IPv4 用 BFD は、スタティック ルーティング、IS-IS、および OSPF を使用してサポートされます。バンドルVLANインターフェイスでBFDセッションを実行すると、VLAN バンドルがアップ状態である限り BFD セッションはアクティブな状態となります。

VLAN バンドルがアクティブであれば、次に示すイベントによって BFD セッションが失敗することはありません。

- コンポーネント リンクの障害。
- 1 つ以上のコンポーネント リンクをホストするラインカードの活性挿抜（OIR）。
- バンドルへのコンポーネント リンクの追加（設定による）。

- バンドルからのコンポーネント リンクの削除（設定による）。
- コンポーネント リンクのシャットダウン。
- RP スイッチオーバー。

（注） VLAN バンドルの設定の詳細については、「*Configuring Link Bundling on the Cisco ASR 9000 Series Router*」」のモジュールを参照してください。

バンドル VLAN で BFD を設定する場合には、次の事項に注意する必要があります。

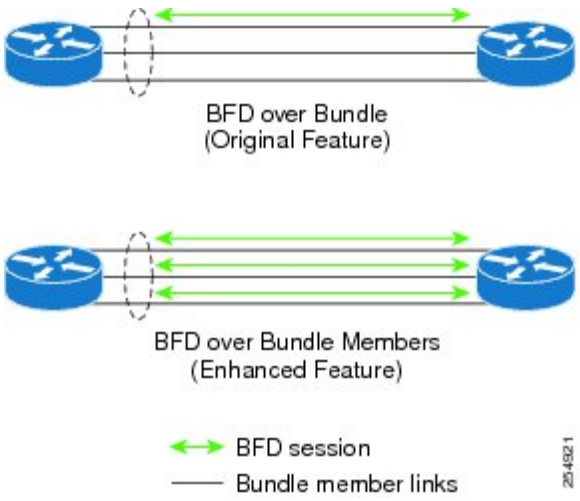
- RP スイッチオーバーの場合、設定されているネクストホップは Routing Information Base（RIB）に登録されます。
- BFD 再起動の場合、スタティック ルートは RIB に残ります。BFD セッションは、BFD の再起動時に再確立されます。

（注） スタティック BFD セッションは、ネクストホップがルータに直接接続されているアドレスプレフィックスを持つピアでサポートされます。

リンク バンドルのメンバ リンク上の BFD

以前のリリースのように単一のバンドルメンバのみではなく、BFD ではリンク上のレイヤ 3 接続をモニタする個々の物理バンドルメンバリンクで BFD セッションをサポートしています（図 37）。

図 *14*：バンドル上の元の *BFD* およびバンドル メンバ リンク アーキテクチャ上の拡張 *BFD* における *BFD* セッション



リンク バンドルで BFD を実行する場合は、そのバンドルの一部である基盤となる各物理インターフェイスで、独立した BFD セッションを実行できます。

BFD がリンク バンドル メンバで実行されているとき、接続の次のレイヤは BFD のインターフェイス状態モニタリングの一部として実質上テストされます。

- レイヤ 1 の物理状態
- レイヤ 2 のリンク アクセス コントロール プロトコル（LACP）状態
- レイヤ 3 の BFD 状態

各バンドル メンバ リンクの BFD エージェントはリンクの状態変更をモニタリングします。バンドル メンバ リンクで実行されているセッションの BFD エージェントはバンドル マネージャと通信します。バンドル マネージャは、メンバ リンクの状態とバンドル全体のアベイラビリティを特定します。メンバ リンクの状態は、そのバンドル用に設定された最小アクティブ リンクまたは最小アクティブ帯域幅のしきい値に基づいて、バンドル全体の状態に影響を及ぼします。

メンバ リンクおよびバンドル ステータスでの BFD 状態変更動作の概要

ここでは、バンドル メンバ リンクの状態がいつアクティブまたはダウンとして特徴づけられるか、およびそれらの全体的なバンドル ステータスへの影響について説明します。

- すでにアクティブであるか、または非アクティブであるバンドル メンバ インターフェイスで BFD を設定できます。インターフェイスで LACP を使用してアップになる BFD セッションの場合は、LACP が配布状態に達している必要があります。

  リンクが LACP 分散状態にあり、BFD セッションがアップである場合、BFD メンバ リンクは「IIR Active」です。

- BFD セッションがダウンである場合は、LACP 状態遷移が受信されない限り、BFD メンバ リンクは「IIR Attached」です。

- リンク バンドル BFD セッションのダウンを宣言する前に、ピアからの BFD 状態変更通知（SCN）の受信遅延を許可するように最大 3600 秒（1 時間）のタイマーを設定できます。設定可能なタイマーは、次の状況に適用されます。

  - BFD セッションの開始（**bfd address-family ipv4 timers start** コマンド）：セッションのアップを宣言するために BFD ピアからの予測される通知が受信されるように、BFD メンバリンクセッションの開始後に見越しておく秒数。その期間の後に SCN が受信されない場合は、BFD セッションのダウンが宣言されます。
  - ネイバーによる BFD 設定の削除の通知（**bfd address-family ipv4 timers nbr-unconfig** コマンド）：BFD ピア間の設定の不一致をすべて解決できるように、BFD 設定が BFD ネイバーによって削除されたことの通知の受信後に見越しておく秒数。指定されたタイマーに達する前に BFD 設定の問題が解決されない場合、BFD セッションのダウンが宣言されます。

- BFD セッションは、次のいずれかが発生すると DOWN 通知を送信します。

- ローカル メンバ リンクで BFD 設定が削除される。

  BFD システムは、設定が削除されたことを隣接ルータのピアに通知します。BFD セッションは、他のバンドル メンバ インターフェイスまたは全体的なバンドル状態に影響を与えることなく、バンドル マネージャから削除されます。

- メンバ リンクがバンドルから削除される。

  バンドルからのメンバ リンクの削除によって、バンドル メンバは強制的に削除されます。BFD セッションは削除され、隣接ルータ上の BFD がそのセッションを NBR_CONFIG_DOWN ではなく、DOWN とマークします。

- 次の場合、DOWN 通知は送信されませんが、内部インフラストラクチャは DOWN が発生したかのようにイベントを処理します。

  - 隣接ルータ上で BFD 設定が削除され、ネイバー設定解除タイマー（設定されている場合）の期限が切れる。

    BFD システムは BFD 設定が隣接ルータから削除されたことをバンドルマネージャに通知し、**bfd timers nbr-unconfig** がリンクで設定されている場合は、タイマーを開始します。タイマーの期限が切れる前にローカル ルータ上で BFD 設定が削除された場合は、タイマーは停止し、ローカル ルータ上の BFD 設定を削除した場合と同じ動作になります。

    タイマーが切れた場合、動作は BFD セッション DOWN 通知の場合と同じです。

  - BFD ピアからの通知が受信される前に、セッション開始タイマーの期限が切れる。

- バンドル メンバの BFD セッションがバンドル マネージャに BFD 状態変更通知を送信します。バンドル メンバ インターフェイスの BFD 状態変更通知がバンドル マネージャによって受信されると、バンドル マネージャは、対応するバンドル インターフェイスが使用可能かどうかを判断します。

- バンドルのアクティブ メンバ リンクの最小数のしきい値は、メンバ リンクの状態に基づいてバンドルがアクティブなままか、ダウンであるかを判断するためにバンドル マネージャによって使用されます。すでにアクティブであるバンドルで BFD が開始された場合、そのバンドルの BFD 状態は、既存のすべてのアクティブ メンバの BFD 状態が既知であるときに宣言されます。

  メンバの状態が変更されるたびに、バンドル マネージャは、アクティブ メンバの数がアクティブ リンクのしきい値の最小数より小さいかどうかを判断します。その場合は、バンドルが DOWN 状態になるか、または DOWN 状態のままになります。アクティブ リンクの数が最小しきい値に達すると、バンドルは UP 状態に戻ります。

- バンドルで別のしきい値を設定できます。そのしきい値は、バンドルマネージャによって使用され、バンドルが DOWN 状態になる前に使用できるアクティブな帯域幅の最小値が決定されます。これを設定するには、**bundle minimum-active bandwidth** コマンドを使用します。

- BFD サーバは、バンドル インターフェイスの状態変更に関するバンドル マネージャからの情報に応答し、そのインターフェイス上のアプリケーションに通知するとともに、システム メッセージや MIB トラップも送信します。

BFD マルチパス セッション

BFD は、GRE トンネル インターフェイスや PWHE インターフェイスなどの仮想インターフェイス経由で、または「マルチホップパスの BFD」の項で説明されているようにマルチホップ離れたインターフェイス間で適用できます。これらのタイプの BFD セッションは、BFD マルチパス セッションと呼ばれます。

宛先への 1 つのパスがアクティブである限り、次のイベントによって BFD マルチパス セッションが失敗する場合も、失敗しない場合もあります。それは、ネゴシエートされた間隔と、フォワーディング プレーンの更新に必要なコンバージェンス時間の関係に依存するためです。

- パスの障害
- 1 つ以上のパスをホストするラインカードのオンライン挿入または削除（OIR）
- パスを構成するリンクの削除（設定による）
- パスを構成するリンクのシャットダウン

マルチパスセッションのパケットを送受信するために使用できる基盤となるメカニズムに対して少なくとも 1 枚のラインカードを有効にするには、**bfd mutlipath include location** *location-id* コマンドを設定する必要があります。

BFD マルチパス セッションが、bfd multipath include の設定から削除されるか、オンラインで削除されるか、またはメンテナンス モードにされようとしているラインカードでホストされている場合、BFD は、そのラインカードでホストされているすべての BFD マルチパス セッションを別のラインカードに移行しようとします。その場合は、RIB からスタティック ルートが削除されてから、BFD セッションが再度確立され、RIB に含まれます。

BFD マルチパスセッションの場合、入力インターフェイスと出力インターフェイスは、ルーティングテーブルの更新に基づいて変更されることがあります。マルチパスセッション BFD パケットを優先的に処理する必要がある場合は、ルータの考えられる入出力インターフェイスを含めて、パス全体の QoS ポリシーを設定する必要があります。

QoS ポリシーは、入力および出力の BFD パケットを優先度レベル 1 のキューまたは優先度レベル 2 のキューに分類する必要があります。同様のアプローチは、BVI での BFD セッションと「バンドル上での VLAN を介した BFD」（つまり、BLB）に適用されます。

例:

```
ipv4 access-list BFD
5 permit udp any any eq 4784
!
class-map match-any BFDCLASS
match access-group ipv4 BFD
!
policy-map BFD
class BFDCLASS
```

```
  priority level 1
  police rate 10 kbps
!
interface GigabitEthernet0/2/0/1
service-policy output BFD
service-policy input BFD
```

PW ヘッドエンドとその設定の詳細については、の「*Implementing Virtual Private LAN Services*」モジュールを参照してください。GRE の詳細については、の「*Implementing MPLS Layer 2 VPNs*」モジュールを参照してください。

マルチホップパスの BFD

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。BFD マルチホップをサポートするアプリケーションには、外部 BGP と内部 BGP があります。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。

BFD マルチホップ機能では、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。**bfd multihop ttl-drop-threshold** コマンドを使用すると、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できます。BFD マルチホップは、BFD シングルホップで現在サポートされてい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。BFD マルチホップでは、グローバル ルーティング テーブルと VRF の両方にある IPv4 アドレスがサポートされます。

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。

MPLS トラフィック エンジニアリング LSP を介した BFD

Cisco IOS XR ソフトウェアの MPLS トラフィック エンジニアリング ラベル スイッチド パス（LSP）機能を介した双方向フォワーディング検出（BFD）は、MPLS ラベルスイッチドパス LSP データプレーンの障害を検出します。BFD 制御パケットに必要なコントロールプレーン処理は、LSP ping メッセージに必要な処理よりも比較的小さいため、BFD を展開すると多数の LSP のデータプレーン障害をより迅速に検出できます。

Cisco IOS XR ソフトウェアでの MPLS TE LSP を介した BFD 実装は、 *RFC 5884*：*MPLS* ラベルスイッチドパス（*LSP*）での *Bidirectional Forwarding Detection*（*BFD*）に基づいています。LSP ping は、MPLS データプレーンの障害を検出し、コントロールプレーンと照合して MPLS LSP データプレーンを確認するための既存のメカニズムです。BFD は、MPLS データプレーンでの

障害の検出に使用できますが、コントロールプレーンと照合するMPLS LSPデータプレーンの確認には使用できません。LSP ping と BFD を組み合わせると、多数の LSP 上でデータプレーン障害を迅速に検出できます。

BFDを高速障害検出トラフィックのブラックホールとして使用することでネットワークの信頼性と稼働時間を引き上げるため、MPLS TE LSP を介した BFD は MPLS をマルチ サービス トランスポートとして展開し、BFDを高速障害検出メカニズムとして使用するネットワークに使用します。

MPLS TE LSP を介した BFD のサポート：

- BFD 非同期モード（BFD エコーモードはサポートされていません）
- IPv4 のみ（MPLS コアは IPv4 であるため）
- BFD パケットで IP DSCP 6 を伝送（インターネット制御）
- TE トンネルの起動、再最適化、およびパスの保護のための BFD の使用（スタンバイおよび FRR）
- 最速の検出時間（100 ms x 3 = 300 ms）
- BFD セッションが起動した後の任意の定期的な LSP ping の検証
- 保留 BFD 障害パスオプションへのダンプニング
- テールエンドからヘッドエンドへの BFD パケットは、次の 2 つの方法で使用されます。
  - テールエンドからヘッドエンドへの BFD パケットは、IP ルーティングされる（IPv4 マルチホップ：ポート番号 4784）。
  - テールエンドからヘッドエンドへの BFD パケットは、テールエンドからヘッドエンドへのラベルパスを使用してMPLS LDPがコアで使用できる場合は、ラベルが切り替わる（ポート番号 3784）。

バンドルインターフェイス上での BFD 用のエコータイマーの設定

エコータイマーの設定では、バンドルメンバリンク上のIPv4 BFDセッションでのエコーパケットの最小間隔を指定できます。エコータイマー値をグローバルに設定するには、**bfd echo ipv4 bundle-per-member minimum-interval** コマンドを使用します。バンドルイーサネット インターフェイスの最小間隔値をローカルに設定するには、**bfd address-family ipv4 echo minimum-interval** を使用します。

（注） この機能は、メンバリンク単位のバンドルを介したシスコの標準的な BFD モードにのみ適用できます。

これらのコマンドの詳細については、『*Cisco ASR 9000 Series Router module of Cisco ASR 9000 Series Aggregation Services Router Routing Command Reference*』の「*BFD Commands*」を参照してください。

次の表に、グローバルおよびローカルのエコー設定の組み合わせによるエコータイマーの動作を示します。

表 *4:* グローバルおよびローカルのエコー設定を使用したエコータイマーの動作

グローバルエコー最小間隔値 コマンド：**bfd echo ipv4 bundle-per-member minimum-interval**	ローカル バンドル イーサネット インターフェイス固有のエコー最小間隔値 コマンド：**bfd address-family ipv4 echo minimum-interval**	グローバルおよびローカルの設定に基づいて **BoB** セッションで使用されるエコー値
未設定	未設定	非同期 * 乗数
グローバル値が非同期 * 乗数より小さい	未設定	非同期 * 乗数
グローバル値が非同期 * 乗数を超えている	未設定	グローバル
未設定	ローカル値が非同期 * 乗数を超えている	ローカル
Not configured	ローカル値が非同期 * 乗数より小さい	非同期 * 乗数
グローバルが設定されている（任意の値）	ローカル値が非同期 * 乗数を超えている	ローカル
グローバルが設定されている（任意の値）	ローカル値が非同期 * 乗数より小さい	非同期 * 乗数

（注）
- テーブル内の乗数は、リモートの乗数値を示します。
- 非同期は、ネゴシエートされた非同期の最小間隔値を示します。

R5.3.0 デバイスで、R5.3.0 よりも前のバージョンで実行されているデバイスとの BoB セッションがある場合は、デフォルトのエコータイマー値を保持するか、両方のデバイスで同じ値を設定することをお勧めします。

論理バンドルを介した双方向転送検出

論理バンドル上の双方向フォワーディング検出（BFD）機能では、RFC 5880 に基づいて、バンドルインターフェイスを介してBFDを実装および配置します。論理バンドル上のBFD（BLB）機能は BVLAN 機能を置き換え、純粋な RFC5880 の方法でバンドル インターフェイス上で BFD を実行する他のプラットフォームとの相互運用性に関する特定の問題を解決します。これらのプラットフォームには、Cisco IOS または Cisco Nexus OS ソフトウェアを実行している他のシスコ製品だけでなく、他のベンダーの製品が含まれています。

BLB は、マルチパス（MP）シングルホップ セッションです。BLB では、セッションが実行されているバンドルインターフェイスに関する限られた知識しか必要としません。これは、BFD がバンドルを 1 つの大きなパイプとして処理するためです。BLB を機能させるには、バンドル インターフェイスの IP アドレス、インターフェイス タイプ、および制限に関する情報だけが必要です。バンドル メンバのリスト、メンバの状態、設定されている最小または最大のバンドル リンクなどの情報は必要ありません。

BLB は、IPv4 アドレス、IPv6 グローバル アドレス、および IPv6 リンクローカル アドレスでサポートされます。ソフトウェアの現在のバージョンは、ラインカードごとに合計 200 のセッションをサポートしています（物理および論理のサブインターフェイスの BFD シングルホップ、バンドルを介した BFD（BoB）および BLB を含む）。ラインカードごとの BFD 制御パケットの最大処理能力も、7000 pps（1 秒あたりのパケット数）へと強化されています。

（注） ISSU は、論理バンドルを介した BFD 機能ではサポートされていません。

論理バンドル上の BFD 機能は、Cisco ASR 9000 シリーズ SPA インターフェイス プロセッサ 700 ではサポートされません。

総称ルーティングカプセル化を介した双方向フォワーディング検出

Generic Routing Encapsulation（GRE）機能を介して Bidirectional Forwarding Detection（BFD）を使用すると、既存の GRE キープアライブメカニズムよりも迅速にネットワーク障害を検出できます。BFD は、エンドポイントが BFD ピアである GRE トンネルを介してセッションを確立します。BFD は障害検出時にトンネルを停止させますが、障害が解消されると、トンネル キープアライブ メカニズムによってトンネルを回復させることができます。BFD は、IPv4 GRE トンネルモードでのみサポートされています。

BFD セッションの送信元と宛先は、GRE トンネルの IPv4 アドレスと同じになります。

トンネルキープアライブが有効になっている場合、GRE トンネルで BFD を有効にすることも、その逆もできません。

GRE トンネリングプロトコルは、さまざまなプロトコルパケットタイプを IP トンネルの内部にカプセル化し、IP インターネットワーク上のリモートポイントにある 2 台のルータ間に仮想ポイントツーポイントリンクを作成します。GRE を使用すると、自身のコア ネットワーク内で MPLS を実行していないサービス プロバイダーが VPN サービスを提供できるようになります。

GRE 上の BFD 機能は、Cisco ASR 9000 シリーズ SPA インターフェイス プロセッサ 700 ではサポートされません。

BFD は、RFC5880 に従った、GRE 番号付きインターフェイス上の IPv4 シングルホップ バージョン 1 非同期モードを提供します。

Generic Routing Encapsulation を介した双方向フォワーディング検出の設定

次の項では、Generic Routing Encapsulation（GRE）を介した双方向フォワーディング検出（BFD）の設定方法を示します。

図 *15 : GRE* を介した *BFD*

設定例

PE1 ルータで次のステップを設定します。

```
Router# configure
Router(config)# bfd
Router(config-bfd)# multipath include location 0/0/CPU0
Router(config-bfd)# exit
Router(config)# interface tunnel-ip 100
Router(config-if)# ipv4 address 10.0.0.1 255.255.255.252
Router(config-if)# tunnel source Loopback 100
Router(config-if)# tunnel destination 10.2.2.2
Router(config-if)# tunnel bfd destination 10.0.0.2
Router(config-if)# tunnel bfd minimum-interval 300
Router(config-if)# tunnel bfd multiplier 5
Router(config-if)# tunnel bfd period 5
Router(config-if)# tunnel bfd retry 2
Router(config-if)# commit
```

PE2 ルータで次のステップを設定します。

```
Router# configure
Router(config)# bfd
Router(config-bfd)# multipath include location 0/0/CPU0
Router(config-bfd)# exit
Router(config)# interface tunnel-ip 100
Router(config-if)# ipv4 address 10.0.0.2 255.255.255.252
Router(config-if)# tunnel source Loopback 100
Router(config-if)# tunnel destination 10.1.1.1
Router(config-if)# tunnel bfd destination 10.0.0.1
Router(config-if)# tunnel bfd minimum-interval 300
Router(config-if)# tunnel bfd multiplier 5
Router(config-if)# tunnel bfd period 5
Router(config-if)# tunnel bfd retry 2
Router(config-if)# commit
```

実行コンフィギュレーション

```
/* The following is the running configuration from PE1 Router */
```

```
bfd
 multipath include location 0/0/CPU0
!
interface tunnel-ip 100
 ipv4 address 100.0.0.1 255.255.255.252
 tunnel source Loopback 100
 tunnel destination 10.2.2.2
 tunnel bfd destination 10.0.0.2
 tunnel bfd minimum-interval 300
 tunnel bfd multiplier 5
 tunnel bfd period 5
 tunnel bfd retry 2

/* The following is the running configuration from PE2 Router */

bfd
 multipath include location 0/0/CPU0
!
interface tunnel-ip 100
 ipv4 address 100.0.0.2 255.255.255.252
 tunnel source Loopback 100
 tunnel destination 10.1.1.1
 tunnel bfd destination 10.0.0.1
 tunnel bfd minimum-interval 300
 tunnel bfd multiplier 5
 tunnel bfd period 5
 tunnel bfd retry 2
```

確認

```
Router# show interfaces tunnel-ip 1
Mon Jul  9 10:54:06.952 IST
tunnel-ip1 is up, line protocol is up
  Interface state transitions: 1
  Hardware is Tunnel
  Internet address is 20.1.1.2/24
  MTU 1500 bytes, BW 100 Kbit (Max: 100 Kbit)
     reliability 255/255, txload 2/255, rxload 2/255
  Encapsulation TUNNEL_IP,  loopback not set,
  Last link flapped 00:03:54
  Tunnel TOS 0
  Tunnel mode GRE IPV4
  Keepalive is enabled, interval 10 seconds, maximum retry 3
  Tunnel source 10.0.0.2, destination 10.1.1.1/32
  Tunnel TTL 255
  Last input 00:00:00, output 00:00:00
  Last clearing of "show interface" counters never
  5 minute input rate 1000 bits/sec, 3 packets/sec
  5 minute output rate 1000 bits/sec, 3 packets/sec
     999 packets input, 75088 bytes, 0 total input drops
     0 drops for unrecognized upper-level protocol
     Received 0 broadcast packets, 0 multicast packets
     1001 packets output, 51380 bytes, 0 total output drops
     Output 0 broadcast packets, 0 multicast packets

Router# show bfd session interface tenGigE 0/1/1/0.200 detail

I/f: TenGigE0/1/1/0.200, Location: 0/0/CPU0
Dest: 10.1.1.2
Src: 10.0.0.2
State: UP for 0d:0h:6m:9s, number of times UP: 1
Session type: PR/V4/SH
Received parameters:
```

```
Version: 1, desired tx interval: 300 ms, required rx interval: 300 ms
Required echo rx interval: 0 ms, multiplier: 3, diag: None
My discr: 2148532226, your discr: 2148335671, state UP, D/F/P/C/A: 0/0/0/1/0
Transmitted parameters:
Version: 1, desired tx interval: 15 ms, required rx interval: 15 ms
Required echo rx interval: 0 ms, multiplier: 3, diag: None
My discr: 2148335671, your discr: 2148532226, state UP, D/F/P/C/A: 0/1/0/1/0
Timer Values:
Local negotiated async tx interval: 300 ms
Remote negotiated async tx interval: 300 ms
Desired echo tx interval: 0 s, local negotiated echo tx interval: 0 ms
Echo detection time: 0 ms(0 ms*3), async detection time: 900 ms(300 ms*3)
Local Stats:
Intervals between async packets:
   Tx: Number of intervals=4, min=1 ms, max=346 s, avg=88 s
       Last packet transmitted 23 s ago
   Rx: Number of intervals=11, min=1 ms, max=346 s, avg=32 s
       Last packet received 23 s ago
Intervals between echo packets:
   Tx: Number of intervals=0, min=0 s, max=0 s, avg=0 s
       Last packet transmitted 0 s ago
   Rx: Number of intervals=0, min=0 s, max=0 s, avg=0 s
       Last packet received 0 s ago
Latency of echo packets (time between tx and rx):
   Number of packets: 0, min=0 ms, max=0 ms, avg=0 ms
Session owner information:
                                    Desired               Adjusted
  Client               Interval   Multiplier Interval   Multiplier
  -------------------- ---------- ---------- ---------- ----------
  tunl_gre_ma           15 ms      3          15 ms      3


Router# show bfd client

Mon Jul  9 10:55:16.025 IST
Name                  Node       Num sessions
--------------------- ---------- --------------
L2VPN_ATOM            0/0/CPU0   0
bundlemgr_distrib     0/0/CPU0   0
object_tracking       0/0/CPU0   0
pim6                  0/0/CPU0   0
pim                   0/0/CPU0   0
tunl_gre_ma           0/0/CPU0   1

Router# show tunnel ip keepalive
Mon Jul  9 10:54:30.005 IST

---- Tunnel GRE Keepalive Database ----

interface tunnel-ip1
 tunnel interface/basecaps state UP/UP
 tunnel ifhandle 0x90
 tunnel source 10.0.0.2
 tunnel destination 10.1.1.1
 tunnel transport vrf id 0x60000000
 tunnel transport vrf table id 0xe0000000
 tunnel ttl 255
 tunnel flags 0x1400
 tunnel keepalive max retries 3
 tunnel keepalive period 10
 tunnel keepalive state 0x2
 tunnel keepalive fail count 0
 tunnel keepalive packets sent 27
```

```
Timestamp of last KA sent Mon Jul  9 10:54:21 2018
tunnel keepalive packets received 24
 Timestamp of last KA received Mon Jul  9 10:54:21 2018
```

関連コマンド

- **bfd minimun-interval**
- **bfd multiplier**
- **tunnel bfd**

双方向フォワーディング検出 IPv6 マルチホップ

双方向フォワーディング検出（BFD）IPv6 マルチホップ機能では、BFD ネイバーを物理的または論理的に複数ホップ離れた場所に配置できる IPv6 マルチホップ BFD セッションが可能になります。BFD ネイバーに到達するために、複数のパスを使用できます。BFD パケットは、対応する BFD セッションをホストしている可能性のあるラインカードで受信されます。あるラインカード内の BFD エージェントが、別のラインカード上の出力インターフェイスから BFD パケットを送信することが必要になる場合があります。

IPv6 マルチホップに対する BFD のサポートは、BFD IPv4 マルチホップと同様です。BFD IPv6 マルチホップは、ASR 9000 イーサネット ラインカードおよび ASR 9000 拡張イーサネット ラインカードでサポートされます。

BFD IPv6 マルチホップ機能は、Cisco ASR 9000 シリーズ SPA インターフェイス プロセッサ 700 ではサポートされません。

BFD IPV6 マルチホップでは、BFD ネイバーが常に 1 ホップ離れていて、ラインカード内の BFD エージェントが常に同じラインカード上のローカルインターフェイス経由で BFD パケットを受信または送信する、単一パス IPv6 BFD セッションの制約事項が解消されます。

IPv6 マルチホップ リンクの BFD スイッチング メカニズムは、BFD パケットがあるエンド ポイント ノードから他のエンド ポイント ノードに送信されたときに使用されます。BFD パンティング メカニズムは、BFD パケットがリモート エンド ポイント ノードで受信されたときに使用されます。

疑似回線ヘッドエンドを介した BFD

疑似回線ヘッドエンド（BFDoPWHE）を介した Bidirectional Forwarding Detection機能により、カスタマーエッジ（CE）から擬似回線ヘッドエンド（S-PE）のリンクを介した BFD サポートが可能になり、eBGP ネイバー間のパスに沿って迅速に障害を検出できます。

PWHE を介した BFD は、ASR 9000 拡張イーサネットラインカードでのみサポートされています。

PWHE を介した BFD は次をサポートしています。

- CE と PWHE PE 間のエンドツーエンドの障害検出のための擬似回線単位の BFD セッション

- BFDv4 for IPv4 と BFDv6 for IPv6（スタティックと BGP）
- PWHE を介した BFD 非同期モード
- 擬似回線 VC タイプ 4 およびタイプ 5

PWHE を動作させるには、PWHE 汎用インターフェイスリストに含まれているラインカードのいずれかで BFD エージェントをホストする必要があります。BFD マルチパスは、汎用インターフェイスリストに含まれているラインカード用に設定する必要があります。

BFD マルチパスセッションをホストして、 PWHE を介した BFD を有効にするために特定のラインカードを含めるには、**bfd multipath include location** *node-id* コマンドを使用します。

サテライトインターフェイスを介した BFD

サテライトインターフェイスを介した双方向フォワーディング検出（BFD）機能は、サテライトラインカード上で BFD をサポートします。サテライトインターフェイスは、仮想（バンドル）インターフェイスと呼ばれています。BFD はマルチパス インフラストラクチャを使用して、サテライトラインカード上の BFD をサポートします。サテライトを介した BFD はマルチパス（MP）シングルホップセッションであり、IPv4 アドレス、IPv6 グローバルアドレス、および IPv6 リンクローカルアドレスでサポートされています。サテライトを介した BFD は、ASR 9000 拡張イーサネットラインカーでのみサポートされており、また、非同期モードでサポートされています。エコーモードでは、サテライトを介した BFD はサポートされていません。

（注）

- **bfd multipath include location** *node-id* コマンドは ASR 9000 イーサネットラインカードではサポートされていません。したがって、サテライトインターフェイスを介した BFD 機能は、ASR 9000 イーサネットラインカードでは動作しません。
- サテライトインターフェイスを介した BFD は、nV エッジシステムではサポートされていません。
- nV サテライトのアクセスポートバンドルは、シャーシ間リンク（ICL）がバンドル（バンドルトポロジを介したバンドル）でもある場合、バンドルを介した BFD（BoB）をサポートしません。

IRB を介した BFD

VLAN でルータを使用するには、ルータが VLAN ヘッダーを維持した状態で、あるインターフェイスから別のインターフェイスにフレームを転送可能である必要があります。レイヤ 3（ネットワーク層）プロトコルをルーティングするようにルータを設定する場合は、フレームを受信するインターフェイスで VLAN と MAC レイヤが終端します。MAC 層のヘッダーは、ルータがネットワーク層プロトコルをブリッジングする場合に維持できます。ただし、通常のブリッジングでも VLAN ヘッダーは終端されます。

Cisco IOS XR ソフトウェア リリース 5.1.0 以降の Integrated Routing Bridging（IRB）機能を使用すると、同じインターフェイス上の同じネットワーク層プロトコルをルーティングし、ブリッジングするようにルータを設定できます。これにより、VLANヘッダーは、あるインターフェイスから別のインターフェイスへのルータを通過する間、フレームで維持されるようになります。IRB は、ブリッジグループ仮想インターフェイス（BVI）により、ブリッジドドメインとルーテッドドメイン間でルーティングする機能を提供します。BVI は、ルータ内の仮想インターフェイスであり、ブリッジングをサポートしないが、ルータ内のルーテッドインターフェイスに相当するブリッジ グループを代表する、正常なルーテッド インターフェイスのように動作します。BVI のインターフェイス番号は、仮想インターフェイスが代表するブリッジ グループの番号です。この番号が BVI とブリッジグループ間のリンクになります。

BVI はルーテッドインターフェイスとしてブリッジグループを代表するため、ネットワーク層アドレスのようなレイヤ 3（L3）特性のみにより設定する必要があります。同様に、プロトコルのブリッジングのために設定されたインターフェイスは、どのような L3 特性によっても設定してはなりません。

IRB を介した BFD はマルチパスのシングルホップセッションです。BFD マルチパスセッションでは、BFD を仮想インターフェイス上か、または複数ホップ離れたインターフェイス間に適用できます。Cisco IOS XR ソフトウェアの BFD マルチホップは、「*RFC 5883*：マルチホップパスの双方向転送検出（*BFD*）（*RFC 5883—Bidirectional Forwarding Detection (BFD) for Multihop Paths*）」に基づいています。IRB を介した BFD は、IPv4 アドレス、IPv6 グローバルアドレス、および IPv6 リンクローカルアドレスでサポートされています。IRB を介した BFD は非同期モードでのみサポートされており、エコーモードはサポートされていません。IRB を介した BFD 機能は、ASR 9000 拡張イーサネットラインカードでのみサポートされています。

メンバリンク単位のバンドルを介した BFD

メンバリンク単位のバンドルを介した BFD（BoB）モードは、異なるプラットフォーム間で相互運用可能なリンク集約（LAG）メンバリンクの標準ベースの高速障害検出機能です。これにより、Cisco または IETF 標準のいずれかを使用するめのメンバリンク単位のモードを選択するオプションが提供されます。この機能は、Cisco ASR 9000 拡張イーサネットラインカードでのみサポートされています。

（注）
- システム内のすべてのバンドルは、任意の単一時点で複数のモードに属することができます。
- バンドルを介した BoB を設定するためのグローバルコマンドは、リリース 5.3.0 までのみ使用できます。5.3.1 以降のリリースでは、バンドル単位でバンドルを介した BFD の CISCO/IETF モードのサポートを設定するオプションが提供されています。

- Cisco モードでは CDP MAC を使用しますが、IETF モードでは IANA によって割り当てられた MAC が使用されます。
- バンドルを介した Cisco BFD セッションでは宛先 UDP ポートとして 3784 が使用され、バンドルを介した IETF BFD セッションでは宛先 UDP ポートとして 6784 が使用されます。

制限事項

次の制限は、メンバリンク単位のバンドルを介した BFD モード機能に適用されます。

- Cisco ASR 9000 拡張イーサネットラインカードでのみサポートされます。
- BFD エコーモードはサポートされていません。
- IPv6 は IETF モードでサポートされており、シスコモードではサポートされていません。
- モードの変更は、新しいセッションにのみ適用されます。既存のセッションに対してモードの変更を適用するには、セッションを削除してから再作成します。
- メンバインターフェイス上の BFD セッションは、1 つのモード（Cisco モードまたは IETF モード）にのみ属することができます。同じバンドル内でのモードの混在はサポートされていません。

バンドルを介した BFD の CISCO/IETF モードのバンドル単位でのサポート

BFD over Bundle（BoB）モードは、異なるプラットフォーム間で相互運用可能なリンク アグリゲーション（LAG）メンバー リンクの標準ベースの高速障害検出です。バンドルごとの BoB サポートでは、さまざまなシステムでリロードやプロセスの再起動を必要とせずに、バンドルごとに Cisco または IETF 標準を選択するオプションが提供されます。デフォルトは Cisco モードです。

（注） バンドルを介した CISCO/IETF BoB を設定するための以前のリリースで使用可能なグローバル レベルのコマンドはリリース 5.3.1 以降、廃止されています。スムーズにアップグレードできるようにするため、バンドルをインターフェイスレベルで設定することをお勧めします。

- Cisco モードでは CDP MAC を使用しますが、IETF モードでは IANA によって割り当てられた MAC が使用されます。
- バンドルを介した Cisco BFD セッションでは宛先 UDP ポートとして 3784 が使用され、バンドルを介した IETF BFD セッションでは宛先 UDP ポートとして 6784 が使用されます。

制約事項

次の制限はバンドルを介した BFD モード 機能に適用されます。

- Cisco ASR 9000 拡張イーサネットラインカードでのみサポートされます。
- BFD モードの変更（Cisco から IETF およびその逆）は、バンドルの BFD 状態が「ダウン（down）」または「動作不能（nonoperational）」の場合にのみ実行されます。

（注） BoB を動作不能にするには、 **no bfd address-family ipv4 fast-detect** コマンドを使用します。また、特定のバンドルでシャットダウンを設定することによっても、バンドルを「ダウン」状態に設定できます。

- バンドルが新しい BFD モードの変更を受け入れるようにするには、既存の BFD セッションをダウンさせてから、再作成する必要があります。
- BFD エコーモードは、バンドルを介した IETF BFD（BoB）セッションではサポートされていません。

BFD ダンプニング

双方向フォワーディング検出（BFD）は、ネイバーに対する到達可能性の障害を迅速に認識して通知するために、ルーティングプロトコルで使用されているメカニズムです。BFD でクライアントの到達可能性ステータスの変更が検出されると、そのネイバーがすぐに通知を受けます。小さい障害であってもコンバージェンスに影響を与えないよう、ルーティングテーブルの変更を最小限にすることが重要になる場合があります。過剰にフラップが発生する不安定なリンクは、ネットワークの他のデバイスにかなりの処理リソースを消費させ、ルーティングプロトコルでフラッピング ンクの状態との同期が失われる原因になる可能性があります。

BFD ダンプニング機能によって、設定可能な指数的遅延メカニズムが導入されます。このメカニズムは、BFD でのリモートノードの到達可能性イベントのフラッピングによる過度な影響を抑えるように設計されています。BFD ダンプニング機能を使用して、ネットワークオペレータは自動的に特定の BFD セッションをダンプニングして BFD クライアントへの過剰な通知を防ぐことができ、このようにして、ネットワークが不必要に不安定にならないようにします。BFD クライアントへの通知のダンプニングにより、モニタリング中のセッションがフラッピングを停止して安定するまで BFD の通知を抑制します。

BFD ダンプニング機能を、特に高速インターフェイスでルーティング クライアントとともに設定すると、ネットワーク全体のコンバージェンス時間と安定性が向上します。BFD ダンプニングは、IPv4/シングルホップ/マルチホップ、Multiprotocol Label Switching-Transport Profile（MPLS-TP）、Pseudo Wire（PW）Virtual Circuit Connection Verification（VCCV）を含む、BFD セッションのすべてのタイプに適用できます。

BFD セッションダンプニング

BFD テンプレート レベル（シングルホップとマルチホップの両方のテンプレート）で BFD ダンプニング機能を設定できます。ダンプニングは、BFD テンプレートを使用するすべてのセッションに適用されます。セッションをダンプニング市内場合は、新しいセッションに対してダンプニングしない新しい BFD テンプレートを使用する必要があります。デフォルトでは、テンプレートのダンプニング機能は有効になっていません。

BFD ハードウェアオフロード

双方向フォワーディング検出（BFD）ハードウェアオフロード機能を使用すると、ASR 9000 拡張イーサネットラインカードのネットワーク処理ユニットに対する非同期 BFD 送信（Tx）と受信（Rx）をオフロードできます。BFD ハードウェアオフロードにより拡張性が改善し、ルーティングテーブル再計算のために迅速な障害検出パケット（メッセージ）をルーティングプロトコルに送信することで、全体的なネットワーク コンバージェンス時間が短縮されます。

次の非同期 BFD セッションは、ASR 9000 拡張イーサネットラインカード上のネットワーク処理ユニットにオフロードされます。

- 物理インターフェイスと VLAN インターフェイスを介した BFD IPv4 セッション。
- 物理インターフェイスと VLAN インターフェイスを介した BFD IPv6 セッション。
- MPLS TP LSP シングルパス（SP） セッションを介した BFD。

BFD ハードウェア オフロード モードは、**hw-module bfd-hw-offload enable** コマンドを使用して ASR 9000 拡張イーサネットラインカードで有効になります。cXR デバイスの管理モードと、eXR デバイスのグローバルコンフィギュレーションモードで **hw-module bfd-hw-offload enable** コマンドを設定します。設定を有効にするには、コマンドを実行した後にラインカードをリロードします。

（注） BFD ハードウェアオフロードモードを有効にした後は、ASR 9000 拡張イーサネットラインカードをリロードする必要があります。

BFD ハードウェアオフロードは、BFD セッションの 7 つのタイマー間隔をサポートしています。サポートされる最小タイマー間隔は 3.3 ミリ秒、最大タイマー間隔は 30 秒です。次の表に、タイマー間隔によって異なる、サポート対象 BFD セッション数の詳細を示します。

BFD セッション	タイマー間隔	ラインカードでサポートされているセッション数	ネットワーク処理ユニットでサポートされているセッション
IPv4、IPv6、MPLS-TP	3.3 ミリ秒	600	300
IPv4、IPv6	15 ミリ秒	2000	1000
IPv4、IPv6	50 ミリ秒	8000	3000
IPv4、IPv6	300 ミリ秒	8000	3000
IPv4、IPv6	1 秒	8000	3000
IPv4、IPv6	2 秒	8000	3000
IPv4、IPv6	30 秒	8000	3000

制約事項

- ハードウェアオフロードされたセッションは、エコーモードをサポートしていません。
- BFD セッションがサポートしているタイマー間隔は 7 つのみです。
- In-Service Software Upgrade（ISSU）は、BFD ハードウェアオフロードされたセッションをサポートしていません。
- バンドルメンバリンクを介してハードウェアオフロードされた BFD は、Cisco モードをサポートしていません。

BFD オブジェクト トラッキング

オブジェクト トラッキングは、リモート IP アドレスの到達可能性を追跡する BFD をサポートするように拡張されました。これにより、BFD は数ミリ秒程度で検出を実行できるため、完全な検出と HSRP スイッチオーバーを 1 秒未満の時間内で実行できます。

BFD の設定方法

BFD 設定時の注意事項

BFD を設定する前に、次の注意事項を考慮してください。

- BFD を使用した FRR/TE、FRR/IP、および FRR/LDP は、POS インターフェイスおよびイーサネット インターフェイスでサポートされます。
- Cisco IOS XR ソフトウェアで BFD ネイバーを確立するには、BFD をダイナミック ルーティング プロトコルの下で、またはスタティック ルートを使用して設定する必要があります。
- BFD セッションの 1 秒当たりのパケット数（pps）での最大レートはラインカードに依存します。BFD をサポートする複数のラインカードがある場合、システム単位の BFD セッションの最大レートは、サポートされるラインカードレートにラインカードの数を掛けた値です。

  BFD スケール値を確認するには、 **show bfd summary** コマンドを使用します。
- バンドル内のメンバの最大数は 64 です。
- BFD を OSPF とともに使用する場合は、次の注意事項を考慮してください。
  - BFD がネイバーから指定ルータ（DR）またはバックアップ DR（BDR）へのセッションを確立するのは、ネイバーの状態がフルである場合だけです。
  - BFD は、DR 以外のネイバー間にセッションを確立しません（たとえば、その OSPF 状態がどちらも双方向である場合）。

注意 特定のインターフェイスで BFD を Unicast Reverse Path Forwarding（uRPF）と併用している場合は、**echo disable** コマンドを使用して、そのインターフェイスでのエコーモードを無効にする必要があります。そうしないと、エコーパケットは拒否されます。詳細については、エコーモードの無効化」の項を参照してください。IPv4 インターフェイスでの IPv4 uRPF のチェックを有効または無効にするには、インターフェイス コンフィギュレーション モードで **[no] ipv4 verify unicast source reachable-via** コマンドを使用します。

（注） **echo disable** コマンドは、論理バンドル（BLB）を介した BFD ではサポートされていません。

ダイナミック ルーティング プロトコルの下での、またはスタティック ルートを使用した BFD の設定

BGP ネイバーでの BFD のイネーブル化

BFDは、ネイバー単位またはインターフェイス単位でイネーブルにすることができます。このタスクでは、隣接ルータで BGP の BFD をイネーブルにする方法について説明します。インターフェイスごとに BFD を有効にするには、「インターフェイスでの OSPF への BFD の有効化」の項にある手順を使用します。

（注） BFD 近接ルータの設定は、BGP でのみサポートされます。

手順の概要

1. **configure**
2. **router bgp** *autonomous-system-number*
3. **bfd minimum-interval** *milliseconds*
4. **bfd multiplier** *multiplier*
5. **neighbor** *ip-address*
6. **remote-as** *autonomous-system-number*
7. **bfd fast-detect**
8. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *autonomous-system-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 120`	BGP コンフィギュレーション モードを開始します。このモードでは、BGP ルーティング プロセスの設定を行えます。 現在のルータの *autonomous-system-number* を取得するには、EXEC モードで **show bgp** コマンドを使用します。
ステップ 3	**bfd minimum-interval** *milliseconds* 例： `RP/0/RSP0/cpu 0: router(config-bgp)# bfd minimum-interval 6500`	BFD の最小間隔を設定します。有効値の範囲は 15 ～ 30000 ミリ秒です。
ステップ 4	**bfd multiplier** *multiplier* 例： `RP/0/RSP0/cpu 0: router(config-bgp)# bfd multiplier 7`	BFD 係数を設定します。
ステップ 5	**neighbor** *ip-address* 例： `RP/0/RSP0/cpu 0: router(config-bgp)# neighbor 172.168.40.24`	BGP ルーティングのためにルータをネイバー コンフィギュレーション モードにして、ネイバーの IP アドレスを BGP ピアとして設定します。 この例では、IP アドレス172.168.40.24 を BGP ピアとして設定しています。
ステップ 6	**remote-as** *autonomous-system-number* 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# remote-as 2002`	ネイバーを作成し、そのネイバーをリモート自律システムに割り当てます。 この例では、設定されるリモート自律システムは 2002 です。
ステップ 7	**bfd fast-detect** 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# bfd fast-detect`	ローカルネットワーキング装置と、ステップ 5 で IP アドレスを BGP ピアとして設定したネイバー間での BFD を有効にします。 ステップ 5 の例では、IP アドレス 172.168.40.24 が BGP ピアとして設定されています。この例では、ローカルネットワーキング装置とネイバー 172.168.40.24 間での BFD が有効になります。
ステップ 8	**commit**	

インターフェイスでの OSPF への BFD の有効化

次に、Open Shortest Path First（OSPF）での BFD を特定のインターフェイスで設定する手順について説明します。この方法の手順は、コマンド モードが異なる点を除き、IS-IS および MPLS-TE での BFD を設定する手順と共通です。

（注） インターフェイス単位での BFD の設定は、OSPF、OSPFv3、IS-IS、MPLS-TE でのみサポートされます。OSPFv3 インターフェイスでの BFD の設定については、特定インターフェイスでの OSPFv3 の BFD の有効化を参照してください。

手順の概要

1. **configure**
2. **bfd multipath include location***node-id*
3. **router ospf** *process-name*
4. **bfd minimum-interval** *milliseconds*
5. **bfd multiplier** *multiplier*
6. **area** *area-id*
7. **interface** *type interface-path-id*
8. **bfd fast-detect**
9. **commit**
10. **show run router ospf**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**bfd multipath include location***node-id* 例： `RP/0/RSP0/cpu 0: router(config)# bfd multipath include location 0/0/CPU0`	（任意）インターフェイス上の指定されたバンドルに BFD マルチパスを有効にします。この手順は、バンドルインターフェイスに必要です。 （注） • この手順は、バンドルインターフェイスにメンバリンクがあるすべてのラインカードに対して繰り返す必要があります。
ステップ 3	**router ospf** *process-name* 例： `RP/0/RSP0/cpu 0: router(config)# router ospf 0`	OSPF コンフィギュレーション モードを開始します。このモードでは、OSPF ルーティング プロセスの設定を行えます。 現在のルータの process-name を取得するには、EXEC コンフィギュレーション モードで **show ospf** コマンドを使用します。

	コマンドまたはアクション	目的
		（注） • IS-IS または MPLS-TE での BFD を設定するには、対応するコンフィギュレーション モードを開始します。たとえば、MPLS-TE の場合は、MPLS-TE コンフィギュレーション モードを開始します。
ステップ 4	**bfd minimum-interval** *milliseconds* 例： `RP/0/RSP0/cpu 0: router(config-ospf)# bfd minimum-interval 6500`	BFD の最小間隔を設定します。有効値の範囲は 15 ～ 30000 ミリ秒です。 この例では、BFD の最小間隔を 6500 ミリ秒に設定しています。
ステップ 5	**bfd multiplier** *multiplier* 例： `RP/0/RSP0/cpu 0: router(config-ospf)# bfd multiplier 7`	BFD 係数を設定します。 この例では、BFD 係数を 7 に設定しています。
ステップ 6	**area** *area-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf)# area 0`	Open Shortest Path First（OSPF）領域を設定します。 *area-id* を OSPF エリア識別子に置き換えます。
ステップ 7	**interface** *type interface-path-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf-ar)# interface gigabitEthernet 0/3/0/1`	インターフェイス コンフィギュレーション モードを開始して、インターフェイス名と *rack/slot/module/port* 表記を指定します。 • この例では、モジュラ サービス カード スロット 3 にあるギガビット イーサネット インターフェイスを示しています。
ステップ 8	**bfd fast-detect** 例： `RP/0/RSP0/cpu 0: router(config-ospf-ar-if)# bfd fast-detect`	隣接する転送エンジン間のパスで障害を検出するために、BFD をイネーブルにします。
ステップ 9	**commit**	
ステップ 10	**show run router ospf** 例： `RP/0/RSP0/cpu 0: router(config-ospf-ar-if)# show run router ospf`	適切なインターフェイスで BFD がイネーブルになっていることを確認します。

特定インターフェイスでの OSPFv3 の BFD の有効化

次に、OSPFv3 での BFD を特定のインターフェイスで設定する手順について説明します。この方法の手順は、コマンド モードが異なる点を除き、IS-IS および MPLS-TE での BFD を設定する手順と共通です。

（注） インターフェイス単位での BFD の設定は、OSPF、OSPFv3、IS-IS、MPLS-TE でのみサポートされます。OSPF インターフェイスでの BFD の設定については、インターフェイスでの OSPF への BFD の有効化を参照してください。

手順の概要

1. **configure**
2. **router ospfv3** *process-name*
3. **bfd minimum-interval** *milliseconds*
4. **bfd multiplier** *multiplier*
5. **area** *area-id*
6. **interface** *type interface-path-id*
7. **bfd fast-detect**
8. **commit**
9. **show run router ospfv3**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router ospfv3** *process-name* 例： `RP/0/RSP0/cpu 0: routerconfig)# router ospfv3 0`	OSPFv3 コンフィギュレーション モードを開始します。このモードでは、OSPFv3 ルーティングプロセスの設定を行えます。 現在のルータの process name を取得するには、EXEC モードで **show ospfv3** コマンドを使用します。 （注） • IS-IS または MPLS-TE での BFD を設定するには、対応するコンフィギュレーション モードを開始します。たとえば、MPLS-TE の場合は、MPLS-TE コンフィギュレーション モードを開始します。
ステップ 3	**bfd minimum-interval** *milliseconds* 例：	BFD の最小間隔を設定します。有効値の範囲は 15 ～ 30000 ミリ秒です。

	コマンドまたはアクション	目的
	`RP/0/RSP0/cpu 0: router(config-ospfv3)# bfd minimum-interval 6500`	この例では、BFD の最小間隔を 6500 ミリ秒に設定しています。
ステップ 4	**bfd multiplier** *multiplier* 例： `RP/0/RSP0/cpu 0: router(config-ospfv3)# bfd multiplier 7`	BFD 係数を設定します。 この例では、BFD 係数を 7 に設定しています。
ステップ 5	**area** *area-id* 例： `RP/0/RSP0/CPU0:router(config-ospfv3)# area 0`	OSPFv3 領域を設定します。 *area-id* は、OSPFv3 エリア識別子に置き換えます。
ステップ 6	**interface** *type interface-path-id* 例： `RP/0/RSP0/cpu 0: router(config-ospfv3-ar)# interface gigabitEthernet 0/1/5/0`	インターフェイス コンフィギュレーション モードを開始して、インターフェイス名と *rack/slot/module/port* 表記を指定します。 • この例では、モジュラ サービス カード スロット 1 にあるギガビット イーサネット インターフェイスを示しています。
ステップ 7	**bfd fast-detect** 例： `RP/0/RSP0/cpu 0: router(config-ospfv3-ar-if)# bfd fast-detect`	隣接する転送エンジン間のパスで障害を検出するために、BFD をイネーブルにします。
ステップ 8	**commit**	
ステップ 9	**show run router ospfv3** 例： `RP/0/RSP0/cpu 0: router(config-ospfv3-ar-if)#show run router ospfv3`	適切なインターフェイスでBFDが有効になっていることを確認します。

スタティック ルートでの BFD のイネーブル化

次に、スタティック ルートでの BFD をイネーブルにする手順について説明します。

（注） バンドル VLAN セッションは、間隔 250 ms、係数 3 の場合のみに制限されます。これよりも強力なパラメータは使用できません。

手順の概要

1. **configure**
2. **router static**
3. **address-family ipv4 unicast** *address nexthop* **bfd fast-detect** [**minimum-interval** *interval*] [**multiplier** *multiplier*]
4. **vrf** *vrf-name*
5. **address-family ipv4 unicast** *address nexthop* **bfd fast-detect**
6. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router static** 例： `RP/0/RSP0/cpu 0: router(config)# router static`	スタティック ルート コンフィギュレーション モードを開始します。このモードでは、スタティック ルーティングの設定を行えます。
ステップ 3	**address-family ipv4 unicast** *address nexthop* **bfd fast-detect** [**minimum-interval** *interval*] [**multiplier** *multiplier*] 例： `RP/0/RSP0/cpu 0: router(config-static)# address-family ipv4 unicast 0.0.0.0/0 2.6.0.1 bfd fast-detect minimum-interval 1000 multiplier 5`	指定の IPV4 ユニキャスト宛先アドレスプレフィックスおよびフォワーディング ネクストホップ アドレスで BFD 高速検出を有効にします。 • ネクストホップが確実に同じ hello 間隔で割り当てられるようにするには、オプションの **minimum-interval** キーワードと引数を含めます。*interval* 引数は、間隔をミリ秒単位で指定する数字に置き換えてください。有効値の範囲は 10 ～ 10,000 です。 • ネクストホップが確実に同じ検出乗数で割り当てられるようにするには、オプションの **multiplier** キーワード引数を含めます。*multiplier* 引数は、検出係数を指定する数字に置き換えてください。有効値の範囲は 1 ～ 10 です。 （注） バンドル VLAN セッションは、間隔 250 ms、係数 3 の場合のみに制限されます。これよりも強力なパラメータは使用できません。
ステップ 4	**vrf** *vrf-name* 例： `RP/0/RSP0/cpu 0: router(config-static)# vrf vrf1`	VPN ルーティングおよび転送（VRF）インスタンスを指定して、その VRF に対するスタティック ルート コンフィギュレーション モードを開始します。

	コマンドまたはアクション	目的
ステップ 5	**address-family ipv4 unicast** *address nexthop* **bfd fast-detect** 例： `RP/0/RSP0/cpu 0: router(config-static-vrf)# address-family ipv4 unicast 0.0.0.0/0 2.6.0.2`	指定の IPV4 ユニキャスト宛先アドレスプレフィックスおよびフォワーディング ネクストホップ アドレスで BFD 高速検出を有効にします。
ステップ 6	**commit**	

IPv6 静的ルートでの BFD の有効化

次に、IPv6 静的ルートでの BFD の有効化を説明する設定例を示します。

```
RP/0/RSP0/cpu 0: router# configure
RP/0/RSP0/cpu 0: router(config)# router static
RP/0/RSP0/cpu 0: router(config-static)# address-family ipv6 unicast 1011:17e4::1/128
ab11:15d2::2 bfd fast-detect minimum-interval 50 multiplier 3
RP/0/RSP0/cpu 0: router(config-static)# commit
```

バンドル メンバ リンクでの BFD の設定

バンドルメンバリンクで BFD を設定するための前提条件

バンドル メンバである物理インターフェイスは、間にスイッチを使用せずにピア ルータ間で直接接続している必要があります。

バンドルの BFD 宛先アドレスの指定

バンドルの BFD 宛先アドレスを指定するには、次の手順を実行します。

手順の詳細

手順の概要

1. **configure**
2. **interface Bundle-Ether** | **Bundle-POS**] *bundle-id*
3. **bfd address-family ipv4 destination***ip-address*
4. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**interface Bundle-Ether**	**Bundle-POS**] *bundle-id* 例：	指定したバンドル ID のインターフェイス コンフィギュレーション モードを開始します。

	コマンドまたはアクション	目的
	`RP/0/RSP0/cpu 0: router(config)# interface Bundle-Ether 1`	
ステップ 3	**bfd address-family ipv4 destination***ip-address* 例： `RP/0/RSP0/cpu 0: router(config-if)# bfd address-family ipv4 destination 10.20.20.1`	接続されたリモート システムでバンドル インターフェイスに割り当てられたプライマリ IPv4 アドレスを指定します。ここで、*ip-address* はドット区切りの 10 進数形式（A.B.C.D）の 32 ビットの IP アドレスです。
ステップ 4	**commit**	

バンドル メンバの BFD セッションのイネーブル化

バンドル メンバー リンクで BFD セッションをイネーブルにするには、次の手順を実行します。

手順の概要

1. **configure**
2. **interface Bundle-Ether** | **Bundle-POS**] *bundle-id*
3. **bfd address-family ipv4 fast-detect**
4. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**interface Bundle-Ether**	**Bundle-POS**] *bundle-id* 例： `RP/0/RSP0/cpu 0: router(config)# interface Bundle-Ether 1`	指定したバンドル ID のインターフェイス コンフィギュレーション モードを開始します。
ステップ 3	**bfd address-family ipv4 fast-detect** 例： `RP/0/RSP0/cpu 0: router(config-if)# bfd address-family ipv4 fast-detect`	バンドル メンバー リンクで IPv4 BFD セッションをイネーブルにします。
ステップ 4	**commit**	

アクティブ バンドルを維持するための最小しきい値の設定

バンドルマネージャは、メンバーリンクの状態に基づいて、バンドルが始動できるまたはアップのまま維持できる、またはダウンしているかどうかを判断するために2つの設定可能な最小しきい値を使用します。

- アクティブ リンクの最小数
- 使用可能な最小アクティブ帯域幅

メンバの状態が変更されるたびに、バンドル マネージャは、アクティブ メンバの数または使用可能な帯域幅が最小値より小さいかどうかを判断します。その場合は、バンドルが DOWN 状態になるか、または DOWN 状態のままになります。アクティブ リンクの数または使用可能な帯域幅がいずれかの最小しきい値に達すると、バンドルは UP 状態に戻ります。

最小バンドルしきい値を設定するには、次の手順を実行します。

手順の概要

1. **configure**
2. **interface Bundle-Ether** *bundle-id*
3. **bundle minimum-active bandwidth** *kbps*
4. **bundle minimum-active links** *links*
5. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**interface Bundle-Ether** *bundle-id* 例： `RP/0/RSP0/cpu 0: router(config)# interface Bundle-Ether 1`	指定したバンドル ID のインターフェイス コンフィギュレーション モードを開始します。
ステップ 3	**bundle minimum-active bandwidth** *kbps* 例： `RP/0/RSP0/cpu 0: router(config-if)# bundle minimum-active bandwidth 580000`	バンドルを始動またはアップのままにできるようにする前に必要な最小帯域幅を設定します。範囲は 1 から、プラットフォームやバンドル タイプによって異なる数値までです。
ステップ 4	**bundle minimum-active links** *links* 例： `RP/0/RSP0/cpu 0: router(config-if)# bundle minimum-active links 2`	バンドルを始動またはアップのままにできる前に必要なアクティブ リンク数を設定します。指定できる範囲は 1 ～ 32 です。

	コマンドまたはアクション	目的
		（注） • すでにアクティブであるバンドルで BFD が開始された場合、そのバンドルの BFD 状態は、既存のすべてのアクティブ メンバの BFD 状態が既知であるときに宣言されます。
ステップ 5	**commit**	

バンドルの BFD パケット送信間隔と障害検出時間の設定

バンドル メンバー リンク上の BFD セッションの BFD 非同期パケット間隔と障害検出時間は、バンドル上の **bfd address-family ipv4 minimum-interval** および **bfd address-family ipv4 multiplier** インターフェイス設定コマンドの組み合わせを使用して設定されます。

BFD 制御パケット間隔は、**bfd address-family ipv4 minimum-interval** コマンドを使用して直接設定されます。BFD エコー パケット間隔およびすべての障害検出時間は、これらのコマンドの間隔および係数の値を組み合わせて決定されます。詳細については、BFD パケット間隔と障害検出を参照してください。

バンドル メンバ リンクで BFD 非同期モード制御およびエコー パケットの最小送信間隔と障害検出時間を設定するには、次の手順を実行します。

手順の詳細

手順の概要

1. **configure**
2. **interface Bundle-Ether** | **Bundle-POS**] *bundle-id*
3. **bfd address-family ipv4 minimum-interval** *milliseconds*
4. **bfd address-family ipv4 multiplier** *multiplier*
5. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**interface Bundle-Ether**	**Bundle-POS**] *bundle-id* 例： `RP/0/RSP0/cpu 0: router(config)# interface Bundle-Ether 1`	指定したバンドル ID のインターフェイス コンフィギュレーション モードを開始します。
ステップ 3	**bfd address-family ipv4 minimum-interval** *milliseconds* 例：	

	コマンドまたはアクション	目的
	`RP/0/RSP0/cpu 0: router(config-if)#bfd address-family ipv4 minimum-interval 2000` （注） • バンドル メンバー リンクで IPv4 BFD セッションの非同期モード制御パケットの最小間隔（ミリ秒単位）を指定します。範囲は 15 ～ 30000 です。このコマンドでは 15 ミリの最小値を設定できますが、Cisco ASR 9000 シリーズ ルータでサポートされる最小値は 50 ミリ秒です。	
ステップ 4	**bfd address-family ipv4 multiplier** *multiplier* 例： `RP/0/RSP0/cpu 0: router(config-if)#bfd address-family ipv4 multiplier 30`	バンドル メンバ リンクの IPv4 BFD セッションの BFD 制御、エコー パケット障害検出時間およびエコーパケットの送信間隔を決定するために、最小間隔とともに係数として使用する値を指定します。範囲は 2 ～ 50 です。デフォルトは 3 です。 （注） • このコマンドでは2の最小値を設定できますが、サポートされる最小値は3です。
ステップ 5	**commit**	

バンドルのタイマーを使用した BFD 状態変更通知の許容可能な遅延の設定

リンク バンドル メンバの BFD セッションのダウンを宣言する前に、ピアからの BFD SCN の受信の遅延を許可するために次の 2 つの設定可能なタイマーが BFD システムによってサポートされています。

- BFD セッションの開始
- ネイバーによる BFD 設定の削除

これらのタイマーの動作方法やその他の BFD 状態変更動作の詳細については、メンバ リンクおよびバンドル ステータスでの BFD 状態変更動作の概要を参照してください。

ピアからの BFD SCN の受信の遅延を許可するタイマーを設定するには、次の手順を実行します。

手順の概要

1. **configure**
2. **interface Bundle-Ether** | **Bundle-POS**] *bundle-id*
3. **bfd address-family ipv4 timers start** *seconds*
4. **bfd address-family ipv4 timers nbr-unconfig** *seconds*

5. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**interface Bundle-Ether** | **Bundle-POS**] *bundle-id* 例： `RP/0/RSP0/cpu 0: router(config)# interface Bundle-Ether 1`	指定したバンドル ID のインターフェイス コンフィギュレーション モードを開始します。
ステップ 3	**bfd address-family ipv4 timers start** *seconds* 例： `RP/0/RSP0/cpu 0: router(config-if)#`	セッションのアップを宣言できるように、BFD メンバ リンク セッションの開始後、BFD ピアからの予測される通知が受信されるのを待つ秒数を指定します。その期間の後に SCN が受信されない場合は、BFD セッションのダウンが宣言されます。範囲は 60 ～ 3600 です。（Cisco IOS XR Release 4.0 および 4.0.1 では、使用可能な最小値は 30 ですが、これはお勧めできません）。
ステップ 4	**bfd address-family ipv4 timers nbr-unconfig** *seconds* 例： `RP/0/RSP0/cpu 0: router(config-if)#`	BFD ピア間の設定の不一致を解決できるように、BFD 設定が BFD ネイバーによって削除されたことの通知の受信後に待機する秒数を指定します。指定されたタイマーに達する前に BFD 設定の問題が解決されない場合、BFD セッションのダウンが宣言されます。範囲は 30 ～ 3600 です。
ステップ 5	**commit**	

バンドル単位の バンドル CISCO/IETF モードを介した BFD のサポートの設定

バンドル CISCO/IETF モードを介した BFD のサポートをバンドル単位で設定するには、次の手順を実行します。

始める前に

BFD モードの変更（Cisco から IETF およびその逆）は、バンドルが新たに作成されるか、またはバンドルの BFD 状態が「ダウン（down）」または「BoB 動作不能（BoB nonoperational）」の場合にのみ実行されます。

（注） この手順は、リリース 5.3.1 以降に適用されます。

手順の概要

1. **configure**
2. **interface Bundle-Ether** *bundle-id*
3. **no bfd address-family ipv4 fast-detect**
4. **commit**
5. **bfd mode { cisco | ietf }**
6. **bfd address-family ipv4 fast-detect**
7. **commit**
8. **show bundle bundle-ether** *bundle-id*

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**interface Bundle-Ether** *bundle-id* 例： `RP/0/RSP0/cpu 0: router(config)# interface Bundle-Ether 1`	指定したバンドル ID のインターフェイス コンフィギュレーション モードを開始します。
ステップ 3	**no bfd address-family ipv4 fast-detect** 例： `RP/0/RSP0/cpu 0: router(config-if)# no bfd address-family ipv4 fast-detect`	指定したバンドルで IPv4 BFD セッションを無効にします。
ステップ 4	**commit**	
ステップ 5	**bfd mode { cisco | ietf }** 例： `RP/0/RSP0/cpu 0: router(config-if)# bfd mode ietf`	指定したバンドルのバンドルを介した BFD に Cisco モードまたは IETF モードを有効にします。デフォルトは **cisco** です。
ステップ 6	**bfd address-family ipv4 fast-detect** 例： `RP/0/RSP0/cpu 0: router(config-if)# bfd address-family ipv4 fast-detect`	指定したバンドルで IPv4 BFD セッションを有効にします。
ステップ 7	**commit**	
ステップ 8	**show bundle bundle-ether** *bundle-id*	選択されたバンドル モードが表示されます。

モードを確認するための show コマンドの出力例

次に、バンドルモードが選択された **show bundle bundle-ether** コマンドの出力例を示します。

```
RP/0/RP0/CPU0:R3-PE3#sh bundle bundle-ether 4301

Bundle-Ether4301
  Status:                                  Up
  Local links {active/standby/configured}:   2 / 0 / 2
  Local bandwidth {effective/available}:     20000000 (20000000) kbps
  MAC address (source):                    0014.1c00.0003 (Chassis pool)
  Inter-chassis link:                      No
  Minimum active links / bandwidth:        1 / 1 kbps
  Maximum active links:                    64
  Wait while timer:                        2000 ms
  Load balancing:                          Default
  LACP:                                    Operational
    Flap suppression timer:                Off
    Cisco extensions:                      Disabled
  mLACP:                                   Not configured
  IPv4 BFD:                                Operational
    State:                                 Up
    Mode:                                            ietf  ####------ this is the
mode cisco/ietf .
    Fast detect:                           Enabled
    Start timer:                           60 s
    Neighbor-unconfigured timer:           60 s
    Preferred min interval:                150 ms
    Preferred multiple:                    3
    Destination address:                   101.43.1.1

  Port                  Device           State        Port ID         B/W, kbps
  --------------------  ---------------  -----------  --------------  ----------
  Te0/5/0/4             Local            Active       0x8000, 0x0012    10000000
      Link is Active
  Te0/7/0/8             Local            Active       0x8000, 0x0006    10000000
      Link is Active
```

次のタスク

バンドルが新しい BFD モードの変更を受け入れるようにするには、既存の BFD セッションをダウンさせてから、再作成する必要があります。

BFD ピアへの転送パスをテストするためのエコーモードの有効化

BFD エコーモードは、次のインターフェイスではデフォルトで有効になっています。

- BFD バンドル インターフェイスのメンバ リンク上の IPv4 の場合。
- 最小間隔が 2 秒未満である他の物理インターフェイス上の IPv4 の場合。

（注） **bfd minimum-interval** コマンドを使用して物理インターフェイスで2秒より長いBFD最小間隔を設定した場合、エコーモードをサポートして有効にするには、この間隔を 2 秒未満に変更する必要があります。これは、エコーモードを常にサポートするバンドルメンバ リンクには適用されません。

デフォルトのエコー パケット送信元アドレスの上書き

エコー パケット送信元アドレスを指定しないと、BFD はエコー パケットのデフォルト送信元アドレスとして出力インターフェイスの IP アドレスを使用します。

3.9.0 よりも前の Cisco IOS XR リリースでは、エコーパケット送信元アドレスのデフォルト IP アドレスをルータ ID として指定されたアドレスに変更するために、**router-id** コマンドを使用してローカルルータ ID を設定することを推奨します。

Cisco IOS XR Release 3.9.0 以降では、BFD で **echo ipv4 source** コマンドを使用するか、インターフェイス BFD コンフィギュレーション モードを使用して、エコーパケット送信元アドレスとして使用する IP アドレスを指定できます。

ルータ全体で、または特定のインターフェイスについて、BFD のエコー パケットのデフォルト IP 送信元アドレスを上書きできます。

BFD に対するグローバルでのエコー パケット送信元アドレスの指定

ルータの BFD に対してグローバルにエコー パケット送信元 IP アドレスを指定するには、次の手順を実行します。

手順の概要

1. **configure**
2. **bfd**
3. **echo ipv4 source** *ip-address*
4. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**bfd** 例： `RP/0/RSP0/cpu 0: router(config)# bfd`	BFD コンフィギュレーション モードを開始します。

	コマンドまたはアクション	目的
ステップ 3	**echo ipv4 source** *ip-address* 例： `RP/0/RSP0/cpu 0: router(config-bfd)# echo ipv4 source 10.10.10.1`	BFDエコーパケットで送信元アドレスとして使用されるIPv4アドレスを指定します。ここで、*ip-address* はドット区切りの10進数形式（A.B.C.D）の32ビットの IP アドレスです。
ステップ 4	**commit**	

個々のインターフェイスまたはバンドルのエコー パケット送信元アドレスの指定

個々の BFD インターフェイスまたはバンドルのエコー パケット送信元 IP アドレスを指定するには、次の手順を実行します。

手順の概要

1. **configure**
2. **bfd**
3. **interface** type interface-path-id
4. **echo ipv4 source** *ip-address*
5. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**bfd** 例： `RP/0/RSP0/cpu 0: router(config)# bfd`	BFD コンフィギュレーション モードを開始します。
ステップ 3	**interface** type interface-path-id 例： `RP/0/RSP0/cpu 0: router(config-bfd)# interface gigabitEthernet 0/1/5/0`	特定のインターフェイスまたはバンドルに対して BFDインターフェイス コンフィギュレーション モードを開始します。BFD インターフェイス コンフィギュレーション モードでは、個別のインターフェイスまたはバンドルでIPv4アドレスを指定できます。
ステップ 4	**echo ipv4 source** *ip-address* 例： `RP/0/RSP0/cpu 0: router(config-bfd)# echo ipv4 source 10.10.10.1`	BFDエコーパケットで送信元アドレスとして使用されるIPv4アドレスを指定します。ここで、*ip-address* はドット区切りの10進数形式（A.B.C.D）の32ビットの IP アドレスです。
ステップ 5	**commit**	

エコー遅延検出に基づいた BFD セッション ティアダウンの設定

Cisco IOS XR 4.0.1 以降では、設定されたエコー遅延許容値を超えた BFD セッションを停止するように非バンドル インターフェイスの BFD セッションを設定できます。

エコー遅延検出を使用して BFD セッション ティアダウンを設定するには、次の手順を実行します。

エコー遅延検出を有効にする前に、BFD設定でエコーモードがサポートされていることを確認してください。

エコー遅延検出はバンドル インターフェイスではサポートされません。

手順の詳細

手順の概要

1. **configure**
2. **bfd**
3. **echo latency detect** [**percentage** *percent-value* [**count** *packet-count*]
4. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**bfd** 例： `RP/0/RSP0/cpu 0: router(config)# bfd`	BFD コンフィギュレーション モードを開始します。
ステップ 3	**echo latency detect** [**percentage** *percent-value* [**count** *packet-count*] 例： `RP/0/RSP0/cpu 0: router(config-bfd)# echo latency detect`	BFDセッション中のエコーパケットの遅延検出を有効にします。ここで、各値は次のとおりです。 • **percentage** *percent-value*：不正遅延として検出するエコー障害検出時間のパーセンテージを指定します。範囲は 100 ～ 250 です。デフォルトは 100 です。 • **count** *packet-count*：BFD セッションをダウンさせる、不正遅延で受信される連続したパケットの数を指定します。指定できる範囲は 1 ～ 10 です。デフォルトは 1 です。
ステップ 4	**commit**	

エコー パスと遅延の検証までの BFD セッション開始の遅延

Cisco IOS XR Release 4.0.1 以降、非バンドルインターフェイスの BFD セッションを開始する前に、エコーパケットパスが動作していて、設定された遅延しきい値内であることを確認できます。

（注） エコー起動検証は、バンドル インターフェイスではサポートされません。

BFD のエコー起動検証を設定するには、次の手順を実行します。

始める前に

エコー起動検証を有効にする前に、BFD設定でエコーモードがサポートされていることを確認してください。

手順の概要

1. **configure**
2. **bfd**
3. **echo startup validate** [**force**]
4. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**bfd** 例： `RP/0/0RP0RSP0/CPU0:router(config)# bfd`	BFD コンフィギュレーション モードを開始します。
ステップ 3	**echo startup validate** [**force**] 例： `RP/0/0RP0RSP0/CPU0:router(config-bfd)# echo startup validate`	BFD セッションを開始する前に、エコー パケット パスの検証を有効にします。その場合、BFD セッションの状態変更を可能にする前に設定済みの遅延内に送信が成功するかどうかを確認するために、リンク上でエコーパケットが定期的に送信されます。 **force** キーワードが設定されていない場合、ローカル システムは、次の条件が満たされた場合にエコー起動検証を実行します。 • ローカル ルータは、エコーを実行できます（このセッションでは、エコーが有効になっています）。

	コマンドまたはアクション	目的
		• リモートルータは、エコーを実行できます（リモートシステムから受信された制御パケットには、ゼロ以外の「Required Min Echo RX Interval」値が含まれています）。 force キーワードが設定されている場合、ローカルシステムは、次の条件を満たす場合にエコー起動検証を実行します。 • ローカルルータは、エコーを実行できます（このセッションでは、エコーが有効になっています）。 • リモートルータのエコー機能は考慮されません（リモートシステムから受信された制御パケットには、ゼロまたはゼロ以外の「Required Min Echo RX Interval」値が含まれています）。
ステップ 4	**commit**	

エコーモードの無効化

BFD は、特定の環境ではエコー モードでの非同期動作をサポートしません。BFD を次のアプリケーションまたは条件で使用する場合、エコーモードを無効にする必要があります。

- uRPF での BFD (IPv4)
- BFD バンドル インターフェイスに複数のラックにまたがるメンバ リンクがある場合の、ラック リロードおよび活性挿抜（OIR）をサポートするため。

（注） 最小間隔が 2 秒より長い場合、BFD エコーモードは、物理インターフェイスの BFD に対して自動的に無効になります。最小間隔は、BFD バンドル メンバ リンクのエコー モードには影響を与えません。BFD エコーモードはまた、バンドル VLAN および IPv6 の BFD に対しても自動的に無効になります（グローバルおよびリンクローカル アドレッシング）。

ルータ全体で、または特定のインターフェイスについて、BFDのエコーモードを無効にすることができます。

ルータでのエコーモードの無効化

ルータでエコーモードをグローバルに無効にするには、次の手順を実行します。

手順の詳細

手順の概要

1. **configure**
2. **bfd**
3. **echo disable**
4. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**bfd** 例： `RP/0/RSP0/cpu 0: router(config)# bfd`	BFD コンフィギュレーション モードを開始します。
ステップ 3	**echo disable** 例： `RP/0/RSP0/cpu 0: router(config-bfd)# echo disable`	ルータでエコーモードを無効にします。
ステップ 4	**commit**	

個々のインターフェイスまたはバンドルでのエコーモードの無効化

次に、インターフェイスまたはバンドルでエコーモードを無効にする手順について説明します。

手順の概要

1. **configure**
2. **bfd**
3. **interface** *type interface-path-id*
4. **echo disable**
5. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**bfd** 例：	BFD コンフィギュレーション モードを開始します。

	コマンドまたはアクション	目的
	`RP/0/RSP0/cpu 0: router(config)# bfd`	
ステップ 3	**interface** *type interface-path-id* 例： `RP/0/RSP0/cpu 0: router(config-bfd)# interface gigabitEthernet 0/1/5/0`	特定のインターフェイスまたはバンドルに対して BFDインターフェイス コンフィギュレーション モードを開始します。BFD インターフェイス コンフィギュレーション モードでは、個別のインターフェイスまたはバンドルでエコーモードを無効にすることができます。
ステップ 4	**echo disable** 例： `RP/0/RSP0/cpu 0: router(config-bfd-if)# echo disable`	指定された個別のインターフェイスまたはバンドルでエコーモードを無効にします。
ステップ 5	**commit**	

BFD ダンプニングを使用した BFD セッション フラッピングの最小化

BFD セッション フラッピングを制御するために BFD ダンプニングを設定するには、次の手順を実行します。

手順の概要

1. **configure**
2. **bfd**
3. **dampening** [**bundle-member**] {**initial-wait** | **maximum-wait** | **secondary-wait**} *milliseconds*
4. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**bfd** 例： `RP/0/RSP0/cpu 0: router(config)# bfd`	BFD コンフィギュレーション モードを開始します。
ステップ 3	**dampening** [**bundle-member**] {**initial-wait** | **maximum-wait** | **secondary-wait**} *milliseconds* 例： `RP/0/RSP0/cpu 0: router(config-bfd)# dampening initial-wait 30000`	フラッピングを制御するための BFDセッション開始の遅延（ミリ秒単位）を指定します。 **maximum-wait** の値は、**initial-wait** の値よりも大きい値にする必要があります。

	コマンドまたはアクション	目的
		ダンプニング値は、バンドル メンバ インターフェイスおよび非バンドルインターフェイスに対して定義できます。
ステップ 4	**commit**	

IPv6 チェックサムサポートの有効化および無効化

デフォルトでは、UDP パケットの IPv6 チェックサム計算はルータの BFD に対して有効になっています。

ルータ全体で、または特定のインターフェイスのどちらかの BFD の IPv6 チェックサムサポートを無効にすることができます。IPv6 チェックサムサポートが一方のルータでは有効になっているが、もう一方のルータでは無効になっている場合は、不良構成が発生する可能性があります。そのため、両方のルータで IPv6 チェックサムサポートを有効または無効にする必要があります。

ここでは、次のことについて説明します。

（注） コマンドラインインターフェイス（CLI）は、BFD設定とBFDインターフェイス コンフィギュレーションでは若干異なります。BFD 設定の場合、**disable** キーワードはオプションではありません。そのため、そのモードで BFD 設定を有効にするには、コマンドの **no** 形式を使用する必要があります。

ルータでの BFD の IPv6 チェックサム計算の有効化および無効化

ルータで IPv6 チェックサム計算をグローバルに有効または無効にするには、次の手順を実行します。

手順の概要

1. **configure**
2. **bfd**
3. **ipv6 checksum** [**disable**]
4. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**bfd** 例：	BFD コンフィギュレーション モードを開始します。

	コマンドまたはアクション	目的
	`RP/0/RSP0/cpu 0: router(config)# bfd`	
ステップ 3	**ipv6 checksum** [**disable**] 例： `RP/0/RSP0/cpu 0: router(config-bfd-if)# ipv6 checksum disable`	インターフェイスでのIPv6チェックサムサポートを有効にします。無効にするには、**disable** キーワードを使用します。
ステップ 4	**commit**	

個々のインターフェイスまたはバンドルの BFD の IPv6 チェックサム計算の有効化と無効化

次に、インターフェイスまたはバンドルで IPv6 チェックサム計算を有効または無効にする手順について説明します。

手順の詳細

手順の概要

1. **configure**
2. **bfd**
3. **interface** *type interface-path-id*
4. **ipv6 checksum** [**disable**]
5. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**bfd** 例： `RP/0/RSP0/cpu 0: router(config)# bfd`	BFD コンフィギュレーション モードを開始します。
ステップ 3	**interface** *type interface-path-id* 例： `RP/0/RSP0/cpu 0: router(config-bfd)# interface gigabitEthernet 0/1/5/0`	特定のインターフェイスまたはバンドルに対して BFDインターフェイス コンフィギュレーション モードを開始します。

	コマンドまたはアクション	目的
ステップ 4	**ipv6 checksum** [**disable**] 例： `RP/0/RSP0/cpu 0: router(config-bfd-if)# ipv6 checksum`	インターフェイスでのIPv6チェックサムサポートを有効にします。無効にするには、**disable**キーワードを使用します。
ステップ 5	**commit**	

BFD カウンタのクリアおよび無効化

次に、BFD パケット カウンタの表示およびクリアの手順について説明します。特定ノードまたは特定インターフェイスでホストされている BFD セッションのパケット カウンタをクリアすることができます。

手順の概要

1. **show bfd counters**[**ipv4 | all**] **packet interface** *type interface-path-id*] **location** *node-id*
2. **clear bfd counters** [**ipv4 | ipv6 |all**] **packet** [**interface** *type interface-path-id*] **location** *node-id*
3. **show bfd counters** [**[ipv4 | ipv6 | all**] **packet** [**interface** *type interface-path-id*] **location** *node-id*

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**show bfd counters**[**ipv4	all**] **packet interface** *type interface-path-id*] **location** *node-id* 例： `RP/0/RSP0/cpu 0: router#show bfd counters all packet location 0/3/cpu0`	IPv4 パケット、IPv6 パケット、またはすべてのパケットの BFD カウンタを表示します。
ステップ 2	**clear bfd counters** [**ipv4	ipv6	all**] **packet** [**interface** *type interface-path-id*] **location** *node-id* 例： `RP/0/RSP0/cpu 0: router# clear bfd counters all packet location 0/3/cpu0`	IPv4 パケット、IPv6 パケット、またはすべてのパケットの BFD カウンタをクリアします。
ステップ 3	**show bfd counters** [**[ipv4	ipv6	all**] **packet** [**interface** *type interface-path-id*] **location** *node-id* 例： `RP/0/RSP0/cpu 0: router# show bfd counters all packet location 0/3/cpu0`	IPv4 パケット、IPv6 パケット、またはすべてのパケットのBFDカウンタがクリアされていることを確認します。

バンドル上のBFD（BoB）と論理バンドル上のBFD（BLB）の間の共存設定

BoB と BLB の間の共存メカニズムを設定するには、次のタスクを実行します。

始める前に

MP BFD セッションのホストを許可するには、1 つ以上のラインカードを設定する必要があります。ラインカードが搭載されていない場合、ラインカードグループは形成されず、その結果、BFD MP セッションは作成されません。グループサイズと番号のデフォルト設定では、2 行以上の **bfd multiple-paths include location** *node-id* コマンドと有効なラインカードを、グループの形成と BFD MP セッションの確立を開始するためのアルゴリズムの設定に追加する必要があります。

次に、設定例を示します。

```
(config)#bfd multipath include location 0/0/CPU0
(config)#bfd multipath include location 0/1/CPU0
```

手順の概要

1. **configure**
2. **bfd**
3. 次のいずれかのコマンドを使用します。
   - **bundle coexistence bob-blb inherit**
   - **bundle coexistence bob-blb logical**
4. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**bfd** 例： `RP/0/RSP0/cpu 0: router(config)#bfd`	双方向フォワーディング検出（BFD）を設定し、グローバル BFD コンフィギュレーション モードを開始します。

ステップ 3	次のいずれかのコマンドを使用します。
	• **bundle coexistence bob-blb inherit**
	• **bundle coexistence bob-blb logical**
	例：
	`RP/0/RSP0/cpu 0: router(config-bfd)#bundle coexistence bob-blb inherit`
	または
	`RP/0/RSP0/cpu 0: router(config-bfd)#bundle coexistence bob-blb logical`
		BoB と BLB の間の共存メカニズムを設定します。
		• **inherit**：「継承された」共存モードを設定するには、**inherit** キーワードを使用します。
		• **logical**：「論理的な」共存モードを設定するには、**logical** キーワードを使用します。

	コマンドまたはアクション	目的
ステップ 4	**commit**	

BFD IPv6 マルチホップの設定

eBGP ネイバーの BFD IPv6 マルチホップの設定

eBGP ネイバーの BFD IPv6 マルチホップを設定するには、次のタスクを実行します。

手順の概要

1. **configure**
2. **bfd multipath include location** *node-id*
3. **router bgp** *as-number*
4. **neighbor** *ip-address* **ebgp-multihop** *ttl-value*
5. **neighbor** *ip-address* **bfd fast-detect**
6. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**bfd multipath include location** *node-id* 例： `RP/0/RSP0/cpu 0: router(config)#bfd multipath include location 0/7/CPU0`	BFDマルチホップセッションをホストするための指定されたラインカードを含めます。
ステップ 3	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 65001`	BGP コンフィギュレーション モードを開始します。
ステップ 4	**neighbor** *ip-address* **ebgp-multihop** *ttl-value* 例： `RP/0/RSP0/cpu 0: router(config-bgp)#neighbor 21:1:1:1:1:1:1:2 ebgp-multihop 255`	外部 BGP（eBGP）ネイバーとのマルチホップ ピアリングを有効にします。
ステップ 5	**neighbor** *ip-address* **bfd fast-detect** 例： `RP/0/RSP0/cpu 0: router(config-bgp)#neighbor 21:1:1:1:1:1:1:2 bfd fast-detect`	eBGP ネイバーの IP アドレスを指定し、BFD の迅速な検出を有効にします。
ステップ 6	**commit**	

iBGP ネイバーの BFD IPv6 マルチホップの設定

iBGP ネイバーの BFD IPv6 マルチホップを設定するには、次のタスクを実行します。

手順の概要

1. **configure**
2. **bfd multipath include location** *node-id*
3. **router bgp** *as-number*
4. **neighbor** *ip-address* **bfd fast-detect**
5. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**bfd multipath include location** *node-id* 例： `RP/0/RSP0/cpu 0: router(config)#bfd multipath include location 0/7/CPU0`	BFD マルチホップ セッションをホストするための指定されたラインカードを含めます。
ステップ 3	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)#router bgp 65001`	BGP コンフィギュレーション モードを開始します。
ステップ 4	**neighbor** *ip-address* **bfd fast-detect** 例： `RP/0/RSP0/cpu 0: router(config-bgp)#neighbor 21:1:1:1:1:1:1:2`	iBGP ネイバーの IP アドレスを指定し、BFD の迅速な検出を有効にします。
ステップ 5	**commit**	

MPLS トラフィック エンジニアリング LSP を介した BFD の設定

TE トンネルを介した BFD に対する BFD パラメータの有効化

TE トンネルの BFD は、トンネルで BFD パラメータを設定することにより、ヘッドエンドで有効になります。すでに起動しているトンネルで BFD が有効になっている場合、TE は、トンネルをダウンさせる前に、起動タイムアウトを待機します。デフォルトでは、TE トンネル上で BFD は無効になっています。BFD パラメータを設定し、TE トンネルを介した BFD を有効にするには、次のタスクを実行します。

（注） BFD は、CPU 使用率の変動を避けるために LSP ping メッセージを 50 PPS 未満に制限することで、BFD セッションの作成ペースを調整します。

手順の概要

1. **configure**
2. **interface tunnel-te** *interface-number*
3. **bfd fast-detect**
4. **bfd minimum-interval***milliseconds*
5. **bfd multiplier** *number*
6. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**interface tunnel-te** *interface-number* 例： `RP/0/RSP0/cpu 0: router(config)#interface tunnel-te 65535`	MPLS トラフィック エンジニアリング（MPLS TE）トンネルインターフェイスを設定し、MPLS TE トンネルインターフェイス コンフィギュレーション モードを開始します。
ステップ 3	**bfd fast-detect** 例： `RP/0/RSP0/cpu 0: router(config-if)#bfd fast-detect`	BFD 高速検出を有効にします。
ステップ 4	**bfd minimum-interval***milliseconds* 例： `RP/0/RSP0/cpu 0: router(config-if)#bfd minimum-interval 2000`	hello 間隔をミリ秒単位で設定します。 hello 間隔の範囲は 100 ～ 30000 ミリ秒です。デフォルトの hello 間隔は 100 ミリ秒です。
ステップ 5	**bfd multiplier** *number* 例： `RP/0/RSP0/cpu 0: router(config-if)#bfd multiplier 5`	BFD 乗数検出を設定します。 BFD 乗数の範囲は 3 ～ 10 です。デフォルトの BFD 乗数は 3 です。
ステップ 6	**commit**	

次のタスク

BFD の起動タイムアウト間隔を設定します。

LSP がシグナリングされ、BFD セッションが作成されると、TE は BFD セッションが起動するまでの時間を指定できるようになります。BFD セッションがタイムアウト内に起動しない場合、LSP は切断されます。そのため、BFD 起動タイムアウトを設定する必要があります

BFD 起動タイムアウトの設定

BFD起動タイムアウト間隔を設定するには、次の手順を実行します。デフォルトの起動タイムアウト間隔は 60 秒です。

始める前に

BFD は、MPLS TE トンネルインターフェイスで有効にする必要があります。

手順の概要

1. **configure**
2. **interface tunnel-te** *interface-number*
3. **bfd bringup-timeout** *seconds*
4. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**interface tunnel-te** *interface-number* 例： `RP/0/RSP0/cpu 0: router(config)#interface tunnel-te 65535`	MPLS トラフィック エンジニアリング（MPLS TE）トンネルインターフェイスを設定し、MPLS TE トンネルインターフェイス コンフィギュレーション モードを開始します。
ステップ 3	**bfd bringup-timeout** *seconds* 例： `RP/0/RSP0/cpu 0: router(config-if)#bfd bringup-timeout 2400`	BFD セッションが起動するのを待機する時間間隔（秒単位）を有効にします。 起動タイムアウトの範囲は 6 ～ 3600 秒です。デフォルトの起動タイムアウト間隔は 60 秒です。
ステップ 4	**commit**	

次のタスク

BFD ダンプニングパラメータを設定して、TE トンネルを起動し、ネットワーク内のシグナリングの変化を回避します。

TE トンネルの BFD ダンプニングの設定

BFD セッションの起動に失敗すると、TE は失敗したパスオプションを使用してすぐにオフに戻り、ネットワーク内でのシグナリングの変化を回避します。

ダンプニング間隔を設定して TE トンネルを起動するには、次の手順を実行します。

始める前に

- BFD は、MPLS TE トンネルインターフェイスで有効にする必要があります。

- BFD 起動タイムアウト間隔は、**bfd bringup-timeout** コマンドを使用して設定する必要があります。

手順の概要

1. **configure**
2. **interface tunnel-te** *interface-number*
3. **bfd dampening initial-wait** *milliseconds*
4. **bfd dampening maximum-wait** *milliseconds*
5. **bfd dampening secondary-wait** *milliseconds*
6. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**interface tunnel-te** *interface-number* 例： `RP/0/RSP0/cpu 0: router(config)#interface tunnel-te 65535`	MPLS トラフィック エンジニアリング（MPLS TE）トンネルインターフェイスを設定し、MPLS TE トンネルインターフェイス コンフィギュレーション モードを開始します。
ステップ 3	**bfd dampening initial-wait** *milliseconds* 例： `RP/0/RSP0/cpu 0: router(config-if)#bfd dampening initial-wait 360000`	トンネルが起動するまでの初期遅延間隔を設定します。 初期起動待機遅延間隔の範囲は 1 ～ 518400000 ミリ秒です。デフォルトの初期待機間隔は 16000 ミリ秒です。 （注） このオプションを使用すると、以前にシグナリングされた帯域幅を持つ TE トンネルが起動します。
ステップ 4	**bfd dampening maximum-wait** *milliseconds* 例： `RP/0/RSP0/cpu 0: router(config-if)#bfd dampening maximum-wait 700000`	トンネルが起動するまでの最大遅延間隔を設定します。 最大起動待機遅延の時間間隔の範囲は 1 ～ 518400000 ミリ秒です。デフォルトの初期待機間隔は 600000 ミリ秒です。 （注） このオプションを使用すると、設定された帯域幅を持つ TE トンネルが起動します。
ステップ 5	**bfd dampening secondary-wait** *milliseconds* 例：	トンネルを起動するまでのセカンダリ遅延間隔を設定します。

	コマンドまたはアクション	目的
	`RP/0/RSP0/cpu 0: router(config-if)#bfd dampening secondary-wait 30000`	セカンダリ起動待機遅延時間間隔の範囲は 1 ～ 518400000 ミリ秒です。デフォルトのセカンダリ待機間隔は 20000 ミリ秒です。
ステップ 6	**commit**	

次のタスク

定期的な LSP ping オプションを設定します。

定期的な LSP ping 要求の設定

BFD セッションが起動した後、BFD TLV で定期的な LSP ping 要求を送信するように設定するには、次のタスクを実行します。

始める前に

BFD は、MPLS TE トンネルインターフェイスで有効にする必要があります。

手順の概要

1. **configure**
2. **interface tunnel-te** *interface-number*
3. 次のいずれかのコマンドを使用します。
   - **bfd lsp-ping interval** *300*
4. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**interface tunnel-te** *interface-number* 例： `RP/0/RSP0/cpu 0: router(config)#interface tunnel-te 65535`	MPLS トラフィック エンジニアリング（MPLS TE）トンネルインターフェイスを設定し、MPLS TE トンネルインターフェイス コンフィギュレーション モードを開始します。
ステップ 3	次のいずれかのコマンドを使用します。 • **bfd lsp-ping interval** *300* 例： `RP/0/RSP0/cpu 0: router(config-if)#bfd lsp-ping interval 300` または	LSP ping 要求の定期的な間隔を設定するか、または LSP ping 要求を無効にします。 • **interval** *seconds*：定期的な LSP ping 要求の間隔を秒単位で設定します。間隔の範囲は 60 ～ 3600 秒です。デフォルトの間隔は 120 秒です。 • **disable**: 定期的な LSP ping 要求を無効にします。

	コマンドまたはアクション	目的
	`RP/0/RSP0/cpu 0: router(config-if)#bfd lsp-ping disable`	定期的な LSP ping 要求は、デフォルトで有効になっています。ping 要求のデフォルトの間隔は 120 秒です。BFD は LSP ping を 1 秒あたり 50 ping（PPS）に調整します。そのため、ping 間隔は受け入れられます。ただし、60 ～ 3600 秒の間隔を設定しない限り、これは保証されません。
ステップ 4	**commit**	

次のタスク

テールエンドで BFD を設定します。

テールエンドでの BFD の設定

LSP セッションを介したすべての BFD に BFD 最小間隔と BFD 乗数パラメータを設定するには、テール エンド グローバルコンフィギュレーション コマンドを使用します。範囲とデフォルト値は、BFD ヘッドエンド設定値と同じです。BFD は、ヘッドエンドの最小間隔とテールエンドの最小間隔の間で設定された最大値を使用します。

テールエンドで BFD を設定するには、次のタスクを実行します。

手順の概要

1. **configure**
2. **mpls traffic-eng bfd lsp tailminimum-interval** *milliseconds*
3. **mpls traffic-eng bfd lsp tailmultiplier** *number*
4. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**mpls traffic-eng bfd lsp tailminimum-interval** *milliseconds* 例： `RP/0/RSP0/cpu 0: router(config)#mpls traffic-eng bfd lsp tail minimum-interval 20000`	hello 間隔をミリ秒単位で設定します。 hello 間隔の範囲は 100 ～ 30000 ミリ秒です。デフォルトの hello 間隔は 100 ミリ秒です。
ステップ 3	**mpls traffic-eng bfd lsp tailmultiplier** *number* 例： `RP/0/RSP0/cpu 0: router(config)#mpls traffic-eng bfd lsp tail multiplier 5`	BFD 乗数検出を設定します。 BFD 乗数検出の範囲は 3 ～ 10 です。デフォルトの BFD 乗数は 3 です。
ステップ 4	**commit**	

次のタスク

BFD マルチパスセッションをホストするように指定されたラインカードを含めるように **bfd multipath include location** *node-id* コマンドを設定します。

ラインカードでの LSP セッションを介した BFD の設定

フロントエンドとテールエンドの両方での LSP セッションを介した BFD は、次の設定が有効になっているラインカード上でホストされます。

手順の概要

1. **configure**
2. **bfd**
3. **multipath include location** *node-id*
4. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**bfd** 例： `RP/0/RSP0/cpu 0: router(config)# bfd`	BFD コンフィギュレーション モードを開始します。

ステップ 3	**multipath include location** *node-id*
	例：
	`RP/0/RSP0/cpu 0: router(config-bfd)# multipath include location 0/1/CPU0`
		特定のライン カードに BFD マルチパスを設定します。
		bfd multipath include を使用して、1 枚以上のラインカードを設定する必要があります。次に例を示します。
		`bfd`
		` multipath include location 0/1/CPU0`
		` multipath include location 0/2/CPU0`
		ヘッドエンドとテールエンドの両方での LSP セッションを介した BFD は、ラインカード上でホストされます。ヘッドエンドとテールエンドの両方での LSP セッションを介した BFD は、内部選択メカニズムに従ってラインカード 0/1/CPU0 と 0/2/CPU0 に配信されます。

ステップ 4	**commit**	

BFD オブジェクト トラッキングの設定

手順の概要

1. **configure**
2. **track** *track-name*
3. **type bfdrtr rate** *tx-rate*
4. **debounce***debounce*
5. **interface** *if-name*
6. **destaddress** *dest_addr*
7. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**track** *track-name* 例： `RP/0/RSP0/cpu 0: router(config)# track track1`	トラック コンフィギュレーション モードを開始します。 • *track-name*：トラッキングの対象となるオブジェクト名を指定します。
ステップ 3	**type bfdrtr rate** *tx-rate* 例： `RP/0/RSP0/cpu 0: router(config-track)# type bfdrtr rate 4`	tx_rate：BFD がリモート エンティティをプローブする時間（ミリ秒）
ステップ 4	**debounce***debounce* 例： `RP/0/RSP0/cpu 0: router(config-if)# debounce 10`	debounce：BFD が OT に通知するまでにステータスが一致しなければならない連続する BFD プローブの数
ステップ 5	**interface** *if-name* 例： `RP/0/RSP0/cpu 0: router(config-track-line-prot)# interface atm 0/2/0/0.1`	if_name：BFD がリモート BFD のステータスをチェックするために使用する送信元のインターフェイス名。
ステップ 6	**destaddress** *dest_addr* 例： `RP/0/RSP0/cpu 0: router(config-if)#destaddress 1.2.3.4`	dest_addr：トラッキングされるリモート BFD エンティティの IPV4 アドレス。
ステップ 7	**commit**	

BFD を設定するための設定例

BGP を介した BFD：例

次に、自律システム 65000 とネイバー 192.168.70.24 間での BFD を設定する例を示します。

```
RP/0/RSP0/cpu 0: router#configure
RP/0/RSP0/cpu 0: router(config)#router bgp 65000
RP/0/RSP0/cpu 0: router(config-bgp)#bfd multiplier 2
RP/0/RSP0/cpu 0: router(config-bgp)#bfd minimum-interval 20
RP/0/RSP0/cpu 0: router(config-bgp)#neighbor 192.168.70.24
RP/0/RSP0/cpu 0: router(config-bgp-nbr)#remote-as 2
RP/0/RSP0/cpu 0: router(config-bgp-nbr)#bfd fast-detect
RP/0/RSP0/cpu 0: router(config-bgp-nbr)#commit
RP/0/RSP0/cpu 0: router(config-bgp-nbr)#end
RP/0/RSP0/cpu 0: router#show run router bgp
```

OSPF を介した BFD：例

次に、ギガビット イーサネット インターフェイスで OSPF での BFD を有効にする例を示します。

```
RP/0/RSP0/cpu 0: router#configure
RP/0/RSP0/cpu 0: router(config)#router ospf 0
RP/0/RSP0/cpu 0: router(config-ospf)#area 0
RP/0/RSP0/cpu 0: router(config-ospf-ar)#interface gigabitEthernet 0/3/0/1
RP/0/RSP0/cpu 0: router(config-ospf-ar-if)#bfd fast-detect
RP/0/RSP0/cpu 0: router(config-ospf-ar-if)#commit
RP/0/RSP0/cpu 0: router(config-ospf-ar-if)#end

RP/0/RSP0/cpu 0: router#show run router ospf

router ospf 0
area 0
interface GigabitEthernet0/3/0/1
bfd fast-detect
```

次に、ギガビット イーサネット インターフェイスで OSPFv3 での BFD を有効にする例を示します。

```
RP/0/RSP0/cpu 0: router#configure
RP/0/RSP0/cpu 0: router(config)#router ospfv3 0
RP/0/RSP0/cpu 0: router(config-ospfv3)#bfd minimum-interval 6500
RP/0/RSP0/cpu 0: router(config-ospfv3)#bfd multiplier 7
RP/0/RSP0/cpu 0: router(config-ospfv3-ar)#area 0
RP/0/RSP0/cpu 0: router(config-ospfv3-ar)#interface gigabitethernet 0/1/5/0
RP/0/RSP0/cpu 0: router(config-ospfv3-ar-if)#bfd fast-detect
RP/0/RSP0/cpu 0: router(config-ospfv3-ar-if)#commit
RP/0/RSP0/cpu 0: router(config-ospfv3-ar-if)#end

RP/0/RSP0/cpu 0: router#show run router ospfv3
router ospfv3
```

```
area 0
interface GigabitEthernet0/1/5/0
bfd fast-detect
```

静的ルートを介した BFD：例

次に、IPv4 静的ルートでの BFD を有効にする例を示します。この例では、BFD セッションは、ネクストホップ 10.3.3.3 が到達可能になると、このネクストホップで確立されます。

```
RP/0/RSP0/cpu 0: router#configure
RP/0/RSP0/cpu 0: router(config)#router static
RP/0/RSP0/cpu 0: router(config-static)#address-family ipv4 unicast
RP/0/RSP0/cpu 0: router(config-static)#10.2.2.0/24 10.3.3.3 bfd fast-detect
RP/0/RSP0/cpu 0: router(config-static)#end
```

次に、IPv6 静的ルートでの BFD を有効にする例を示します。この例では、BFD セッションは、ネクスト ホップ 2001:0DB8:D987:398:AE3:B39:333:783 が到達可能になると、このネクスト ホップで確立されます。

```
RP/0/RSP0/cpu 0: router#configure
RP/0/RSP0/cpu 0: router(config)#router static
RP/0/RSP0/cpu 0: router(config-static)#address-family ipv6 unicast
RP/0/RSP0/cpu 0: router(config-static)#2001:0DB8:C18:2:1::F/64
2001:0DB8:D987:398:AE3:B39:333:783 bfd fast-detect minimum-interval 150 multiplier 4
RP/0/RSP0/cpu 0: router(config-static)#end

RP/0/RSP0/cpu 0: router#show run router static address-family ipv6 unicast
```

バンドル VLAN での BFD：例

次に、バンドル VLAN で BFD を設定する例を示します。

```
RP/0/RSP0/cpu 0: router#configure
RP/0/RSP0/cpu 0: router(config)#bfd
RP/0/RSP0/cpu 0: router(config-bfd)#multipath include location 0/0/CPU0
RP/0/RSP0/cpu 0: router(config-bfd)#exit

RP/0/RSP0/cpu 0: router#configure
RP/0/RSP0/cpu 0: router(config)#interface Bundle-ether 1
RP/0/RSP0/cpu 0: router(config-if)#bundle maximum-active links 1
RP/0/RSP0/cpu 0: router(config-if)#exit
!
RP/0/RSP0/cpu 0: router(config)#interface TenGigE 0/1/0/1
RP/0/RSP0/cpu 0: router(config-if)#bundle id 1 mode active
RP/0/RSP0/cpu 0: router(config-if)#exit
!
RP/0/RSP0/cpu 0: router(config)#interface TenGigE 0/2/0/1
RP/0/RSP0/cpu 0: router(config-if)#bundle id 1 mode active
RP/0/RSP0/cpu 0: router(config-if)#exit
!
RP/0/RSP0/cpu 0: router(config)#interface Bundle-Ether1.2
RP/0/RSP0/cpu 0: router(config-if)#ipv4 address 172.16.2.1 255.255.255.0
RP/0/RSP0/cpu 0: router(config-if)#encapsulation dot1q 2
```

```
RP/0/RSP0/cpu 0: router(config-if)#exit
!
RP/0/RSP0/cpu 0: router(config)#interface Bundle-Ether1.1
RP/0/RSP0/cpu 0: router(config-if)#ipv4 address 172.16.1.1 255.255.255.0
RP/0/RSP0/cpu 0: router(config-if)#encapsulation dot1q 1
!
RP/0/RSP0/cpu 0: router(config)#router static
RP/0/RSP0/cpu 0: router(config-static)#address-family ipv4 unicast
RP/0/RSP0/cpu 0: router(config-static-afi)#10.2.1.0/24 172.16.1.2 bfd fast-detect
minimum-interval 250
RP/0/RSP0/cpu 0: router(config-static-afi)#10.2.2.0/24 172.16.2.2 bfd fast-detect
minimum-interval 250
RP/0/RSP0/cpu 0: router(config-static-afi)#10.2.3.0/24 172.16.3.2 bfd fast-detect
minimum-interval 250
RP/0/RSP0/cpu 0: router(config-static-afi)#exit
RP/0/RSP0/cpu 0: router(config-static)#exit
!
```

ブリッジグループ仮想インターフェイスを介した BFD：例

次に、ピアノードと uut ノードの設定の例を示します。BVI インターフェイスがデフォルトテーブルではなく VRF にあることがわかります。

```
interface BVI100
vrf cctv1 <<<<<<<<<
```

次に、ピアノードの例を示します。

```
l2vpn
 bridge group bg
  bridge-domain bd
   interface Bundle-Ether1.100
   !
   routed interface BVI100
  !
 !
!
router vrrp
 interface BVI100
  bfd minimum-interval 15
  address-family ipv4
   vrrp 100
    address 192.168.1.254
    bfd fast-detect peer ipv4 192.168.1.2
   !
  !
 !
!
router ospf 100
vrf cctv1
  router-id 192.168.1.1
  area 0
   interface BVI100
   !
  !
 !
!
interface BVI100
```

```
 vrf cctv1
 ipv4 address 192.168.1.1 255.255.255.0
!
interface GigE0/1/0/10
 bundle id 1 mode active
 no shut
!
interface Bundle-Ether1
no shut
!
interface Bundle-Ether1.100 l2transport
 encapsulation dot1q 100
 rewrite ingress tag pop 1 symmetric

!
bfd multipath include loc 0/1/cpu0

interface MgmtEth0/RSP1/CPU0/0
 ipv4 address 7.37.19.20 255.255.0.0
 no shutdown
!
router static
 address-family ipv4 unicast
  0.0.0.0/0 7.37.0.1
```

次に、uut ノードの例を示します。

```
l2vpn
 bridge group bg
  bridge-domain bd
   interface Bundle-Ether1.100
   !
   routed interface BVI100
  !
 !
!
router vrrp
 interface BVI100
  bfd minimum-interval 15
  address-family ipv4
   vrrp 100
    address 192.168.1.254
    bfd fast-detect peer ipv4 192.168.1.1
   !
  !
 !
!
router ospf 100
 vrf cctv1
  router-id 192.168.1.2
  area 0
   interface BVI100
   !
  !
 !
!
interface BVI100
 vrf cctv1
 ipv4 address 192.168.1.2 255.255.255.0
!
```

```
interface GigE0/1/0/0
 bundle id 1 mode active
 no shut
 !
 interface Bundle-Ether1
 no shut
!
interface Bundle-Ether1.100 l2transport
 encapsulation dot1q 100
 rewrite ingress tag pop 1 symmetric

!
bfd multipath include location 0/1/CPU0
```

バンドル メンバ リンクでの BFD：例

次に、イーサネット バンドル インターフェイスのメンバ リンクで BFD を設定する例を示します。

```
bfd
 interface Bundle-Ether4
  echo disable
 !
 interface GigabitEthernet0/0/0/2.3
  echo disable
 !
!
interface GigabitEthernet0/0/0/3 bundle id 1 mode active
interface GigabitEthernet0/0/0/4 bundle id 2 mode active
interface GigabitEthernet0/1/0/2 bundle id 3 mode active
interface GigabitEthernet0/1/0/3 bundle id 4 mode active
interface Bundle-Ether1
 ipv4 address 192.168.1.1/30
 bundle minimum-active links 1
!
interface Bundle-Ether1.1
 ipv4 address 192.168.100.1/30
 encapsulation dot1q 1001
!
interface Bundle-Ether2
 bfd address-family ipv4 destination 192.168.2.2
 bfd address-family ipv4 fast-detect
 bfd address-family ipv4 min 83
 bfd address-family ipv4 mul 3
 ipv4 address 192.168.2.1/30
 bundle minimum-active links 1
!
interface Bundle-Ether3
 bfd address-family ipv4 destination 192.168.3.2
 bfd address-family ipv4 fast-detect
 bfd address-family ipv4 min 83
 bfd address-family ipv4 mul 3
 ipv4 address 192.168.3.1/30
 bundle minimum-active links 1
!
interface Bundle-Ether4
 bfd address-family ipv4 destination 192.168.4.2
 bfd address-family ipv4 fast-detect
 bfd address-family ipv4 min 83
 bfd address-family ipv4 mul 3
```

```
 ipv4 address 192.168.4.1/30
 bundle minimum-active links 1
!
interface GigabitEthernet 0/0/0/2
 ipv4 address 192.168.10.1/30
!
interface GigabitEthernet 0/0/0/2.1
 ipv4 address 192.168.11.1/30
ipv6 address beef:cafe::1/64
 encapsulation dot1q 2001
!
interface GigabitEthernet 0/0/0/2.2
 ipv4 address 192.168.12.1/30
 encapsulation dot1q 2002
!
interface GigabitEthernet 0/0/0/2.3
 ipv4 address 192.168.13.1/30
 encapsulation dot1q 2003
!
router static
 address-family ipv4 unicast
  10.10.11.2/32 192.168.11.2 bfd fast-detect minimum-interval 250 multiplier 3
  10.10.12.2/32 192.168.12.2 bfd fast-detect minimum-interval 250 multiplier 3
  10.10.13.2/32 192.168.13.2 bfd fast-detect minimum-interval 250 multiplier 3
  10.10.100.2/32 192.168.100.2 bfd fast-detect minimum-interval 250 multiplier 3
!
 address-family ipv6 unicast
  babe:cace::2/128 beef:cafe::2 bfd fast-detect minimum-interval 250 multiplier 3
 !
```

エコー パケット送信元アドレス：例

次に、ルータ上のすべての BFD セッションの BFD エコー パケットの送信元アドレスとして IP アドレス 10.10.10.1 を指定する例を示します。

```
RP/0/RSP0/cpu 0: router#configure
RP/0/RSP0/cpu 0: router(config)#bfd
RP/0/RSP0/cpu 0: router(config-bfd)#echo ipv4 source 10.10.10.1
```

次に、個々のギガビット イーサネット インターフェイス上の BFD エコー パケットの送信元アドレスとして IP アドレス 10.10.10.1 を指定する例を示します。

```
RP/0/RSP0/cpu 0: router#configure
RP/0/RSP0/cpu 0: router(config)#bfd
RP/0/RSP0/cpu 0: router(config-bfd)#interface gigabitethernet 0/1/0/0
RP/0/RSP0/cpu 0: router(config-bfd-if)#echo ipv4 source 10.10.10.1
```

次に、個々の Packet-over-SONET（POS）インターフェイスの BFD エコー パケットの送信元アドレスとして、IP アドレス 10.10.10.1 を指定する例を示します。

```
RP/0/RSP0/cpu 0: router#configure
RP/0/RSP0/cpu 0: router(config)#bfd
RP/0/RSP0/cpu 0: router(config-bfd)#interface pos 0/1/0/0
RP/0/RSP0/cpu 0: router(config-bfd-if)#echo ipv4 source 10.10.10.1
```

エコー遅延検出：例

次の例では、BFD 最小間隔は 50 ms、また BFD セッションの係数は 3 とします。

次に、1 のパケットカウントに対するエコー障害期間（I x M）の 100% のデフォルト値を使用してエコー遅延検出を有効にする例を示します。この例では、ラウンドトリップ遅延が 150 ms を超える 1 つのエコー パケットが検出されると、セッションはダウンします。

```
RP/0/RSP0/cpu 0: router#configure
RP/0/RSP0/cpu 0: router(config)#bfd
RP/0/RSP0/cpu 0: router(config-bfd)#echo latency detect
```

次に、1 のパケットカウントに対するエコー障害期間の 200%（2 倍）に基づいてエコー遅延検出を有効にする例を示します。この例では、ラウンドトリップ遅延が 300 ms を超える 1 つのパケットを検出すると、セッションはダウンします。

```
RP/0/RSP0/cpu 0: router#configure
RP/0/RSP0/cpu 0: router(config)#bfd
RP/0/RSP0/cpu 0: router(config-bfd)#echo latency detect percentage 200
```

次に、3 のパケットカウントに対するエコー障害期間の 100% に基づいてエコー遅延検出を有効にする例を示します。この例では、ラウンドトリップ遅延が 150 ms を超える 3 つの連続したエコー パケットが検出されると、セッションはダウンします。

```
RP/0/RSP0/cpu 0: router#configure
RP/0/RSP0/cpu 0: router(config)#bfd
RP/0/RSP0/cpu 0: router(config-bfd)#echo latency detect percentage 100 count 3
```

エコー起動検証：例

次に、最後に受信された制御パケットにゼロ以外の「Required Min Echo RX Interval」値が含まれている場合に、非バンドル インターフェイスの BFD セッションのエコー起動検証を有効にする例を示します。

```
RP/0/RSP0/cpu 0: router#configure
RP/0/RSP0/cpu 0: router(config)#bfd
RP/0/RSP0/cpu 0: router(config-bfd)#echo startup validate
```

次に、最後の制御パケット内の「Required Min Echo RX Interval」値には関係なく、非バンドル インターフェイスの BFD セッションのエコー起動検証を有効にする例を示します。

```
RP/0/RSP0/cpu 0: router#configure
RP/0/RSP0/cpu 0: router(config)#bfd
RP/0/RSP0/cpu 0: router(config-bfd)#echo startup validate force
```

BFD エコーモードの無効化：例

次に、ルータでエコーモードを無効にする例を示します。

```
RP/0/RSP0/cpu 0: router#configure
RP/0/RSP0/cpu 0: router(config)#bfd
RP/0/RSP0/cpu 0: routerr(config-bfd)#echo disable
```

次に、インターフェイスでエコーモード無効にする例を示します。

```
RP/0/RSP0/cpu 0: router#configure
RP/0/RSP0/cpu 0: router(config)#bfd
RP/0/RSP0/cpu 0: router(config-bfd)#interface gigabitethernet 0/1/0/0
RP/0/RSP0/cpu 0: router(config-bfd-if)#echo disable
```

BFD ダンプニング：例

次に、BFD バンドル メンバの BFD セッション開始の初期および最大遅延を設定する例を示します。

```
RP/0/RSP0/cpu 0: router#configure
RP/0/RSP0/cpu 0: router(config)#bfd
RP/0/RSP0/cpu 0: router(config-bfd)#dampening bundle-member initial-wait 8000
RP/0/RSP0/cpu 0: router(config-bfd)#dampening bundle-member maximum-wait 15000
```

次に、非バンドル インターフェイスの BFD に対するデフォルトの initial-wait を変更する例を示します。

```
RP/0/RSP0/cpu 0: router#configure
RP/0/RSP0/cpu 0: router(config)#bfd
RP/0/RSP0/cpu 0: router(config-bfd)#dampening initial-wait 30000
RP/0/RSP0/cpu 0: router(config-bfd)#dampening maximum-wait 35000
```

BFD IPv6 チェックサム：例

次に、ルータのすべての BFD セッションの UDP パケットの IPv6 チェックサム計算を無効にする例を示します。

```
RP/0/RSP0/cpu 0: router#configure
RP/0/RSP0/cpu 0: router(config)#bfd
RP/0/RSP0/cpu 0: router(config-bfd)#ipv6 checksum disable
```

次に、ルータのすべての BFD セッションの UDP パケットの IPv6 チェックサム計算を再び有効にする例を示します。

```
RP/0/RSP0/cpu 0: router#configure
RP/0/RSP0/cpu 0: router(config)#bfd
RP/0/RSP0/cpu 0: router(config-bfd)#no ipv6 checksum disable
```

次に、個々のインターフェイスのBFDセッションのエコーモードを有効にする例を示します。

```
RP/0/RSP0/cpu 0: router#configure
RP/0/RSP0/cpu 0: router(config)#bfd
RP/0/RSP0/cpu 0: router(config-bfd)#interface gigabitethernet 0/1/0/0
RP/0/RSP0/cpu 0: router(config-bfd-if)#ipv6 checksum
```

次に、個々のインターフェイスのBFDセッションのエコーモードを無効にする例を示します。

```
RP/0/RSP0/cpu 0: router#configure
RP/0/RSP0/cpu 0: router(config)#bfd
RP/0/RSP0/cpu 0: router(config-bfd)#interface gigabitethernet 0/1/0/0
RP/0/RSP0/cpu 0: router(config-bfd-if)#ipv6 checksum disable
```

Cisco IOS および Cisco IOS XR ソフトウェアを実行しているルータの BFD ピア：例

次に、Cisco IOS ソフトウェアを実行しているルータ 1 のルータインターフェイスで BFD を設定し、**bfd neighbor** コマンドを使用してインターフェイスの IP アドレス 192.0.2.1 をルータ 2 の BFD ピアとして指定する例を示します。ルータ 2 は、Cisco IOS XR ソフトウェアを実行していて、**router static** コマンドと **address-family ipv4 unicast** コマンドを使用して、IP アドレス 192.0.2.2 でルータ 1 のインターフェイスに戻るパスを指定します。

ルータ 1（Cisco IOS ソフトウェア）

```
RP/0/RSP0/cpu 0: router#configure
RP/0/RSP0/cpu 0: router(config)#interface GigabitEthernet8/1/0
RP/0/RSP0/cpu 0: router(config-if)#description to-TestBed1 G0/0/0/0
RP/0/RSP0/cpu 0: router(config-if)#ip address 192.0.2.2 255.255.255.0
RP/0/RSP0/cpu 0: router(config-if)#bfd interval 100 min_rx 100 multiplier 3
RP/0/RSP0/cpu 0: router(config-if)#bfd neighbor 192.0.2.1
```

ルータ 2（Cisco IOS XR ソフトウェア）

```
RP/0/RSP0/cpu 0: router#configure
RP/0/RSP0/cpu 0: router(config)#router static
RP/0/RSP0/cpu 0: router(config-static)#address-family ipv4 unicast
RP/0/RSP0/cpu 0: router(config-static-afi)#10.10.10.10/32 192.0.2.2 bfd fast-detect
RP/0/RSP0/cpu 0: router(config-static-afi)#exit
RP/0/RSP0/cpu 0: router(config-static)#exit
RP/0/RSP0/cpu 0: router(config)#interface GigabitEthernet0/0/0/0
RP/0/RSP0/cpu 0: router(config-if)#ipv4 address 192.0.2.1 255.255.255.0
```

BFD IPv6 マルチホップの設定：例

eBGP ネイバーの BFD IPv6 マルチホップの設定：例

次に、eBGP ネイバーの BFD IPv6 マルチホップを設定する例を示します。

```
bfd
 multipath include location 0//CPU0
!
router bgp 65001
 neighbor 21:1:1:1:1:1:1:2
  bfd fast-detect
  ebgp-multiphop 255
```

iBGP ネイバーの BFD IPv6 マルチホップの設定：例

次に、iBGP ネイバーの BFD IPv6 マルチホップを設定する例を示します。

```
bfd
 multipath include location 0/7/CPU0
!
router bgp 65001
 neighbor 21:1:1:1:1:1:1:2
  bfd fast-detect
```

MPLS TE LSP を介した BFD：例

次に、MPLS TE LSP を介した BFD の設定方法の例を示します。

MPLS TE トンネルヘッドエンド設定を介した BFD：例

次に、ヘッドエンドで MPLS TE トンネルを介した BFD を設定する例を示します。

```
bfd multipath include loc 0/1/CPU0
mpls oam
interface tunnel-te 1 bfd fast-detect
interface tunnel-te 1
 bfd minimum-interval
 bfd multiplier
 bfd bringup-timeout
 bfd lsp-ping interval 60
 bfd lsp-ping disable
 bfd dampening initial-wait    (default 16000 ms)
 bfd dampening maximum-wait    (default 600000 ms)
 bfd dampening secondary-wait  (default 20000 ms)
 logging events bfd-status
```

MPLS TE トンネルテールエンド設定を介した BFD：例

次に、テールエンドで MPLS TE トンネルを介した BFD を設定する例を示します。

```
bfd multipath include loc 0/1/CPU0
mpls oam
mpls traffic-eng bfd lsp tail multiplier 3
mpls traffic-eng bfd lsp tail minimum-interval 100
```

次の作業

BFDは、複数のプラットフォームでサポートされます。これらのコマンドの詳細については、プラットフォームに対応する『*Cisco IOS XR Routing Command Reference*』および『*Cisco IOS XR MPLS Command Reference*』の関連の章を参照してください。

http://www.cisco.com/en/US/products/ps5845/prod_command_reference_list.html

- 『*BGP Commands on Cisco IOS XR Software*』
- 『*IS-IS Commands on Cisco IOS XR Software*』
- 『*OSPF Commands on Cisco IOS XR Software*』
- 『*Static Routing Commands on Cisco IOS XR Software*』
- 『*MPLS Traffic Engineering Commands on Cisco IOS XR Software*』

その他の参考資料

ここでは、Cisco IOS XR ソフトウェアの BFD の実装に関する関連資料について説明します。

関連資料

関連項目	マニュアル タイトル
BFD コマンド：コマンド構文の詳細、コマンド モード、コマンド履歴、デフォルト、使用上のガイドライン、例	*Routing Command Reference for Cisco ASR 9000 Series Routers*
QoS パケット分類の設定	*Modular QoS Configuration Guide for Cisco ASR 9000 Series Routers*

標準

標準	タイトル
この機能でサポートされる新規の標準または変更された標準はありません。また、既存の標準のサポートは変更されていません。	—

RFC

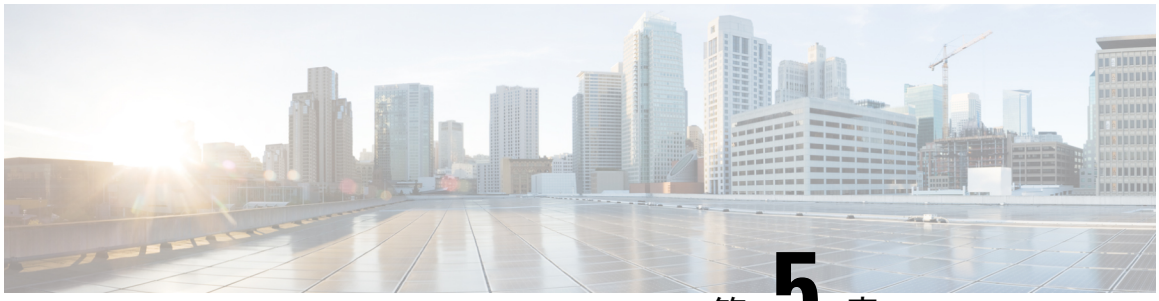
RFC	タイトル
rfc5880_bfd_base	『*Bidirectional Forwarding Detection*』、2010 年 6 月
rfc5881_bfd_ipv4_ipv6	『*BFD for IPv4 and IPv6 (Single Hop)*』、2010 年 6 月
rfc5883_bfd_multihop	『*BFD for Multihop Paths*』、2010 年 6 月

MIB

MIB	MIB のリンク
—	Cisco IOS XR ソフトウェアを使用して MIB の場所を特定してダウンロードするには、次の URL にある Cisco MIB Locator を使用して、[Cisco Access Products] メニューからプラットフォームを選択します。 https://mibs.cloudapps.cisco.com/ITDIT/MIBS/servlet/index

シスコのテクニカル サポート

説明	リンク
シスコのテクニカル サポート Web サイトでは、製品、テクノロジー、ソリューション、技術的なヒント、およびツールへのリンクなどの、数千ページに及ぶ技術情報が検索可能です。Cisco.com に登録済みのユーザは、このページから詳細情報にアクセスできます。	http://www.cisco.com/techsupport



第 5 章

EIGRP の実装

Enhanced Interior Gateway Routing Protocol（EIGRP）は、シスコによって開発された IGRP の拡張バージョンです。このモジュールでは、Cisco IOS XR ソフトウェアを使用して基本 EIGRP 設定を実装するために理解する必要がある概念と作業について説明します。EIGRP はディスタンス ベクトル ルーティング テクノロジーを採用しているため、ルータはネットワーク全体でのルータとリンクのすべての関係を認識する必要がありません。各ルータは対応する距離の宛先をアドバタイズし、ルート受信時に距離を調整し、ネイバー ルートへ情報を伝搬します。

次の機能に関連する EIGRP の設定については、このモジュールの関連資料 （375 ページ）を参照してください。

- マルチプロトコル ラベル スイッチング（MPLS）レイヤ 3 バーチャル プライベート ネットワーク（VPN）
- Site of Origin（SoO）のサポート

（注） Cisco IOS XR ソフトウェアの EIGRP の詳細とこのモジュールに示す EIGRP コマンドの詳細な説明については、 *Routing Command Reference for Cisco ASR 9000 Series Routers*の「*EIGRP Commands*」の章を参照してください。設定作業の実行時に使用する可能性があるその他のコマンドに関するドキュメントを見つけるには、オンラインのCisco IOS XR ソフトウェア マスター コマンド索引を検索してください。

EIGRP の実装の機能履歴

リリース	変更内容
リリース 3.7.2	この機能が導入されました。
リリース 4.3.0	ワイド メトリック サポート機能が追加されました。
リリース 6.0.1	Site of Origin（SoO）属性のサポートが追加されました。

- EIGRP の実装の前提条件 （336 ページ）
- EIGRP の実装での制約事項 （336 ページ）

EIGRP の実装の前提条件

適切なタスク ID を含むタスク グループに関連付けられているユーザ グループに属している必要があります。このコマンド リファレンスには、各コマンドに必要なタスク ID が含まれます。ユーザ グループの割り当てが原因でコマンドを使用できないと考えられる場合、AAA 管理者に連絡してください。

EIGRP の実装での制約事項

Cisco IOS XR ソフトウェア のこのバージョンで EIGRP を実行する場合には、次の制約事項があります。

- 最大 4 つの EIGRP プロセスのインスタンスがサポートされます。
- EIGRP プロセス名に使用できる文字は、**@.#:-_** のみです。
- 簡易ネットワーク管理プロトコル（SNMP）MIB はサポートされていません。
- ネットワーク コマンドがないため、インターフェイスのスタティック ルートは EIGRP に自動的に再配布されません。
- 接続ルートおよび静的ルートの再配布には、メトリック設定が必要です（**default-metric** コマンドまたはルートポリシーのいずれかを使用）。
- 自動要約はデフォルトでは無効になっています。
- スタブ リーク マップはサポートされていません。

EIGRP の実装に関する情報

EIGRP を実装するには、次の概念を理解する必要があります。

EIGRP 機能の概要

Enhanced Interior Gateway Routing Protocol（EIGRP）は、さまざまなトポロジとメディアに適した内部ゲートウェイ プロトコルです。EIGRP は拡張性が高く、最小限のネットワーク トラフィックで非常に短いコンバージェンス時間を実現します。

通常の動作では EIGRP のネットワーク リソース使用率は非常に低くなります。安定したネットワークでは hello パケットだけが送信されます。トポロジの変更が生じた場合、ルーティン

グテーブル全体ではなくルーティングテーブルの変更だけが伝搬されます。この伝搬により、ネットワーク上でのルーティング プロトコル自体による負荷が減ります。EIGRP は、ネットワーク トポロジ変更においても短いコンバージェンス時間を実現します。

EIGRPにおける距離情報は、使用可能な帯域幅、遅延、負荷率、リンク信頼性の総合情報として表され、またコンバージェンスプロパティと運用効率が向上します。リンク特性を細かく調整することで最適なパスが実現します。

EIGRP に採用されているコンバージェンス テクノロジーは、SRI International で行われた調査に基づき、Diffusing Update Algorithm（DUAL）と呼ばれるアルゴリズムを採用しています。このアルゴリズムは、ルート計算中のどの時点でもループが発生しないようにし、トポロジ変更に関与するすべてのデバイスを同時に同期できるようにします。トポロジ変更の影響を受けないルータは、再計算に含まれません。DUAL を使用した場合のコンバージェンス時間は、既存の他のルーティング プロトコルのコンバージェンス時間に匹敵します。

EIGRP の機能

EIGRP は、次の機能を提供します。

- 高速コンバージェンス：DUAL アルゴリズムにより、現在利用可能なルーティング プロトコルと同様にルーティング情報を迅速にコンバージできます。
- 部分アップデート：宛先の状態が変化した場合、EIGRPは、ルーティングテーブルの内容全体を送信するのではなく、差分アップデートを送信します。この機能により、EIGRPパケットに必要な帯域幅が最小限に抑えられます。
- ネイバー探索メカニズム：隣接ルータの学習に使用される簡単な hello メカニズムです。これはプロトコルに依存しません。
- 可変長サブネット マスク（VLSM）
- 任意のルート集約
- スケーリング：EIGRP は大規模なネットワークに合わせて拡張します。

Cisco IOS XR の実装でサポートされている主な機能を次に示します。

- Site of Origin（SoO）およびボーダー ゲートウェイ プロトコル（BGP）コスト コミュニティ サポートによるプロバイダーエッジ（PE）- カスタマーエッジ（CE）プロトコルのサポート。
- MPLS に対する PECE プロトコル サポート。

EIGRP コンポーネント

EIGRP には、次の 4 つの基本コンポーネントがあります。

- ネイバー探索またはネイバー回復
- Reliable Transport Protocol

- DUAL 有限状態マシン
- プロトコル依存モジュール

ネイバー探索およびネイバーの回復：直接接続されたネットワーク上の他のルータに関する情報を動的に取得するために、ルータで使用されるプロセスです。また、ネイバーが到達不能または動作不能になっていることを検出するためにも使用されます。ネイバー探索およびネイバーの回復は、サイズの小さな hello パケットを定期的に送信することにより、わずかなオーバーヘッドで行われます。hello パケットを受信しているかぎり、Cisco IOS XR ソフトウェアはネイバーがダウンせずに機能しているものと判定します。ネイバーが正常に動作していることが確認されると、隣接ルータとの間でルーティング情報を交換できます。

Reliable Transport Protocol：EIGRP パケットをすべてのネイバーに確実に、順序どおりに配信します。マルチキャストパケットとユニキャストパケットが混在した伝送もサポートされます。EIGRP パケットには、確実に送信する必要があるものと、その必要がないものがあります。効率化のため、信頼性は必要時にのみ提供されます。たとえば、マルチキャスト機能を備えているマルチアクセス ネットワーク（イーサネットなど）では、すべてのネイバーに対して個別に hello パケットを高信頼性で送信する必要はありません。そのため、EIGRP は、1 つのマルチキャスト hello を送信し、パケットに確認応答が必要ないという通知をそのパケットに含めます。その他のタイプのパケット（アップデートなど）には、確認応答が必要であり、そのことをパケットで示します。信頼性の高い転送では、確認応答のないパケットの保留中にすばやくマルチキャストパケットを送信できます。これにより、さまざまな速度のリンクが存在する場合でも短いコンバージェンス時間を維持できます。

DUAL 有限状態マシンには、すべてのルート計算の決定プロセスが組み込まれており、すべてのネイバーによってアドバタイズされたすべてのルートが追跡されます。DUAL は距離情報（メトリックともいう）を使用して、効率的な、ループのないパスを選択し、DUAL は到達可能条件の計算に基づいてルーティングテーブルに挿入するルートを選択します。後継ルータは、宛先への最小コスト パス（ルーティング ループに関連しないことが保証されている）を持つ、パケット転送に使用される隣接ルータです。フィジブル サクセサはないが、宛先をアドバタイジングするネイバーがある場合、再計算が行われ、この結果、新しい後継ルータが決定されます。ルートの再計算に必要な時間は、コンバージェンス時間に影響します。再計算は、プロセッサに高い負荷を与えます。したがって、不要な再計算を行わないことを推奨します。トポロジが変更されると、DUAL はフィジブル サクセサの有無を調べます。適切なフィジブル サクセサが存在する場合は、それらを探して使用し、不要な再計算を回避します。

プロトコル依存モジュールは、ネットワーク層プロトコル固有のタスクを実行します。たとえば EIGRP モジュールでは、IP で暗号化されている EIGRP パケットの送信と受信を行います。また、EIGRP パケットを解析したり、DUAL に受信した新しい情報を通知したりします。EIGRP は DUAL にルーティング決定を行うように要求しますが、結果は IP ルーティング テーブルに格納されます。EIGRP は、他の IP ルーティング プロトコルによって取得したルートの再配信も行います。

EIGRP 設定のグループ化

Cisco IOS XR ソフトウェアは、ルータ EIGRP コンフィギュレーション モードですべての EIGRP 設定（EIGRP に関連するインターフェイス設定も含む）をグループ化します。EIGRP 設定全

体を表示するには、**show running-config router eigrp** コマンドを使用します。このコマンドの出力には、設定されている EIGRP インスタンスの実行設定（インターフェイス割り当てとインターフェイス属性を含む）が表示されます。

EIGRP コンフィギュレーション モード

ここでは各コンフィギュレーション モードの開始方法について説明します。現行のモードで**?** コマンドを入力することで、そのモードで使用可能なコマンドを表示できます。

ルータ コンフィギュレーション モード

次に、ルータ コンフィギュレーション モードを開始する例を示します。

```
RP/0/RSP0/cpu 0: router# configuration
RP/0/RSP0/cpu 0: router(config)# router eigrp 100
RP/0/RSP0/cpu 0: router(config-eigrp)#
```

VRF コンフィギュレーション モード

次に、VRF コンフィギュレーション モードを開始する例を示します。

```
RP/0/RSP0/cpu 0: router# configuration
RP/0/RSP0/cpu 0: router(config)# router eigrp 100
RP/0/RSP0/cpu 0: router(config-eigrp)# vrf customer1
RP/0/RSP0/cpu 0: router(config-eigrp-vrf)#
```

IPv4 アドレス ファミリ コンフィギュレーション モード

次に、IPv4 アドレス ファミリ コンフィギュレーション モードを開始する例を示します。

```
RP/0/RSP0/cpu 0: router# configuration
RP/0/RSP0/cpu 0: router(config)# router eigrp 100
RP/0/RSP0/cpu 0: router(config-eigrp)# address-family ipv4
RP/0/RSP0/cpu 0: router(config-eigrp-af)#
```

IPv4 VRF アドレス ファミリ コンフィギュレーション モード

次に、IPv4 VRF アドレス ファミリ コンフィギュレーション モードを開始する例を示します。

```
RP/0/RSP0/cpu 0: router# configuration
RP/0/RSP0/cpu 0: router(config)# router eigrp 100
RP/0/RSP0/cpu 0: router(config-eigrp)# vrf customer1
RP/0/RSP0/cpu 0: router(config-eigrp-vrf)# address-family ipv4
RP/0/RSP0/cpu 0: router(config-eigrp-vrf-af)#
```

インターフェイス コンフィギュレーション モード

次に、IPv4 アドレス ファミリ コンフィギュレーション モードでインターフェイス コンフィギュレーション モードを開始する例を示します。

```
RP/0/RSP0/cpu 0: router# configuration
RP/0/RSP0/cpu 0: router(config)# router eigrp 100
RP/0/RSP0/cpu 0: router(config-eigrp)# address-family ipv4
RP/0/RSP0/cpu 0: router(config-eigrp-af)# interface GigabitEthernet 0/3/0/0
RP/0/RSP0/cpu 0: router(config-eigrp-af-if)#
```

EIGRP インターフェイス

EIGRP インターフェイスは次のいずれかのタイプとして設定できます。

- アクティブ：接続されたプレフィックスをアドバタイズし、隣接関係を形成します。これはデフォルトのインターフェイス タイプです。
- パッシブ：接続されたプレフィックスをアドバタイズしますが、隣接関係は形成しません。インターフェイスをパッシブに設定するには、passive コマンドを使用します。パッシブ インターフェイスはむやみに使用せず、EIGRP ドメインに挿入する必要がある重要なプレフィックス（ループバック アドレスなど）に対して使用してください。アドバタイズする必要がある接続プレフィックスが多数ある場合は、代わりに適切なポリシーを使用した接続ルートの再配布を使用してください。

EIGRP プロセスへの再配布

他のプロトコルからのルートを EIGRP に再配布できます。ルートポリシーは、**redistribute** コマンドで設定できます。メトリックが必要です。**default-metric** コマンドか、またはルートを EIGRP にインポートする **redistribute** コマンドを使用して設定されたルートポリシーで設定します。

ルート ポリシーでは、宛先、発信元プロトコル、ルート タイプ、ルート タグなどの属性に基づいてルートをフィルタリングできます。VRF で再配布が設定されている場合は、ルーティング情報ベース（RIB）内のルートに接続されている拡張コミュニティがEIGRPによって取得されます。MPSL VPN のバック ドア リンクが存在する場合は、ルーティング ループを除外するために SoO が使用されます。

EIGRP ルーティングのメトリックの重み付け

EIGRP は宛先ネットワークへのパスで最小の帯域幅を使用し、ルーティング メトリックを計算するために合計遅延を使用します。**metric weights** コマンドを使用して、EIGRP のルーティングおよびメトリック計算のデフォルト動作を調整できます。たとえば、この調整によって、人工衛星との送信を可能にするためにシステムの動作をチューニングすることができます。EIGRP メトリックのデフォルトは、大半のネットワークで最適なパフォーマンスを実現できるよう、慎重に選択されています。

デフォルトでは、EIGRP複合メトリックは、特定のルートのセグメント遅延と（拡張およびインバートされた）最小セグメント帯域幅の合計である 32 ビットになります。同種メディアのネットワークでは、このメトリックは1ホップカウントまで減少します。混合メディア（FDDI、イーサネット、および毎秒 9600 ビットから T1 まで多様なレートのシリアル回線）のネットワークでは、最低メトリックのルートが、宛先までの最適なパスになります。

K 値の不一致

K 値（EIGRP メトリック）の不一致があると、ネイバー関係を確立することができなくなり、ネットワークのコンバージェンスに悪影響を与えることがあります。次の例で、2 つの EIGRP ピア（ルータ A とルータ B）間のこの動作について説明します。

K 値が一致しないため、ルータ B のコンソールに次のエラー メッセージが表示されます。

```
  RP/0/RSP0/CPU0:Mar 13 08:19:55:eigrp[163]:%ROUTING-EIGRP-5-NBRCHANGE:IP-EIGRP(0)
1:Neighbor 11.0.0.20 (GigabitEthernet0/6/0/0) is down: K-value mismatch
```

次の 2 つのシナリオで、このエラー メッセージが表示される可能性があります。

- 同じリンク上に 2 台のルータが接続されており、ネイバー関係を確立するよう設定されている。ただし、各ルータには異なる K 値が設定されている。

  次の設定がルータ A に適用されています。K 値は **metric weights** コマンドで変更されます。帯域幅計算を調整するために、*k1* 引数に値 2 が入力されます。遅延計算を調整するために、*k3* 引数に値 1 が入力されます。

  ```
  hostname ROUTER-A!
  interface GigabitEthernet0/6/0/0
   ipv4 address 10.1.1.1 255.255.255.0

  router eigrp 100
   metric weights 0 2 0 1 0 0
   interface GigabitEthernet0/6/0/0
  ```

  次の設定がルータ B に適用されています。ただし、**metric weights** コマンドは適用されておらず、デフォルトの K 値が使用されています。デフォルトの K 値は、1、0、1、0、および 0 です。

  ```
  hostname ROUTER-B!
  interface GigabitEthernet0/6/0/1
   ipv4 address 10.1.1.2 255.255.255.0

  router eigrp 100
   interface GigabitEthernet0/6/0/1
  ```

  帯域幅計算はルータ A で 2 に設定され、ルータ B で 1（デフォルト）に設定されます。この設定により、これらのピアがネイバー関係を形成できなくなります。

- 2 つのピアのうち、いずれかが「goodbye」メッセージを送信したのに、受信側のルータがこのメッセージをサポートしていない場合に、K 値の不一致エラー メッセージが表示されることがあります。この場合には、受信側のルータが、このメッセージを K 値の不一致と解釈します。

goodbye メッセージ

goodbye メッセージは、EIGRP ネットワークのコンバージェンスを改善するために設計された機能です。goodbye メッセージは、EIGRP ルーティングプロセスがシャットダウンしたときに、隣接するピアに、近い将来トポロジの変更が生じることを知らせるブロードキャストです。この機能により、ホールドタイマーの期限が切れた後でピアがトポロジの変更を検出した場合よりも効率よく、EIGRP ピアがネイバー関係を同期化および再計算する機能をサポートできます。

goodbye メッセージを受信すると、サポートされているリリースを実行しているルータによって、次のメッセージが表示されます。

```
RP/0/RSP0/CPU0:Mar 13 09:13:17:eigrp[163]:%ROUTING-EIGRP-5-NBRCHANGE: IP-EIGRP(0) 1:
Neighbor 10.0.0.20 (GigabitEthernet0/6/0/0) is down: Interface Goodbye received
```

goodbye メッセージをサポートしていないソフトウェア リリースを実行している Cisco ルータは、このメッセージが、K 値の不一致であると誤った解釈をして、次のメッセージを表示することがあります。

```
RP/0/RSP0/CPU0:Mar 13 09:13:17:eigrp[163]:%ROUTING-EIGRP-5-NBRCHANGE: IP-EIGRP(0) 1:
Neighbor 10.0.0.20 (GigabitEthernet0/6/0/0) is down: K-value mismatch
```

（注） サポートしていないピアが goodbye メッセージを受信しても、通常のネットワーク処理を中断することはありません。ホールドタイマーの期限が切れると、サポートしていないピアはセッションを終了します。送信元がリロードした後も、送信側および受信側のルータは正常に再コンバージェンスします。

EIGRP パケットに使用されているリンク帯域幅のパーセンテージ

デフォルトでは、**bandwidth** インターフェイス コンフィギュレーション コマンドでの設定に従い、リンク帯域幅の最大 50 % が EIGRP パケットに消費されます。異なるレベルのリンク使用率が必要な場合、または設定されている帯域幅が実際のリンク帯域幅と一致しない場合は（ルート メトリックの計算に影響するように設定されている場合があります）、この値を変更できます。

EIGRP プロセスの浮動サマリー ルート

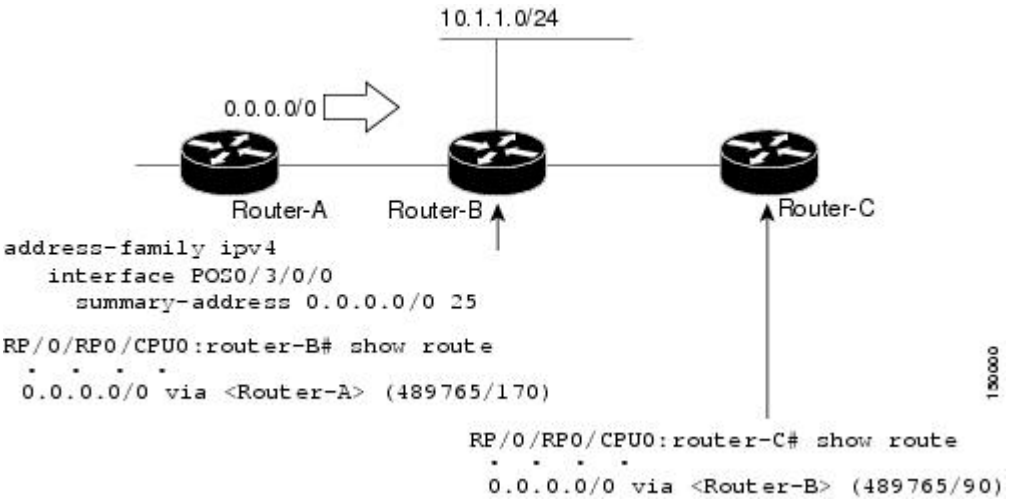
summary-address コマンドを設定するときには浮動サマリールートも使用できます。浮動サマリー ルートは、インターフェイス レベルでデフォルトのルートおよびアドミニストレーティブディスタンスを適用することによって作成されます。この拡張機能の動作を次のシナリオで説明します。

図 16 : ルータ B へ適用される浮動サマリー ルート （343 ページ） に、ルータ A、ルータ B、およびルータ C の 3 つのルータがあるネットワークを示します。ルータ A は、ネットワーク内の他の場所からデフォルトルートを学習し、ルータ B にこのルートをアドバタイズします。ルータ B は、デフォルトのサマリールートだけがルータ C にアドバタイズされるように設定

されています。デフォルトのサマリールートは、次の設定を使用して、ルータ B のインターフェイス 0/1 に適用されます。

```
RP/0/RSP0/cpu 0: router(config)# router eigrp 100
RP/0/RSP0/cpu 0: router(config-eigrp)# address-family ipv4
RP/0/RSP0/cpu 0: router(config-eigrp-af)# interface GigabitEthernet 0/3/0/0
RP/0/RSP0/cpu 0: router(config-eigrp-af-if)# summary-address 100.0.0.0 0.0.0.0
```

図 *16*：ルータ *B* へ適用される浮動サマリー ルート



ルータ B のデフォルトのサマリールートの設定は、ルータ C への 0.0.0.0/0 サマリールートを送信し、10.1.1.0/24 ルートを含む他のすべてのルートをルータ C にアドバタイズしないようにします。ただし、この設定では、ルータ B のローカル破棄ルートも生成されます。このルートは、アドミニストレーティブ ディスタンスが 5 のヌル 0 インターフェイスへの 0.0.0.0/0 のルートです。このルートが作成された場合、EIGRP が学習したデフォルト ルートよりも優先されます。このためルータ B は、通常では 0.0.0.0/0 ルートに到達する宛先に到達できなくなります。

この問題は、ルータ C に接続しているルータ B 上のインターフェイスに対して浮動サマリールートを適用することによって解決されます。浮動サマリールートを適用するには、次のコマンド行を使用して、ルータ B のインターフェイス上でデフォルトのサマリールートに対してアドミニストレーティブ ディスタンスを関連付けます。

```
RP/0/RSP0/cpu 0: router(config-if)# summary-address 100 0.0.0.0 0.0.0.0 250
```

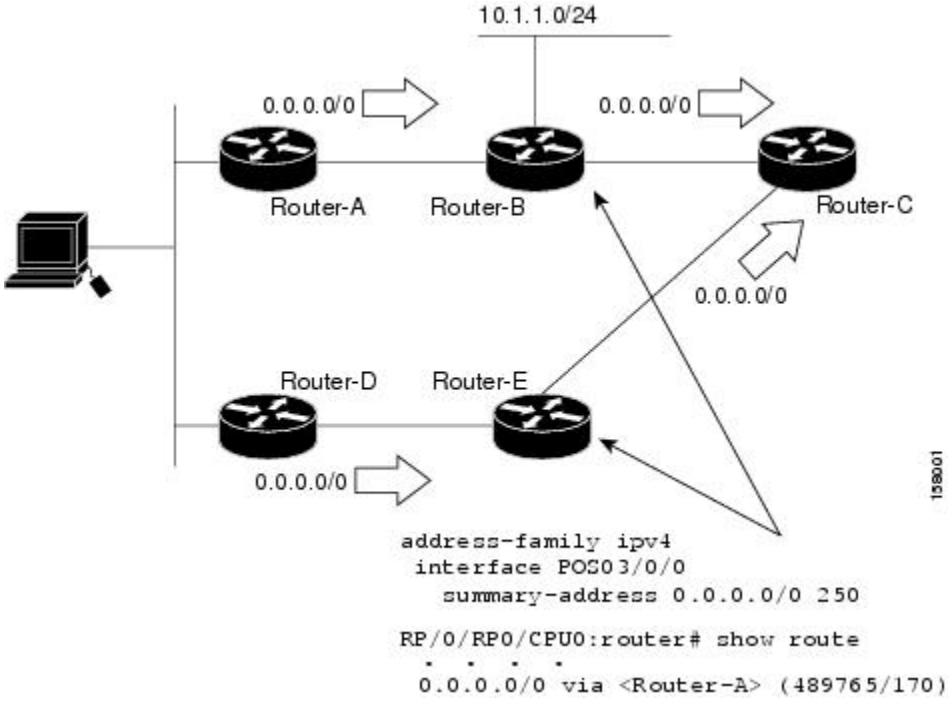
このコマンド行で適用されたアドミニストレーティブ ディスタンス 250 は、ルータ B で生成された破棄ルートに割り当てられています。ルータ A からの 0.0.0.0/0 は EIGRP を介して学習され、ローカル ルーティング テーブルにインストールされます。ルータ C へのルーティングが復元されます。

ルータ A がルータ B に対する接続を失っても、ルータ B がルータ C へ継続してデフォルトルートをアドバタイズしていると、トラフィックは、ルータ B へ割り当てられている宛先へ、その

まま到達することが可能です。ただしルータ A に到達するネットワークまたはルータ A の背後にあるネットワークを宛先とするトラフィックは、ルータ B に到達するとドロップされます。

図 17: デュアルホーム接続リモートへ適用される浮動サマリー ルート（344 ページ） に、コアからの 2 つの接続、ルータ A およびルータ D を持つネットワークを示します。両方のルータで、ルータ C に接続するインターフェイスで浮動サマリールートが設定されています。ルータ E とルータ C の間の接続に障害が発生しても、ネットワークは正常な処理を続行します。すべてのトラフィックは、ルータ B を通ってルータ C に流れ、ルータ A およびルータ D へ接続されているホストへ到達します。

図 *17*: デュアルホーム接続リモートへ適用される浮動サマリー ルート



ただしルータ D とルータ E の間のリンクで障害が発生した場合、ルータ E へのリンク（ルータ C へのリンク以外）が少なくとも 1 つアクティブであるかぎり、ルータ E は引き続きデフォルト ルート（0.0.0.0/0）をルータ C にアドバタイズするため、ネットワークではトラフィックがブラックホールにダンプされることがあります。このシナリオではルータ C は引き続きルータ E にトラフィックを転送しますが、ルータ E はトラフィックをドロップするためブラックホールが形成されます。この問題を防ぐには、ネットワーク セグメント間の出力点が 1 つのみのシングルホーム接続リモート ルータまたはエリアでのみ、アドミニストレーティブ ディスタンスを使用してサマリーアドレスを設定してください。（ネットワークのセグメント間の）出力点が 2 つ以上存在する場合には、フローティング デフォルト ルートを設定すると、ブラック ホールが形成される原因となることがあります。

EIGRP プロセスのスプリット ホライズン

スプリット ホライズンは、EIGRP アップデート パケットとクエリー パケットの送信を制御します。スプリットホライズンがインターフェイスでイネーブルになると、アップデートパケットとクエリー パケットは、このインターフェイスがネクスト ホップとなる宛先には送信されません。この方法でアップデート パケットとクエリー パケットを制御すると、ルーティング ループが発生する可能性が低くなります。

デフォルトでは、スプリットホライズンはすべてのインターフェイスでイネーブルになっています。

スプリットホライズンは、ルート情報が、その情報の発信元となるインターフェイスからルータによってアドバタイズされないようにします。通常、特にリンクが切断された場合には、この動作によって複数のルーティングデバイス間の通信が最適化されます。ただし、非ブロードキャスト ネットワーク（フレーム リレーや SMDS など）を使用している場合は、この動作では不十分な状況が発生する可能性があります。このような場合は、EIGRP を設定したネットワークを含め、スプリット ホライズンを無効にする必要が生じることもあります。

EIGRP プロセスの hello 間隔と保留時間の調整

hello パケットの間隔と保留時間は調整することができます。

ルーティング デバイスは、定期的に hello パケットを相互に送信して、直接接続されたネットワーク上の他のルータをダイナミックに学習します。この情報は、ネイバーを検出したり、ネイバーが到達不能または動作不能になったことを学習したりするために使用されます。デフォルトでは、hello パケットは 5 秒間隔で送信されます。

自律システム番号によって指定された特定の EIGRP ルーティングプロセスの保留時間を、指定したインターフェイスに対して設定できます。保留時間は、helloパケットでアドバタイズされ、送信元を有効と見なす時間の長さをネイバーに示します。デフォルトの保留時間は、hello 間隔の 3 倍（15 秒）です。

EIGRP プロセスのスタブ ルーティング

EIGRP スタブ ルーティング機能は、ネットワークの安定性を高め、リソース利用率を抑え、スタブ ルータ構成を簡素化します。

スタブ ルーティングは一般にハブ アンド スポーク型のネットワーク トポロジで使用されます。ハブ アンド スポーク ネットワークでは、1 つ以上のエンド（スタブ）ネットワークが 1 台のリモート ルータ（スポーク）に接続され、そのリモート ルータは 1 つ以上のディストリビューション ルータ（ハブ）に接続されています。リモート ルータは、1つ以上のディストリビューション ルータにのみ隣接しています。リモート ルータへ流れる IP トラフィックのルートは、ディストリビューション ルータ経由のルートのみです。このタイプの設定は、ディストリビューション ルータが直接 WAN に接続されている WAN トポロジで使用されるのが一般的です。ディストリビューション ルータは、さらに多くのリモート ルータに接続できます。ディストリビューション ルータが 100 台以上のリモート ルータに接続されていることも、よくあります。ハブ アンド スポーク型トポロジでは、リモート ルータがすべての非ローカル トラ

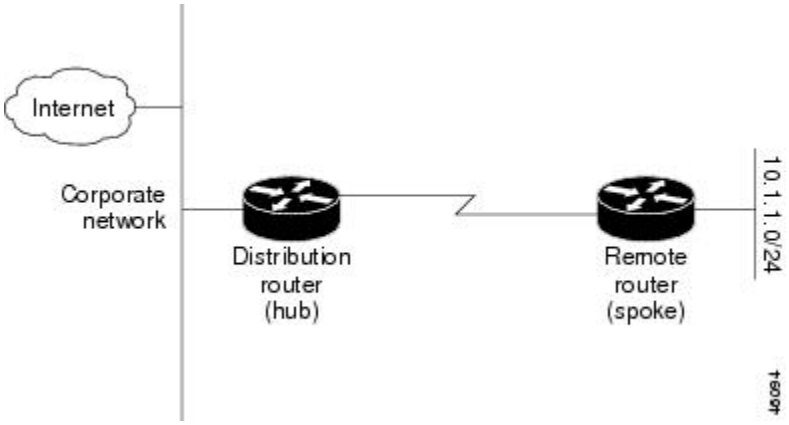
フィックをディストリビューション ルータに転送する必要があります。これにより、リモート ルータが完全なルーティング テーブルを保持する必要はなくなります。一般に、ディストリビューション ルータはデフォルト ルート以外の情報をリモート ルータに送信する必要はありません。

EIGRP スタブ ルーティング機能を使用する場合、EIGRP を使用するように、ディストリビューション ルータおよびリモート ルータを設定し、さらにリモート ルータだけをスタブとして設定する必要があります。指定されたルートのみが、リモート（スタブ）ルータから伝播されます。スタブ ルータは、サマリー、接続されているルート、再配布されたスタティック ルート、外部ルート、および内部ルートに対するクエリーすべてに、応答として「inaccessible」というメッセージを返します。スタブとして設定されているルータは、自身のスタブ ルータとしてのステータスを報告するために、特殊なピア情報パケットがすべての隣接ルータに送信されます。

スタブ ルータの状態を通知するパケットを受信した隣接ルータは、ルートについてはスタブ ルータに照会しません。また、スタブ ピアを持つルータは、そのピアについては照会しません。スタブ ルータは、ディストリビューション ルータを使用して適切なアップデートをすべてのピアに送信します。

図 *18 :* 単純なハブ アンド スポーク ネットワーク

この図は、単純なハブ アンド スポーク型設定を示しています。



ルートがリモート ルータにアドバタイズされることを、スタブ ルーティング機能自体が回避することはありません。図 18 : 単純なハブ アンド スポーク ネットワーク （346 ページ） の例では、リモート ルータはディストリビューション ルータだけを通じて企業ネットワークおよびインターネットにアクセスできます。この例では、リモート ルータが完全なルート テーブルを保有しても機能面での意味はありません。これは、企業ネットワークとインターネットへのパスは常にディストリビューション ルータを経由するためです。ルート テーブルが大きくなると、リモート ルータに必要なメモリ量が減るだけです。帯域幅とメモリは、ディストリビューション ルータのルートを集約およびフィルタリングすることによって節約できます。リモート ルータは、宛先に関係なく、ディストリビューション ルータにすべての非ローカル トラフィックを送信する必要があるため、他のネットワークから学習されたルートを受け取る必要がありません。真のスタブ ネットワークが望ましい場合、ディストリビューション ルータがリモート ルータにデフォルト ルートだけを送信するように設定する必要があります。EIGRP スタブ ルーティング機能では、ディストリビューション ルータでの集約を自動的に有効にし

ません。ほとんどの場合、ネットワーク管理者が、ディストリビューションルータにサマライズを設定する必要があります。

スタブ機能がない場合、ディストリビューションルータからリモートルータに送信されたルートがフィルタリングまたは集約された後でも、問題が発生することがあります。企業ネットワーク内でルートが失われると、EIGRP はクエリーをディストリビューション ルータに送信することがあります。ルートがサマライズされている場合でも、ディストリビューションルータが代わりにリモートルータにクエリーを送信します。WAN リンクを介したディストリビューション ルータとリモート ルータ間の通信に問題がある場合、EIGRP Stuck In Active（SIA）状態が発生し、ネットワークのどこかで不安定になる可能性があります。EIGRP スタブルーティング機能を使用することにより、ネットワーク管理者はリモートルータへクエリーが送信されないようにできます。

EIGRP プロセスのルート ポリシー オプション

ルートポリシーは、 **route-policy** キーワードと **end-policy** キーワードで囲まれた一連のステートメントと式によって構成されます。個別のコマンド（1 行に 1 つのコマンド）の集合ではなく、ルートポリシー内のステートメントには相互に関連するコンテキストがあります。そのため、個別のコマンドを各行に記すのではなく、各ポリシーまたはセットは独立した設定オブジェクトとして、1 つのユニットとして使用、入力、操作できます。

ポリシー設定の各行は論理サブユニットです。 **then** 、 **else** 、 **end-policy** キーワードの後ろには、少なくとも 1 つの新しい行を続ける必要があります。（EIGRP コンテキストの）AS パス セット、コミュニティ セット、拡張コミュニティ セット、またはプレフィックス セットを参照するパラメータ リストと名前ストリングを閉じる括弧の後には改行が必要です。ルート ポリシーまたはプレフィックスセットの定義の前に、改行を1つ以上挿入する必要があります。新しい行はポリシー式の論理ユニットの最後に記される必要があります。他の場所に記すことはできません。

これはルート ポリシーに EIGRP メトリックを設定するコマンドです。

```
RP/0/RSP0/cpu 0: router(config-rpl)# set eigrp-metric bandwidth delay reliability loading
 mtu
```

これは、ルート ポリシーに EIGRP オフセット リストの機能を提供するコマンドです。

```
RP/0/RSP0/cpu 0: router(config-rpl)# add eigrp-metric bandwidth delay reliability loading
 mtu
```

ルート ポリシーは、EIGRP ですべてのステートメントが特定の EIGRP 接続点に適用可能な場合にのみ使用することができます。次のコマンドはルート ポリシーを受け入れます。

- **default-information allowed**：match ステートメントが宛先に対して許可されます。set ステートメントは使用できません。
- **route-policy**：match ステートメントが宛先、ネクストホップ、およびタグに対して許可されます。set ステートメントが eigrp-metric および tag に対して許可されます。

- **redistribute**：match ステートメントが宛先、ネクストホップ、ソースプロトコル、タグ、およびルートタイプに対して許可されます。set ステートメントが eigrp-metric および tag に対して許可されます。

タグ設定範囲は、内部ルートの場合は 0 ～ 255、外部ルートの場合は 0 ～ 4294967295 です。

EIGRP レイヤ 3 VPN PE-CE Site-of-Origin

EIGRP MPLS と IP VPN PE-CE Site-of-Origin（SoO）機能により、マルチプロトコル ラベル スイッチング（MPLS）および IP バーチャル プライベート ネットワーク（VPN）トラフィックを、EIGRP ネットワークに対してサイト単位でフィルタリングする機能が実現しました。SoO フィルタリングはインターフェイス レベルで設定され、これを使用して MPLS および IP VPN トラフィックを管理し、複雑で複合的なネットワーク トポロジにおいて過渡的なルーティング ループが発生しないようにできます。

Site-of-Origin 拡張コミュニティでのルータの相互運用

SoO 拡張コミュニティを設定すると、この機能をサポートしているルータで、各ルートの発信元のサイトを識別できるようになります。この機能が有効になっていると、PE または CE ルータ上の EIGRP ルーティング プロセスは、受信したそれぞれのルートを SoO 拡張コミュニティに対してチェックし、次の条件に基づいてフィルタリングします。

- BGP または CE ルータから受信したルートに、受信インターフェイス上の SoO と一致する SoO 値が含まれている。
  - あるルートが、関連付けられている SoO 値とともに受信され、その値が、受信インターフェイス上で設定されている SoO 値と一致している場合、そのルートは、別の PE ルータまたはバックドア リンクから学習されたものであるため、フィルタで除外されます。この動作は、ルーティング ループを回避するために設計されています。
- CE ルータから受信したルートに、一致しない SoO 値が設定されている。
  - あるルートが、関連付けられている SoO 値とともに受信され、その値が、受信インターフェイス上で設定されている SoO 値と一致しない場合、そのルートは、BGP へ再配布されるように EIGRP トポロジ テーブルに受け入れられます。
  - ルートがすでに EIGRP トポロジ テーブルにインストールされているが、別の SoO 値が関連付けられている場合は、そのルートが BGP へ再配布されるときにトポロジ テーブルの SoO 値が使用されます。
- CE ルータから受信したルートに、SoO 値が含まれていない。
  - 受信したルートに SoO 値がない場合、そのルートは EIGRP トポロジ テーブルに受け入れられます。ルートが BGP へ再配布される前に、ネクスト ホップ CE ルータに到達するために使用されるインターフェイスの SoO 値がそのルートに付加されます。

SoO 拡張コミュニティをサポートする BGP および EIGRP ピアがこれらのルートを受信する場合には、関連付けられている SoO 値も受信します。次に、これらの値を、SoO 拡張

コミュニティをサポートしている他の BGP および EIGRP ピアへ渡します。このフィルタリングは、過渡的なルートが発信元サイトから再学習されないように、つまり過渡的なルーティング ループが発生しないようにする目的で設計されています。

BGP コストコミュニティ、EIGRP、BGP、および RIB を組み合わせることで、MPLS VPN コアを介したパスがバック ドア リンクよりも優先されるようになります。

MPLS、IP VPN、SoO の設定については、『*Cisco ASR 9000 Series Aggregation Services Router MPLS Configuration Guide*』の「*Implementing MPLS Layer 3 VPNs*」を参照してください。

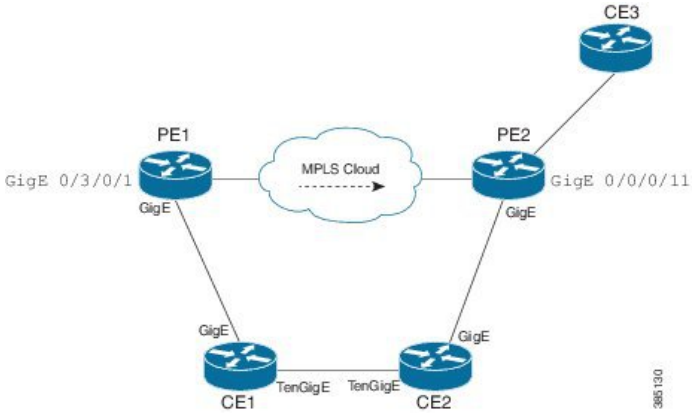
SoO 一致条件を使用したルート操作

EIGRP ネットワークの SoO 設定は、ルーティングポリシーの SoO 一致条件を使用してルートを操作するために使用できます。PE ルータの出力インターフェイスは、リモート PE ルータの SoO 設定に基づいてルートを比較および操作するために使用されます。

トポロジ

次のトポロジでは、CE1、CE2、および CE3 がカスタマーエッジルータです。PE1 と PE2 は、プロバイダーエッジルータです。デフォルトでは、CE1 は CE3 に到達するために PE1 > PE2 を使用します。CE2 を使用して CE3 に到達するように CE1 を設定するには、PE1 によってアドバタイズされるメトリックを増やす必要があります。

PE1 でのルーティングポリシーでは、SoO の一致条件を使用して、PE2 を介して CE3 から受信したルートを操作します。この機能を追加すると、PE1 はルートを CE1 にアドバタイズする間にメトリックを増やすことができます。



Configuration:

```
/*SoO tag is assigned on PE2 router*/

router(config)#interface GigabitEthernet0/0/0/11
router (config-if)#site-of-origin 33.33.33.33:33

/* A route-policy defined on PE1 */

router(config)#route-policy test
router(config-rpl)#if extcommunity soo matches-any (33.33.33.33:33) then
router(config-rpl-if)#set eigrp-metric 2121212121 333333333 245 250 1455
```

```
router(config-rpl-if)#endif
router(config-rpl)#end-policy
router (config)#commit
router (config)#

router(config)#interface GigabitEthernet0/3/0/1
router (config-if)#route-policy test out
```

確認：

```
/*A route with poor metric advertised by PE1 is installed into CE1's routing table for
SoO of site C3. */

router#show eigrp topology 6:6::1/128

IPv6-EIGRP AS(100): Topology entry for 6:6::1/128
  State is Passive, Query origin flag is 1, 1 Successor(s), FD is 15539149614794, RIB
is 4294967295 Routing Descriptor Blocks: fe80::226:98ff:fe24:5109
(GigabitEthernet0/0/0/15), from fe80::226:98ff:fe24:5109, Send flag is 0x0
Composite metric is (15539149614794/15539148304382), Route is Internal Vector metric:
Minimum bandwidth is 1000000 Kbit
Total delay is 237108596182784 picoseconds
Reliability is 245/255
Load is 250/255
Minimum MTU is 1455
Hop count is 2
Originating router is 2.2.2.2
Extended Community:
SoO:33.33.33.33:33
```

（注）

この機能は、IPv4 と IPv6 の両方に適用されます。

すべてのタイプの SoO（IP アドレス、ASN2、ASN4）がサポートされています。

キーチェーンを使用した EIGRP v4/v6 認証

キーチェーンを使用した EIGRP 認証により、インターフェイス単位で EIGRP プロトコル パケットを認証できる機能が導入されました。EIGRP ルーティング認証は、1 つ以上のインターフェイスですべての EIGRP プロトコル トラフィックを Message Digest 5（MD5）認証に基づいて認証するメカニズムを提供します。

EIGRP ルーティング認証では Cisco IOS XR ソフトウェア セキュリティ キーチェーン インフラストラクチャを使用して秘密キーが格納および取得され、インターフェイス単位で着信トラフィックと発信トラフィックが認証されます。

EIGRP ワイド メトリック計算

Cisco IOS XR Enhanced Interior Gateway Routing Protocol（EIGRP）の実装は、ワイド メトリックの計算を実行するように拡張されました。この拡張機能は、高帯域幅のインターフェイスをサポートするためのものです。

ワイド メトリックの計算機能をサポートするために、新しい EIGRP コマンドが追加され、既存の EIGRP コマンドも拡張されました。

- **metric rib-scale**：このコマンドが導入されました。
- **metric**：**picoseconds** キーワードが追加されました。
- **metric weights**：*k6* 定数のサポートが追加されました。
- **show eigrp interfaces**：このコマンドの出力は、関連するワイドメトリックに関する情報を表示するように変更されました。
- **show eigrp neighbors** ：このコマンドの出力は、関連するワイドメトリックに関する情報を表示するように変更されました。
- **show eigrp topology**：このコマンドの出力は、関連するワイドメトリックに関する情報を表示するように変更されました。
- **show protocols**：このコマンドの出力は、関連するワイドメトリックに関する情報を表示するように変更されました。

（注） ネットワーク内に IOS と IOS-XR PE デバイスの組み合わせがある場合は、IOS-XR PE デバイスで EIGRP ワイドメトリックを無効にする必要があります。これは、IOS と IOS-XR の間の L3VPN 設計でメトリックを計算する方法であるためです。

EIGRP マルチインスタンス

Enhanced Interior Gateway Routing Protocol（EIGRP）マルチインスタンス機能を使用すると、複数のプロセスインスタンスで異なるルーティングインスタンスを処理し、同じ VRF にサービスを提供することができます。各プロセスインスタンスは、その下で設定されているルーティングインスタンスを処理します。複数の EIGRP プロセスインスタンスの実装では、自律システム番号に加えて、仮想名を使用して EIGRP を設定できます。

BFD に対する EIGRP サポート

EIGRP はリンク障害を検出するために双方向フォワーディング検出（BFD）をサポートしています。

BFD は隣接する転送エンジン間でパス内で発生した障害を低いオーバーヘッドで短時間に検出します。BFD では、あらゆるメディアおよびあらゆるプロトコル レイヤでの障害検出に単一のメカニズムを使用でき、広範な検出時間とオーバーヘッドに対応できます。障害の迅速な検出が可能なため、リンクやネイバーの障害発生時にもただちに障害に対応することができます。

EIGRP の実装方法

ここでは、次のタスクの手順を示します。

（注） 設定の変更を保存するには、システムでプロンプトが表示されたら、変更を確定する必要があります。

EIGRP ルーティングの有効化

このタスクでは、EIGRP ルーティングを有効にし、EIGRP ルーティングプロセスを確立します。

始める前に

IP アドレスを設定する前に EIGRP を設定できますが、1 つ以上の IP アドレスが設定されていない場合には EIGRP ルーティングは行われません。

手順の概要

1. **configure**
2. **router eigrp** *as-number*
3. **address-family** { **ipv4** }
4. **router-id** *id*
5. **default-metric** *bandwidth delay reliability loading mtu*
6. **distance** *internal-distance external-distance*
7. **interface** *type interface-path-id*
8. **holdtime** seconds
9. **bandwidth-percent** *percent*
10. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router eigrp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router eigrp 100`	ルーティングプロセスの自律システム番号を指定して、EIGRP ルーティングプロセスを設定します。
ステップ 3	**address-family** { **ipv4** } 例：	アドレス ファミリ コンフィギュレーション モードを開始します。

	コマンドまたはアクション	目的
	`RP/0/RSP0/cpu 0: router(config-eigrp)# address-family ipv4`	
ステップ 4	**router-id** *id* 例： `RP/0/RSP0/cpu 0: router(config-eigrp)# router-id 172.20.1.1`	（任意）EIGRP プロセスのルータ ID を設定します。 （注） **router-id** コマンドを使用して、ルータ ID として明示的に一意の 32 ビット数値を指定することをお勧めします。この処理によって、インターフェイス アドレスの設定に関係なく、EIGRP が機能することが保証されます。
ステップ 5	**default-metric** *bandwidth delay reliability loading mtu* 例： `RP/0/RSP0/cpu 0: router(config-eigrp-af)# default-metric 1000 100 250 100 1500`	（任意）EIGRP プロセスのメトリックを設定します。
ステップ 6	**distance** *internal-distance external-distance* 例： `RP/0/RSP0/cpu 0: router(config-eigrp-af)# distance 80 130`	（任意）2 つのアドミニストレーティブ ディスタンス（内部ディスタンスと外部ディスタンス）を使用できるようにします。あるノードへの適切なルートになります。
ステップ 7	**interface** *type interface-path-id* 例： `RP/0/RSP0/cpu 0: router(config-eigrp-af)# interface GigabitEthernet 0/1/0/0`	EIGRP ルーティング プロトコルを実行するインターフェイスを定義します。
ステップ 8	**holdtime** seconds 例： `RP/0/RSP0/cpu 0: router(config-eigrp-af-if)# holdtime 30`	（任意）インターフェイスの保留時間を設定します。 （注） RP のフェールオーバーの際にノンストップ フォワーディングを行うには、ネイバーの数が増加するため、デフォルト値よりも大きい保留時間を設定することを推奨します。VRF 全体で 256 台のネイバーがある場合は、60 秒を推奨します。
ステップ 9	**bandwidth-percent** *percent* 例： `RP/0/RSP0/cpu 0: router(config-eigrp-af-if)# bandwidth-percent 75`	（任意）インターフェイス上で EIGRP が使用可能な帯域幅のパーセンテージを設定します。

	コマンドまたはアクション	目的
ステップ 10	**commit**	

EIGRP プロセスのルート集約の設定

この作業では、EIGRP プロセスのルート集約を設定します。

指定したインターフェイスにサマリー集約アドレスを設定できます。より具体的なルートがルーティングテーブルにある場合、EIGRPは、より具体的なすべてのルートの最小に等しいメトリックを持つインターフェイスからのサマリー アドレスをアドバタイズします。

始める前に

（注） インターフェイスからのデフォルトルート（0.0.0.0）を生成するのに、**summary-address** サマライズコマンドは使用しないでください。このコマンドを使用すると、アドミニストレーティブ ディスタンスが 5 で、ヌル 0 インターフェイスへの EIGRP サマリー デフォルト ルートが作成されます。このデフォルトルートのアドミニストレーティブディスタンスの値が小さいと、ルーティングテーブル内の他のネイバーから学習されたデフォルトルートにこのルートが置き換えられてしまうことがあります。ネイバーから学習したデフォルト ルートがサマリーのデフォルト ルートと置き換えられてしまった場合や、デフォルト ルートとしてサマリー ルートしか存在しない場合、デフォルトルートを宛先とするすべてのトラフィックはルータから発信されず、ヌル 0 インターフェイスへ送信されてドロップされてしまいます。

所定のインターフェイスからのデフォルトルートだけを送信するようにするには、**route-policy** コマンドを使用することを推奨します。

手順の概要

1. **configure**
2. **router eigrp** *as-number*
3. **address-family** { **ipv4** }
4. **route-policy** *name* **out**
5. **interface** *type interface-path-id*
6. **summary-address** *ip-address* { */ length* | *mask* } [*admin-distance*]
7. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router eigrp** *as-number* 例：	ルーティングプロセスのAS番号を指定して、EIGRP ルーティングプロセスを設定します

	コマンドまたはアクション	目的
	`RP/0/RSP0/cpu 0: router(config)# router eigrp 100`	
ステップ 3	**address-family** { **ipv4** } 例： `RP/0/RSP0/cpu 0: router(config-eigrp)# address-family ipv4`	アドレスファミリのコンフィギュレーションモードを入力します。
ステップ 4	**route-policy** *name* **out** 例： `RP/0/RSP0/cpu 0: router(config-eigrp-af)# route-policy FILTER_DEFAULT out`	EIGRP ネイバーにアドバタイズされる更新または EIGRP ネイバーから受信する更新にルーティング ポリシーを適用します。
ステップ 5	**interface** *type interface-path-id* 例： `RP/0/RSP0/cpu 0: router(config-eigrp-af)# interface GigabitEthernet 0/1/0/0`	EIGRP ルーティング プロトコルを実行するインターフェイスを定義します。
ステップ 6	**summary-address** *ip-address* { */ length* | *mask* } [*admin-distance*] 例： `RP/0/RSP0/cpu 0: router(config-eigrp-af-if)# summary-address 192.168.0.0/16 95`	指定された EIGRP インターフェイスのサマリー集約アドレスを設定します。
ステップ 7	**commit**	

EIGRP の再配布ルート

この作業では、ルートを再配布し、ルートの数に制限を設定し、ノンストップ フォワーディングのタイマーを設定する方法について説明します。

手順の概要

1. **configure**
2. **router eigrp** *as-number*
3. **address-family** { **ipv4** }
4. **redistribute** {{ **bgp** | **connected** | **isis** | **ospf** | **rip** | **static** } [*as-number*]} [**route-policy** *name*]
5. **redistribute maximum-prefix** *maximum* [*threshold*] [[**dampened**] [**reset-time** *minutes*] [**restart** *minutes*] [**restart-count** *number*] | [**warning-only**]]
6. **timers nsf route-hold** *seconds*
7. **maximum paths** *maximum*

8. **maximum-prefix** *maximum* [*threshold*] [[**dampened**] [**reset-time** *minutes*] [**restart** *minutes*] [**restart-count** *number*] | [**warning-only]]**
9. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router eigrp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router eigrp 100`	ルーティングプロセスのAS番号を指定して、EIGRPルーティングプロセスを設定します。
ステップ 3	**address-family** { **ipv4** } 例： `RP/0/RSP0/cpu 0: router(config-eigrp)# address-family ipv4`	アドレスファミリのコンフィギュレーションモードを入力します。
ステップ 4	**redistribute** {{ **bgp** | **connected** | **isis** | **ospf** | **rip** | **static** } [*as-number*]} [**route-policy** *name*] 例： `RP/0/RSP0/cpu 0: router(config-eigrp-af)# redistribute bgp 100`	指定したプロトコルと AS 番号から EIGRP プロセスにルートを再配布します。任意で、ルートポリシーを指定してEIGRPプロセスに再配布ルートを取り込むこともできます。
ステップ 5	**redistribute maximum-prefix** *maximum* [*threshold*] [[**dampened**] [**reset-time** *minutes*] [**restart** *minutes*] [**restart-count** *number*]	[**warning-only**]] 例： `RP/0/RSP0/cpu 0: router(config-eigrp-af)# redistribute maximum-prefix 5000 95 warning-only`	EIGRPプロセスに再配布されるプレフィックスの最大数を制限します。 **注意** 再起動回数のしきい値を超えると、`clear eigrp neighbors` コマンドを使用して標準のピアリング、再配布、またはその両方の再確立が必要になります。
ステップ 6	**timers nsf route-hold** *seconds* 例： `RP/0/RSP0/cpu 0: router(config-eigrp-af)# timers nsf route-hold 120`	NSF 対応 EIGRP ルータが非アクティブ ピアのルートを維持する時間を決定するタイマーを設定します。
ステップ 7	**maximum paths** *maximum* 例： `RP/0/RSP0/cpu 0: router(config-eigrp-af)# maximum paths 10`	EIGRPがサポートできる並列ルートの最大数を制御します。

	コマンドまたはアクション	目的
ステップ 8	**maximum-prefix** *maximum* [*threshold*] [[**dampened**] [**reset-time** *minutes*] [**restart** *minutes*] [**restart-count** *number*]	[**warning-only]]** 例： `RP/0/RSP0/cpu 0: router(config-eigrp-af)# maximum-prefix 50000`	EIGRP がアドレスファミリで受け入れるプレフィックス数を制限します。
ステップ 9	**commit**	

ルート ポリシーの作成と EIGRP プロセスへのアタッチ

ここでは、ルート ポリシーを定義し、EIGRP プロセスにアタッチする方法について説明します。

ルートポリシーの定義は、**route-policy** コマンドと *name* 引数、その後に続く一連のオプションのポリシーステートメントで構成され、**end-policy** コマンドで閉じられます。

ルート ポリシーはルーティング プロトコルのルートに適用されてはじめて役に立ちます。

手順の概要

1. **configure**
2. **route-policy** *name*
3. **set eigrp-metric** *bandwidth delay reliability load mtu*
4. end-policy
5. **commit**
6. **configure**
7. **router eigrp** *as-number*
8. **address-family** { **ipv4** }
9. **route-policy** *route-policy-name* { **in** | **out** }
10. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**route-policy** *name* 例： `RP/0/RSP0/cpu 0: router(config)# route-policy IN-IPv4`	ルート ポリシーを定義して、ルート ポリシー コンフィギュレーション モードを開始します。
ステップ 3	**set eigrp-metric** *bandwidth delay reliability load mtu*	（任意）EIGRP メトリック属性を設定します。

	コマンドまたはアクション	目的
	例： `RP/0/RSP0/cpu 0: router(config-rpl)# set eigrp metric 42 100 200 100 1200`	
ステップ 4	end-policy 例： `RP/0/RSP0/cpu 0: router(config-rpl)# end-policy`	ルートポリシーの定義を終了して、ルートポリシーコンフィギュレーション モードを終了します。
ステップ 5	**commit**	
ステップ 6	**configure**	
ステップ 7	**router eigrp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router eigrp 100`	ルーティングプロセスの自律システム番号を指定して、EIGRP ルーティングプロセスを設定します。
ステップ 8	**address-family** { **ipv4** } 例： `RP/0/RSP0/cpu 0: router(config-eigrp)# address-family ipv4`	アドレス ファミリ コンフィギュレーション モードを開始します。
ステップ 9	**route-policy** *route-policy-name* { **in** | **out** } 例： `RP/0/RSP0/cpu 0: router(config-eigrp-af)# route-policy IN-IPv4 in`	EIGRP ネイバーにアドバタイズされる更新または EIGRP ネイバーから受信する更新にルーティングポリシーを適用します。
ステップ 10	**commit**	

EIGRP プロセスのスタブ ルーティングの設定

この作業では、ディストリビューション ルータおよびリモート ルータでスタブ ルーティングに EIGRP プロセスを使用するように設定します。

始める前に

（注） EIGRP スタブ ルーティングは、リモート ルータでのみ使用してください。スタブ ルータは、コア中継トラフィックが通過しないネットワーク コアまたはディストリビューション レイヤに接続されたルータとして定義されます。スタブ ルータがディストリビューション ルータ以外の EIGRP ネイバーを持つことはできません。この制約事項を無視すると、望ましくない動作が発生します。

手順の概要

1. **configure**
2. **router eigrp** *as-number*
3. **address-family** { **ipv4** }
4. **stub** [**receive-only** | {[**connected**] [**redistributed**] [**static**] [**summary**]}]
5. **commit**
6. **show eigrp** [**ipv4**] **neighbors** [*as-number*] [**detail**] [*type interface-path-id* | **static**]

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router eigrp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router eigrp 100`	ルーティングプロセスの自律システム番号を指定して、EIGRP ルーティングプロセスを設定します。
ステップ 3	**address-family** { **ipv4** } 例： `RP/0/RSP0/cpu 0: router(config-eigrp)# address-family ipv4`	アドレス ファミリ コンフィギュレーション モードを開始します。
ステップ 4	**stub** [**receive-only**	{[**connected**] [**redistributed**] [**static**] [**summary**]}] 例： `RP/0/RSP0/cpu 0: router(config-eigrp-af)# stub receive-only`	ルータを EIGRP のスタブとして設定します。
ステップ 5	**commit**	
ステップ 6	**show eigrp** [**ipv4**] **neighbors** [*as-number*] [**detail**] [*type interface-path-id*	**static**] 例：	EIGRP のスタブ ルータとしてリモート ルータが設定されていることを確認します。 出力の最後の行は、リモート ルータまたはスポーク ルータのスタブ ステータスを示します。

	コマンドまたはアクション	目的
	`RP/0/RSP0/cpu 0: router# show eigrp neighbors detail`	

PE-CE プロトコルとしての EIGRP の設定

プロバイダーエッジ（PE）に EIGRP を設定して、EIGRP を使用してプロバイダーエッジとカスタマーエッジ間の（PE-CE）通信を確立するには、次のタスクを実行します。

手順の概要

1. **configure**
2. **router eigrp** *as-number*
3. **vrf** *vrf-name*
4. **address-family** { **ipv4** }
5. **router-id** *router-id*
6. **autonomous-system** *as-number*
7. **redistribute** {{ **bgp** | **connected** | **isis** | **ospf** | **ospfv3** | **rip** | **static** } [*as-number* | *instance-name*]} [**route-policy** *name*]
8. **interface** *type interface-path-id*
9. **site-of-origin** { *as-number:number* | *ip-address* **:** *number* }
10. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router eigrp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router eigrp 100`	ルーティングプロセスの自律システム番号を指定して、EIGRP ルーティングプロセスを設定します。
ステップ 3	**vrf** *vrf-name* 例： `RP/0/RSP0/cpu 0: router(config-eigrp)# vrf vrf_A`	VPN ルーティングおよび転送（VRF）インスタンスを設定します。
ステップ 4	**address-family** { **ipv4** } 例： `RP/0/RSP0/cpu 0: router(config-eigrp-vrf)# address-family ipv4`	VRF アドレス ファミリ コンフィギュレーション モードを開始します。

	コマンドまたはアクション	目的
ステップ 5	**router-id** *router-id* 例： `RP/0/RSP0/cpu 0: router(config-eigrp-vrf-af)# router-id 33`	EIGRP プロセスのルータ ID を設定します。
ステップ 6	**autonomous-system** *as-number* 例： `RP/0/RSP0/cpu 0: router(config-eigrp-vrf-af)# autonomous-system 2`	EIGRP ルーティングプロセスを VRF インスタンスで実行するよう設定します。 （注） VRF 設定において、VRF インターフェイスを起動するように自律システムを設定する必要があります。
ステップ 7	**redistribute** { { **bgp** | **connected** | **isis** | **ospf** | **ospfv3** | **rip** | **static** } [*as-number* | *instance-name*] } [**route-policy** *name*] 例： `RP/0/RSP0/cpu 0: router(config-eigrp-vrf-af)# redistribute bgp 100`	1 つのルーティング ドメインから EIGRP にルートを注入します。
ステップ 8	**interface** *type interface-path-id* 例： `RP/0/RSP0/cpu 0: router(config-eigrp-vrf-af)# interface gigabitEthernet 0/1/5/0`	ルーティング プロトコルが実行される EIGRP のインターフェイスを設定します。
ステップ 9	**site-of-origin** { *as-number:number* | *ip-address* **:** *number* } 例： `RP/0/RSP0/cpu 0: router(config-eigrp-vrf-af-if)# site-of-origin 3:4`	EIGRP インターフェイスに site-of-origin（SoO）フィルタリングを設定します。
ステップ 10	**commit**	

BGP ルートの EIGRP への再配布

BGP ルートを EIGRP に再配布するには、次のタスクを実行します。

通常 EIGRP ルートは、ルートに付加された拡張コミュニティ情報を伴って BGP に再配布されます。BGP は VPN バックボーン上で、BGP 拡張コミュニティ属性内にエンコードされた EIGRP 固有情報と共にルートを伝送します。ピアリング カスタマー サイトがルートを受信した後に、EIGRP は BGP ルートを再配布し、BGP 拡張コミュニティ情報を抽出して元のカスタマー サイトと同じルートを再構築します。

BGP ルートを EIGRP に再配布する場合、受信側のプロバイダーエッジ（PE）EIGRP ルータは BGP 拡張コミュニティ情報を探します。この情報を受信した場合、元の EIGRP ルートを再構築するためにその情報が使用されます。情報がない場合、EIGRP は設定されたデフォルトのメトリック値を使用します。

メトリック値が BGP 拡張コミュニティから得られず、デフォルトのメトリックも設定されていない場合には、PE EIGRP によるカスタマーエッジ（CE）ルータへのルートのアドバタイズは行われません。BGP が BGP に再配布されるときには、メトリックが拡張コミュニティとして BGP プレフィックスに追加されない場合があります（EIGRP が相手ルータで実行されていない場合など）。この場合、EIGRP は「no-metrics」オプションを伴って BGP に再配布されます。

手順の概要

1. **configure**
2. **router eigrp** *as-number*
3. **vrf** *vrf-name*
4. **address-family** { **ipv4** }
5. **redistribute** {{ **bgp** | **connected** | **isis** | **ospf** | **ospfv3** | **rip** | **static** } [*as-number* | *instance-name*]} [**route-policy** *name*]
6. **route-policy** *route-policy-name* { **in** | **out** }
7. **default-metric** *bandwidth delay reliability loading mtu*
8. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router eigrp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router eigrp 100`	ルーティングプロセスの自律システム番号を指定して、EIGRP ルーティングプロセスを設定します。
ステップ 3	**vrf** *vrf-name* 例： `RP/0/RSP0/cpu 0: router(config-eigrp)# router eigrp 100`	VRF インスタンスを設定します。
ステップ 4	**address-family** { **ipv4** } 例： `RP/0/RSP0/cpu 0: router(config-eigrp-vrf)# address-family ipv4`	VRF アドレスファミリ コンフィギュレーション モードを開始します。

	コマンドまたはアクション	目的
ステップ 5	**redistribute** {{ **bgp** | **connected** | **isis** | **ospf** | **ospfv3** | **rip** | **static** } [*as-number* | *instance-name*]} [**route-policy** *name*] 例： `RP/0/RSP0/cpu 0: router(config-eigrp-vrf-af)# redistribute bgp 100`	1 つのルーティング ドメインから EIGRP にルートを注入します。
ステップ 6	**route-policy** *route-policy-name* { **in** | **out** } 例： `RP/0/RSP0/cpu 0: router(config-eigrp-vrf-af)# route-policy policy_A in`	EIGRP ネイバーにアドバタイズされる更新または EIGRP ネイバーから受信する更新にルーティング ポリシーを適用します。
ステップ 7	**default-metric** *bandwidth delay reliability loading mtu* 例： `RP/0/RSP0/cpu 0: router(config-eigrp-vrf-af)# default-metric 1000 100 250 100 1500`	EIGRP のメトリックを設定します。
ステップ 8	**commit**	

EIGRP ルーティングのモニタリング

このセクションのコマンドは、隣接ルータとの隣接関係における変更をロギングし、ルーティング システムの安定性を監視して、問題を検出しやすくするために使用します。

手順の概要

1. **configure**
2. **router eigrp** *as-number*
3. **address-family** [**ipv4**]
4. **log-neighbor-changes**
5. **log-neighbor-warnings**
6. **commit**
7. **clear eigrp** [*as-number*] [**vrf** { *vrf* \| **all** }] [**ipv4**] **neighbors** [*ip-address* \| *type interface-path-id*]
8. **clear eigrp** [*as-number*] [**vrf** { *vrf* \| **all** }] [**ipv4**] **topology** [*prefix mask*] [*prefix* / *length*]
9. **show eigrp** [*as-number*] [**vrf** { *vrf* \| **all** }] [**ipv4**] **accounting**
10. **show eigrp** [*as-number*] [**vrf** { *vrf* \| **all** }][**ipv4**] **interfaces** [*type interface-path-id*] [**detail**]
11. **show eigrp** [*as-number*] [**vrf** { *vrf* \| **all** }] [**ipv4**] **neighbors** [**detail**] [*type interface-path-id* \| **static**]
12. **show protocols eigrp** [**vrf** *vrf-name*]

13. **show eigrp** [*as-number*] [**vrf** { *vrf* | **all** }] [**ipv4**] **topology** [*ip-address mask*] [**active** | **all-links** | **detail-links** | **pending** | **summary** | **zero-successors**]
14. **show eigrp** [*as-number*] [**vrf** { *vrf* | **all** }] [**ipv4**] **traffic**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router eigrp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router eigrp 100`	ルーティングプロセスの自律システム番号を指定して、EIGRP ルーティングプロセスを設定します。
ステップ 3	**address-family** [**ipv4**] 例： `RP/0/RSP0/cpu 0: router(config-eigrp)# address-family ipv4`	アドレス ファミリ コンフィギュレーション モードを開始します。
ステップ 4	**log-neighbor-changes** 例： `RP/0/RSP0/cpu 0: router(config-eigrp-af)# log-neighbor-changes`	EIGRP ネイバーとの隣接関係に関する変更のロギングを有効にします。
ステップ 5	**log-neighbor-warnings** 例： `RP/0/RSP0/cpu 0: router(config-eigrp-af)# log-neighbor-warnings`	EIGRP ネイバーの警告メッセージのロギングを有効にします。
ステップ 6	**commit**	
ステップ 7	**clear eigrp** [*as-number*] [**vrf** { *vrf*	**all** }] [**ipv4**] **neighbors** [*ip-address*	*type interface-path-id*] 例： `RP/0/RSP0/cpu 0: routerr# clear eigrp 20 neighbors GigabitEthernet 0/1/0/0`	EIGRP および VPN ネイバーエントリを適切なテーブルから削除します。
ステップ 8	**clear eigrp** [*as-number*] [**vrf** { *vrf*	**all** }] [**ipv4**] **topology** [*prefix mask*] [*prefix / length*] 例： `RP/0/RSP0/cpu 0: router# clear eigrp topology`	EIGRP および VRF トポロジエントリを適切なタブから削除します。

	コマンドまたはアクション	目的
ステップ 9	**show eigrp** [*as-number*] [**vrf** { *vrf* | **all** }] [**ipv4**] **accounting** 例： `RP/0/RSP0/cpu 0: router# show eigrp vrf all accounting`	EIGRP プロセスのプレフィックス アカウンティング情報を表示します。
ステップ 10	**show eigrp** [*as-number*] [**vrf** { *vrf* | **all** }][**ipv4**] **interfaces** [*type interface-path-id*] [**detail**] 例： `RP/0/RSP0/cpu 0: router# show eigrp interfaces detail`	EIGRP に設定されているインターフェイスに関する情報を表示します。
ステップ 11	**show eigrp** [*as-number*] [**vrf** { *vrf* | **all** }] [**ipv4**] **neighbors** [**detail**] [*type interface-path-id* | **static**] 例： `RP/0/RSP0/cpu 0: router# show eigrp neighbors 20 detail static`	EIGRP によって検出されたネイバーを表示します。
ステップ 12	**show protocols eigrp** [**vrf** *vrf-name*] 例： `RP/0/RSP0/cpu 0: router# show protocols eigrp`	EIGRP プロセスの設定に関する情報を表示します。
ステップ 13	**show eigrp** [*as-number*] [**vrf** { *vrf* | **all** }] [**ipv4**] **topology** [*ip-address mask*] [**active** | **all-links** | **detail-links** | **pending** | **summary** | **zero-successors**] 例： `RP/0/RSP0/cpu 0: router# show eigrp topology 10.0.0.1 253.254.255.255 summary`	EIGRP トポロジ テーブル内のエントリを表示します。
ステップ 14	**show eigrp** [*as-number*] [**vrf** { *vrf* | **all** }] [**ipv4**] **traffic** 例： `RP/0/RSP0/cpu 0: router# show eigrp traffic`	送受信された EIGRP パケットの数を表示します。

EIGRP 認証キーチェーンの設定

EIGRP インターフェイスで認証キーチェーンを設定するには、次のタスクを実行します。

デフォルトの VRF での IPv4/IPv6 インターフェイスの認証キーチェーンの設定

デフォルトの VRF で IPv4/IPv6 インターフェイスの認証キーチェーンを設定するには、次のタスクを実行します。

手順の概要

1. **configure**
2. **router eigrp** *as-number*
3. **address-family** { **ipv4** | **ipv6** }
4. **interface** *type interface-path-id*
5. **authentication keychain** *keychain-name*
6. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router eigrp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router eigrp 100`	ルーティングプロセスの自律システム番号を指定して、EIGRP ルーティングプロセスを設定します。
ステップ 3	**address-family** { **ipv4** | **ipv6** } 例： `RP/0/RSP0/cpu 0: router(config-eigrp)# address-family ipv4`	VRF アドレスファミリ コンフィギュレーション モードを開始します。
ステップ 4	**interface** *type interface-path-id* 例： `RP/0/RSP0/cpu 0: router(config-eigrp-af)# interface gigabitEthernet 0/1/5/0`	ルーティング プロトコルが実行される EIGRP のインターフェイスを設定します。
ステップ 5	**authentication keychain** *keychain-name* 例： `RP/0/RSP0/cpu 0: router(config-eigrp-af-if)# authentication keychain`	MD5 アルゴリズムに基づいて、インターフェイス上のすべての EIGRP プロトコル トラフィックを認証します。
ステップ 6	**commit**	

デフォルト以外の VRF での IPv4/IPv6 インターフェイスの認証キーチェーンの設定

デフォルト以外の VRF で IPv4/IPv6 インターフェイスの認証キーチェーンを設定するには、次のタスクを実行します。

手順の概要

1. **configure**
2. **router eigrp** *as-number*
3. **vrf** *vrf-name*
4. **address-family** { **ipv4** | **ipv6** }
5. **interface** *type interface-path-id*
6. **authentication keychain** *keychain-name*
7. **commit**

手順の詳細

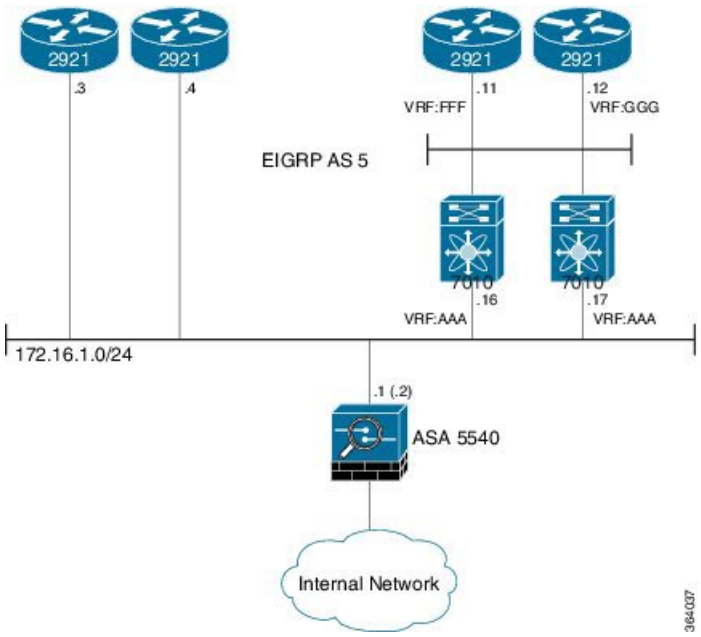
	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router eigrp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router eigrp 100`	ルーティングプロセスの自律システム番号を指定して、EIGRP ルーティングプロセスを設定します。
ステップ 3	**vrf** *vrf-name* 例： `RP/0/RSP0/cpu 0: router(config-eigrp)# vrf vrf1`	VRF インスタンスを作成し、VRF コンフィギュレーション モードを開始します。
ステップ 4	**address-family** { **ipv4** | **ipv6** } 例： `RP/0/RSP0/cpu 0: router(config-eigrp-vrf)# address-family ipv4`	VRF アドレスファミリ コンフィギュレーション モードを開始します。
ステップ 5	**interface** *type interface-path-id* 例： `RP/0/RSP0/cpu 0: router(config-eigrp-vrf-af)# interface gigabitEthernet 0/1/5/0`	EIGRP が実行されるインターフェイスを設定します。
ステップ 6	**authentication keychain** *keychain-name* 例： `RP/0/RSP0/cpu 0: router(config-eigrp-vrf-af-if)# authentication keychain`	MD5 アルゴリズムに基づいて、インターフェイス上のすべての EIGRP プロトコル トラフィックを認証します。
ステップ 7	**commit**	

ユニキャストネイバーの設定

EIGRP は通常、ルーティングの更新をブロードキャストまたはマルチキャストします。セキュリティ上の理由から、EIGRP ルーティングプロセスで静的ネイバーの設定を選択して、EIGRP がユニキャストを使用して強制的に指定されたネイバーと通信させることができます。特定のインターフェイスを介した静的ネイバー関係を指定すると、EIGRP は指定したインターフェイス上でのマルチキャスト EIGRP パケットの処理を無効にします。これにより、EIGRP は、EIGRP ルーティングの処理中に定義された静的ネイバーをも打つインターフェイス上で受信したマルチキャスト EIGRP トラフィックの送信も処理も行わなくなります。

ネイバーが隣接していない場合、EIGRP 情報の交換に通常の EIGRP ピアリングメカニズムは使用できません。このタイプのネットワークをサポートするため、EIGRP は neighbor コマンドを提供します。これにより、リモートネイバーが設定され、ユニキャストパケット伝送によりセッションが確立されます。ただし、ネットワーククラウドを介して EIGRP 情報を交換する必要があるフォワーダの数が増えると、ユニキャスト EIGRP ネイバーの定義が管理しにくくなることがあります。各ネイバーは手動で設定される必要があるので、運用コストが増加します。これらのトポロジの展開、設定管理の容易化、運用コストの削減を促進するために、動的ネイバー機能は、リモートユニキャスト（「リモートネイバー」という）の動的な検出をサポートします。リモートネイバーがサポートされているため、EIGRP を 1 つ以上のリモートネイバーとピアリングさせることができます。この情報はデバイスを設定する時点では認識されている必要がないため、設定管理が軽減されます。

次に示すトポロジでは、ASA がハブとして動作し、他のルータ（2921、7010）はスポークとして機能します。2921 と 7010 は互いにピアリングせず、けっしてパケット（データトラフィック）がこのパス内でルーティング（ASA > 2921.3 > 2921.4）されないようにする必要があります。このタイプのネットワークをサポートするために、EIGRP では、静的ネイバーを設定し、ユニキャストパケット伝送を使用してセッションを確立することができます。そのため、このトポロジでは、2921 と 7010 は neighbor コマンドを使用して ASA とピアリングし、ASA はリモートネイバーを動的に検出するように設定されます。



リモート ネイバー セッション ポリシー

リモート ユニキャスト リッスンまたはリモート マルチキャスト グループ ネイバー設定を使用しているときは、EIGRP ネイバーの IP アドレスが事前に定義されておらず、ネイバーは何ホップも離れた場所にある可能性があります。この設定のデバイスは、有効な HELLO パケットを送信するデバイスとピア関係を確立できます。セキュリティ上の理由から、このオープンな側面はポリシー機能を有効なデバイスへのピアリングに限定し、リソース消費を抑えるため、ネイバー数を制限する必要があります。この機能は、次の手動で設定されたパラメータを使用して実行され、即座に有効になります。

- **ネイバー フィルタ リスト**

  remote-neighbors コマンドで使用できる、オプションの allow-list キーワードにより、アクセスリスト（アクセスコントロールリスト）を使用して、EIGRP ネイバー接続が受け入れられる可能性のあるリモート IP アドレスを指定できます。allow-list キーワードを使用しないと、すべての IP アドレス（permit any）が受け入れられます。アクセスコントロールリスト（ACL）は、次の条件の IPv4 または IPv6 IP アドレスの範囲を定義します。

  - アクセスリストの IP アドレスに一致する送信元 IP アドレスを持つネイバーは、ユーザ設定に基づいて許可（または拒否）されます。
  - allow-list キーワードが指定されないと、どの IP アドレスも許可されます（permit any）。
  - allow-list キーワードは、リモート マルチキャスト グループおよびユニキャスト リッスン ネイバーに対してのみサポートされます。静的、リモート静的、またはローカル ネイバーでは使用できません。
  - 指定したアクセスリストに一致しない EIGRP 入力パケットは拒否されます。

- **最大リモートネイバー**

remote-neighbors コマンドで使用可能な、オプションの max-neighbors キーワードにより、EIGRP がリモートネイバー設定を使用して作成できるリモートネイバーの最大数を指定できます。設定に対しリモートネイバーの最大数が作成されたら、EIGRP はその構成のすべての後続の接続試行を拒否します。このオプションは、デバイスのリソースに負荷をかけようとして多くのリモートネイバーを作成しようとする DoS 攻撃からの保護を支援します。max-neighbors 設定オプションには、次の条件があります。

- このオプションは、マルチキャスト グループまたはユニキャスト リッスン ネイバーに対してのみサポートされます。ローカル、静的、またはリモート静的ネイバーでは使用できません。
- デフォルトの最大値はありません。リモート ネイバーの最大数を指定しないと、リモート ネイバーの数は、使用可能なメモリと帯域幅により制限されます。
- リモートネイバーの最大数を現在のセッション数以下に減らすと、ネイバー数が新しい制限に達するまで、ネイバーはドロップされます（順不同）。

- **ネイバーのフィルタリストと最大リモートネイバー数の設定の変更**

  allow-list または max-neighbors 設定が変更されたときに、新しい設定では許可されていないすべての既存のリモート EIGRP セッションは自動的かつ即座に削除されます。新しい設定によって引き続き許可されている既存のネイバーは影響を受けません。

ネイバーの用語について

ネイバー タイプについて説明する際には、次の用語が使用されます。

- **ローカル ネイバー**：共有サブネット（または共通サブネット）に隣接し、パケット交換用のリンクローカル マルチキャスト アドレスを使用するネイバー。これは、EIGRP のネイバーのデフォルトタイプです。
- **静的ネイバー**：通信にユニキャストを使用し、1 つのホップ先で、共通サブネット上にあり、その IP アドレスが neighborip-address コマンドを使用して指定されたネイバー。
- **リモート ネイバー**：リモートの静的ネイバーを含む、複数のホップ先のネイバー。
- **リモートグループ**：複数のホップ先にあり、neighbor コマンドを使用して手動で設定されたアドレスを持たず、パケット交換にマルチキャスト グループ アドレスを使用するネイバー。
- **リモート静的ネイバー**：通信にユニキャストを使用し、複数のホップ先にあり、IPアドレスが neighborip-address コマンドを使用して指定されたネイバー。
- **リモート ユニキャスト リッスン（または単にユニキャストリッスン）**：通信にユニキャストを使用し、複数のホップ先にあり、IP アドレスが neighbor ip-address コマンドを使用して設定されていないネイバー。
- **リモートダイナミック**：複数のホップ先にあり、そのアドレスが neighbor ip-address コマンドで設定されていないネイバー。つまり、リモート マルチキャスト グループまたはリモートユニキャストリッスンネイバーであるが、リモート静的ネイバーではありません。

リモート ユニキャスト リッスン（ポイントツーポイント）ネイバー

複数のリモート ネイバーが 1 つのハブとピアリングする設定（ポイントツーポイント）の場合、remote-neighbors コマンドを使用して、ハブをリモート ユニキャスト リッスン ピアリング

用に設定でき、ハブのリモート ネイバーの IP アドレスを手動で設定する必要なく、リモートネイバーはハブとピアリングできます。このコマンドを使用して設定すると、ハブデバイスは次を実行します。

- すべてのユニキャスト伝送の送信元の IP アドレスとしてインターフェイス IP アドレスを使用します。この IP アドレスはルーティング可能なものでなければなりません。
- ハブとピアリングするネイバーは、neighbor ip-address loopback loopback-interface-number remote maximum-hops コマンドを使用して設定する必要があります。この ip-address はローカル デバイス インターフェイスのユニキャストアドレスです。
- remote-neighbor コマンドで指定されたインターフェイスのユニキャスト HELLO パケットをリッスンします。
- ユニキャスト HELLO パケットが allow-list キーワードを使用して設定された IP アドレス範囲にある場合は受け入れ、または許可リストが定義されていなければ、すべてのユニキャスト HELLO パケットを受け入れます。
- ユニキャスト HELLO パケットも送信していて、ユニキャスト許可リストにより許可されている（または許可リストが定義されていなければ、すべてのネイバー）、すべてのネイバーからのマルチキャスト HELLO パケットを拒否します。
- ネイバー関係が確立されると、パケット転送のためのリモート ネイバーの IP アドレスを使用して、通常のネイバーの確立を開始します。
- インターフェイス上に remote-neighbor コマンドが設定されている場合、ルータは、そのインターフェイスにネイバーが 1 つ以上ある場合で、HELLO を受信していたネイバーに対してのみ、HELLO の送信をそのインターフェイス上で開始します。
- インターフェイス上では、ダイナミックネイバーがすでに存在しており、リモートネイバーユニキャストリッスンが設定されている場合は、既存のネイバーの関係が切断され、それ以降はユニキャストネイバーの関係のみが許可されます。

リモートネイバーの制約事項

1 つのユニキャストアドレスは、特定のアドレスファミリに対して 1 つのリモートの静的ネイバーにのみ設定できます。別のインターフェイスで同じユニキャストアドレスを使用して 2 番目のリモートの静的ネイバーは設定できません。異なるアドレスファミリでのリモートネイバーの EIGRP 設定は無制限です。

1 つのインターフェイスが、1 つの single unicast-listen remote-neighbors コマンドと、任意の数の静的ネイバーとリモートの静的ネイバー（それぞれ異なるユニキャストを使用）を使用して 1 つのアドレスファミリに設定できます。

リモートネイバー設定の継承と優先順位

neighbor <*address*> コマンドまたは neighbor <*address*> remote コマンドを使用して設定した静的ネイバーは、remote-neighbors コマンドの結果として作成されたリモート ネイバーよりも優先されます。着信ユニキャスト EIGRP 接続のリモートアドレスが、静的ネイバーと、リモート ユニキャスト リッスン ネイバーのアクセスリストの両方に一致する場合、静的ネイバーが使用され、リモート ユニキャスト リッスン ネイバーは作成されません。同じリモートアドレスのリモート ネイバーがすでに存在する間に新しい静的ネイバーを設定すると、EIGRP は自動的にリモート ユニキャスト リッスン ネイバーを削除します。

リモート ユニキャスト ネイバーの設定方法

EIGRP ユニキャストネイバーを設定する場合は、同じ自律システム内で動作する EIGRP ルーティングプロセスのネイバー関係の両端（ハブとスポーク）に neighbor ステートメントが必要です。

始める前に

ユニキャストリッスンモードを使用しているときに、IP 接続（到達可能性）がリモートピアリングを実行する必要があるデバイス間に存在していることを確認します。

手順の概要

1. **configure**
2. **router eigrp AS Number**
3. **address-family { ipv4 | ipv6 }**
4. **interfacetype interface-path-id**
5. **remote-neighbor unicast-listen**{[**allow-list**route policy name][**max-neighbors**maximum remote peers]}
6. **commit**
7. **sh run router eigrp**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router eigrp AS Number** 例： `RP/0/RSP0/CPU0:HUB(config)#router eigrp 100`	EIGRP ルータインスタンスを有効にします。
ステップ 3	**address-family { ipv4 | ipv6 }** 例： `RP/0/RSP0/CPU0:HUB(config-eigrp)#address-family ipv6`	IPv4 または IPv6 のいずれかのアドレス ファミリ ユニキャストを指定し、アドレス ファミリのコンフィギュレーション サブモードを開始します。
ステップ 4	**interfacetype interface-path-id** 例： `RP/0/RSP0/CPU0:HUB(config-eigrp-af)#int g0/0/0/3`	インターフェイスを設定して、インターフェイス コンフィギュレーション モードを開始します。
ステップ 5	**remote-neighbor unicast-listen**{[**allow-list**route policy name][**max-neighbors**maximum remote peers]} 例： `RP/0/RSP0/CPU0:HUB(config-eigrp-af-if)# remote-neighbor unicast-listen`	リモートネイバーがすべてのリモート IP アドレスからのインバウンド接続を受け入れるようにする EGRP プロセスを設定します。 • EIGRP ネイバー接続が受け入れられる可能性のあるリモート IP アドレスを指定するアクセスリスト（アクセスコントロールリスト）を使用す

	コマンドまたはアクション	目的
		るための **allow-list** キーワード。allow-list キーワードを使用しないと、すべてのIPアドレスが受け入れられます。 • リモート ネイバーの最大数を指定する **max-neighbors** キーワード。数を指定しないと、リモート ネイバーの最大数は、使用可能なメモリと帯域幅により制限されます。
ステップ 6	**commit**	
ステップ 7	**sh run router eigrp**	

EIGRP リモート ユニキャスト ネイバーの設定

次に、隣接関係に関与する両方のデバイス（ハブとスポーク）を設定する例を示します。

```
RP/0/RSP0/CPU0:HUB(config)#router eigrp 100
RP/0/RSP0/CPU0:HUB(config-eigrp)#address-family ipv4
RP/0/RSP0/CPU0:HUB(config-eigrp-af)#int g0/0/0/3
RP/0/RSP0/CPU0:HUB(config-eigrp-af-if)#exit
RP/0/RSP0/CPU0:HUB(config-eigrp-af)#interface gigabitEthernet 0/0/0/3
RP/0/RSP0/CPU0:HUB(config-eigrp-af-if)#remote-neighbor unicast-listen
RP/0/RSP0/CPU0:HUB(config-eigrp-af-if)#commit

RP/0/RSP0/CPU0:spoke(config)#router eigrp 100
RP/0/RSP0/CPU0:spoke(config-eigrp)#address-family ipv4
RP/0/RSP0/CPU0:spoke(config-eigrp-af)#interface g0/0/0/3
RP/0/RSP0/CPU0:spoke(config-eigrp-af-if)#neighbor 21.21.21.1 remote 10
RP/0/RSP0/CPU0:spoke(config-eigrp-af-if)#commit

RP/0/RSP0/CPU0:spoke#sh run router eigrp
Fri Aug  8 08:47:48.556 UTC
router eigrp 100
address-family ipv4
interface GigabitEthernet0/0/0/3
neighbor 21.21.21.1  remote 10! !!
```

EIGRP の実装の設定例

ここでは、次の設定例について説明します。

基本的な EIGRP 実装の設定：例

次に、EIGRP と着信ルートをフィルタするポリシーを設定する例を示します。これは、ネイバーが 1 台だけであるものの、接続された他のサブネットをアドバタイズするルータの一般的な設定です。

```
router eigrp 144
  address-family ipv4
    metric maximum-hops 20
    router-id 10.10.9.4
    route-policy GLOBAL_FILTER_POLICY in
    log-neighbor-changes
    log-neighbor-warnings
    interface Loopback0
    !
    interface GigabitEthernet 0/2/0/0
      passive-interface
    !
    interface GigabitEthernet 0/6/0/0
      hello-interval 8
      hold-time 30
      summary-address 10.0.0.0 255.255.0.0
      !
```

EIGRP スタブ動作の設定：例

次に、EIGRP スタブを設定する例を示します。スタブ動作では、接続ルート、サマリー ルート、およびスタティック ルートだけをネイバーにアドバタイズできます。

```
router eigrp 200
  address-family ipv4
    stub connected static summary
    router-id 172.16.82.22
    log-neighbor-changes
    log-neighbor-warnings
    redistribute connected route-policy CONN_POLICY
    interface GigabitEthernet0/6/0/0
      passive-interface
      neighbor 10.0.0.31
    !
    interface GigabitEthernet0/6/0/1
      passive-interface
      neighbor 10.0.1.21
      !
    !
  !
```

プレフィックス制限のある EIGRP PE-CE 構成の設定：例

次に、PE-CE プロトコルとして動作するように PE ルータ上で EIGRP を設定する例を示します。この設定は VRF CUSTOMER_1 の下にあります。最大プレフィックスは一般的に、1 組のカスタマールートによってEIGRPプロセスが過負荷にならないようにするために設定します。

```
router eigrp 500
  vrf CUSTOMER_1
    address-family ipv4
      timers nsf route-hold 300
      router-id 172.16.6.11
      maximum-prefix 450 70
      default-metric 200000 10000 195 10 1500
      log-neighbor-changes
```

```
    log-neighbor-warnings
    redistribute maximum-prefix 350 70
    redistribute bgp 1.65500 route-policy SITE_1_POLICY
    interface GigabitEthernet 0/4/0/5
    neighbor 10.22.1.1
    !
  !
!
```

EIGRP 認証キーチェーンの設定：例

次に、デフォルト以外の VRF で IPv4 インターフェイスの認証キーチェーンを設定する例を示します。

```
config
router eigrp 100
vrf vrf1
address-family ipv4
interface POS 0/1/0/0
authentication keychain key1
```

次に、デフォルトの VRF で IPv6 インターフェイスの認証キーチェーンを設定する例を示します。

```
config
router eigrp 100
address-family ipv6
interface POS 0/1/0/0
authentication keychain key2
```

その他の参考資料

ここでは、EIGRP の実装に関する参考資料について説明します。

関連資料

関連項目	マニュアル タイトル
EIGRP コマンド：コマンド構文の詳細、コマンドモード、コマンド履歴、デフォルト設定、使用に関する注意事項、および例	*Routing Command Reference for Cisco ASR 9000 Series Routers*
EIGRP 機能向けの MPLS VPN のサポート情報	*MPLS Configuration Guide for Cisco ASR 9000 Series RoutersMPLS Configuration Guide for Cisco NCS 560 Series Routers*の「*Implementing MPLS Layer 3 VPNs* module and *Implementing MPLS Layer 2 VPNs*」のモジュール

関連項目	マニュアル タイトル
EIGRP 機能向けの Site of Origin（SoO）のサポート情報	*MPLS Configuration Guide for Cisco ASR 9000 Series RoutersMPLS Configuration Guide for Cisco NCS 560 Series Routers*の「*Implementing MPLS Traffic Engineering on* Cisco ASR 9000 シリーズ ルータ 」のモジュール
MIB リファレンス	『*Cisco ASR 9000 Series Aggregation Services Router MIB Specification Guide*』

標準

標準	タイトル
この機能でサポートされる新規の標準または変更された標準はありません。また、既存の標準のサポートは変更されていません。	—

MIB

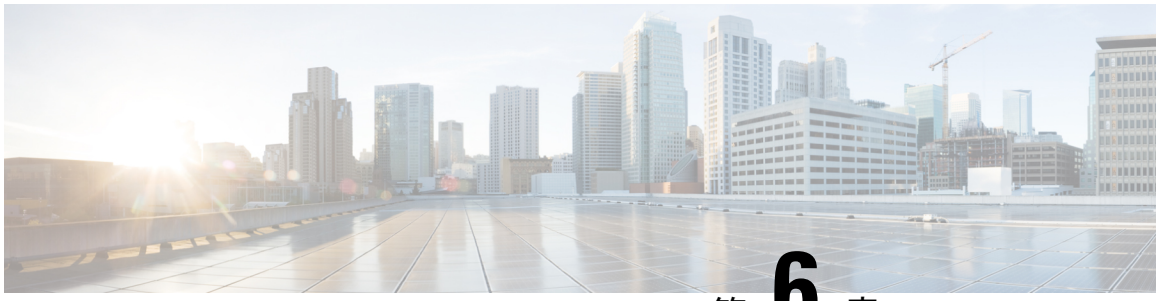
MIB	MIB のリンク
—	Cisco IOS XR ソフトウェアを使用して MIB の場所を特定してダウンロードするには、次の URL にある Cisco MIB Locator を使用して、[Cisco Access Products] メニューからプラットフォームを選択します。 https://mibs.cloudapps.cisco.com/ITDIT/MIBS/servlet/index

RFC

RFC	タイトル
この機能がサポートする新しい RFC または変更された RFC はありません。また、この機能は既存の規格に対するサポートに影響を及ぼしません。	—

シスコのテクニカル サポート

説明	リンク
シスコのテクニカル サポート Web サイトでは、製品、テクノロジー、ソリューション、技術的なヒント、およびツールへのリンクなどの、数千ページに及ぶ技術情報が検索可能です。Cisco.com に登録済みのユーザは、このページから詳細情報にアクセスできます。	http://www.cisco.com/techsupport



第 6 章

IS-IS の実装

Integrated Intermediate System-to-Intermediate System（IS-IS）、インターネット プロトコル バージョン 4（IPv4）は、標準ベースの内部ゲートウェイ プロトコル（IGP）です。Cisco ソフトウェアは、国際標準化機構（ISO）/International Engineering Consortium（IEC）10589 および RFC 1195 に記載されている IP ルーティング機能を実装し、IP バージョン 6（IPv6）向けに標準拡張のシングル トポロジおよびマルチトポロジ IS-IS を追加しています。

このモジュールでは、Cisco IOS XR ネットワークで IS-IS（IPv4 および IPv6）を実装する方法について説明します。

（注） 現在は、デフォルトの VRF のみがサポートされています。VPNv4、VPNv6 および VPN ルーティング/転送（VRF）のアドレス ファミリ、L3VPN およびマルチキャストは、今後のリリースでサポートされる予定です。

IS-IS の実装の前提条件

適切なタスク ID を含むタスク グループに関連付けられているユーザ グループに属している必要があります。このコマンド リファレンスには、各コマンドに必要なタスク ID が含まれます。ユーザ グループの割り当てが原因でコマンドを使用できないと考えられる場合、AAA 管理者に連絡してください。

IS-IS の実装

同じ物理インターフェイス上に複数の IS-IS インスタンスを存在させることができます。ただし、同じ物理インターフェイスを共有するすべてのインスタンスに異なるインスタンス ID を設定する必要があります。

または、 dot1q サブインターフェイスを作成して、dot1q サブインターフェイスそれぞれを異なる IS-IS インスタンスに設定することもできます。

（注） **show configuration** コマンドの出力の結果を表示するには、1 または 2 のレベル（**no max-metric level {1|2}**）でのみ **no max-metric** コマンドを設定します。それ以外の場合、最大メトリック設定は出力に表示されません。この動作は、ルータに設定をコミットする前に確認されます。

IS-IS の実装の設定例

ここでは、次の設定例について説明します。

シングルトポロジ IS-IS for IPv6 の設定：例

次に、single-topology モードのイネーブル化の例を示します。IS-IS インスタンスが作成され、NET が定義され、インターフェイス上で IPv6 が IPv4 とともに設定され、IPv4 リンク トポロジが IPv6 で使用されます。

この設定は、POS インターフェイス 0/3/0/0 が IPv4 アドレスと IPv6 アドレスの両方の隣接関係を形成できるようにします。

```
router isis isp
 net 49.0000.0000.0001.00
 address-family ipv6 unicast
  single-topology
 interface POS0/3/0/0
  address-family ipv4 unicast
  !
  address-family ipv6 unicast
  !
  exit
!
interface POS0/3/0/0
 ipv4 address 10.0.1.3 255.255.255.0
 ipv6 address 2001::1/64
```

マルチトポロジ IS-IS for IPv6 の設定：例

次に、IPv6 に設定されているマルチトポロジ IS-IS を示します。

```
router isis isp
 net 49.0000.0000.0001.00
 interface POS0/3/0/0
  address-family ipv6 unicast
  metric-style wide level 1
  exit
!
interface POS0/3/0/0
 ipv6 address 2001::1/64
```

複数インスタンス間での IS-IS ルートの再配布：例

次に、**set- attached-bit** コマンドと **redistribute** コマンドの使用例を示します。レベル 1 に制限されたインスタンス「1」とレベル 2 に制限されたインスタンス「2」の 2 つのインスタンスが設定されています。

再配布を使用してレベル 1 のインスタンスからレベル 2 のインスタンスにルートが伝播します。レベル 1 のルートが優先されるように、レベル 2 インスタンスのアドミニストレーティブディスタンスが明示的に大きく設定されていることに注目してください。

レベル 1 インスタンスはレベル 2 インスタンスへの再配布ルートであることから、レベル 1 インスタンスには attached ビットが設定されています。このため、インスタンス「1」はエリアからバックボーンへ到達するための適切な候補になります。

```
router isis 1
  is-type level-2-only
 net 49.0001.0001.0001.0001.00
 address-family ipv4 unicast
  distance 116
  redistribute isis 2 level 2
!
interface GigabitEthernet 0/3/0/0
 address-family ipv4 unicast
!
!
router isis 2
 is-type level-1
 net 49.0002.0001.0001.0002.00
 address-family ipv4 unicast
  set
-attached
-bit

 !
 interface GigabitEthernet 0/1/0/0
  address-family ipv4 unicast
```

ルートのタギング：例

次に、ルートのタギングの例を示します。

```
route-policy isis-tag-55
end-policy
```

```
!
route-policy isis-tag-555
  if destination in (5.5.5.0/24 eq 24) then
    set tag 555
    pass
  else
    drop
  endif
end-policy
!
router static
 address-family ipv4 unicast
  0.0.0.0/0 2.6.0.1
  5.5.5.0/24 Null0
 !
!
router isis uut
 net 00.0000.0000.12a5.00
 address-family ipv4 unicast
  metric-style wide
  redistribute static level-1 route-policy isis-tag-555
  spf prefix-priority critical tag 13
  spf prefix-priority high tag 444
  spf prefix-priority medium tag 777
```

IS-IS 過負荷ビット無効化の設定：例

次に、IS-IS 過負荷ビット無効化をアクティブにする例を示します。

```
config
 mpls traffic-eng path-selection ignore overload
```

次に、IS-IS 過負荷ビット無効化を非アクティブにする例を示します。

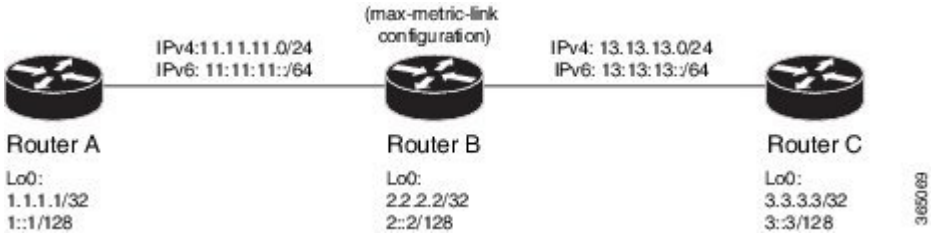
```
config
 no mpls traffic-eng path-selection ignore overload
```

例：ルータの過負荷状態を処理するための IS-IS の設定

この項では、過負荷ビットを設定せずに、ルータの過負荷状態を処理するための IS-IS の設定例について説明します。

ルータが IS-IS 過負荷ビットで設定されている場合、過負荷ビットが設定されているときはルーティングプロセスに参加しますが、トラフィックを転送しません（直接接続されたインターフェイスへのトラフィックを除く）。過負荷動作を IS-IS に設定するには、過負荷ビットを設定せずに、**max-link-metric** ステートメントを設定します。このステートメントを設定することにより、ルータはルーティングプロセスに参加し、最後の手段である中継ノードとして使用されます。

図 *19 :*



始める前に

特定のトポロジのルータインターフェイスの設定に精通していることを確認します。

手順の概要

1. トポロジに示すように、ルータ A、B、および C を設定します。
2. ルータ A、B、および C で、IS-IS と対応するネットアドレスを設定します。
3. ルータ A、B、および C のループバック インターフェイスで IPv4 と IPv6 のアドレスファミリを設定します。
4. ルータインターフェイスでリンクメトリックを設定します。
5. ルータ A、B、および C のルートプレフィックスを表示して、設定を確認します。
6. **max-link-metric** ステートメントを設定する前に、ルータ B にリンクメトリックを確認します。
7. ルータ B に **max-link-metric** ステートメントを設定します。
8. 設定をコミットします。
9. ルータ B のリンクメトリックの変更を確認します。
10. （任意）ルータ A と C のルートプレフィックスの変更を確認します。

手順の詳細

ステップ 1 トポロジに示すように、ルータ A、B、および C を設定します。

次の IP アドレスを使用します。

- **Router A Loopback0**: 1.1.1.1/32 and 1::1/128
- **Router A -> Router B**: 11.11.11.2/24 and 11:11:11::2/64
- **Router B Loopback0**: 2.2.2.2/32 and 2::2/128
- **Router B -> Router A**: 11.11.11.1/24 and 11:11:11::1/64
- **Router B-> Router C**: 13.13.13.1/24 and 13:13:13::1/64
- **Router C Loopback0**: 3.3.3.3/32 and 3::3/128
- **Router C-> Router B**: 13.13.13.2/24 and 13:13:13::2/64

ステップ 2 ルータ A、B、および C で、IS-IS と対応するネットアドレスを設定します。

例：

```
!Router A
RP/0/0/CPU0:RouterA(config)# router isis ring
RP/0/0/CPU0:RouterA(config-isis)# net 00.0000.0000.0001.00
RP/0/0/CPU0:RouterA(config-isis)# address-family ipv4 unicast
RP/0/0/CPU0:RouterA(config-isis)# metric-style wide
RP/0/0/CPU0:RouterA(config-isis-af)# exit

!Router B
RP/0/0/CPU0:RouterB(config)# router isis ring
RP/0/0/CPU0:RouterB(config-isis)# net 00.0000.0000.0002.00
RP/0/0/CPU0:RouterB(config-isis)# address-family ipv4 unicast
RP/0/0/CPU0:RouterB(config-isis-af)# exit

!Router C
RP/0/0/CPU0:RouterC(config)# router isis ring
RP/0/0/CPU0:RouterC(config-isis)# net 00.0000.0000.0003.00
RP/0/0/CPU0:RouterC(config-isis)# address-family ipv4 unicast
RP/0/0/CPU0:RouterA(config-isis)# metric-style wide
RP/0/0/CPU0:RouterC(config-isis-af)# exit
```

ステップ 3 ルータ A、B、および C のループバック インターフェイスで IPv4 と IPv6 のアドレスファミリを設定します。

例：

```
RP/0/0/CPU0:Router(config-isis)# interface loopback0
RP/0/0/CPU0:Router(config-isis-if)# address-family ipv4 unicast
RP/0/0/CPU0:Router(config-isis-if-af)# exit
RP/0/0/CPU0:Router(config-isis-if)# address-family ipv6 unicast
RP/0/0/CPU0:Router(config-isis-if-af)# exit
RP/0/0/CPU0:Router(config-isis-if)# exit
RP/0/0/CPU0:Router(config-isis)#
```

ステップ 4 ルータインターフェイスでリンクメトリックを設定します。

例：

```
! Configuration for Router A Interface GigabitEthernet 0/0/0/0 with Router B is shown here.
Similarly, configure other router interfaces.
RP/0/0/CPU0:RouterA(config-isis)# interface GigabitEthernet 0/0/0/0
RP/0/0/CPU0:RouterA(config-isis-if)# address-family ipv4 unicast
RP/0/0/CPU0:RouterA(config-isis-if-af)# metric 10
RP/0/0/CPU0:RouterA(config-isis-if-af)# exit
RP/0/0/CPU0:RouterA(config-isis-if)# address-family ipv6 unicast
RP/0/0/CPU0:RouterA(config-isis-if-af)# exit
RP/0/0/CPU0:RouterA(config-isis-if)# exit
RP/0/0/CPU0:RouterA(config-isis)#
```

ステップ 5 ルータ A、B、および C のルートプレフィックスを表示して、設定を確認します。

例：

```
! The outputs for Router A are shown here. Similarly, view the outputs for Routers B and C.
RP/0/0/CPU0:RouterA# show route
Tue Oct 13 13:55:18.342 PST

Codes: C - connected, S - static, R - RIP, B - BGP, (>) - Diversion path
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - ISIS, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, su - IS-IS summary null, * - candidate default
```

```
       U - per-user static route, o - ODR, L - local, G  - DAGR, l - LISP
       A - access/subscriber, a - Application route
       M - mobile route, (!) - FRR Backup path

Gateway of last resort is not set

L    1.1.1.1/32 is directly connected, 00:03:40, Loopback0
i L1 2.2.2.2/32 [115/20] via 11.11.11.2, 00:01:27, GigabitEthernet0/0/0/0
i L1 3.3.3.3/32 [115/30] via 11.11.11.2, 00:01:27, GigabitEthernet0/0/0/0
C    11.11.11.0/24 is directly connected, 00:03:39, GigabitEthernet0/0/0/0
L    11.11.11.1/32 is directly connected, 00:03:39, GigabitEthernet0/0/0/0
i L1 13.13.13.0/24 [115/20] via 11.11.11.2, 00:01:27, GigabitEthernet0/0/0/0
i L1 15.15.15.0/24 [115/30] via 11.11.11.2, 00:01:27, GigabitEthernet0/0/0/0


RP/0/0/CPU0:RouterA# show route ipv6
Tue Oct 13 14:00:55.758 PST

Codes: C - connected, S - static, R - RIP, B - BGP, (>) - Diversion path
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - ISIS, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, su - IS-IS summary null, * - candidate default
       U - per-user static route, o - ODR, L - local, G  - DAGR, l - LISP
       A - access/subscriber, a - Application route
       M - mobile route, (!) - FRR Backup path

Gateway of last resort is not set

L    1::1/128 is directly connected,
      00:09:17, Loopback0
i L1 2::2/128
      [115/20] via fe80::e9:45ff:fe22:5326, 00:00:05, GigabitEthernet0/0/0/0
i L1 3::3/128
      [115/30] via fe80::e9:45ff:fe22:5326, 00:00:05, GigabitEthernet0/0/0/0
C    11:11:11::/64 is directly connected,
      00:09:16, GigabitEthernet0/0/0/0
L    11:11:11::1/128 is directly connected,
      00:09:16, GigabitEthernet0/0/0/0
i L1 13:13:13::/64
      [115/20] via fe80::e9:45ff:fe22:5326, 00:00:05, GigabitEthernet0/0/0/0
i L1 15:15:15::/64
      [115/30] via fe80::e9:45ff:fe22:5326, 00:00:05, GigabitEthernet0/0/0/0
```

ステップ 6 **max-link-metric** ステートメントを設定する前に、ルータ B にリンクメトリックを確認します。

例：

```
RP/0/0/CPU0:RouterB# show isis database
Tue Oct 13 13:56:44.077 PST

No IS-IS RING levels found
IS-IS ring (Level-1) Link State Database
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime  ATT/P/OL
RouterB.00-00           * 0x00000005   0x160d        1026            0/0/0
  Area Address: 00
  NLPID:        0xcc
  NLPID:        0x8e
  MT:           Standard (IPv4 Unicast)
  MT:           IPv6 Unicast                                     0/0/0
  Hostname:     RouterB
  IP Address:   2.2.2.2
```

```
  IPv6 Address: 2::2

  Metric: 10          IS RouterB.01
  Metric: 10          IS RouterA.00
  Metric: 10          IP 2.2.2.2/32
  Metric: 10          IP 11.11.11.0/24
  Metric: 10          IP 13.13.13.0/24
  Metric: 10          MT (IPv6 Unicast) IS-Extended RouterB.01
  Metric: 10          MT (IPv6 Unicast) IS-Extended RouterA.00
  Metric: 10          MT (IPv6 Unicast) IPv6 2::2/128
  Metric: 10          MT (IPv6 Unicast) IPv6 11:11:11::/64
  Metric: 10          MT (IPv6 Unicast) IPv6 13:13:13::/64
RouterB.01-00              0x00000001   0xc8df        913             0/0/0
  Metric: 0           IS RouterB.00
  Metric: 0           IS RouterC.00
  Metric: 0           IS-Extended RouterB.00
  Metric: 0           IS-Extended RouterC.00

 Total Level-1 LSP count: 2     Local Level-1 LSP count: 1
```

出力で、IS-IS プロトコルが動作していること、および表示されているリンクメトリック（**Metric: 10**）が設定どおりであることを確認します。

ステップ 7 ルータ B に **max-link-metric** ステートメントを設定します。

例：

```
RP/0/0/CPU0:RouterB(config)# router isis ring
RP/0/0/CPU0:RouterB(config-isis)# max-link-metric
RP/0/0/CPU0:RouterB(config-isis)# exit
RP/0/0/CPU0:RouterB(config)#
```

ステップ 8 設定をコミットします。

例：

```
RP/0/0/CPU0:RouterB(config)# commit
```

ステップ 9 ルータ B のリンクメトリックの変更を確認します。

例：

```
RP/0/0/CPU0:RouterB# show isis database
Tue Oct 13 13:58:36.790 PST

No IS-IS RING levels found
IS-IS ring (Level-1) Link State Database
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime  ATT/P/OL
RouterB.00-00           * 0x00000006   0x0847        1171            0/0/0
  Area Address: 00
  NLPID:        0xcc
  NLPID:        0x8e
  MT:           Standard (IPv4 Unicast)
  MT:           IPv6 Unicast                                     0/0/0
  Hostname:     RouterB
  IP Address:   2.2.2.2
  IPv6 Address: 2::2
  Metric: 63          IS RouterB.01
  Metric: 63          IS RouterA.00
  Metric: 63          IP 2.2.2.2/32
  Metric: 63          IP 11.11.11.0/24
  Metric: 63          IP 13.13.13.0/24
  Metric: 16777214    MT (IPv6 Unicast) IS-Extended RouterB.01
  Metric: 16777214    MT (IPv6 Unicast) IS-Extended RouterA.00
  Metric: 16777214    MT (IPv6 Unicast) IPv6 2::2/128
```

```
  Metric: 16777214   MT (IPv6 Unicast) IPv6 11:11:11::/64
  Metric: 16777214   MT (IPv6 Unicast) IPv6 13:13:13::/64
RouterB.01-00             0x00000001   0xc8df        800             0/0/0
  Metric: 0          IS RouterB.00
  Metric: 0          IS RouterC.00
  Metric: 0          IS-Extended RouterB.00
  Metric: 0          IS-Extended RouterC.00

 Total Level-1 LSP count: 2     Local Level-1 LSP count: 1
```

出力で、最大リンクメトリック（IPv4 の場合は **63**、IPv6 の場合は **16777214** ）が指定されたリンクに割り当てられていることを確認します。

ステップ 10 （任意）ルータ A と C のルートプレフィックスの変更を確認します。

例：

```
! The outputs for Router A are shown here. Similarly, view the outputs on Router C.
RP/0/0/CPU0:RouterA# show route
Tue Oct 13 13:58:59.289 PST

Codes: C - connected, S - static, R - RIP, B - BGP, (>) - Diversion path
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - ISIS, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, su - IS-IS summary null, * - candidate default
       U - per-user static route, o - ODR, L - local, G  - DAGR, l - LISP
       A - access/subscriber, a - Application route
       M - mobile route, (!) - FRR Backup path

Gateway of last resort is not set

L    1.1.1.1/32 is directly connected, 00:07:21, Loopback0
i L1 2.2.2.2/32 [115/73] via 11.11.11.2, 00:00:50, GigabitEthernet0/0/0/0
i L1 3.3.3.3/32 [115/83] via 11.11.11.2, 00:00:50, GigabitEthernet0/0/0/0
C    11.11.11.0/24 is directly connected, 00:07:20, GigabitEthernet0/0/0/0
L    11.11.11.1/32 is directly connected, 00:07:20, GigabitEthernet0/0/0/0
i L1 13.13.13.0/24 [115/73] via 11.11.11.2, 00:00:50, GigabitEthernet0/0/0/0
i L1 15.15.15.0/24 [115/83] via 11.11.11.2, 00:00:50, GigabitEthernet0/0/0/0


RP/0/0/CPU0:RouterA# show route ipv6
Tue Oct 13 14:00:06.616 PST

Codes: C - connected, S - static, R - RIP, B - BGP, (>) - Diversion path
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - ISIS, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, su - IS-IS summary null, * - candidate default
       U - per-user static route, o - ODR, L - local, G  - DAGR, l - LISP
       A - access/subscriber, a - Application route
       M - mobile route, (!) - FRR Backup path

Gateway of last resort is not set

L    1::1/128 is directly connected,
      00:08:28, Loopback0
i L1 2::2/128
      [115/16777224] via fe80::e9:45ff:fe22:5326, 00:01:58, GigabitEthernet0/0/0/0
i L1 3::3/128
      [115/16777234] via fe80::e9:45ff:fe22:5326, 00:01:58, GigabitEthernet0/0/0/0
C    11:11:11::/64 is directly connected,
      00:08:27, GigabitEthernet0/0/0/0
```

```
L    11:11:11::1/128 is directly connected,
      00:08:27, GigabitEthernet0/0/0/0
i L1 13:13:13::/64
      [115/16777224] via fe80::e9:45ff:fe22:5326, 00:01:58, GigabitEthernet0/0/0/0
i L1 15:15:15::/64
      [115/16777234] via fe80::e9:45ff:fe22:5326, 00:01:58, GigabitEthernet0/0/0/0
```

出力で、ルーティングテーブルの最大メトリック設定の影響（**[115/73]** と **[115/83]**）を確認します。

IS-IS は、過負荷ビットを設定せずにルータの過負荷状態を処理するように正常に設定されています。

次の作業

他の IP ルーティング プロトコルを実装するには、 *Routing Configuration Guide for Cisco ASR 9000 Series Routers*の次のドキュメント モジュールを参照してください。

- 「OSPF の実装」
- 「BGP の実装」
- 「EIGRP の実装」
- 「RIP の実装」

その他の参考資料

ここでは、IS-IS の実装に関する参考資料について説明します。

関連資料

関連項目	マニュアル タイトル
IS-IS コマンド：コマンド構文の詳細、コマンド モード、コマンド履歴、デフォルト設定、使用に関する注意事項、および例	*Routing Command Reference for Cisco ASR 9000 Series Routers*
MPLS TE 機能情報	*MPLS Configuration Guide for Cisco ASR 9000 Series RoutersMPLS Configuration Guide for Cisco NCS 560 Series Routers*の「 *Implementing MPLS Traffic Engineering on Cisco ASR 9000 Series Router* 」のモジュール
双方向フォワーディング検出（BFD）	*Interface and Hardware Component Configuration Guide for Cisco ASR 9000 Series Routers* および *Interface and Hardware Component Command Reference for Cisco ASR 9000 Series Routers*

標準

標準	タイトル
Draft-ietf-isis-ipv6-05.txt	『*Routing IPv6 with IS-IS*』（Christian E. Hopps）
Draft-ietf-isis-wg-multi-topology-06.txt	『*M-ISIS: Multi Topology (MT) Routing in IS-IS*』（Tony Przygienda、Naiming Shen、Nischal Sheth）
Draft-ietf-isis-traffic-05.txt	『*IS-IS Extensions for Traffic Engineering*』（Henk Smit、Toni Li）
Draft-ietf-isis-restart-04.txt	『*Restart Signaling for IS-IS*』（M. Shand、Les Ginsberg）
Draft-ietf-isis-igp-p2p-over-lan-05.txt	『*Point-to-point operation over LAN in link-state routing protocols*』（Naiming Shen）
Draft-ietf-rtgwg-ipfrr-framework-06.txt	『*IP Fast Reroute Framework*』（M. Shand、S. Bryant）
Draft-ietf-rtgwg-lf-conv-frmwk-00.txt	『*A Framework for Loop-free Convergence*』（M. Shand、S. Bryant）

MIB

MB	MIB のリンク
—	Cisco IOS XR ソフトウェアを使用して MIB の場所を特定してダウンロードするには、次の URL にある Cisco MIB Locator を使用して、[Cisco Access Products] メニューからプラットフォームを選択します。 https://mibs.cloudapps.cisco.com/ITDIT/MIBS/servlet/index

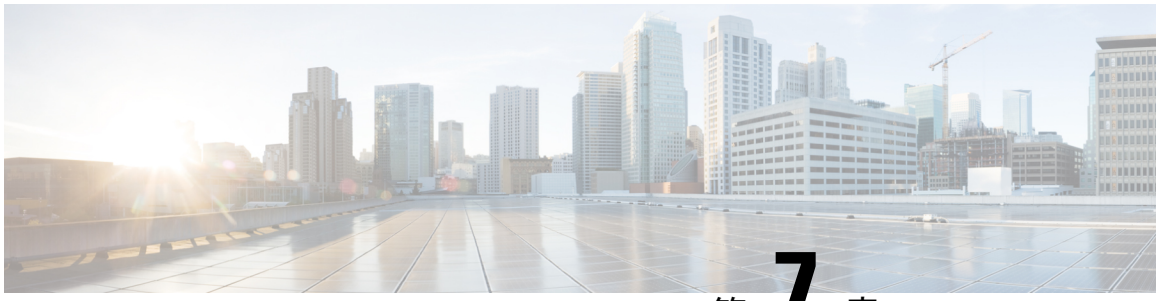
RFC

RFC	タイトル
RFC 1142	『OSI IS-IS Intra-domain Routing Protocol』
RFC 1195	『Use of OSI IS-IS for Routing in TCP/IP and Dual Environments』
RFC 2763	『Dynamic Hostname Exchange Mechanism for IS-IS』
RFC 2966	『Domain-wide Prefix Distribution with Two-Level IS-IS』
RFC 2973	『IS-IS Mesh Groups』
RFC 3277	『IS-IS Transient Blackhole Avoidance』

RFC	タイトル
RFC 3373	『Three-Way Handshake for IS-IS Point-to-Point Adjacencies』
RFC 3567	『IS-IS Cryptographic Authentication』
RFC 4444	『IS-IS Management Information Base』

シスコのテクニカル サポート

説明	リンク
シスコのテクニカル サポート Web サイトでは、製品、テクノロジー、ソリューション、技術的なヒント、およびツールへのリンクなどの、数千ページに及ぶ技術情報が検索可能です。Cisco.com に登録済みのユーザは、このページから詳細情報にアクセスできます。	http://www.cisco.com/techsupport



第 7 章

OSPF の実装

Open Shortest Path First（OSPF）は、Internet Engineering Task Force（IETF）の OSPF ワーキング グループによって開発された内部ゲートウェイ プロトコル（IGP）です。OSPF は特に IP ネットワーク向けに設計されており、IP サブネット化、および外部から取得したルーティング情報のタギングをサポートしています。また、OSPF を使用すると、パケットの送受信時にパケット認証が可能になり、IP マルチキャストが使用されます。

OSPF Version 3（OSPFv3）は OSPF Version 2 を拡張し、IPv6 ルーティング プレフィックスのサポートを提供します。

このモジュールでは、Cisco ASR 9000 シリーズ ルータで OSPF の両方のバージョンを実装するために必要な概念と作業について説明します。特に記載のないかぎり、用語「OSPF」は両方のバージョンのルーティング プロトコルを意味します。

（注） Cisco IOS XR ソフトウェアの OSPF についての詳細情報、およびこのモジュールに記載されている OSPF コマンドの詳細説明については、このモジュールの関連資料 （491 ページ）の項を参照してください。設定作業を実行中に表示されることのある他のコマンドのドキュメントを検索するには、オンラインで *Cisco ASR 9000 Series Aggregation Services Router Commands Master List* を検索してください。

OSPF の実装の機能履歴

リリース	変更内容
リリース 3.7.2	この機能が導入されました。
リリース 3.9.0	次の機能に対するサポートが追加されました。 • OSPFv2 SPF プレフィックスのプライオリティ付け。 • IP 高速再ルーティング ループフリー代替の計算 • OSPF バージョン 3 のウォームスタンバイ

リリース	変更内容
リリース 4.2.0	次の機能に対するサポートが追加されました。 • プレフィックス単位の OSPFv2 高速再ルーティングの計算 • OSPFv3 ノンストップルーティング（NSR）
リリース 4.3.0	次の機能に対するサポートが追加されました。 • OSPFv2 VRF Lite • OSPFv3 タイマーの更新
リリース 5.3.0	次の機能に対するサポートが追加されました。 • OSPFv2 セグメントルーティング トポロジに依存しない高速再ルーティング • ASR 9000 拡張イーサネットラインカード用 64 ECMP
リリース 5.3.2	次の機能に対するサポートが追加されました。 • BFD ダンプニングの OSPF ストリクトモードのサポート • OSPF FIB ダウンロード通知
リリース 6.0.1	次の機能がサポートされました。 • 過剰パントフロートラップ処理

OSPF の実装の前提条件

次に、Cisco IOS XR ソフトウェアで OSPF を実装するための前提条件を示します。

- 適切なタスク ID を含むタスク グループに関連付けられているユーザ グループに属している必要があります。このコマンド リファレンスには、各コマンドに必要なタスク ID が含まれます。ユーザ グループの割り当てが原因でコマンドを使用できないと考えられる場合、AAA 管理者に連絡してください。

- OSPFv3 の設定作業では、IPv6 のアドレッシングと基本概念について精通していることを前提としています。IPv6 のルーティングとアドレッシングについては、 *IP Addresses and Services Configuration Guide for Cisco ASR 9000 Series Routers*の「*Implementing Network Stack IPv4 and IPv6 on* Cisco ASR 9000 シリーズ ルータ 」のモジュールを参照してください。
- インターフェイスで OSPFv3 をイネーブルにする前に、次の手順を実行する必要があります。
  - ご使用の IPv6 ネットワークに対する OSPF ネットワーク戦略と計画を完成させます。たとえば、複数のエリアが必要かどうかを決定します。
  - インターフェイスで IPv6 をイネーブルにします。
- 認証（IP セキュリティ）の設定はオプションの作業です。認証を設定する場合、プレーンテキスト認証と Message Digest 5（MD5）認証のどちらを設定するかについて、また、認証をエリア全体に適用するか特定のインターフェイスに適用するかについて最初に決定する必要があります。

OSPF の実装に関する情報

OSPF を実装するには、次の概念を理解する必要があります。

OSPF 機能の概要

OSPF は、IP 用のルーティング プロトコルです。これは、ディスタンスベクトル プロトコルではなく、リンクステート プロトコルです。リンクステート プロトコルは、送信元マシンと宛先マシンを接続するリンクの状態に基づいて、ルーティングの決定を行います。リンク ステートは、インターフェイスと、その隣接ネットワーキングデバイスとの関係を説明するものです。インターフェイス情報には、インターフェイスの IP アドレス、ネットワーク マスク、接続されているネットワークの種類、そのネットワークに接続されているルータなどがあります。この情報は、さまざまなタイプのリンクステートアドバタイズメント（LSA）によって伝播します。

ルータは受信した LSA データの集まりをリンクステートデータベースに格納します。このデータベースにはこのルータのリンクの LSA データが含まれます。ダイクストラ アルゴリズムが採用されている場合、データベースの内容からデータが抽出されて OSPF ルーティングテーブルが作成されます。データベースとルーティングテーブルの違いは、データベースにはすべての raw データが含まれており、ルーティングテーブルには特定のルータ インターフェイス ポートを介した既知の宛先への最短パスのリストが含まれていることです。

OSPF は大規模ネットワークにまで拡張できるため、IGP として適しています。エリアを使用してネットワークをより管理しやすい大きさに分割するとともに、ネットワークに階層を導入します。ルータはネットワークの 1 つのエリアまたは複数のエリアに接続されます。エリア内のすべてのネットワーキング デバイスは、デバイスが属するエリア内のみのリンク ステートがすべて揃った、同じデータベース情報を維持します。ネットワーク内のすべてのリンク ス

テートについての情報は持ちません。エリア内のルータ間におけるデータベース情報の合意はコンバージェンスと呼ばれます。

ドメイン内レベルで、OSPF は Intermediate System-to-Intermediate System（IS-IS）を使用して取得したルートを取り込むことができます。OSPF ルートを IS-IS に伝達することもできます。ドメイン間レベルで、OSPF はボーダー ゲートウェイ プロトコル（BGP）を使用して取得したルートを取り込むことができます。OSPF ルートを BGP に伝達することもできます。

Routing Information Protocol（RIP）とは異なり、OSPF は定期的なルーティング アップデートを送信しません。OSPF ルータはネイバーになると、データベースを交換および同期することによって隣接関係を確立します。その後、変更されたルーティング情報だけが伝播されます。エリア内のすべてのルータは自分のリンクのコストとステートをアドバタイズします。この情報は LSA 内で送られます。このステート情報は、1 ホップ先のすべての OSPF ネイバーに送られます。その後すべての OSPF ネイバーは、ステート情報を変更せずに送信します。このフラッディング プロセスは、エリア内のすべてのデバイスが同じリンクステート データベースを持つまで続けられます。

宛先への最適なルートを決定するために、宛先へのルートに含まれるリンクのすべてのコストがソフトウェアによって合計されます。各ルータが別のネットワーキング デバイスからルーティング情報を受信した後で、Shortest Path First（SPF）アルゴリズムが実行されて、データベース内の各宛先ネットワークへの最適なパスが計算されます。

OSPF を実行しているネットワーキング デバイスは、ネットワーク内のトポロジの変化を検出して、リンクステート アップデートをネイバーにフラッディングし、新しいトポロジ ビューをすぐに収束させます。ネットワーク内の各 OSPF ルータは、すぐに再び同じトポロジ ビューを持ちます。OSPF は、同じ宛先に対する複数の等コストのパスを許容します。すべてのリンクステート情報がフラッディングされて SPF 計算に使用されるため、複数の等コスト パスが計算されてルーティングに使用されることがあります。

ブロードキャスト ネットワークおよび非ブロードキャスト マルチアクセス（NBMA）ネットワークでは、指定ルータ（DR）またはバックアップ DR が LSA フラッディングを実行します。ポイントツーポイントネットワークでは、フラッディングは単にインターフェイスからネイバーに直接送信されます。

OSPF は直接 IP の上で実行され、TCP やユーザ データグラム プロトコル（UDP）を使用しません。OSPF はパケット ヘッダーおよび LSA のチェックサムを使用してそれ自体でエラー訂正を実行します。

OSPFv3 は、基本概念は OSPF Version 2 と同じですが、IPv6 の拡大されたアドレス サイズのサポートが追加されています。IPv6 のアドレスとプレフィックスを伝送するために新しい LSA タイプが作成され、個々の IP サブネット ベースではなく、個々のリンク ベースでプロトコルが実行されます。

OSPF は通常多くの内部ルータ間の調整を必要とします。このようなルータには、複数のエリアに接続されたエリア境界ルータ（ABR）や、他のソース（IS-IS、BGP、静的ルートなど）からの再ルーティングを OSPF トポロジに伝達する自律システム境界ルータ（ASBR）があります。OSPF ベースのルータまたはアクセス サーバの最小設定では、すべてのデフォルト パラメータ値、およびエリアに割り当てられたインターフェイスが使用され、認証は行われません。環境をカスタマイズする場合は、すべてのルータの調和が取れた設定が必要です。

Cisco IOS XR ソフトウェアの OSPF 実装でサポートされる主要機能

Cisco IOS XR ソフトウェアの OSPF 実装は、Internet RFC 2328 および RFC 2740 で詳述されている OSPF Version 2 および OSPF Version 3 の仕様にそれぞれ準拠しています。

Cisco IOS XR ソフトウェア の実装でサポートされている主な機能を次に示します。

- 階層：CLI 階層がサポートされます。
- 継承：CLI 継承がサポートされます。
- スタブ エリア：スタブ エリアの定義がサポートされています。
- NSF：ノンストップ フォワーディングがサポートされています。
- SPF スロットリング：Shortest Path First スロットリング機能がサポートされています。
- LSA スロットリング：LSA スロットリング機能がサポートされています。
- 高速コンバージェンス：SPF および LSA のスロットル タイマーが設定されると高速コンバージェンスが設定されます。OSPF LSA スロットリング機能は、ネットワークが不安定な間、OSPF での LSA アップデートを低速化するためのダイナミック メカニズムを提供します。さらに LSA スロットリングは、LSA のレート制限をミリ秒単位で指定することにより、OSPF コンバージェンス時間の短縮が可能になります。
- ルート再配布：任意の IP ルーティング プロトコルを使用して学習されたルートを、別の IP ルーティング プロトコルで再配布できます。
- 認証：エリア内の隣接ルータ間でのプレーン テキスト認証および MD5 認証がサポートされています。
- ルーティングインターフェイスパラメータ：サポートされる設定可能なパラメータには、インターフェイス出力コスト、再送信インターバル、インターフェイス送信遅延、ルータプライオリティ、ルータの「dead」インターバルと hello インターバル、認証キーなどがあります。
- 仮想リンク：仮想リンクがサポートされています。
- Not-So-Stubby Area（NSSA）：RFC 1587 がサポートされます。
- デマンド回線上の OSPF：RFC 1793 がサポートされています。

Cisco IOS XR ソフトウェアの OSPFv3 と OSPFv2 の比較

OSPFv3 プロトコルの大半は OSPFv2 と同じです。OSPFv3 は RFC 2740 に記載されています。

Cisco IOS XR ソフトウェアの OSPFv3 プロトコルと OSPFv2 プロトコルの主な相違点は、次のとおりです。

- OSPFv2 を拡張した OSPFv3 では、IPv6 ルーティング プレフィックスとサイズの大きい IPv6 アドレスのサポートを提供しています。

- NBMA インターフェイスを OSPFv3 で使用する場合、ユーザは、ネイバーのリストを使用してルータを手動で設定する必要があります。隣接ルータはネイバーに接続されたインターフェイスのリンク ローカル アドレスによって識別されます。
- OSPFv2 とは異なり、複数の OSPFv3 プロセスをリンク上で実行できます。
- OSPFv3 の LSA は、「アドレスとマスク」ではなく、「プレフィックスとプレフィックス長」として表現されます。
- ルータ ID は IPv6 アドレスとは無関係な 32 ビットの数値です。

OSPF の階層 CLI および CLI 継承

Cisco IOS XR ソフトウェア には、階層 CLI および CLI 継承で構成される新しい OSPF コンフィギュレーションの基礎が導入されています。

階層 CLI とは、定義された階層レベル（ルータ レベル、エリア レベル、インターフェイス レベルなど）で、ネットワーク コンポーネント情報がグループ化されたものです。階層 CLI を使うと、OSPF の設定、メンテナンス、トラブルシューティングをより簡単に行えます。コンフィギュレーション コマンドが一緒に階層コンテキストに表示されると、視覚的な検査が簡単になります。階層 CLI はサポートされる CLI 継承自体に備わっています。

CLI 継承を使うと、エリアやインターフェイスのパラメータを明示的に設定する必要がありません。Cisco IOS XR ソフトウェアでは、同じエリアのインターフェイスのパラメータだけを 1 つのコマンドで設定できます。また、エリア コンフィギュレーション レベルやルータ OSPF コンフィギュレーション レベルなどの高い階層レベルからパラメータ値を継承できます。

たとえば、インターフェイスの hello interval 値は、IF ステートメントの優先順位によって次のように決まります。

インターフェイス コンフィギュレーション レベルで hello interval コマンドが設定されている場合は、インターフェイスに設定されている値を使用します。

エリア コンフィギュレーション レベルで hello interval コマンドが設定されている場合は、エリアに設定されている値を使用します。

ルータ コンフィギュレーション レベルで hello interval コマンドが設定されている場合は、ルータ設定されている値を使用します。

その他の場合は、コマンドのデフォルト値を使用します。

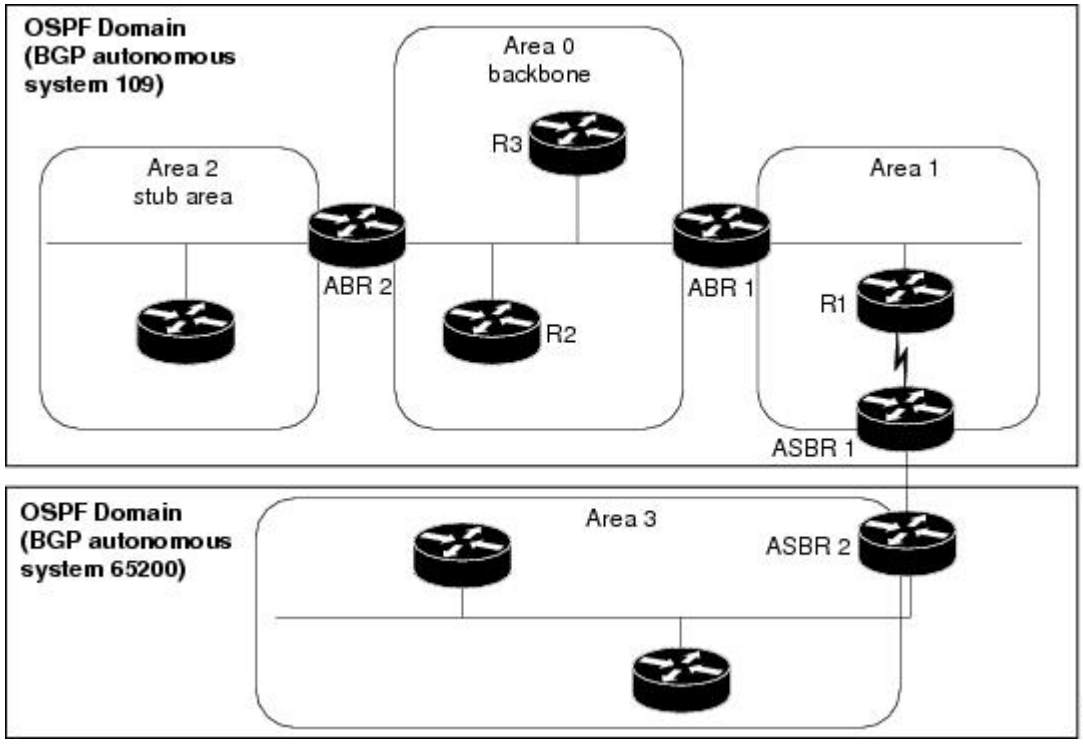
ヒント　階層 CLI および CLI 継承を理解すると、設定時間を大幅に短縮できます。これらの基礎を理解するには、OSPF Version 2 の異なる階層レベルでの認証の設定 （433 ページ）を参照してください。また、Cisco IOS XR ソフトウェアの例については、OSPF の実装の設定例 （481 ページ）を参照してください。

OSPF ルーティング コンポーネント

OSPF を実装する前に、ルーティング コンポーネントの概要とその使用目的を把握する必要があります。これらは自律システム、エリア タイプ、内部ルータ、ABR、および ASBR で構成されます。

図 *20 : OSPF* ルーティング コンポーネント

次の図に、OSPF ネットワークのトポロジのルーティング コンポーネントを示します。



自律システム

自律システムは、同じ管理制御下で、相互にルーティング情報を共有するネットワークの集合です。自律システムは、ルーティング ドメインとも呼ばれます。図 20 : OSPF ルーティング コンポーネント （395 ページ）に、109 と 65200 の 2 つの自律システムを示します。自律システムは 1 つまたは複数の OSPF エリアで構成されます。

エリア

エリアでは、自律システムをより小さく管理しやすいネットワークや隣接ネットワークのセットに再分割できます。図 20 : OSPF ルーティング コンポーネント （395 ページ）で示されるように、自律システム 109 はエリア 0、エリア 1、エリア 2 の 3 つのエリアから構成されます。

OSPF は 1 つのエリアのトポロジをその他の自律システムから見えないようにします。1 つのエリアのネットワーク トポロジはそのエリア内のルータにのみ認識されます。OSPF ルーティングがエリア内にある場合、そのルーティングはエリア内ルーティングと呼ばれます。この

ルーティングは、ネットワークにフラッディングするリンク ステート情報量を制限して、ルーティング トラフィックを少なくします。各ルータのトポロジ情報のサイズも小さくし、各ルータの処理と必要なメモリを節約します。

また、エリア内のルータはエリア外の詳細なネットワーク トポロジを見ることはできません。このようにトポロジ情報の開示が制限されているため、自律システム全体が1つのルーティング ドメインであるときに、エリア間のトラフィック フローを制御して、ルーティング トラフィックを少なくすることができます。

バックボーン エリア

バックボーンエリアは、自律システムの複数エリア間でルーティング情報を配布する役割を担当します。エリアの外で発生する OSPF ルーティングをエリア間ルーティングと呼びます。

エリアのプロパティはすべてバックボーン自体にあります。これは、バックボーンだけにあるABR、ルータ、ネットワークで構成されます。図 20 : OSPF ルーティング コンポーネント （395 ページ）に示されるように、エリア 0 は OSPF バックボーン エリアです。すべての OSPF バックボーン エリアでは、0.0.0.0 の ID が予約されています。

スタブ エリア

スタブ エリアは、ルート アドバタイズメントや、エリアの外部にあるネットワークの詳細情報を受け入れないエリアです。スタブエリアには、通常、エリアと他の自律システムとのインターフェイスになるルータが 1 つだけがあります。スタブ ABR は、外部の宛先への単一のデフォルト ルートをスタブ エリアにアドバタイズします。スタブ エリア内のルータはエリア外の宛先および自律システムに対してこのルートを使用します。この関係は、エリアにフラッディングされた外部 LSA を格納するためにも使用される LSA データベースのスペースを節約します。図 20 : OSPF ルーティング コンポーネント （395 ページ）で、エリア 2 は ABR 2 を経由してのみ到達するスタブ エリアです。エリア 0 はスタブ エリアにはできません。

Not So Stubby Area

Not-So-Stubby Area（NSSA）はスタブ エリアに似ています。NSSA はコアからエリアへとタイプ 5 の外部 LSA をフラッディングしませんが、限定的に自律システム外部ルートをエリア内にインポートできます。

NSSA は、再配布によって、タイプ 7 の自律システムの外部ルートを NSSA エリア内部にインポートできます。これらのタイプ 7 の LSA は、NSSA の ABR によってタイプ 5 の LSA に変換され、ルーティング ドメイン全体にフラッディングされます。変換中は集約とフィルタリングがサポートされます。

異なるルーティング プロトコルを使用するリモート サイトに OSPF を使用する中央サイトを接続する必要があるネットワーク管理者であれば、管理を簡素化するために NSSA 使用します。

スタブエリアにはリモートサイトのルートが再配布されないため、NSSA が実装される前は、企業サイトの境界ルータとリモート ルータ間の接続に OSPF スタブ エリアを利用できず、2 つのルーティング プロトコルを維持する必要がありました。RIP のようなシンプルなプロトコルを実行して再配布を処理する方法が一般的でした。NSSA が実装されたことで、企業ルータと

リモート ルータ間のエリアを NSSA として定義することにより、NSSA で OSPF を拡張してリモート接続をカバーできます。エリア 0 を NSSA にすることはできません。

ルータ

OSPF ネットワークは ABR、ASBR、内部ルータで構成されます。

エリア境界ルータ

エリア境界ルータ（ABR）は複数のエリアのネットワークに直接接続する複数のインターフェイスを持つルータです。ABR は OSPF アルゴリズムのコピーを個別に実行し、バックボーン エリアを含む、アタッチされる各エリアに対する個別のルーティング データを保持します。また、ABR はアタッチされたエリアの設定の集約をバックボーン エリアに送り、バックボーン エリアではこの情報を自律システム内の他の OSPF エリアに配布します。図 20 : OSPF ルーティング コンポーネント （395 ページ）には 2 つの ABR があります。ABR 1 はバックボーン エリアに対するエリア 1 のインターフェイスとなります。ABR 2 はスタブ エリアであるエリア 2 に対するバックボーン エリア 0 のインターフェイスとなります。

自律システム境界ルータ（ASBR）

自律システム境界ルータ（ASBR）を使用すると、1 つの自律システムから別のシステムに接続できるようになります。ASBR は自律システム ルーティング情報を他の自律システムの境界ルータと交換します。自律システム内のすべてのルータは、その自律システムの境界ルータに到達する方法を情報として保有しています。

ASBR は、BGP などの他のプロトコルから外部ルーティング情報をインポートして、それらをネットワークに AS-External（ASE）タイプ 5 LSA として再配布できます。Cisco IOS XR ルータが ASBR の場合、コンテンツの VIP アドレスを自律システムの外部ルートとしてアドバタイズするようにルータを設定できます。このようにして、ASBR は OSPF ネットワーク内のルータに外部ネットワークに関する情報をフラッディングします。

ASBR ルートは、タイプ 1 またはタイプ 2 の ASE としてアドバタイズできます。タイプ 1 とタイプ 2 ではコストの計算方法が異なります。タイプ 2 ASE では、同じ宛先への複数パスを比較するとき、外部コスト（メトリック）のみが考慮されます。タイプ 1 ASE では、外部コストと ASBR に到達するためのコストの組み合わせが使用されます。タイプ 2 の外部コストがデフォルトであり、常に OSPF ルートよりコストがかかるため、OSPF ルートが存在しない場合にのみ使用されます。

内部ルータ

内部ルータ（図 20 : OSPF ルーティング コンポーネント （395 ページ） の R1 など）は 1 領域に接続されます（たとえば、すべてのインターフェイスは同じエリアに存在します）。

OSPF プロセスおよびルータ ID

OSPF プロセスは、物理ルータで OSPF を実行している論理ルーティング エンティティです。システム管理者（Cisco IOS XR ソフトウェアの所有者と呼ばれる）が物理ボックスをパーティ

ションで個別のルータに区切ることができる論理ルーティング機能がありますが、その機能とこの論理ルーティング エンティティを混同しないでください。

物理ルータは複数の OSPF プロセスを実行できます。ただし、複数のプロセスを実行するのは、複数の OSPF ドメインに接続する場合のみです。各プロセスにはそれぞれのリンクステート データベースがあります。ルーティング テーブルのルートはリンクステート データベースから計算されます。ルートが再配布されないかぎり、1 つの OSPF プロセスは別の OSPF プロセスとルートを共有しません。

各 OSPF プロセスは、ルータ ID で識別されます。ルータ ID はルーティング ドメイン全体で一意である必要があります。OSPF はルータ ID を優先度の高い順に次の送信元から取得します。

- デフォルトでは、OSPF プロセスが初期化されると、チェックポイント データベースに router-id があるかどうかをチェックします。
- ルータ コンフィギュレーション モードで OSPF router-id コマンドで指定された 32 ビット数値。（この値には任意の 32 ビット値を指定できます。このルータのインターフェイスに割り当てられた IPv4 アドレス以外のアドレスを設定できます。また、ルーティング可能な IPv4 アドレスでなくてもかまいません）。
- ITAL が選択した router-id。
- OSPF プロセスが実行されているインターフェイスのプライマリ IPv4 アドレス。OSPF インターフェイスの最初のインターフェイス アドレスが選択されます。

ルータ コンフィギュレーション モードで router-id コマンドを使用してルータ ID を設定することを推奨します。個別の OSPF プロセスは同じルータ ID を共有できますが、その場合、それらのプロセスは同じ OSPF ルーティング ドメインには存在できません。

サポート対象 OSPF ネットワーク タイプ

OSPF は異なるメディアを次のタイプのネットワークに分類します。

- NBMA ネットワーク
- ポイントツーポイント ネットワーク（POS）
- ブロードキャスト ネットワーク（ギガビット イーサネット）
- ポイントツーマルチポイント

ブロードキャストまたは NBMA ネットワークで Cisco IOS XR ネットワークを設定できます。たとえば、ユーザのネットワークにあるルータでマルチキャストアドレッシングがサポートされない場合に、この機能を使用してブロードキャスト ネットワークを NBMA ネットワークとして設定できます。

OSPF のルート認証方法

OSPF Version 2 は 2 種類の認証（プレーン テキスト認証と MD5 認証）をサポートします。デフォルトでは、認証はイネーブルになっていません（RFC 2178 ではヌル認証と呼ばれます）。

OSPF バージョン 3 では、キーロールオーバーを除くすべてのタイプの認証がサポートされています。

プレーン テキスト認証

プレーンテキスト認証（タイプ 1 認証とも呼ばれる）では、物理メディアを移動するパスワードを使用します。この認証は、アクセス権限を持ないユーザや、ネットワークに接続するパスワードを使用できないユーザでも簡単に見ることができます。そのため、プレーンテキスト認証はセキュリティで保護されません。プレーンテキスト認証はOSPFインターフェイスの誤った実装や設定ミスにより、間違った OSPF パケットが送信されることを防止できる場合があります。

MD5 認証

MD5 認証はセキュリティで保護されます。パスワードは物理メディアに移動されません。その代わり、ルータでは MD5 を使用して、OSPF パケットとキーのメッセージ ダイジェストが生成され、このメッセージ ダイジェストが物理メディアに送信されます。MD5 認証を使用すると、未認証または悪意のあるルーティング アップデートをルータで受け取らないようにできますが、トラフィックを迂回させることによってネットワーク セキュリティが危険にさらされる可能性があります。

（注） MD5 認証では複数のキーがサポートされています。キー番号をキーに関連付ける必要があります。

OSPF 認証のメッセージ ダイジェスト管理。

認証方法

プロセスまたはエリアの全体に対して、またはインターフェイスまたは仮想リンク上に認証を指定できます。インターフェイスまたは仮想リンクには 1 種類の認証だけを設定でき、両方とも設定することはできません。エリアまたはプロセス用に設定されたインターフェイスまたは仮想リンクのオーバーライド認証用に設定された認証。

エリアのすべてのインターフェイスで同じ認証タイプを使用する場合、エリア コンフィギュレーション サブモードで **authentication** コマンドを使用すると（また、エリア全体で MD5 認証と HMAC SHA 256 認証を使用する場合は **message-digest** キーワードを指定すると）、設定するコマンドを少なくすることができます。この方法を使用すると、各インターフェイスに認証を指定するときに必要なコマンドよりも少ないコマンドで設定できます。

キー ロールオーバー

OSPF 隣接関係（およびトポロジ）を中断することなく、操作用ネットワークで MD5 キーを変更するために、キー ロールオーバー メカニズムがサポートされています。ネットワーク管理者が新しいキーを複数のネットワーキング デバイスに設定するとき、異なるデバイスで新しいキーと古いキーの両方が使用されていることがあります。インターフェイスに新しいキーが設定されている場合、ソフトウェアから 2 つの同じパケットのコピーが送信されます。それぞれ

のパケットは古いキーと新しいキーによって認証されます。ソフトウェアではどのデバイスが新しいキーの使用を開始したかを追跡し、すべてのネイバーで新しいキーが使用されていることを検出すると、重複パケットの送信を停止します。次に、ソフトウェアでは古いキーを廃棄します。ネットワーク管理者は、各ルータの各コンフィギュレーションファイルから古いキーを削除する必要があります。

OSPF のネイバーおよび隣接関係

セグメントを共有するルータ（2 つのインターフェイス間のレイヤ 2 リンク）は、そのセグメント上でネイバー同士となります。OSPF では Hello プロトコルをネイバー探索およびキープアライブ メカニズムとして使用します。Hello プロトコルでは定期的に hello パケットを各インターフェイスで送受信します。hello パケットは、インターフェイス上のすべての既知の OSPF ネイバーをリストします。ルータがネイバーの Hello パケット内に自身がリストされていることを認識すると、それらのルータはネイバー同士となります。2 つのルータがネイバーになると、データベースの交換や同期化を行うことができるようになります。これにより、隣接が作成されます。ブロードキャストおよび NBMA ネットワークのすべての隣接ルータに隣接があります。

BFD ダンプニングの OSPF ストリクトモードのサポート

ストリクトモードは、OSPF BFD 操作モードの 1 つで、BFD セッションが起動するまでネイバーをダウン状態に保持します。ネイバーノードのステータスは、BFD セッションの起動待機中として **show ospf neighbor** コマンドの出力に表示されます。これにより、クライアントプロトコルは、BFD の宣言された状態とは無関係に動作しなくなります。

制約事項

- ストリクトモードおよび非ストリクトモードの動作モードは互換性がなく、OSPF がネイバー関係を形成しなくなります。一方のノードでストリクトモードを設定し、もう一方のノードでデフォルトモードまたは非ストリクトモードを設定することはできません。両方の BFD ネイバーが、ストリクトモードをサポートする IOS-XR イメージを実行している必要があります。ただし、設計上、追加の BFD クライアントがすでに BFD セッションを開始していて、OSPF が唯一のイニシエータではない場合、ネイバー関係が形成されることがあります。
- BFD の依存関係により、ストリクトモードで動作している OSPF では、ネイバーの確立と完全な隣接関係が遅延する可能性があります。

ストリクトモードの有効化

ストリクトモードは、異なるレベル（プロセス、エリア、およびインターフェイス）で有効にすることができます。次に、インターエイス上で Open Shortest Path First（OSPF）に対して BFD ストリクトモードを有効にする手順について説明します。

手順の概要

1. **configure**
2. **router ospf** *process-name*
3. **area** *area-id*
4. **interface** *type interface-path-id*
5. **bfd fast-detect strict-mode**
6. **commit**
7. **show ospf interface** *type interface-path-id*

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure** 例： `RP/0/RSP0/cpu 0: router# configure`	グローバル コンフィギュレーション モードを開始します。
ステップ 2	**router ospf** *process-name* 例： `RP/0/RSP0/cpu 0: router(config)# router ospf 1`	OSPF コンフィギュレーションモードを開始します。このモードでは、OSPF ルーティングプロセスの設定を行えます。 現在のルータの process-name を取得するには、EXEC コンフィギュレーション モードで **show ospf** コマンドを使用します。
ステップ 3	**area** *area-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf)# area 0`	Open Shortest Path First（OSPF）領域を設定します。 *area-id* を OSPF エリア識別子に置き換えます。
ステップ 4	**interface** *type interface-path-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf-ar)# interface gigabitEthernet 0/3/0/1`	インターフェイス コンフィギュレーション モードを開始して、インターフェイス名と *rack/slot/module/port* 表記を指定します。 この例では、モジュラ サービス カード スロット 3 にあるギガビット イーサネット インターフェイスを示しています。
ステップ 5	**bfd fast-detect strict-mode** 例： `RP/0/RSP0/cpu 0: router(config-ospf-ar-if)# bfd fast-detect strict-mode`	BFD セッションが起動状態になるまで、ストリクトモードを有効にし、ネイバーセッションをダウンさせておきます。
ステップ 6	**commit**	実行コンフィギュレーションに変更をコミットします。
ステップ 7	**show ospf interface** *type interface-path-id* 例：	適切なインターフェイスでストリクトモードが有効になっていることを確認します。

	コマンドまたはアクション	目的
	RP/0/RSP0/cpu 0: router(config-ospf-ar-if)#show ospf interface gigabitEthernet 0/3/0/1	

BFD ストリクトモード：例

次に、ギガビット イーサネット インターフェイスで OSPF での BFD ストリクトモードを有効にし、OSPF インターフェイス情報を確認する例を示します。BFD ストリクトモードが有効になっている場合、**Mode** の値は **Strict** と表示されます。デフォルトでは、 **Mode** の値が **Default** と表示されます。

```
RP/0/RSP0/cpu 0: router#configure
RP/0/RSP0/cpu 0: router(config)#router ospf 0
RP/0/RSP0/cpu 0: router(config-ospf)#area 0
RP/0/RSP0/cpu 0: router(config-ospf-ar)#interface gigabitEthernet 0/3/0/1
RP/0/RSP0/cpu 0: router(config-ospf-ar-if)#bfd fast-detect strict-mode
RP/0/RSP0/cpu 0: router(config-ospf-ar-if)#commit
RP/0/RSP0/cpu 0: router(config-ospf-ar-if)#end
RP/0/RSP0/cpu 0: router#show ospf interface gigabitEthernet 0/3/0/1

GigabitEthernet0/3/0/1 is up, line protocol is up
  Internet Address 10.1.1.2/24, Area 0
  Process ID 1, Router ID 2.2.2.2, Network Type BROADCAST, Cost: 1
  Transmit Delay is 1 sec, State DR, Priority 1, MTU 1500, MaxPktSz 1500
  BFD enabled, BFD interval 150 msec, BFD multiplier 3, Mode: Strict
  Designated Router (ID) 2.2.2.2, Interface address 10.1.1.2
  No backup designated router on this network
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    Hello due in 00:00:07:358
  Index 1/1, flood queue length 0
  Next 0(0)/0(0)
  Last flood scan length is 1, maximum is 1
  Last flood scan time is 0 msec, maximum is 0 msec
  LS Ack List: current length 0, high water mark 1
  Neighbor Count is 1, Adjacent neighbor count is 0
  Suppress hello for 0 neighbor(s)
  Multi-area interface Count is 0
```

次に、 **show ospf neighbor** コマンドの出力例を示します# は、ネイバーが BFD セッションが起動するのを待機していることを示します。

```
RP/0/RSP0/cpu 0: router#show ospf neighbor

Neighbors for OSPF 1

Neighbor ID     Pri   State           Dead Time   Address         Interface
1.1.1.1         0     DOWN/DROTHER    00:00:33    10.1.1.3/24
GigabitEthernet0/3/0/1#

Total neighbor count: 1
```

OSPF FIB ダウンロード通知

OSPF FIB ダウンロード通知によって、ラインカードのリロード後に入力トラフィックのドロップが長期間にわたって最小化されます。また、この機能はデフォルトで有効になっています。

Open Shortest Path First（OSPF）は、インターフェイステーブル属性ライブラリ（ITAL）を介してルーティング情報ベース（RIB）に登録され、すべてのルートが転送情報ベース（FIB）にダウンロードされるまで、インターフェイスがダウン状態のままになります。OSPF は、リロードされたライン カード上のすべてのルートが RIB/FIB を介してダウンロードされると、インターフェイス アップ通知を取得します。

RIB は、以下の場合に登録クライアントに通知を提供します。

- ノードが失われた。
- ノードが作成された。
- ノードの FIB アップロードが完了した。

OSPF の指定ルータ（DR）

ポイントツーポイント ネットワークおよびポイントツーマルチポイント ネットワーク上では、Cisco IOS XR ソフトウェアによってルーティング アップデートがすぐ隣のネイバーにフラッディングされます。DR またはバックアップ DR（BDR）はありません。すべてのルーティング情報が各ルータにフラッディングされます。

OSPF は、1 つのルータを DR に、もう 1 つのルータを BDR に選択することで、ブロードキャスト セグメントまたは NBMA セグメント上でのみ、セグメント上で交換される情報量を最小化します。このため、セグメント上のルータには、情報交換のための中央接続ポイントがあります。各ルータは、セグメント上の他の各ルータとルーティング アップデートを交換するのではなく、DR および BDR と情報を交換します。DR および BDR は、情報を他のルータに中継します。 ブロードキャスト ネットワーク セグメントでは、ネットワーク セグメントにあるすべての OSPF ルータがでリッスンしているマルチキャスト IP アドレスに、DR および BDR からそれらの OSPF アップデートが送信されることによって、OSPF パケットの数が大幅に削減されます。

ソフトウェアによってセグメント上の各ルータのプライオリティが確認され、DR および BDR となるルータが決定されます。最も高いプライオリティのルータが DR として選択されます。プライオリティが同じ場合、よりの高位ルータ ID を持つルータが優先されます。DR が選択されると、BDR も同様の方法で選択されます。プライオリティが 0 に設定されているルータは、DR または BDR になる資格がありません。

OSPF のデフォルト ルート

タイプ 5（ASE）LSA が生成され、スタブ エリアを除くすべてのエリアにフラッディングされます。スタブ エリアにあるルータから、スタブ エリア外の宛先にパケットをルーティングで

きるようにするために、スタブ エリアにアタッチされている ABR によってデフォルト ルートが挿入されます。

デフォルト ルートのコストは 1 です（デフォルト）。または、default-cost コマンドに指定されている値によって決まります。

OSPF Version 2 のリンクステート アドバタイズメント タイプ

次の各 LSA タイプには、個別の目的があります。

- ルータ LSA（タイプ 1）：1 つのエリア内にルータが持つリンクと各リンクのコストを表します。これらの LSA は、エリア内でのみフラッディングされます。LSA は、QoS（Quality of Service）に基づいてルータがパスを計算できるかどうか、ルータが ABR または ASBR のどちらであるか、ルータが仮想リンクの一端であるかどうかを示します。また、タイプ 1 の LSA は、スタブ ネットワークへのアドバタイズにも使用されます。
- ネットワーク LSA（タイプ 2）：マルチアクセス ネットワーク セグメントにアタッチされているすべてのルータに関するリンク ステートとコストの情報を表します。この LSA ではネットワーク セグメントにアタッチされているインターフェイスを持つすべてのルータを一覧にします。この LSA のコンテンツを生成して追跡するのは、ネットワーク セグメントの指定ルータの仕事です。
- ABR のサマリー LSA（タイプ 3）：他のエリア内のルータ（エリア間ルート）に内部ネットワークをアドバタイズします。タイプ 3 の LSA は、1 つのネットワークを表すことも、1 つのプレフィックスに集約された一連のネットワークを表すこともあります。サマリー LSA を生成するのは ABR だけです。
- ASBR のサマリー LSA（タイプ 4）：ASBR および ASBR に到達するまでのコストをアドバタイズします。外部ネットワークにアクセスしようとするルータは、これらのアドバタイズメントを使用して、ネクスト ホップへの最適パスを決定します。ABR はタイプ 4 LSA を生成します。
- 自律システム外部 LSA（タイプ 5）：別の自律システムからルートを再配布します。通常は別のルーティング プロトコルから OSPF に再配布します。
- 自律システム外部 LSA（タイプ 7）：外部ルート情報を NSSA 内で伝搬するために提供されます。タイプ 7 LSA は NSSA で生成およびアドバタイズできます。NSSA はタイプ 5 LSA を受信または生成しません。タイプ 7 LSA は 1 つの NSSA 内でのみアドバタイズされます。境界ルータによってバックボーン エリアや他のエリアにフラッディングされることはありません。
- 内部エリア プレフィックス LSA（タイプ 9）：ルータは各ルータまたは中継ネットワークに複数の内部エリア プレフィックス LSA を生成できます。それぞれの内部エリア プレフィックス LSA には固有のリンクステート ID があります。それぞれの内部エリア プレフィックス LSA のリンクステート ID には、ルータ LSA またはネットワーク LSA に対する関係と、スタブおよび中継ネットワークのプレフィックスが記されています。
- エリア ローカル スコープ（タイプ 10）：Opaque LSA は関連付けられているエリアの境界を越えてフラッディングされません。

- リンクステート（タイプ 11）：LSA は AS を通してフラッディングされます。タイプ 11 LSA のフラッディング スコープは、AS-External（タイプ 5）LSA のフラッディング スコープと同じです。タイプ 5 LSA と同様、タイプ 11 Opaque LSA がスタブ エリア内の隣接ルータからスタブ エリアに受信されると、LSA は拒否されます。タイプ 11 Opaque LSA には、次のような属性があります。
  - LSA はすべての中継エリアを超えてフラッディングされます。
  - LSA はバックボーンからのスタブ エリアにはフラッディングされません
  - LSA はルータから、ルータが接続されたスタブ エリアには発信されません。

OSPFv3 のリンクステート アドバタイズメント タイプ

次の各 LSA タイプには、個別の目的があります。

- ルータ LSA（タイプ 1）：リンク ステートおよびエリアに対するルータ リンクのコストを表します。これらの LSA は、エリア内でのみフラッディングされます。LSA は、ルータが ABR または ASBR のどちらであるか、および仮想リンクの一端であるかどうかを示します。また、タイプ 1 の LSA は、スタブ ネットワークへのアドバタイズにも使用されます。OSPFv3 では、これらの LSA はアドレス情報を持たず、ネットワーク プロトコルに依存しません。OSPFv3 では、ルータ インターフェイス情報は複数のルータ LSA 間で拡散されます。受信者は、SPF 計算を実行する前に、特定のルータから発信されたすべてのルータ LSA を連結する必要があります。
- ネットワーク LSA（タイプ 2）：マルチアクセス ネットワーク セグメントにアタッチされているすべてのルータに関するリンク ステートとコストの情報を表します。この LSA ではネットワーク セグメントにアタッチされているインターフェイスを持つすべての OSPF ルータを一覧にします。ネットワーク セグメントに選択された指定ルータだけが、セグメントのネットワーク LSA を生成して追跡できます。OSPFv3 では、ネットワーク LSA はアドレス情報を持たず、ネットワーク プロトコルに依存しません。
- ABR のエリア間プレフィックス LSA（タイプ 3）：他のエリア内のルータ（エリア間ルート）に内部ネットワークがアドバタイズされます。タイプ 3 の LSA は、1 つのネットワークを表すことも、1 つのプレフィックスとして集約された一連のネットワークを表すこともあります。ABR はタイプ 3 LSA だけを生成します。OSPFv3 では、これらの LSA のアドレスは「address および mask」ではなく「prefix および prefix length」で表されます。デフォルト ルートは、長さが 0 のプレフィックスとして表現されます。
- ASBR のエリア間ルータ LSA（タイプ 4）：ASBR および ASBR に到達するまでのコストをアドバタイズします。外部ネットワークにアクセスしようとするルータは、これらのアドバタイズメントを使用して、ネクスト ホップへの最適パスを決定します。ABR はタイプ 4 LSA を生成します。
- 自律システム外部 LSA（タイプ 5）：別の自律システムからルートを再配布します。通常は別のルーティング プロトコルから OSPF に再配布します。OSPFv3 では、これらの LSA のアドレスは「address および mask」ではなく「prefix および prefix length」で表されます。デフォルト ルートは、長さが 0 のプレフィックスとして表現されます。

- 自律システム外部 LSA（タイプ 7）：外部ルート情報を NSSA 内で伝搬するために提供されます。タイプ 7 LSA は NSSA で生成およびアドバタイズできます。NSSA はタイプ 5 LSA を受信または生成しません。タイプ 7 LSA は 1 つの NSSA 内でのみアドバタイズされます。境界ルータによってバックボーンエリアや他のエリアにフラッディングされることはありません。
- リンク LSA（タイプ 8）：リンクローカル フラッディング スコープを持ち、関連付けられているリンクを超えてフラッディングすることはありません。リンク LSA は、リンクまたはネットワーク セグメントに接続されている他のすべてのルータに対してルータのリンクローカルアドレスを提供し、リンクに接続されている他のルータに、そのリンクに関連付ける IPv6 プレフィックスのリストを通知します。また、ルータが Options ビットの集まりをアサートして、リンクの起点となるネットワーク LSA と関連付けできるようにします。
- 内部エリア プレフィックス LSA（タイプ 9）：ルータは各ルータまたは中継ネットワークに複数の内部エリア プレフィックス LSA を生成できます。それぞれの内部エリア プレフィックス LSA には固有のリンクステート ID があります。それぞれの内部エリア プレフィックス LSA のリンクステート ID には、ルータ LSA またはネットワーク LSA に対する関係と、スタブおよび中継ネットワークのプレフィックスが記されています。

新しく定義された LSA のほとんどすべてに、アドレス プレフィックスが存在します。プレフィックスは、Prefix Length、Prefix Options、および Address Prefix の 3 つのフィールドで表現されます。OSPFv3 では、これらの LSA のアドレスは「address および mask」ではなく「prefix および prefix length」で表されます。デフォルト ルートは、長さが 0 のプレフィックスとして表現されます。

エリア間プレフィックス LSA およびエリア内プレフィックス LSA では、すべての IPv6 プレフィックス情報が伝送されます。IPv4 ではこの情報はルータ LSA およびネットワーク LSA に含まれます。特定の LSA（ルータ LSA、ネットワーク LSA、エリア間ルータ LSA、およびリンク LSA）の Options フィールドは、IPv6 で OSPF をサポートするために 24 ビットに拡張されています。

OSPFv3 では、エリア間プレフィックス LSA、エリア間ルータ LSA、および自律システム外部 LSA のリンクステート ID の機能は、リンクステート データベースの個々の部分を識別することだけです。OSPF Version 2 ではリンクステート ID で表されたアドレスまたはルータ ID はすべて、OSPFv3 では LSA の本体で伝送されます。

OSPF の仮想リンクおよび中継エリア

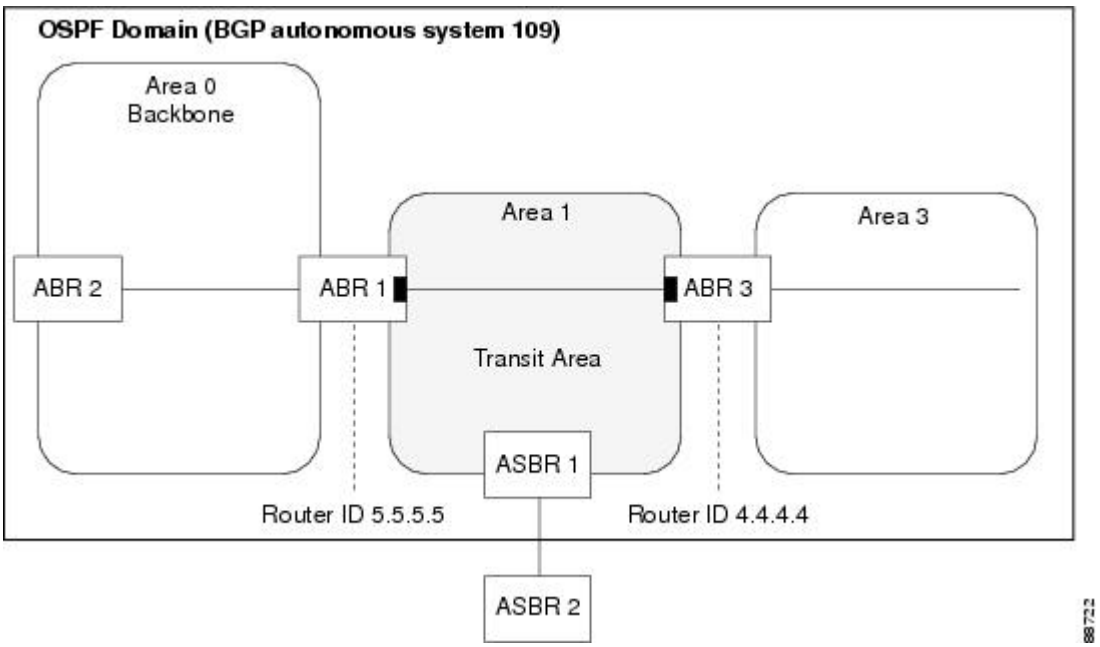
OSPF では、すべてのエリアからのルーティング情報は、ABR によって最初にバックボーンエリアに集約されます。次に、同じ ABR は受信したその情報をアタッチされているエリアに伝播します。このような階層型のルーティング情報の配信では、すべてのエリアがバックボーンエリア（エリア 0）に接続する必要があります。エリアを定義する必要がある場合もありますが、エリア 0 には物理的に接続することはできません。そのような場合の例として、会社で OSPF エリアが含まれる新しい取得を行う場合やエリア 0 自体がパーティション化されている場合が挙げられます。

エリアをエリア 0 に接続できない場合、そのエリアとエリア 0 の間で仮想リンクを設定する必要があります。仮想リンクの 2 つのエンドポイントは ABR であり、仮想リンクは両方のルータで設定する必要があります。2 つのルータが属する、バックボーン以外の共通エリアは中継エリアと呼ばれます。仮想リンクは、他の仮想エンドポイント（他の ABR）の中継エリアとルータ ID を指定します。

仮想リンクはスタブ エリアまたは NSSA から設定することはできません。

図 *21 :* エリア *0* への仮想リンク

この図はエリア 3 からエリア 0 への仮想リンクを示します。



パッシブ インターフェイス

パッシブとしてインターフェイスを設定すると、ネイバーへのルーティングアップデートの送信が無効になるため、隣接関係はOSPF で形成されません。ただし、特定のサブネットはOSPF ネイバーに引き続きアドバタイズされます。インターフェイスでの OSPF プロトコル動作の送信を抑制するには、適切なモードで **passive** コマンドを使用します。

ホストを持つLANセグメントをネットワークの残りに接続しているが、ルータ間のトランジットリンクになるように作られていないインターフェイスでは、パッシブ設定を使用することを推奨します。

MPLS VPN の OSPFv2 模造リンクのサポート

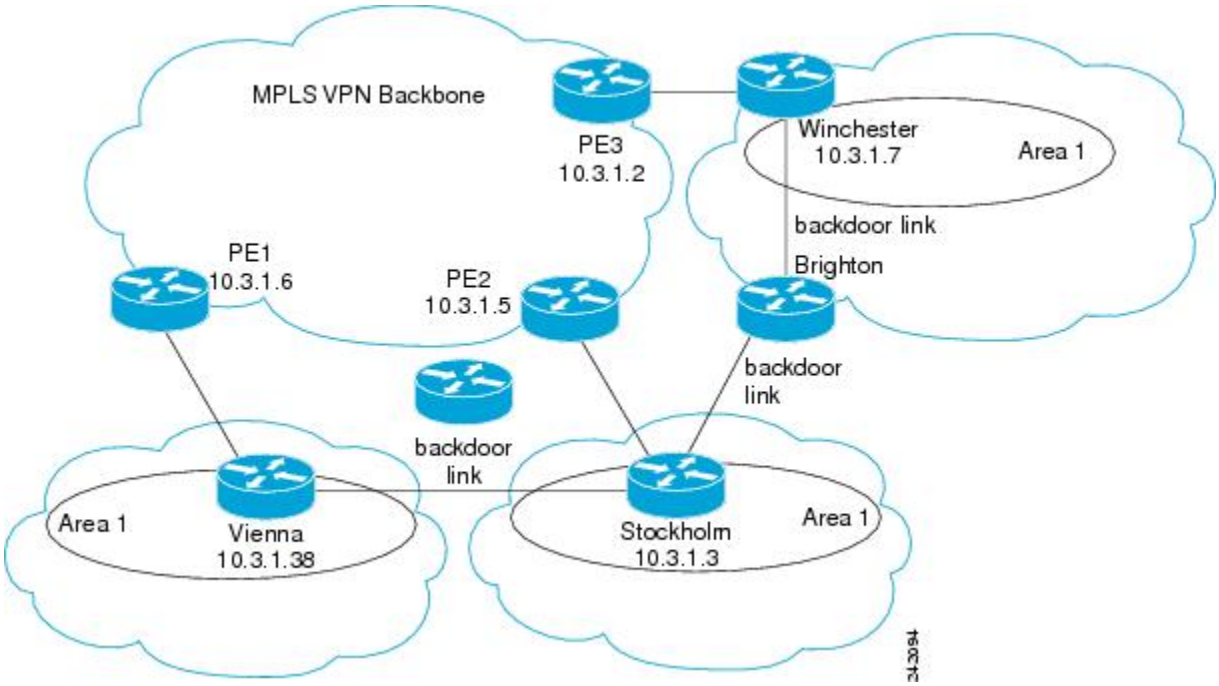
MPLS VPN 環境では、複数の VPN クライアント サイトを同じ OSPF エリアで接続できます。これらのサイトがバックドアリンクを経由して接続され（エリア内リンク）、VPNバックボーンに接続されている場合、プロバイダーエッジルータは VPN バックボーンを経由して学習し

た OSPF ルートを、バックドアリンクを経由してアドバタイズされたエリア内ルートよりも優先順位が低いエリア間ルートまたは外部ルートとしてアドバタイズするため、すべてのトラフィックは、VPN バックボーンではなくバックドアリンクを通過します。

MPLS VPN でのこの OSPF のデフォルトの動作を修正するには、2 つのプロバイダーエッジ（PE）ルータ間に模造リンクを設定して、MPLS VPN バックボーンを介してサイトを接続します。模造リンクは、PE ルータ間のエリア内（番号なしのポイントツーポイント）接続を表します。エリア内のその他すべてのルートは模造リンクを確認して、リモートサイトへのエリア内 Shortest Path First（SPF）ルートを計算するために使用します。トラフィックがバックドアリンクと模造リンクのどちらで送信されるかを決定するために、各模造リンクとともにコストを設定する必要があります。

設定された送信元と宛先のアドレスは、模造リンクのエンドポイントとして機能します。送信元と宛先の IP アドレスは VRF に属し、ボーダーゲートウェイ プロトコル（BGP）によってホストルートとしてリモート PE ルータにアドバタイズされる必要があります。模造リンク エンドポイント アドレスは、OSPF によってアドバタイズされないことが必要です。

図 *22 : OSPF* クライアント サイト間のバックドア パス



たとえば、図 22 : OSPF クライアント サイト間のバックドア パス （408 ページ）には 3 つのクライアント サイトがあり、それぞれにバックドア リンクがあります。各サイトはエリア 1 コンフィギュレーション内で OSPF を実行するため、サイト間のすべてのルーティングは MPLS VPN バックボーンではなく、バックドア リンク間のエリア内パスに従います。

サイト間のバックドア リンクがバックアップの目的でのみ使用される場合、望ましくないトラフィック フローが作成されるため、バックボーン リンクを介するデフォルト ルートの選択は受け入れられません。MPLS バックボーンを介して目的のパス選択を確立するには、イングレ

スとイーグレス PE ルータ間に追加の OSPF エリア内（模造リンク）リンクを作成する必要があります。

模造リンクが必要なのは、同じ OSPF エリアに属し、OSPF バックドア リンクを共有する 2 つの VPN サイト間です。サイト間にバックドア リンクがない場合、模造リンクは不要です。

図 *23* : 接続されている *OSPF* クライアント サイトへの *PE* ルータ間の模造リンク

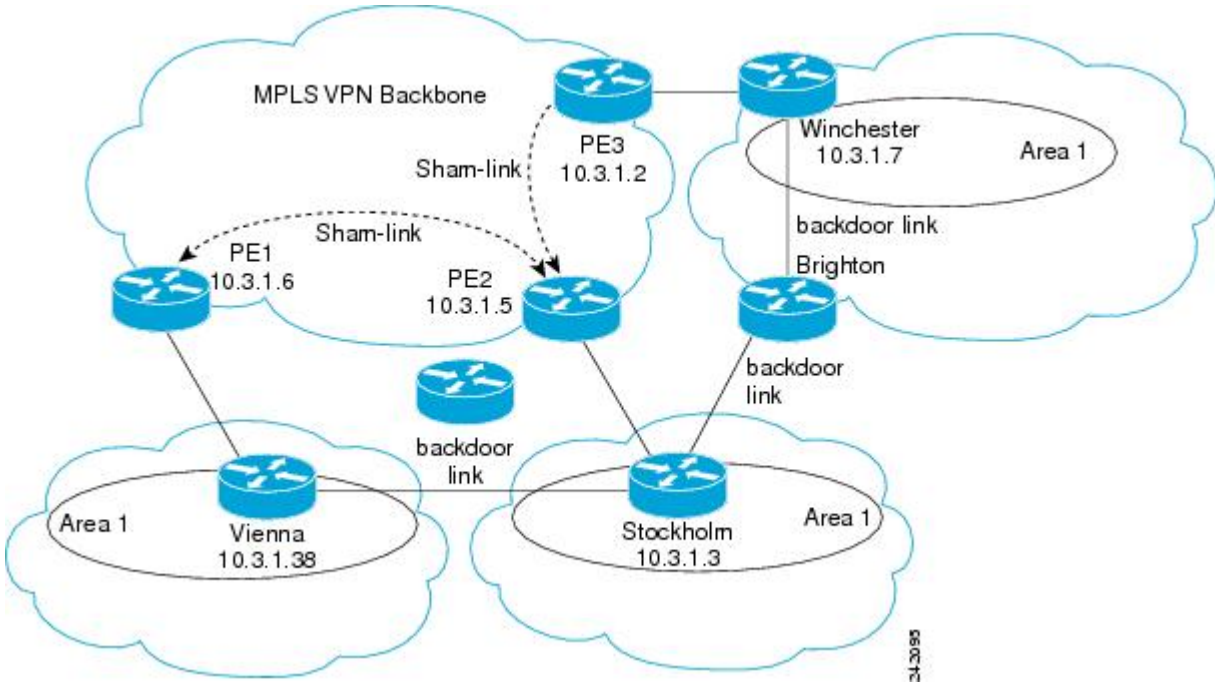


図 23 : 接続されている OSPF クライアント サイトへの PE ルータ間の模造リンク （409 ページ）には、模造リンク設定が必要な MPLS VPN トポロジが示されています。VPN クライアントには 3 つのサイトがあり、それぞれにバックドア リンクがあります。1 つは PE-1 と PE-2 の間、もう 1 つは PE-2 と PE-3 の間に、2 つの模造リンクが設定されています。PE-1 と PE-3 の間に模造リンクは必要ありません。これは、これらのサイト間にはバックドア リンクがないためです。

PE ルータ間に模造リンクが設定されてる場合、PE ルータは模造リンクを介して学習した OSPF ルートによって仮想ルーティングおよび転送（VRF）テーブルを入力します。これらの OSPF ルートには、BGP ルートよりも大きいアドミニストレーティブディスタンスがあります。BGP ルートが利用可能な場合は、より大きいアドミニストレーティブディスタンスを持つこれらの OSPF ルートよりも優先されます。

MPLS VPN の OSPFv3 模造リンクのサポート

OSPFv3 模造リンクは、2 つの PE 間の単一ポイントツーポイント接続としての VPN バックボーンを表します。OSPFv3 では、仮想リンクと同様に、ポイントツーポイントの番号なしインターフェイスとして模造リンクを処理します。OSPFv3 模造リンクを設定する際は、模造リンクのリモートエンドポイントへのルートが VRF RIB に存在することを確認します。

リモートエンドポイントへのルートが存在している場合は、模造リンクインターフェイスが起動します。模造リンクのリモートエンドポイントへのルートがVRF RIBから削除された場合、OSPFv3 は再配布コールバックを受信し、模造リンクをダウンさせます。

模造リンクを介したグレースフルリスタートの手順

OSPFv3 は、スイッチまたはプロセスの再起動時に、模造リンクを他のインターフェイスとして処理します。OSPFv3 は、設定されているすべての模造リンクが稼働していることを前提としており、それらに隣接関係を形成しようとします。

スイッチオーバーの前に模造リンクがダウンしている場合、OSPFv3 はHello パケットをリモートエンドポイントに送信します。最終コンバージェンス信号を RIB から受信すると、OSPFv3 は、RIB 内に設定された各模造リンクの BGP ルートに基づいて、模造リンクをアップまたはダウンのいずれかで保持します。

OSPFv3 は、BGP コンバージェンスが完了した後にのみ、模造リンクを介して上位の AD ルートをインストールします。

ECMP と OSPFv3 の模造リンク

等コストマルチパス（ECMP）メカニズムは、プレフィックスに複数の iBGP パスがある場合に、模造リンク上のトラフィックのロードバランシングに使用されます。模造リンクパスとバックドアパスのコストが同じである場合、模造リンクパスとバックドアパスの間のECMP はサポートされません。

OSPF SPF プレフィックスのプライオリティ設定

OSPF SPF のプレフィックスのプライオリティ設定機能によって、ルートのインストール中に、高速モードで、管理者が重要なプレフィックスを収束できます。

多くのプレフィックスがルーティング情報ベース（RIB）および転送情報ベース（FIB）にインストールされる必要がある場合、SPF 中の、最初のプレフィックスから最後のプレフィックスまでの更新期間が、かなりの長さになることがあります。

時間依存のトラフィック（VoIP など）が他のトラフィック フローとともに同じルータを通過する可能性があるネットワークでは、SPF 中の、これらの時間に依存するプレフィックスの RIB および FIB アップデートを優先することが重要です。

OSPF SPF のプレフィックスのプライオリティ設定機能によって、SPF 計算中に RIB にインストールされる重要なプレフィックスに、管理者が優先順位を付けることが可能になります。重要なプレフィックスは、領域ごとに同じルート タイプのプレフィックス内で高速で収束します。RIB およびFIB のインストール前に、ルートとプレフィックスは指定したルート ポリシーに基づいて OSPF ローカル RIB のさまざまなプライオリティ バッチ キューに割り当てられます。RIB プライオリティ バッチ キューはプライオリティの高い順から「critical」、「high」、「medium」、「low」に分類されます。

イネーブルの場合、次のプレフィックス プライオリティで RIB 更新シーケンスが変更されます。

Critical > High > Medium > Low

プレフィックス プライオリティが設定されると、デフォルトでは/32プレフィックスは優先されなくなり、より高いプライオリティポリシーに一致しない場合は、lowプライオリティキューに配置されます。ルート ポリシーは、/32 が高いプライオリティのキュー（High プライオリティ、または Medium プライオリティ）に保持されるように考案する必要があります。

プライオリティはルート ポリシーを使用して指定されます。このルート ポリシーは、IP アドレスまたはルート タグに基づいて照会することができます。SPF 中に、指定したルート ポリシーに対してプレフィックスがチェックされ、適切な RIB バッチ プライオリティ キューに割り当てられます。

これらは、このシナリオの例です。

- high プライオリティ ルート ポリシーだけを指定した場合は、medium プライオリティに対してルート ポリシーは設定されません。
  - 許可されたプレフィックスは、high プライオリティ キューに配置されます。
  - /32 を含む一致しないプレフィックスは、lowプライオリティキューに配置されます。
- high プライオリティと medium プライオリティの両方のルート ポリシーが指定され、critical プライオリティにマップが指定されない場合
  - high プライオリティのルートポリシーに一致する許可されたプレフィックスは、high プライオリティ キューに配置されます。
  - medium プライオリティのルート ポリシーに一致する許可されたプレフィックスは、medium プライオリティ キューに配置されます。
  - /32 を含む一致しないプレフィックスは、lowプライオリティキューに移動されます。
- critical プライオリティと high プライオリティの両方のルート ポリシーが指定されており、medium プライオリティにマップが指定されていない場合
  - critical プライオリティのルート ポリシーに一致する許可されたプレフィックスは、critical プライオリティ キューに配置されます。
  - high プライオリティのルートポリシーに一致する許可されたプレフィックスは、high プライオリティ キューに配置されます。
  - /32 を含む一致しないプレフィックスは、lowプライオリティキューに配置されます。
- medium プライオリティ ルートポリシーだけが指定され、high プライオリティまたは critical プライオリティにマップが指定されていない場合
  - medium プライオリティのルート ポリシーに一致する許可されたプレフィックスは、medium プライオリティ キューに割り当てられます。
  - /32 を含む一致しないプレフィックスは、lowプライオリティキューに配置されます。

[no] spf prefix-priority route-policy *rpl* コマンドを使用して、SPF 中に OSPF プレフィックス インストールのプライオリティをグローバル RIB で設定します。

SPF プレフィックスのプライオリティ設定は、デフォルトではディセーブルです。ディセーブル モードでは、/32 プレフィックスは他のプレフィックスよりも前にグローバル RIB にインストールされます。SPF プライオリティ設定がイネーブルの場合、ルートは route-policy 基準に対して照会され、SPF プライオリティ セットに基づいて適切なプライオリティ キューに割り当てられます。/32 を含む一致しないプレフィックスは、low プライオリティのキューに配置されます。

すべての /32 を high プライオリティ キューまたは medium プライオリティ キューで処理する必要がある場合、次の 1 つのルート マップを設定します。

```
prefix-set ospf-medium-prefixes
  0.0.0.0/0 ge 32
  end-set
```

OSPF のルート再配布

再配布により、異なるルーティング プロトコルを使用してルーティング情報を交換できます。この手法を使用すると、複数のルーティング プロトコルに接続を広げることができます。**redistribute** コマンドでは、OSPF からの再配布ではなく、OSPF プロセスへの再配布が制御されることに注意することが重要です。OSPF のルート再配布の例については、OSPF の実装の設定例 （481 ページ）を参照してください。

OSPF Shortest Path First スロットリング

OSPF SPF スロットリングにより、SPF スケジューリングをミリ秒間隔で設定して、ネットワークが不安定な場合に SPF 計算を遅らせることができます。トポロジ変化が発生した場合、Shortest Path Tree（SPT）を再計算するように SPF がスケジューリングされます。SPF が 1 回実行されると、複数のトポロジ変化イベントが発生します。

SPF 計算の実行間隔は、ネットワークのトポロジ変化の頻度に応じて動的に選択されます。ユーザ指定値の範囲内で、間隔は選択されます。ネットワーク トポロジが不安定な場合、トポロジが安定するまで、SPF スロットリング機能は SPF スケジューリング間隔を長目に計算します。

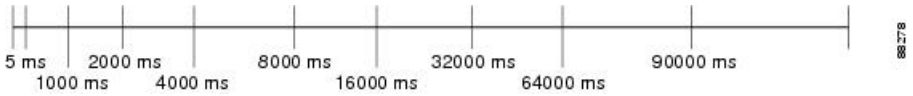
SPF の計算は、timers throttle spf コマンドで設定した間隔で実行されます。待機期間とは、次の SPF 計算が実行されるまで待機する時間のことです。計算を行うたびに、待機期間はその前の期間の 2 倍の長さになり、指定された最大待機期間に達するまでそれが行われます。

SPF タイミングについて、例を使用して説明します。この例では、開始時の間隔は 5 ミリ秒（ms）、初回待機時間は 1000 ミリ秒、最大待機期間は 90,000 ミリ秒に設定されます。

```
timers spf 5 1000 90000
```

図 *24 : timers spf* コマンドで設定される *SPF* の計算間隔

次の図に、ある待機期間中に少なくとも 1 回のトポロジ変化イベントを受信する場合の、SPF 計算の実行間隔を示します。

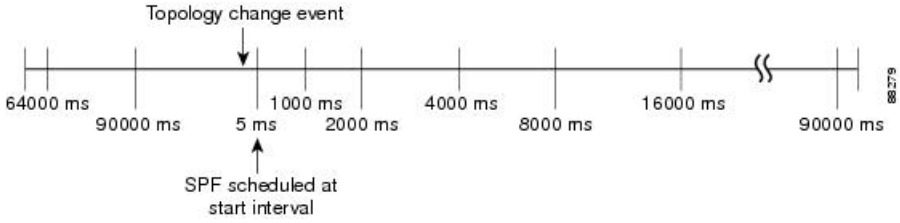


前の待機期間中に少なくとも 1 回のトポロジ変化イベントを受信すると、SPF 計算の待機期間が 2 倍になることに注意してください。最大待機期間に達すると、トポロジが安定し、待機期間中にイベントを受信しなくなるまで、待機期間が変化しなくなります。

現在の待機期間の経過後に、最初のトポロジ変化イベントを受信した場合は、開始時待機期間として指定されている時間だけ SPF 計算が遅延されます。その後の待機期間は、動的パターンに従います。

最大待機期間の開始後に、最初のトポロジ変化イベントが発生した場合、SPF 計算は開始時待機期間で再びスケジューリングされ、その後の待機期間は timers throttle spf コマンドで指定されたパラメータに従ってリセットされます。図 25 : トポロジ変化イベント後のタイマー間隔のリセット （413 ページ） では、最大待機期間の開始後にトポロジ変化イベントを受信して、SPF 間隔がリセットされることに注意してください。

図 *25 :* トポロジ変化イベント後のタイマー間隔のリセット



OSPF Version 2 のノンストップ フォワーディング

OSPF バージョン 2 用の Cisco IOS XR ソフトウェア NSF では、プロセスの再起動 またはフェールオーバー後にルーティングプロトコル情報を保存しながら、既知のルートを通ってデータパケットの転送が継続されるようにできます。NSF を使用すると、ピア ネットワーキング デバイスでルーティング フラップが発生しません。プロセスの再起動またはフェールオーバー中、データトラフィックはインテリジェント ラインカードを介して転送されますが、スタンバイ ルートプロセッサ（RP）では、障害が発生した RP からの制御と見なします。プロセスの再起動中にラインカードのアップ状態が維持され、アクティブ RP の転送情報ベース（FIB）が最新状態に維持される機能が、Cisco IOS XR ソフトウェア NSF の動作にとって非常に重要です。

OSPF などのルーティング プロトコルは、アクティブ RP または DRP 上でのみ実行され、隣接ルータからルーティング アップデートを受信します。OSPF NSF 対応ルータがプロセスの再起動を実行する場合、リンクステートデータベースを OSPF ネイバーと再同期するために、次の 2 つのタスクを実行する必要があります。まず、ネイバー関係をリセットせずに、ネットワーク上の使用可能な OSPF ネイバーを再学習します。次に、ルータはネットワークのリンクステート データベースのコンテンツを再取得します。

RP フェールオーバー後またはプロセスの再起動後にできるだけ迅速に NSF 対応ルータは OSPF NSF 信号を隣接する NSF 対応デバイスに送信します。この信号はフェールオーバー ルータで生成されたリンクローカル LSA の形式になります。ネイバー ネットワーキング デバイスは、この信号をこのルータとのネイバー関係がリセットされるべきでないことを示す指示として認識します。NSF 対応ルータがネットワーク上の他のルータから信号を受信すると、ネイバー リストの再構築を始めます。

ネイバー関係が再構築されると、NSF 対応ルータはすべての NSF 認識ネイバーとデータベースの再同期化を始めます。この時点でルーティング情報は OSPF ネイバーの間で交換されます。交換が完了すると、NSF 対応デバイスはルーティング情報を使用して、失効ルートを削除し、RIB を更新して、新しい転送情報で FIB を更新します。ルータおよび OSPF ネイバー上の OSPF が完全にコンバージされるようになりました。

OSPFv3 のグレースフルシャットダウン

OSPFv3 グレースフル シャットダウン機能により、次の状況でもデータ プレーン機能を維持することができます。

- RP 障害。その結果、バックアップ プロセッサにスイッチオーバーされます
- ソフトウェアのアップグレードまたはダウングレードに伴う再起動などの、計画された OSPFv3 プロセスの再起動
- プロセスのクラッシュに伴う再起動などの、予期しない OSPFv3 プロセスの再起動

また、プロセッサが使用可能なメモリで非常に低いことを示す重大なメモリ イベントが sysmon のウォッチ ドッグ プロセスから受信された場合、OSPFv3 は一方的にシャットダウンするか、または終了状態に入ります。

この機能を使うと、OSPFv3 ルーティング プロトコルが再起動している間に、確立されているルートでノンストップ データ転送が行われます。そのため、この機能により IPv6 転送の可用性が向上します。

グレースフル リスタート操作のモード

ルータがこの機能に使用できる動作モードは、再起動モード、ヘルパー モード、およびプロトコル シャットダウン モードです。

リスタート モード

OSPFv3 プロセスが開始されたときに、グレースフル リスタートを試行する必要があるかどうかを決定します。決定は、グレースフル リスタートがそれまでにイネーブルされているかどうかに基づきます。（OSPFv3 は、ルータの初回の起動時にグレースフル リスタートを試行しません）。OSPFv3 グレースフル リスタートを有効にすると、RIB の消去タイマーがゼロ以外の値に変わります。グレースフルリスタートを有効にして設定する方法については、OSPFv3 グレースフル リスタートの設定 （454 ページ）を参照してください。

グレースフル リスタート中、ルータは OSPFv3 ルートを RIB に入力しません。ルータは再起動前に OSPFv3 が保有していた完全に隣接するネイバーとの完全な隣接関係を立ち上げようと

します。最終的に、OSPFv3 プロセスは、（何らかの理由により）グレースフル リスタートを終了するため、または、グレースフル リスタートを終了したため、プロセスがコンバージされたことを RIB に示します。

再起動モードに関する一般的な詳細を次に示します。動作、特定の制約事項、要件に関するより詳しい情報は、グレースフル リスタートの要件と制約事項（417 ページ）の項に記されています。

- 最後の再起動から間を空けずに OSPFv3 が再起動を試みると、OSPFv3 プロセスは頻繁に繰り返しクラッシュするようになり、新しいグレースフルリスタートの実行が停止します。グレースフルリスタートの許可間隔を制御するには、graceful-restart interval コマンドを使用します。
- 起動する最初のインターフェイスで OSPFv3 がグレースフル リスタートを開始すると、グレースフル リスタートの期間（有効期間）を制限するためにタイマーが起動します。graceful-restart lifetime コマンドを使用して、この期間を設定できます。起動する各インターフェイスで *grace* LSA（タイプ 11）がフラッディングされ、このルータがグレースフル リスタートを試みていることを隣接ルータに示します。ネイバーはヘルパー モードを開始します。
- 再起動中のネイバーから受信した hello パケットの指定ルータとバックアップ指定ルータ パケットの指定ルータ チェックは正しくないため、バイパスされます。

ヘルパー モード

ヘルパー モードは、デフォルトでイネーブルになっています。グレースフル リスタートを試みているルータから（ヘルパー）ルータが grace LSA（タイプ 11）を受け取ると、次のイベントが発生します。

- graceful-restart helper disable コマンドによりヘルパー モードがディセーブルされている場合、ルータは LSA パケットをドロップします。
- ヘルパー モードがイネーブルの場合、次の条件がすべて満たされると、ルータはヘルパー モードを開始します。
  - ローカル ルータ自体がグレースフル リスタートを試みていない。
  - ローカル（ヘルパー）ルータに送信先ネイバーとの完全な隣接関係がある。
  - 受信した LSA の *lsage*（リンク ステートの経過時間）の値が、要求された猶予期間よりも短い。
  - grace LSA の送信元が grace LSA の生成元と同じである。
- ヘルパー モードを開始すると、ルータは一定期間そのヘルパー機能を実行します。この期間は再起動モードにあるルータの有効期間の値から、受信した grace LSA の *lsage* の値を引いた値です。グレースフル リスタートが時間内に成功すると、ヘルパー タイマーが期限切れになる前に停止します。ヘルパー タイマーの期限が切れた場合、再起動しているルータへの隣接関係がダウンし、通常の OSPFv3 機能が再開します。
- デッド タイマーはヘルパー モードにあるルータでは使用できません。

- 次のいずれかの場合に、ヘルパー モードにあるルータはヘルパー機能の実行を停止します。
  - ヘルパー ルータが再起動中のルータとの完全な隣接関係を起動できる。
  - ヘルパー機能のローカル タイマーの有効期限が切れている。

プロトコル シャットダウン モード

このモードでは、OSPFv3 操作は完全に無効になっています。これは、自己生成リンク ステートアドバタイズメント（LSA）をフラッシュすることで達成され、ローカルの OSPFv3 対応インターフェイスが即座に停止し、リンク ステートデータベース（LSDB）がクリアされます。ローカル以外の LSDB エントリは OSPFv3 によって削除され、フラッディング（MaxAged）されません。

プロトコル シャットダウン モードは、**protocol shutdown** コマンドを使用（プロトコル インスタンスが無効になります）して手動で起動できます。または OSPFv3 プロセスのメモリが不足すると起動します。次のイベントは、プロトコル シャットダウンが実行されると発生します。

- ローカル ルータ LSA およびすべてのローカル リンク LSA がフラッシュされます。他の LSA はすべて、ドメイン内の他の OSPFv3 ルータによって最終的にエージ アウトされます。
- まだローカル ルータとフル状態になっていない OSPFv3 ネイバーは、Kill_Nbr イベントとともに停止します。
- 3 秒の遅延後、空の Hello パケットはアクティブな隣接関係がある各ネイバーに即座に送信されます。
  - 空の Hello パケットは、dead_interval が経過するまで定期的に送信されます。
  - dead_interval が経過すると、Hello パケットは送信されなくなります。

Dead Hello インターバルの遅延（4 X Hello インターバル）後、次のイベントが実行されます。

- その OSPFv3 インスタンスからの LSA データベースがクリアされます。
- OSPFv3 によってインストールされた RIB からのすべてのルートが消去されます。

ルータは、プロトコル シャットダウン状態時にネイバーから受信するいずれの OSPF 制御パケットにも応答しません。

プロトコルの復元

プロトコルを復元する方法は、シャットダウンを最初に引き起こしたトリガーに依存します。OSPFv3 が **protocol shutdown** コマンドを使用してシャットダウンされた場合、OSPFv3 を通常の動作に復元するには **no protocol shutdown** コマンドを使用します。OSPFv3 が sysmon からの重要なメモリ メッセージによってシャットダウンされた場合は、十分なメモリがプロセッサに復元されたことを示す sysmon からの通常のメモリ メッセージによって OSPFv3 プロトコルが復元され、通常の動作が再開されます。OSPFv3 が重要なメモリ トリガーによってシャットダ

ウンされた場合は、通常のメモリ レベルがルート プロセッサで復元された際に、手動で再起動する必要があります。これは自動的に復元されません。

次のイベントは、OSPFv3 が復元されると発生します。

1. すべてのOSPFv3インターフェイスが、Helloパケットとデータベース交換を使用してバックアップされます。
2. ローカル ルータおよびリンク LSA が再作成され、アドバタイズされます。
3. ルータは、ネイバーから受信したすべての OSPFv3 制御メッセージに正常に応答します。
4. 他の OSPFv3 ルータから学習されたルートが RIB にインストールされます。

グレースフル リスタートの要件と制約事項

グレースフル リスタート機能をサポートするための要件には、次のようなものがあります。

- グレースフル リスタート中にルータのネイバーと連携します。OSPFv3 が再起動しているルータに対して、各ルータはヘルパーと呼ばれます。
- グレースフルリスタートを実行するルータのすべてのネイバーは、グレースフルリスタートを実行できる必要があります。
- はじめてルータを起動するときには、グレースフル リスタートは実行されません。
- OSPFv3 ネイバー情報とデータベース情報ではチェックポイントが行われません。
- OSPFv3 プロセスは再起動後に隣接関係を再構築します。
- 再起動してもデータベースの一貫性を確保するには、再起動前に OSPFv3 コンフィギュレーションを同じにする必要があります（この要件は、ローカルデータベース内の自己生成情報に適用されます）。動作中に設定が変更されると、グレースフルリスタートが失敗する可能性があります。この場合、データ転送にも影響を与えます。OSPFv3 はすべての LSA を再生成して、データベースをすべてのネイバーと再同期させることによって、操作を再開します。
- グレースフル リスタート中に IPv6 FIB テーブルは変更されませんが、これらのテーブルでは最終的にホールドダウン タイマーを使用して、失効としてルートをマークします。プロトコルには、状態情報とコンバージを再構築するために十分な時間が許されています。
- OSPFv3 を再起動中のルータは、プロセス再起動のデッド インターバル内に OSPFv3 hello を送信する必要があります。隣接関係のデッドタイマーの有効期限が切れる前に、プロトコルはネイバーとの隣接関係を保持できるようになる必要があります。デッドタイマーのデフォルトは 40 秒です。デッド タイマーの有効期限が切れる前に hello が隣接関係に到達しない場合、ルータは隣接関係を切断します。OSPFv3 プロセスの再起動後に hello を送信するために必要な時間よりもデッド タイマーが短く設定されていると、OSPFv3 グレースフル リスタート機能は適切に機能しません。
- 複数ルータでの同時グレースフル リスタート セッションは、1 つのネットワーク セグメントではサポートされていません。複数ルータが再起動モードにあることをルータが判別すると、すべてのローカル グレースフル リスタート操作を停止します。

- この機能では、ルーティング情報ベース（RIB）にある既存の OSPFv3 ルートのパージ時間の変更に利用可能なサポートを活用します。グレースフルリスタートが有効になっている場合、パージタイマーはデフォルトで 90 秒に設定されます。グレースフルリスタートが無効である場合、パージタイマー設定は 0 です。
- この機能には、関連付けられているグレース LSA があります。このリンクスコープ LSA はタイプ 11 です。
- RFC には、OSPFv3 プロセスは再起動中にすべての古い自動送信 LSA をフラッシュする必要があると記されています。ただし、グレースフル リスタート機能を使用すると、ルータはグレースフル リスタート中にこの不明の自動送信 LSA のフラッシュを遅らせます。OSPFv3 は新しい情報を学習して、新しい LSA を構築し、古い LSA と置き換えることができます。遅延が終了すると、すべての古い LSA がフラッシュされます。
- グレースフルリスタートが有効になっている場合、すべてのネイバーの隣接関係の作成時間がシステムデータベース（SysDB）に保存されます。作成時間の保存目的は、OSPFv3 が元の隣接関係作成時間を使用して、再起動後にそのネイバーの稼働時間を遅延できるようにすることです。

OSPF Version 2 のウォーム スタンバイとノンストップ ルーティング

OSPFv2 ウォーム スタンバイは、RP のスイッチオーバー全体でハイ アベイラビリティを実現します。ウォーム スタンバイ拡張機能により、アクティブ RP で実行されているプロセスごとに、スタンバイ RP で開始された、対応するスタンバイ プロセスがあります。スタンバイ OSPF プロセスは、アクティブな OSPF プロセスにパフォーマンスへ影響を与えることなく、OSPF パケットを送受信できます。

ノンストップ ルーティング（NSR）によって、RP フェールオーバー、プロセスの再起動、またはインサービス アップグレードはピア ルータから見えなくなり、パフォーマンスまたは処理への影響が最小限になります。ルーティング プロトコルはルータ間でやり取りされるため、NSR の影響を受けません。NSR はウォーム スタンバイ拡張機能によって構築されます。NSR を使用すると Cisco NSF および IETF グレースフル リスタート プロトコル拡張機能の要件が緩和されます。

NSR は OSPF ではデフォルトで有効になっています。NSR を無効にするには、OSPF コンフィギュレーション モードで **nsr disable** コマンドを使用します。

（注） Hello タイマーの間隔はデフォルトの 10 秒に設定することをお勧めします。設定された Hello 間隔タイマーがデフォルト値より小さい場合、スイッチオーバー中に OSPF セッションがフラップすることがあります。

OSPF バージョン 3 のウォームスタンバイ

この機能を使うと、フェールオーバー（FO）の前にOSPFv3が自動で初期化され、障害が発生する前に機能する準備が整います。また、スイッチオーバー中のダウンタイムを減らすことができます。デフォルトでは、ルータは hello パケットを 40 秒ごとに送信します。

各 OSPF プロセスのウォーム スタンバイ プロセスが、アクティブ ルート プロセッサで実行されている場合、対応する OSPF プロセスはスタンバイ RP で開始する必要があります。この機能のためにコンフィギュレーションを変更する必要はありません。

ウォーム スタンバイは常にイネーブルです。この機能は、IGP としてOSPFv3 を実行しているシステムが RP フェールオーバーを実行するときに有利です。

OSPF の multicast-intact サポート

multicast-intact 機能を使用すると、IGP ショートカットがルータに設定されアクティブな場合に、マルチキャスト ルーティング（PIM）を実行できます。OSPFv2 および IS-IS の両方で multicast-intact 機能がサポートされています。

IGP の multicast-intact は、マルチキャスト ルーティング プロトコル（PIM）と IGP ショートカットがルータで設定されている場合に有効にすることができます。IGP ショートカットは IGP に公開される MPLS トンネルです。IGP はこれらのトンネルを介して、（SPF を基点として）トンネルの出力ルータからのダウンストリームである宛先に IP トラフィックを送信します。PIM は PIM Join を伝播するために IGP ショートカットを使用できません。これは、リバース パス転送（RPF）が単方向トンネルでは機能しないためです。

multicast-intact を IGP で有効にすると、IGP は PIM が使用するパラレル等コストネクストホップまたは代替等コストネクストホップをパブリッシュします。これらのネクストホップは *mcast-intact* ネクストホップと呼ばれます。mcast-intact ネクスト ホップには次の属性があります。

- IGP のショートカットが含まれていないことが保証されます。
- ユニキャスト ルーティングには使用されませんが、PIM によってのみ PIM 送信元への IPv4 ネクスト ホップの検索に使用されます。
- FIB には公開されません。
- multicast-intact が IGP で有効になっている場合、リンクステート アドバタイズメントによって学習されたすべての IPv4 宛先は、RIB への等コスト mcast-intact ネクストホップのセットとともにパブリッシュされます。この属性は、ネイティブ ネクスト ホップに IGP ショートカットがない場合にも適用されます。

OSPF では、最大パス（等コスト ネクスト ホップの数）制限は、ネイティブ ネクスト ホップおよびmcast-intact ネクスト ホップに個別に適用されます。等コスト mcast-intact ネクスト ホップの数は、ネイティブ ネクスト ホップに設定されている数と同じです。

OSPF Version 2 および OSPFv3 でのロード バランシング

ルータは、複数のルーティング プロセス（またはルーティング プロトコル）を使用して特定のネットワークへの複数のルートを認識すると、最短のアドミニストレーティブディスタンスを持つルートをルーティング テーブルにインストールします。同じアドミニストレーティブディスタンスを持つ同じルーティングプロセスを使用して認識された多数のルートから、1つのルートを選択する必要があることもあります。この場合、ルータはその宛先へのコスト（またはメトリック）が最も小さいパスを選択します。各ルーティングプロセスはコストをそれぞれの方法で計算します。コストは、ロードバランシングを実現するために処理が必要なこともあります。

OSPF では、自動的にロードバランシングが実行されます。OSPF により、複数のインターフェイスを通って宛先に到達できること、および各パスのコストが同じであることが検出された場合は、ルーティング テーブルに各パスがインストールされます。同じ宛先へのパスの数は、**maximum-paths**（OSPF） コマンドを指定しない限り、制限されません。

最大パスの範囲は 1 から 8 です。デフォルトの最大パスの数は 8 です。

OSPF Version 2 のマルチエリアの隣接関係

OSPFv2 のマルチエリアの隣接関係機能を使うと、マルチエリアのプライマリ インターフェイスにリンクを設定できるため、リンクをこれらのエリアのエリア内リンクと見なすことができ、より高価なパスより優先されるパスとして設定できます。

この機能は、ポイントツーポイントのアンナンバードリンクを OSPF エリアに確立します。ポイントツーポイント リンクを使うと、そのエリアのトポロジ パスを利用でき、プライマリ隣接関係ではそのリンクを使用して、draft-ietf-ospf-multi-area-adj-06 と同じリンクをアドバタイズします。

マルチエリア インターフェイスの属性と制限を次に示します。

- OSPF の既存のプライマリ インターフェイス上の論理構成体として存在しますが、プライマリ インターフェイス上のネイバー ステートは、マルチエリア インターフェイスと無関係です。
- 隣接ルータ上の対応するマルチエリア インターフェイスとの隣接関係を確立します。マルチエリアとプライマリ インターフェイスの混在はサポートされていません。
- ネイバー ステートがフルの場合、ルータ リンク ステート アドバタイズメント（LSA）のアンナンバードポイントツーポイント リンクを、対応するエリアにアドバタイズします。
- ポイントツーポイント ネットワーク タイプとして作成されます。OSF スピーカーが 2 つだけアタッチされている任意のインターフェイスでは、マルチエリアの隣接関係を設定できます。ネイティブ ブロードキャスト ネットワークの場合、マルチエリア隣接関係のインターフェイスを有効にする **network point-to-point** コマンドを使用して、インターフェイスを OPSF ポイントツーポイント型で設定する必要があります。

- 双方向フォワーディング検出（BFD）の性質をプライマリ インターフェイスから継承します。BFD はマルチエリア インターフェイスでは設定できません。ただし、プライマリ インターフェイスでは設定できます。

マルチエリア インターフェイスは、インターフェイスの性質をそのプライマリ インターフェイスから継承しますが、次のように、マルチエリア インターフェイス コンフィギュレーション モードでインターフェイスの一部の性質を設定できます。

```
RP/0/RSP0/cpu 0: router(config-ospf-ar)# multi-area-interface GigabitEthernet 0/1/0/3
RP/0/RSP0/cpu 0: router(config-ospf-ar-mif)# ?
  authentication       Enable authentication
  authentication-key   Authentication password (key)
  cost                 Interface cost
  cost-fallback        Cost when cumulative bandwidth goes below the theshold
  database-filter      Filter OSPF LSA during synchronization and flooding
  dead-interval        Interval after which a neighbor is declared dead
  distribute-list      Filter networks in routing updates
  hello-interval       Time between HELLO packets
  message-digest-key   Message digest authentication password (key)
  mtu-ignore           Enable/Disable ignoring of MTU in DBD packets
  packet-size          Customize size of OSPF packets upto MTU
  retransmit-interval  Time between retransmitting lost link state advertisements
  transmit-delay       Estimated time needed to send link-state update packet

RP/0/RSP0/cpu 0: router(config-ospf-ar-mif)#
```

OSPF のラベル配布プロトコル IGP 自動設定

ラベル配布プロトコル（LDP）内部ゲートウェイ プロトコル（IGP）自動設定を使うと、OSPF などの IGP インスタンスに使用されているインターフェイスのセットで LDP をイネーブルにする手順を簡略化できます。LDP IGP 自動設定は、多数のインターフェイス（転送に LDP がコアで使用される場合など）および複数の OSPF インスタンスで同時に使用できます。

この機能は、デフォルトの VPN ルーティングおよび転送（VRF）インスタンスとして IPv4 ユニキャスト アドレス ファミリをサポートします。

LDP IGP 自動設定は、LDP の個々のインターフェイス ベースで igp auto-config disable コマンド使用して明示的にディセーブルにすることもできます。これにより、明示的にディセーブルにしたインターフェイスを除くすべての OSPF インターフェイスを LDP で受信できます。

LDP IGP 自動設定については、 *MPLS Configuration Guide for Cisco ASR 9000 Series RoutersMPLS Configuration Guide for Cisco NCS 560 Series Routers*を参照してください。

OSPF 認証のメッセージ ダイジェスト管理

すべてのOSPFルーティング プロトコル交換は認証されます。使用される方法は、認証が設定される方法によって異なります。暗号認証を使用する場合、OSPF ルーティング プロトコルは、Message Digest 5（MD5）認証アルゴリズムを使用してネットワーク内のネイバー間で送信されたパケットを認証します。各 OSPF プロトコル パケットでは、キーを使用して、OSPF パケットの最後に付加されるメッセージ ダイジェストを生成および検証します。メッセージ ダ

イジェストはOSPFプロトコルパケットおよび秘密キーの単方向機能です。各キーは使用されるインターフェイスとキー ID の組み合わせで識別されます。インターフェイスでは、複数のキーが常にアクティブになっています。

キーのロールオーバーを管理し、OSPF の MD5 認証を拡張するには、キーチェーンと呼ばれるキーのコンテナを設定できます。この各キーは、生成/受け取り時間、キー ID、認証アルゴリズムの属性で構成されます。

OSPF の GTSM TTL セキュリティ メカニズム

OSPF は、ネイバーに対するネットワーク、フラッディング リンク ステート アドバタイズメント（LSA）アップデートで、トポロジの変更を検出し、トポロジの新しいビュー上ですばやくコンバージするためにネットワーキング デバイスを必要とするリンク ステート プロトコルです。ただし、ネイバーからの LSA の受信動作中は、ネットワーク攻撃が発生する可能性があります。これは、ユニキャストまたはマルチキャストパケットが仮想リンクの 1 ホップまたは複数ホップ離れて配置されているネイバーから送信されているという確認ができないためです。

仮想リンクについては、OSPF パケットはネットワーク全体の複数ホップを通過して送信されます。したがって、TTL 値は複数回にわたり減少していく可能性があります。このようなリンクの種類では、最小 TTL 値が複数ホップ パケットで許可され受け入れられなければなりません。

複数ホップを通過して送信される無効なソースから発生するネットワーク攻撃をフィルタリングするには、一般 TTL セキュリティ メカニズム（GTSM）の RFC 3682 を使用して、攻撃を防止します。GTSM はリンク ローカル アドレスをフィルタリングして、TTL 値 255 のコンフィギュレーションの 1 ホップ ネイバーとなる隣接関係だけを許可します。IP ヘッダーの TTL 値は OSPF パケットが生成されるときに 255 に設定され、受信された OSPF パケットでデフォルトの GTSM TTL 値 255 またはユーザ設定された GTSM TTL 値に対してチェックされます。このようにして、TTL ホップを超える不正な OSPF パケットをブロックします。

OSPFv2 のパス計算要素

PCE はネットワーク パスやルートをネットワーク図に基づいて計算し、計算上の制限を適用する機能を持つエンティティ（コンポーネント、アプリケーション、ネットワーク ノード）です。

PCE は、PCE アドレスおよびクライアントが MPLS-TE に設定されると実行されます。PCE はその PCE アドレスおよび機能を OSPF に通信して、OSPF はこの情報を PCE ディスカバリ Type-Length-Value（TLV）（タイプ 2）にパッケージ化し、RI LSA を再発信します。OSPF には、すべての RI LSA でルータ機能 TLV（タイプ 1）も含まれます。PCE ディスカバリ TLV には PCE アドレス サブ TLV（タイプ 1）およびパス スコープ サブ TLV（タイプ 2）が含まれます。

PCE アドレス サブ TLV では PCE に到達するために使用される必要がある IP アドレスを指定します。このアドレスは常に到達可能なループ バック アドレスにする必要があります。この TLV は必須であり、PCE ディスカバリ TLV 内に存在する必要があります。パス スコープ サブ

TLV は、PCE パス計算スコープを示します。これは、PCE 機能を参照して計算したり、エリア内ルート、エリア間、AS 間、またはレイヤ TE 間 LSP の計算に参加したりします。

OSPFv2 への PCE 拡張機能には、ルータ情報リンク ステート アドバタイズメント（RI LSA）のサポートが含まれます。OSPFv2 は、すべてのエリアの範囲（LSA タイプ 9、10、および 11）を受信するように拡張されます。ただし、OSPFv2 はエリア範囲タイプ 10 のみを発信します。

パス計算要素機能の詳細については、 *MPLS Configuration Guide for Cisco ASR 9000 Series RoutersMPLS Configuration Guide for Cisco NCS 560 Series Routers*の「*Implementing MPLS Traffic Engineering on* Cisco ASR 9000 シリーズ ルータ 」のモジュールと次の IETF のドラフトを参照してください。

- draft-ietf-ospf-cap-09
- draft-ietf-pce-disco-proto-ospf-00

OSPF IP 高速再ルーティング ループフリー代替

OSPF IP 高速再ルーティング（FRR）ループフリー代替（LFA）の計算では、次の処理がサポートされています。

- IP 転送およびルーティングを使用した高速再ルーティング機能
- 最短時間でラインカードの障害に対処
- デフォルト以外の VRF での OSPFv2 および OSPFv3 IP FRR の機能のサポート

OSPFv3 の管理情報ベース（MIB）

Cisco IOS XR では RFC 5643 に定義されている MIB および OSPFv3 のトラップが完全にサポートされています。RFC 5643 には、IPv6 用の Open Shortest Path First（OSPF）ルーティング プロトコル（OSPF バージョン 3）で使用する管理情報ベース（MIB）のオブジェクトが定義されています。

OSPFv3 MIB の実装は、IETF ドラフト『*Management Information Base for OSPFv3 (draft-ietf-ospf-ospfv3-mib-8)*』に基づきます。RFC 5643 にアップグレードすると、ユーザは新しい MIB をピックアップするように NMS アプリケーションを更新する必要があります。

Cisco IOS XR MIB サポートの詳細については、『*Cisco ASR 9000 Series Aggregation Services Router MIB Specification Guide*』を参照してください。

複数の OSPFv3 インスタンス

SNMPv3 は、複数の OSPFv3 インスタンスに MIB ビューを設定するために使用できる「コンテキスト」を同じシステムでサポートします。

OSPFv2 の VRF-lite サポート

OSPF バージョン 2（OSPFv2）の VRF Lite 機能は、イネーブルになっています。VRF-Lite は、BGP/MPLS ベースのバックボーンがない状態での仮想ルーティングおよび転送（VRF）導入です。VRF-Lite では、個別のプロバイダー エッジ（PE）ルータは VRF インターフェイスを使用して直接接続されています。OSPFv2 の VRF-Lite をイネーブルにするには、VRF コンフィギュレーション モードで **capability vrf-lite** コマンドを設定します。VRF-Lite が設定されている場合、DN ビット処理および自動エリア境界ルータ（ABR）のステータス設定はディセーブルです。

OSPFv3 タイマー リンクステート アドバタイズメントおよび Shortest Path First スロットリングのデフォルト値のアップデート

Open Shortest Path First バージョン 3（OSPFv3）タイマー リンク ステート アドバタイズメント（LSA）、Shortest Path First（SPF）スロットリングのデフォルト値は次のように更新されます。

- **timers throttle lsa all**：*start-interval*：50 ミリ秒および *hold-interval*：200 ミリ秒
- **timers throttle spf** ：*spf-start*：50 ミリ秒、*spf-hold*：200 ミリ秒、*spf-max-wait*：5000 ミリ秒

OSPF の不等コスト マルチパス ロードバランシング

不等コストマルチパス（UCMP）のロードバランシングは、Open Shortest Path First（OSPF）による、異なるコストを使用してトラフィックを複数のパスでバランス良く負荷分散する機能を追加します。UCMP を有効にしないと、OSPF（ECMP）によってベストコストパスのみが検出され、代替の上位コストパスは計算されません。

通常、最短 IGP パスを形成するために、高い帯域幅リンクほど低い IGP メトリックが設定されています。UCMP ロードバランシングが有効になっている場合、IGP は、トラフィックに対してより低い帯域幅のリンク（またはより高いコストのリンク）を使用でき、転送情報ベース（FIB）にこれらのパスをインストールできます。OSPF は FIB の同じ宛先に複数のパスをインストールしますが、各パスには関連付けられた「ロードメトリック/重み」が含まれます。FIB はこのロードメトリック/重みを、高帯域幅パスおよび低帯域幅パスで送信する必要があるトラフィックの量を決定するために使用します。

UCMP の計算は OSPF VRF のコンテキストで実行され、特定の VRF の UCMP 計算を有効にします。デフォルト VRF の場合、設定は OSPF グローバルモードで実行されます。UCMP の設定には、UCMP 計算をプレフィックス リストに存在するプレフィックスに対してのみに制限する prefix-list オプションがあります。prefix-list オプションが指定されていない場合は、UCMP の計算は、OSPF の到達可能プレフィックスに対して実行されます。考慮され、インストールされる UCMP のパスの数は、**variance** 設定を使用して制御されます。variance 値は、ルーティング情報ベース（RIB/FIB）にインストールする対象となる UCMP パス メトリックの範囲を指定し、プライマリ パス メトリックの割合の観点から定義されます。ECMP や UCMP のパスな

どのパスの合計数は、最大パス設定またはプラットフォームの最大パス機能によって制限されます。

UCMP の計算に使用されるインターフェイスから特定のインターフェイスを除外するオプションがあります。特定のインターフェイスが任意のプレフィックスに関して UCMP ネクストホップとして考慮されないようにする場合は、UCMP **exclude interface** コマンドを使用して、インターフェイスを UCMP の計算から除外するように設定します。

UCMP の設定を有効にすると、prefix-list オプションが使用される場合、OSPF は到達可能なすべての OSPF プレフィックスまたはプレフィックスリストで許可されているすべてのプレフィックスに対して UCMP 計算を実行する必要があります。UCMP 計算はプライマリ SPF の後に実行され、ルート計算が完了します。プライマリルート計算の完了時から、設定可能な遅延があり（デフォルトの遅延は 100 ms）、UCMP の計算が開始されます。プライマリ SPF の完了と UCMP 計算の開始の間の遅延を設定するには、UCMP **delay-interval** コマンドを使用します。UCMP の計算は、高速再ルーティングの計算中に実行されます（UCMP の計算を実行するために IPFRR を有効にする必要はありません）。IPFRR が有効になっている場合、高速再ルーティング バックアップ パスは、プライマリ等コストマルチパス（ECMP）パスと UCMP パスの両方に対して計算されます。

UCMP 比率を手動で調整するには、リンクのメトリックを変更する任意のコマンドを使用します。

- インターフェイスコンフィギュレーションモードで帯域幅コマンドを使用する
- リンクの OSPF インターフェイスコストを調整する

OSPF の実装方法

ここでは、次の手順について説明します。

OSPF のイネーブル化

このタスクでは、1 つのルータ ID で OSPF プロセスをイネーブルにするルータで、最小の OSPF コンフィギュレーションを実行し、バックボーンまたはバックボーン以外のエリアを設定し、OSPF を実行する 1 つ以上のインターフェイスを割り当てるする方法を説明します。

始める前に

IP アドレスを設定する前に OSPF を設定することはできますが、IP アドレスが設定されるまで、OSPF はルーティングされません。

手順の概要

1. **configure**
2. 次のいずれかを実行します。
   - **router ospf** *process-name*

- **router ospfv3** *process-name*

3. **router-id** { *router-id* }
4. **area** *area-id*
5. **interface** *type interface-path-id*
6. OSPF を使用する各インターフェイスでステップ 5 を繰り返します。
7. **log adjacency changes** [**detail**] [**disable**]
8. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	

ステップ 2	次のいずれかを実行します。
	• **router ospf** *process-name*
	• **router ospfv3** *process-name*
	例：
	`RP/0/RSP0/cpu 0: router(config)# router ospf 1`
	または
	`RP/0/RSP0/cpu 0: router(config)# router ospfv3 1`
		指定したルーティング プロセスに OSPF ルーティングをイネーブルにし、ルータ コンフィギュレーション モードでルータを配置します。
		または
		指定したルーティング プロセスに OSPFv3 ルーティングをイネーブルにし、router ospfv3 コンフィギュレーション モードでルータを配置します。
		（注） *process-name* 引数は、40 文字未満の英数字です。

ステップ 3	**router-id** { *router-id* } 例： `RP/0/RSP0/cpu 0: router(config-ospf)# router-id 192.168.4.3`	OSPF プロセスのルータ ID を設定します。 （注） 固定 IP アドレスをルータ ID として使用することを推奨します。
ステップ 4	**area** *area-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf)# area 0`	エリア コンフィギュレーション モードを開始し、OSPF プロセスのエリアを設定します。 • バックボーン エリアには 0 のエリア ID があります。 • バックボーン以外のエリアにはゼロではないエリア ID があります。 • *area-id* 引数は、area 1000 や area 0.0.3.232 など、ドット付き 10 進表記または IPv4 アドレス形式で入力できます。ただし、1 つのエリアでは同じ形式を選択する必要があります。IPv4 アドレス形式を使用することを推奨します。

	コマンドまたはアクション	目的
ステップ 5	**interface** *type interface-path-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf-ar)# interface GigabitEthernet 0/1/0/3`	インターフェイス コンフィギュレーション モードを開始して、ステップ 4 で設定したエリアのインターフェイスを 1 つ以上関連付けます。
ステップ 6	OSPF を使用する各インターフェイスでステップ 5 を繰り返します。	—
ステップ 7	**log adjacency changes** [**detail**] [**disable**] 例： `RP/0/RSP0/cpu 0: router(config-ospf-ar-if)# log adjacency changes detail`	（任意）ネイバー変更の通知を要求します。 • デフォルトでは、この機能はイネーブルです。 • ネイバー変更によって生成されたメッセージは通知と見なされます。このメッセージは logging console コマンドで重大度レベル 5 に分類されます。logging console コマンドではどの重大度レベルのメッセージをコンソールに送信するかを制御します。デフォルトでは、すべての重大度レベルのメッセージが送信されます。
ステップ 8	**commit**	

スタブ エリアおよび Not-So-Stubby Area タイプの設定

このタスクでは、OSPF のスタブ エリアおよび NSSA を設定する方法を説明します。

手順の概要

1. **configure**
2. 次のいずれかを実行します。
   - **router ospf** *process-name*
   - **router ospfv3** *process-name*
3. **router-id** { *router-id* }
4. **area** *area-id*
5. 次のいずれかを実行します。
   - **stub** [**no-summary**]
   - **nssa** [**no-redistribution**] [**default-information-originate**] [**no-summary**] [**translate**] [**translate always**]
6. 次のいずれかを実行します。
   - **stub**
   - **nssa**
7. **default-cost** *cost*

8. **commit**
9. スタブ エリアまたは NSSA にある他のすべてのルータでこのタスクを繰り返します。

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	

ステップ 2	次のいずれかを実行します。
	• **router ospf** *process-name*
	• **router ospfv3** *process-name*
	例：
	`RP/0/RSP0/cpu 0: router(config)# router ospf 1`
	または
	`RP/0/RSP0/cpu 0: router(config)# router ospfv3 1`
		指定したルーティング プロセスに OSPF ルーティングをイネーブルにし、ルータ コンフィギュレーション モードでルータを配置します。
		または
		指定したルーティング プロセスに OSPFv3 ルーティングをイネーブルにし、router ospfv3 コンフィギュレーション モードでルータを配置します。
		（注） *process-name* 引数は、40 文字未満の英数字です。

ステップ 3	**router-id** { *router-id* } 例： `RP/0/RSP0/cpu 0: router(config-ospf)# router-id 192.168.4.3`	OSPF プロセスのルータ ID を設定します。 （注） 固定 IP アドレスをルータ ID として使用することを推奨します。
ステップ 4	**area** *area-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf)# area 1`	エリア コンフィギュレーション モードを開始し、OSPF プロセスのバックボーン以外のエリアを設定します。 • *area-id* 引数は、area 1000 や area 0.0.3.232 など、ドット付き 10 進表記または IPv4 アドレス形式で入力できます。ただし、1 つのエリアでは同じ形式を選択する必要があります。IPv4 アドレス形式を使用することを推奨します。
ステップ 5	次のいずれかを実行します。 • **stub** [**no-summary**] • **nssa** [**no-redistribution**] [**default-information-originate**] [**no-summary**] [**translate**] [**translate always**] 例： `RP/0/RSP0/cpu 0: router(config-ospf-ar)# stub no summary` または	非バックボーン エリアをスタブ エリアとして定義します。 • スタブ エリアに送信される LSA の数をさらに減らすために **no-summary** キーワードを指定します。このキーワードにより、ABR がサマリー リンクステート アドバタイズメント（タイプ 3）をスタブエリアに送信しないようにします。 または エリアを NSSA として定義します。

	コマンドまたはアクション	目的

	`RP/0/RSP0/cpu 0: router(config-ospf-ar)# nssa no-redistribution`
	または
	`RP/0/RSP0/cpu 0: router(config-ospf-ar)# nssa translate <type number> always`	

ステップ 6	次のいずれかを実行します。
	• **stub**
	• **nssa**
	例：
	`RP/0/RSP0/cpu 0: router(config-ospf-ar)# stub`
	または
	`RP/0/RSP0/cpu 0: router(config-ospf-ar)# nssa`
		（任意）スタブエリア、およびNSSAエリアに設定されたオプションをオフにします。
		• ステップ 5 でオプションのキーワード（**no-summary** 、 **no-redistribution** 、**default-information-originate** 、および **translate** ）を使用してスタブエリアおよびNSSAエリアを設定した場合、コマンドの **no** 形式を使用するのではなく、**stub** および **nssa** コマンドをこれらのキーワードなしで再度発行する必要があります。
		• たとえば、コマンドの**no nssa default-information-originate** 形式は、NSSA エリアを通常のエリアに変更し、そのエリアの既存の隣接関係を意図せずダウンさせます。

ステップ 7	**default-cost** *cost* 例： `RP/0/RSP0/cpu 0: router(config-ospf-ar)#default-cost 15`	（任意）スタブエリアまたはNSSAに送信されるデフォルト サマリー ルートのコストを指定します。 • このコマンドは NSSA にアタッチされている ABR でのみ使用します。エリア内の他のルータには使用しないでください。 • デフォルトのコストは 1 です。
ステップ 8	**commit**	
ステップ 9	スタブエリアまたはNSSAにある他のすべてのルータでこのタスクを繰り返します。	—

ブロードキャスト ネットワーク以外のネイバーの設定

このタスクでは、非ブロードキャスト ネットワークにネイバーを設定する方法を説明します。このタスクはオプションです。

始める前に

NBMA ネットワークをブロードキャストまたは非ブロードキャストとして構成する場合は、各ルータから各ルータあるいはフルメッシュのネットワークにまで仮想回線があると想定されます。

手順の概要

1. **configure**
2. 次のいずれかを実行します。
   - **router ospf** *process-name*
   - **router ospfv3** *process-name*
3. **router-id** { *router-id* }
4. **area** *area-id*
5. **network** { **broadcast** | **non-broadcast** | { **point-to-multipoint** [**non-broadcast**] | **point-to-point** }}
6. **dead-interval** *seconds*
7. **hello-interval** *seconds*
8. **interface** *type interface-path-id*
9. 次のいずれかを実行します。
   - **neighbor** *ip-address* [**priority** *number*] [**poll-interval** *seconds*][**cost** *number*]
   - **neighbor** *ipv6-link-local-address* [**priority** *number*] [**poll-interval** *seconds*][**cost** *number*] [**database-filter** [**all**]]
10. インターフェイスのすべてのネイバーでステップ 9 を繰り返します。
11. exit
12. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	

ステップ 2	次のいずれかを実行します。
	• **router ospf** *process-name*
	• **router ospfv3** *process-name*
	例：
	`RP/0/RSP0/cpu 0: router(config)# router ospf 1`
	または
	`RP/0/RSP0/cpu 0: router(config)# router ospfv3 1`
		指定したルーティング プロセスに OSPF ルーティングをイネーブルにし、ルータ コンフィギュレーション モードでルータを配置します。
		または
		指定したルーティング プロセスに OSPFv3 ルーティングをイネーブルにし、router ospfv3 コンフィギュレーション モードでルータを配置します。
		（注） *process-name* 引数は、40 文字未満の英数字です。

	コマンドまたはアクション	目的
ステップ 3	**router-id** { *router-id* } 例： `RP/0/RSP0/cpu 0: router(config-ospf)# router-id 192.168.4.3`	OSPF プロセスのルータ ID を設定します。 （注） 固定 IP アドレスをルータ ID として使用することを推奨します。
ステップ 4	**area** *area-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf)# area 0`	エリア コンフィギュレーション モードを開始し、OSPF プロセスのエリアを設定します。 • この例ではバックボーンエリアを設定します。 • *area-id* 引数は、area 1000 や area 0.0.3.232 など、ドット付き 10 進表記または IPv4 アドレス形式で入力できます。ただし、1 つのエリアでは同じ形式を選択する必要があります。IPv4 アドレス形式を使用することを推奨します。
ステップ 5	**network** { **broadcast** | **non-broadcast** | { **point-to-multipoint** [**non-broadcast**] | **point-to-point** } } 例： `RP/0/RSP0/cpu 0: router(config-ospf-ar)# network non-broadcast`	OSPF ネットワーク タイプをそのメディアのデフォルト以外のタイプに設定します。 • この例では、ネットワーク タイプを NBMA に設定します。
ステップ 6	**dead-interval** *seconds* 例： `RP/0/RSP0/cpu 0: router(config-ospf-ar)# dead-interval 40`	（任意）ネイバーのダウンを宣言する前に、ネイバーからの hello パケットを待機する時間を設定します。
ステップ 7	**hello-interval** *seconds* 例： `RP/0/RSP0/cpu 0: router(config-ospf-ar)# hello-interval 10`	（任意）OSPF がインターフェイスで送信する hello パケットの間隔を指定します。 （注） Hello タイマーの間隔はデフォルトの 10 秒に設定することをお勧めします。設定された Hello 間隔タイマーがデフォルト値より小さい場合、スイッチオーバー中に OSPF セッションがフラップすることがあります。
ステップ 8	**interface** *type interface-path-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf-ar)# interface GigabitEthernet 0/2/0/0`	インターフェイス コンフィギュレーション モードを開始して、ステップ 4 で設定したエリアのインターフェイスを 1 つ以上関連付けます。 • この例では、値がインターフェイス レベルで設定されていないため、インターフェイスはブ

	コマンドまたはアクション	目的
		ロードキャスト ネットワーク タイプおよび hello および dead 間隔をそのエリアから継承します。

ステップ 9	次のいずれかを実行します。
	• **neighbor** *ip-address* [**priority** *number*] [**poll-interval** *seconds*][**cost** *number*]
	• **neighbor** *ipv6-link-local-address* [**priority** *number*] [**poll-interval** *seconds*][**cost** *number*] [**database-filter** [**all**]]
	例：
	`RP/0/RSP0/cpu 0: router(config-ospf-ar-if)# neighbor 10.20.20.1 priority 3 poll-interval 15`
	または
	`RP/0/RSP0/cpu 0: router(config-ospf-ar-if)# neighbor fe80::3203:a0ff:fe9d:f3fe`
		ブロードキャスト ネットワーク以外と相互接続する OSPF ネイバーの IPv4 アドレスを設定します。
		または
		OSPFv3 ネイバーのリンクローカル IPv6 アドレスを設定します。
		• *ipv6-link-local-address* 引数は、RFC 2373 に記載されている形式である必要があります。このアドレスは16ビット値を使用する16進数をコロンで区切って指定します。
		• **priority** キーワードでは、このネイバーが DR または BDR になる資格があることをルータに通知します。priority 値は隣接ルータの実際のプライオリティ設定と一致する必要があります。ネイバー プライオリティのデフォルト値はゼロです。このキーワードはポイントツーマルチポイント インターフェイスには適用されません。
		• この **poll-interval** キーワードはポイントツーマルチポイント インターフェイスには適用されません。RFC 1247 では、この値を hello interval よりずっと大きくすることが推奨されています。デフォルトは120秒（2分）です。
		• 特定のコストが設定されていないネイバーは、cost コマンドに基づいてインターフェイスのコストを想定します。ポイントツーマルチポイント インターフェイスでは、機能するキーワードと引数の組み合わせは **cost** *number* だけです。 **cost** キーワードは NBMA には適用されません。
		• **database-filter** キーワードでは OSPF ネイバーへの発信 LSA をフィルタリングします。 **all** キーワードを指定すると、着信および発信 LSA はフィルタリングされます。フィルタリングによりルーティングトポロジが2つのネイバー間でまったく異なるように見え、データトラフィックがブラックホール化やルーティング

	コマンドまたはアクション	目的
		ループを引き起こすことがあるため、十分注意して使用してください。
ステップ 10	インターフェイスのすべてのネイバーでステップ 9 を繰り返します。	—
ステップ 11	exit 例： `RP/0/RSP0/cpu 0: router(config-ospf-ar-if)# exit`	エリア コンフィギュレーション モードを開始します。
ステップ 12	**commit**	

OSPF Version 2 の異なる階層レベルでの認証の設定

このタスクでは、OSPF ルータ プロセスに MD5（セキュア）認証を設定する方法について説明します。プレーン テキスト認証を 1 エリアに設定し、次にクリア テキスト（null）認証を 1 インターフェイスに適用します。

（注） インターフェイス レベルで設定された認証は、エリア レベルおよびルータ プロセス レベルで設定された認証を上書きします。インターフェイスに特別に設定された認証がない場合、そのインターフェイスは認証パラメータ値をより高い階層レベルから継承します。階層および継承の詳細については、OSPF の階層 CLI および CLI 継承 （394 ページ）を参照してください。

始める前に

認証を設定する場合、プレーン テキスト認証または MD5 認証のどちらを設定するかをはじめに決定する必要があります。また、認証の適用対象がプロセス内のすべてのインターフェイスか、全エリアか、特定のインターフェイスかを決定する必要があります。ネットワークに特定のメソッドを使用する場合、それぞれの種類の認証に関する情報については、OSPF のルート認証方法 （398 ページ）を参照してください。

手順の概要

1. **configure**
2. **router ospf** *process-name*
3. **router-id** { *router-id* }
4. **authentication** [**message-digest** | **null**]
5. **message-digest-key** *key-id* **md5** { *key* | **clear** *key* | **encrypted** *key* | **LINE**}
6. **area** *area-id*
7. **interface** *type interface-path-id*
8. 同じ認証を使用して通信する必要があるインターフェイスごとにステップ 7 を繰り返します。

9. **exit**
10. **area** *area-id*
11. **authentication** [**message-digest** | **null**]
12. **interface** *type interface-path-id*
13. 同じ認証を使用して通信する必要があるインターフェイスごとにステップ 12 を繰り返します。
14. **interface** *type interface-path-id*
15. **authentication** [**message-digest** | **null**]
16. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router ospf** *process-name* 例： `RP/0/RSP0/cpu 0: router(config)# router ospf 1`	指定したルーティング プロセスに OSPF ルーティングを有効にし、ルータ コンフィギュレーション モードでルータを配置します。 （注） *process-name* 引数は、40 文字未満の英数字です。
ステップ 3	**router-id** { *router-id* } 例： `RP/0/RSP0/cpu 0: router(config-ospf)# router-id 192.168.4.3`	OSPF プロセスのルータ ID を設定します。
ステップ 4	**authentication** [**message-digest**	**null**] 例： `RP/0/RSP0/cpu 0: router(config-ospf)#authentication message-digest`	OSPF プロセスに対して MD5 認証が有効になります。 • エリアやインターフェイスなどのより低い階層レベルによって変更されないかぎり、この認証タイプはルータ プロセス全体に適用されます。
ステップ 5	**message-digest-key** *key-id* **md5** { *key*	**clear** *key*	**encrypted** *key*	**LINE**} 例： `RP/0/RSP0/cpu 0: router(config-ospf)#message-digest-key 4 md5 yourkey`	OSPF プロセスに対して MD5 認証キーを指定します。 • 隣接ルータが、同じキー ID を保持する必要があります。
ステップ 6	**area** *area-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf)# area 0`	エリア コンフィギュレーション モードを開始して、OSPF プロセスのバックボーン エリアを設定します。

	コマンドまたはアクション	目的

ステップ 7	**interface** *type interface-path-id*
	例：
	`RP/0/RSP0/cpu 0:`
	`router(config-ospf-ar)# interface GigabitEthernet 0/1/0/3`
		インターフェイス コンフィギュレーション モードを開始して、1つ以上のインターフェイスをバックボーン エリアに関連付けます。
		• すべてのインターフェイスは、OSPFプロセスの指定された認証パラメータ値を継承します（ステップ 4、ステップ 5、ステップ6）。

ステップ 8	同じ認証を使用して通信する必要があるインターフェイスごとにステップ 7 を繰り返します。	—
ステップ 9	**exit** 例： `RP/0/RSP0/cpu 0: router(config-ospf-ar)# exit`	エリア OSPF コンフィギュレーション モードを開始します。
ステップ 10	**area** *area-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf)# area 1`	エリア コンフィギュレーション モードを開始し、OSPF プロセスの非バックボーンのエリア 1 を設定します。 • *area-id* 引数は、area 1000 や area 0.0.3.232 など、ドット付き 10 進表記または IPv4 アドレス形式で入力できます。ただし、1 つのエリアでは同じ形式を選択する必要があります。IPv4 アドレス形式を使用することを推奨します。
ステップ 11	**authentication** [**message-digest** | **null**] 例： `RP/0/RSP0/cpu 0: router(config-ospf-ar)# authentication`	セキュリティのないタイプ 1（プレーンテキスト）認証をイネーブルにします。 • 例では、プレーン テキスト認証を（キーワードを指定しないことによって）指定します。インターフェイス コンフィギュレーション モードで **authentication-key** コマンドを使用し、このプレーンテキストパスワードを指定します。
ステップ 12	**interface** *type interface-path-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf-ar)# interface GigabitEthernet 0/1/0/0`	インターフェイス コンフィギュレーション モードを開始して、ステップ7で指定したバックボーン以外のエリア1に1つ以上のインターフェイスを関連付けます。 • 設定されているすべてのインターフェイスがエリア1に対して設定されている認証パラメータ値を継承します。
ステップ 13	同じ認証を使用して通信する必要があるインターフェイスごとにステップ 12 を繰り返します。	—

	コマンドまたはアクション	目的
ステップ 14	**interface** *type interface-path-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf-ar)# interface GigabitEthernet 0/3/0/0`	インターフェイス コンフィギュレーション モードを開始し、1つ以上のインターフェイスを異なる認証タイプに関連付けます。
ステップ 15	**authentication** [**message-digest** | **null**] 例： `RP/0/RSP0/cpu 0: router(config-ospf-ar-if)# authentication null`	ギガビットイーサネット インターフェイス 0/3/0/0 に no authentication を指定し、エリア 1 に指定されたプレーン テキスト認証を上書きします。 • デフォルトでは、同じエリアで設定されるすべてのインターフェイスは、エリアと同じ認証パラメータ値を継承します。
ステップ 16	**commit**	

OSPF に同じ LSA が生成される頻度または受け入れられる頻度の制御

このタスクでは、非常に短い間隔で多数の LSA がフラッディングされる必要がある場合に、ルーティングテーブルの OSPF ルートのコンバージェンス時間を調整する方法を説明します。

手順の概要

1. **configure**
2. 次のいずれかを実行します。
   - **router ospf** *process-name*
   - **router ospfv3** *process-name*
3. **router-id** { *router-id* }
4. ステップ 5 またはステップ 6、または両方のステップを実行して、同じ LSA が送受信される間隔を制御します。
5. **timers lsa refresh** *seconds*
6. **timers lsa min-arrival** *seconds*
7. **timers lsa group-pacing** *seconds*
8. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	次のいずれかを実行します。 • **router ospf** *process-name* • **router ospfv3** *process-name*	指定したルーティング プロセスに OSPF ルーティングをイネーブルにし、ルータ コンフィギュレーション モードでルータを配置します。

	コマンドまたはアクション	目的

	例：
	`RP/0/RSP0/cpu 0: router:router(config)# router ospf 1`
	または
	`RP/0/RSP0/cpu 0: router(config)# router ospfv3 1`
		または
		指定したルーティング プロセスに OSPFv3 ルーティングをイネーブルにし、router ospfv3 コンフィギュレーション モードでルータを配置します。
		（注） *process-name* 引数は、40 文字未満の英数字です。

ステップ 3	**router-id** { *router-id* } 例： `RP/0/RSP0/cpu 0: router(config-ospf)# router-id 192.168.4.3`	OSPF プロセスのルータ ID を設定します。 （注） 固定 IP アドレスをルータ ID として使用することを推奨します。
ステップ 4	ステップ 5 またはステップ 6、または両方のステップを実行して、同じ LSA が送受信される間隔を制御します。	—
ステップ 5	**timers lsa refresh** *seconds* 例： `RP/0/RSP0/cpu 0: router(config-ospf)# timers lsa refresh 1800`	自動送信 LSA をリフレッシュする頻度を秒単位で設定します。 • OSPF および OSPFv3 の両方で、デフォルトは 1800 秒です。
ステップ 6	**timers lsa min-arrival** *seconds* 例： `RP/0/RSP0/cpu 0: router(config-ospf)# timers lsa min-arrival 2`	フラッディング中に特定の OSPF Version 2 LSA の新しいプロセスが受け入れられる頻度を制限します。 • デフォルト値は 1 秒です。
ステップ 7	**timers lsa group-pacing** *seconds* 例： `RP/0/RSP0 /CPU0:router(config-ospf)# timers lsa group-pacing 1000`	OSPF リンクステート LSA がフラッディングのグループに収集される間隔を変更します。 • デフォルトは 240 秒です。
ステップ 8	**commit**	

OSPF のエリア 0 に MD5 認証を使用する仮想リンクの作成

このタスクでは、仮想リンクをバックボーン（エリア 0）に作成して MD5 認証を適用する方法について説明します。説明されている手順は、仮想リンクの各端にある両方の ABR で実行する必要があります。仮想リンクを理解するには、OSPF の仮想リンクおよび中継エリア （406 ページ）を参照してください。

（注） 明示的にエリアパラメータ値を設定したら、インターフェイスの値を上書きして明示的に設定しないかぎり、その値はそのエリアにバインドされているすべてのインターフェイスに継承されます。OSPF Version 2 の MD5 認証を使用して設定された仮想リンク：例 （487 ページ）に例を示します。

始める前に

MD5 認証が設定された仮想リンクをエリア 0 に作成するには、次の前提条件を満たす必要があります。

- ローカル ルータを設定するリンクの反対の隣接ルータのルータ ID が必要です。ルータ ID を取得するためにリモート ルータで show ospf コマンドまたは show ospfv3 コマンドを実行できます。
- 仮想リンクが正常に機能するには、仮想リンクの各端に固定ルータ ID が必要です。ルータ ID は変更されないようにします。デフォルトでルータ ID を割り当てると、変更される可能性があります（ルータ ID の決定方法の説明については、OSPF プロセスおよびルータ ID （397 ページ）を参照してください）。したがって、仮想リンクを設定する前に、次のいずれかのタスクを実行することをお勧めします。
  - ルータ ID を設定するには、router-id コマンドを使用します。この方法を推奨します。
  - ルータが安定したルータ ID を持つために、ループバック インターフェイスを設定します。
- OSPF Version 2 の仮想リンクを設定する前に、プレーン テキスト認証、MD5 認証、認証なし（デフォルト）のうち、どの認証を設定するかを決定する必要があります。認証に関連する追加のタスクを実行する必要があるかどうかに応じて決定します。

（注） プレーンテキスト認証を設定するか、または認証を設定しない場合は、 *Routing Command Reference for Cisco ASR 9000 Series Routers*の「*OSPF Commands on* Cisco ASR 9000 シリーズ ルータ 」のモジュールに記載されている **authentication** コマンドを参照してください。

手順の概要

1. 次のいずれかを実行します。
   - **show ospf** [*process-name*]
   - **show ospfv3** [*process-name*]
2. **configure**
3. 次のいずれかを実行します。
   - **router ospf** *process-name*
   - **router ospfv3** *process-name*

4. **router-id** { *router-id* }
5. **area** *area-id*
6. **virtual-link** *router-id*
7. **authentication message-digest**
8. **message-digest-key** *key-id* **md5** { *key* | **clear** *key* | **encrypted** *key* }
9. 仮想リンクの反対側にあるABRでこのタスクのすべての手順を繰り返します。このルータで仮想リンクに指定する同じキー ID およびキーを指定します。
10. **commit**
11. 次のいずれかを実行します。
    - **show ospf** [*process-name*] [*area-id*] **virtual-links**
    - **show ospfv3** [*process-name*] **virtual-links**

手順の詳細

	コマンドまたはアクション	目的

ステップ 1	次のいずれかを実行します。
	• **show ospf** [*process-name*]
	• **show ospfv3** [*process-name*]
	例：
	`RP/0/RSP0/CPU0:router# show ospf`
	または
	`RP/0/RSP0/CPU0:router# show ospfv3`
		（任意）OSPFルーティング プロセスに関する一般情報を表示します。
		• 出力にはローカル ルータのルータ ID が表示されます。このルータ ID はリンクのもう一端を設定するために必要です。

ステップ 2	**configure**	

ステップ 3	次のいずれかを実行します。
	• **router ospf** *process-name*
	• **router ospfv3** *process-name*
	例：
	`RP/0/RSP0/CPU0:router(config)# router ospf 1`
	または
	`RP/0/RSP0/CPU0:router(config)# router ospfv3 1`
		指定したルーティング プロセスに OSPF ルーティングをイネーブルにし、ルータ コンフィギュレーション モードでルータを配置します。
		または
		指定したルーティング プロセスにOSPFv3ルーティングをイネーブルにし、router ospfv3 コンフィギュレーション モードでルータを配置します。
		（注） *process-name* 引数は、40 文字未満の英数字です。

ステップ 4	**router-id** { *router-id* } 例： `RP/0/RSP0/CPU0:router(config-ospf)# router-id 192.168.4.3`	OSPF プロセスのルータ ID を設定します。 （注） 固定 IPv4 アドレスをルータ ID として使用することを推奨します。

	コマンドまたはアクション	目的
ステップ 5	**area** *area-id* 例： `RP/0/RSP0/CPU0:router(config-ospf)# area 1`	エリア コンフィギュレーション モードを開始し、OSPFプロセスのバックボーン以外のエリアを設定します。 • *area-id* 引数は、area 1000 や area 0.0.3.232 など、ドット付き 10 進表記または IPv4 アドレス形式で入力できます。ただし、1 つのエリアでは同じ形式を選択する必要があります。IPv4 アドレス形式を使用することを推奨します。
ステップ 6	**virtual-link** *router-id* 例： `RRP/0/RSP0/CPU0:router(config-ospf-ar)# virtual-link 10.3.4.5`	OSPF 仮想リンクを定義します。 • を参照してください。
ステップ 7	**authentication message-digest** 例： `RP/0/RSP0/CPU0:router(config-ospf-ar-vl)#authentication message-digest`	この仮想リンクに対して MD5 認証を選択します。
ステップ 8	**message-digest-key** *key-id* **md5** { *key* | **clear** *key* | **encrypted** *key* } 例： `RP/0/RSP0/CPU0:router(config-ospf-ar-vl)#message-digest-key 4 md5 yourkey`	OSPF 仮想リンクを定義します。 • 仮想リンクを理解するには、を参照してください。 • *key-id* 引数は、1 ～ 255 の範囲の数です。 *key* 引数は最大 16 文字の英数字です。仮想リンクの両端のルータには同じキー ID と、OSPF トラフィックをルーティングできるキーが必要です。 • **authentication-key** *key* コマンドは、OSPFv3 ではサポートされていません。 • キーが暗号化されたら、その暗号化を保持する必要があります。
ステップ 9	仮想リンクの反対側にある ABR でこのタスクのすべての手順を繰り返します。このルータで仮想リンクに指定する同じキー ID およびキーを指定します。	—
ステップ 10	**commit**	

	コマンドまたはアクション	目的

ステップ 11	次のいずれかを実行します。
	• **show ospf** [*process-name*] [*area-id*] **virtual-links**
	• **show ospfv3** [*process-name*] **virtual-links**
	例：
	`RP/0/RSP0/CPU0:router# show ospf 1 2 virtual-links`
	または
	`RP/0/RSP0/CPU0:router# show ospfv3 1 virtual-links`
		（任意）OSPF 仮想リンクのパラメータと現在の状態を表示します。

例

次に、**show ospfv3 virtual links** EXEC コンフィギュレーション コマンドで、OSPFv3 ネイバーへの OSPF_VL0 仮想リンクが起動しており、仮想リンクインターフェイスの ID が 2 であり、仮想リンクのエンドポイントの IPv6 アドレスが 2003:3000::1 であることを検証する例を示します。

```
 show ospfv3 virtual-links

Virtual Links for OSPFv3 1

Virtual Link OSPF_VL0 to router 10.0.0.3 is up
  Interface ID 2, IPv6 address 2003:3000::1
  Run as demand circuit
  DoNotAge LSA allowed.
  Transit area 0.1.20.255, via interface GigabitEthernet 0/1/0/1, Cost of using 2
  Transmit Delay is 5 sec, State POINT_TO_POINT,
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    Hello due in 00:00:02
    Adjacency State FULL (Hello suppressed)
    Index 0/2/3, retransmission queue length 0, number of retransmission 1
    First 0(0)/0(0)/0(0) Next 0(0)/0(0)/0(0)
    Last retransmission scan length is 1, maximum is 1
    Last retransmission scan time is 0 msec, maximum is 0 msec

Check for lines:
Virtual Link OSPF_VL0 to router 10.0.0.3 is up
    Adjacency State FULL (Hello suppressed)

State is up and Adjacency State is FULL
```

OSPF ABR でのサブネットワーク LSA の要約

IP アドレスをインターフェイスに割り当てたときに複数のサブネットワークを設定した場合、すべてのサブネットワークが含まれ、ローカルエリアが別のエリアにアドバタイズする 1 つの LSA にソフトウェアを集約することができます。このようにソフトウェアを集約すると LSA

の数を減らすことができるため、ネットワーク リソースを節約できます。この集約はエリア間ルート集約と呼ばれます。これは自律システム内のルートに適用されます。再配布によって OSPF に挿入された外部ルートには適用されません。

このタスクでは、一緒にアドバタイズされる範囲に収まるすべてのサブネットワークを指定することによって、サブネットワークを 1 つの LSA に集約するように OSPF を設定します。このタスクは 1 つの ABR でのみ実行します。

手順の概要

1. **configure**
2. 次のいずれかを実行します。
   - **router ospf** *process-name*
   - **router ospfv3** *process-name*
3. **router-id** { *router-id* }
4. **area** *area-id*
5. 次のいずれかを実行します。
   - **range** *ip-address mask* [**advertise** | **not-advertise**]
   - **range** *ipv6-prefix* / *prefix-length* [**advertise** | **not-advertise**]
6. **interface** *type interface-path-id*
7. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	

ステップ 2	次のいずれかを実行します。
	• **router ospf** *process-name*
	• **router ospfv3** *process-name*
	例：
	`RP/0/RSP0/cpu 0: router(config)# router ospf 1`
	または
	`RP/0/RSP0/cpu 0: router(config)# router ospfv3 1`
		指定したルーティング プロセスに OSPF ルーティングをイネーブルにし、ルータ コンフィギュレーション モードでルータを配置します。
		または
		指定したルーティング プロセスに OSPFv3 ルーティングをイネーブルにし、router ospfv3 コンフィギュレーション モードでルータを配置します。
		（注） *process-name* 引数は、40 文字未満の英数字です。

ステップ 3	**router-id** { *router-id* } 例： `RP/0/RSP0/cpu 0: router(config-ospf)# router-id 192.168.4.3`	OSPF プロセスのルータ ID を設定します。 （注） 固定 IPv4 アドレスをルータ ID として使用することを推奨します。

	コマンドまたはアクション	目的
ステップ 4	**area** *area-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf)# area 10`	エリア コンフィギュレーション モードを開始し、OSPF プロセスのバックボーン以外のエリアを設定します。 • *area-id* 引数は、area 1000 や area 0.0.3.232 など、ドット付き 10 進表記または IPv4 アドレス形式で入力できます。ただし、1 つのエリアでは同じ形式を選択する必要があります。IPv4 アドレス形式を使用することを推奨します。

ステップ 5	次のいずれかを実行します。
	• **range** *ip-address mask* [**advertise** | **not-advertise**]
	• **range** *ipv6-prefix* / *prefix-length* [**advertise** | **not-advertise**]
	例：
	`RP/0/RSP0/cpu 0: router(config-ospf-ar)# range 192.168.0.0 255.255.0.0 advertise`
	または
	`RP/0/RSP0/cpu 0: router(config-ospf-ar)# range 4004:f000::/32 advertise`
		エリア境界で OSPF ルートを統合および集約します。
		• **advertise** キーワードにより、タイプ 3 サマリー LSA のソフトウェアがサブネットワークのアドレス範囲をアドバタイズします。
		• **not-advertise** キーワードによって、ソフトウェアがタイプ 3 サマリー LSA に制限され、この範囲内のサブネットワークは他のエリアには見えません。
		• 最初の例では、ネットワーク 192.168.0.0 のすべてのサブネットワークが ABR によって集約され、バックボーン外のエリアにアドバタイズされます。
		• 2 番目の例では、複数の IPv4 インターフェイスが 192.*x.x* のネットワークに対応しています。

ステップ 6	**interface** *type interface-path-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf-ar)# interface GigabitEthernet 0/2/0/3`	インターフェイス コンフィギュレーション モードを開始して、1 つ以上のインターフェイスをエリアに関連付けます。
ステップ 7	**commit**	

OSPF へのルートの再配布

このタスクでは、IGP（別の OSPF プロセスでも可）から OSPF にルートを再配布します。

始める前に

ルーティングポリシーの設定については、 *Routing Configuration Guide for Cisco ASR 9000 Series Routers*の「*Implementing Routing Policy on Cisco ASR 9000* シリーズ ルータ 」のモジュールを参照してください。

手順の概要

1. **configure**
2. 次のいずれかを実行します。
   - **router ospf** *process-name*
   - **router ospfv3** *process-name*
3. **router-id** { *router-id* }
4. **redistribute** *protocol* [*process-id*] { **level-1** | **level-1-2** | **level-2** } [**metric** *metric-value*] [**metric-type** *type-value*] [**match** { **external** [**1** | **2**]} [**tag** *tag-value*] [**route-policy** *policy-name*]
5. 次のいずれかを実行します。
   - **summary-prefix** *address mask* [**not-advertise**] [**tag** *tag*]
   - **summary-prefix** *ipv6-prefix / prefix-length* [**not-advertise**] [**tag** *tag*]
6. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	

ステップ 2	次のいずれかを実行します。
	• **router ospf** *process-name*
	• **router ospfv3** *process-name*
	例：
	`RP/0/RSP0/cpu 0: router(config)# router ospf 1`
	または
	`RP/0/RSP0/cpu 0: router(config)# router ospfv3 1`
		指定したルーティング プロセスに OSPF ルーティングをイネーブルにし、ルータ コンフィギュレーション モードでルータを配置します。
		または
		指定したルーティング プロセスに OSPFv3 ルーティングをイネーブルにし、router ospfv3 コンフィギュレーション モードでルータを配置します。
		（注） *process-name* 引数は、40 文字未満の英数字です。

ステップ 3	**router-id** { *router-id* } 例： `RRP/0/RSP0/cpu 0: router(config-ospf)# router-id 192.168.4.3`	OSPF プロセスのルータ ID を設定します。 （注） 固定 IPv4 アドレスをルータ ID として使用することを推奨します。
ステップ 4	**redistribute** *protocol* [*process-id*] { **level-1** | **level-1-2** | **level-2** } [**metric** *metric-value*] [**metric-type** *type-value*] [**match** { **external** [**1** | **2**]} [**tag** *tag-value*] [**route-policy** *policy-name*] 例： `RP/0/RSP0/cpu 0: router(config-ospf)# redistribute bgp 100`	1 つのルーティング ドメインから別のルーティング ドメインへの OSPF ルートの再配布 または あるルーティング ドメインから別のルーティング ドメインへ OSPFv3 ルートを再配布します。 • このコマンドを実行すると、定義上ルータが ASBR になります。

	コマンドまたはアクション	目的

	または
	
	`RP/0/RSP0/cpu 0:`
	`router(config-router)#redistribute bgp 110`
		• OSPF は再配布によって学習したすべてのルートを external とタグ付けします。
		• プロトコルとそのプロセス ID（設定されている場合）は、OSPF に再配布されるプロトコルを示します。
		• メトリックは外部ルートに割り当てるコストです。すべてのプロトコルでデフォルトは 20 です。ただし、BGP のデフォルトのメトリックは 1 です。
		• OSPF の例では、BGP 自律システム 1、レベル 1 のルートを OSPF にタイプ 2 外部ルートとして再配布します。
		• OSPFv3 の例では、BGP 自律システム 1、レベル 1 および 2 のルートを OSPF に再配布します。OSPFv3 ルーティング ドメインにアドバタイズされるデフォルト ルートに関連付けられている外部リンク タイプは、タイプ 1 の外部ルートです。
		
		（注） OSPFv3 では RPL はサポートされていません。

ステップ 5	次のいずれかを実行します。
	• **summary-prefix** *address mask* [**not-advertise**] [**tag** *tag*]
	• **summary-prefix** *ipv6-prefix / prefix-length* [**not-advertise**] [**tag** *tag*]
	例：
	
	`RP/0/RSP0/cpu 0: router(config-ospf)#`
	`summary-prefix 10.1.0.0 255.255.0.0`
	または
	
	`RP/0/RSP0/cpu 0: router(config-router)#`
	`summary-prefix 2010:11:22::/32`
		（任意）OSPF の集約アドレスを作成します。
		または
		（任意）OSPFv3 の集約アドレスを作成します。
		• このコマンドは、非 OSPF ルートの外部ルート集約を行います。
		• 集約される外部範囲は隣接している必要があります。異なる 2 台のルータからの重複範囲を集約すると、誤った宛先にパケットが送信される原因となる場合があります。
		• このコマンドはオプションです。これを指定しないと、各ルートはリンクステート データベースに含まれ、LSA にアドバタイズされます。
		• OSPFv2 の例では、集約アドレス 10.1.0.0 にアドレス 10.1.1.0、10.1.2.0、10.1.3.0 などが含まれています。外部の LSA では、アドレス 10.1.0.0 だけがアドバタイズされます。

	コマンドまたはアクション	目的
		• OSPFv3 の例では、集約アドレス 2010:11:22::/32 には 2010:11:22:0:1000::1、2010:11:22:0:2000:679:1、などのアドレスがあります。外部 LSA にはアドレス 2010:11:22::/32 だけがアドバタイズされます。
ステップ 6	**commit**	

OSPF Shortest Path First スロットリングの設定

このタスクでは、SPF スケジューリングをミリ秒間隔で設定し、ネットワークが不安定な場合に SPF 計算を遅らせる方法について説明します。このタスクはオプションです。

手順の概要

1. **configure**
2. 次のいずれかを実行します。
   - **router ospf** *process-name*
   - **router ospfv3** *process-name*
3. **router-id** { *router-id* }
4. **timers throttle spf** *spf-start spf-hold spf-max-wait*
5. **area** *area-id*
6. **interface** *type interface-path-id*
7. **commit**
8. 次のいずれかを実行します。
   - **show ospf** [*process-name*]
   - **show ospfv3** [*process-name*]

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	次のいずれかを実行します。 • **router ospf** *process-name* • **router ospfv3** *process-name* 例： `RP/0/RSP0/cpu 0: router(config)# router ospf 1` または	指定したルーティング プロセスに OSPF ルーティングをイネーブルにし、ルータ コンフィギュレーション モードでルータを配置します。 または 指定したルーティング プロセスに OSPFv3 ルーティングをイネーブルにし、router ospfv3 コンフィギュレーション モードでルータを配置します。

	コマンドまたはアクション	目的
	`RP/0/RSP0/cpu 0: router(config)# router ospfv3 1`	（注） *process-name* 引数は、40 文字未満の英数字です。
ステップ 3	**router-id** { *router-id* } 例： `RP/0/RSP0/cpu 0: router(config-ospf)# router-id 192.168.4.3`	OSPF プロセスのルータ ID を設定します。 （注） 固定 IPv4 アドレスをルータ ID として使用することを推奨します。
ステップ 4	**timers throttle spf** *spf-start spf-hold spf-max-wait* 例： `RP/0/RSP0/cpu 0: router(config-ospf)# timers throttle spf 10 4800 90000`	SPF スロットリング タイマーを設定します。
ステップ 5	**area** *area-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf)#  area 0`	エリア コンフィギュレーション モードを開始し、バックボーン エリアを設定します。 • *area-id* 引数は、area 1000 や area 0.0.3.232 など、ドット付き 10 進表記または IPv4 アドレス形式で入力できます。ただし、1 つのエリアでは同じ形式を選択する必要があります。IPv4 アドレス形式を使用することを推奨します。
ステップ 6	**interface** *type interface-path-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf-ar)# interface GigabitEthernet 0/1/0/3`	インターフェイス コンフィギュレーション モードを開始して、1 つ以上のインターフェイスをエリアに関連付けます。
ステップ 7	**commit**	

ステップ 8	次のいずれかを実行します。
	• **show ospf** [*process-name*]
	• **show ospfv3** [*process-name*]
	例：
	`RP/0/RSP0/cpu 0: router# show ospf 1`
	または
	`RP/0/RSP0/cpu 0: router# RP/0/RP0/CPU0:router# show ospfv3 2`
		（任意）SPF スロットリング タイマーを表示します。

例

次の例では、**show ospf** EXEC コンフィギュレーション コマンドを使用して、初期 SPF スケジュール遅延時間、最小ホールドタイム、最大待機時間が正しく設定されていることを検証します。ルータ タイプおよびルートの再配布などのOSPFプロセスに関する詳細情報が表示されます。

```
show ospf 1

 Routing Process "ospf 1" with ID 192.168.4.3
  Supports only single TOS(TOS0) routes
  Supports opaque LSA
  It is an autonomous system boundary router
  Redistributing External Routes from,
     ospf 2
  Initial SPF schedule delay 5 msecs
  Minimum hold time between two consecutive SPFs 100 msecs
  Maximum wait time between two consecutive SPFs 1000 msecs
  Minimum LSA interval 5 secs. Minimum LSA arrival 1 secs
  Number of external LSA 0. Checksum Sum 00000000
  Number of opaque AS LSA 0. Checksum Sum 00000000
  Number of DCbitless external and opaque AS LSA 0
  Number of DoNotAge external and opaque AS LSA 0
  Number of areas in this router is 1. 1 normal 0 stub 0 nssa
  External flood list length 0
  Non-Stop Forwarding enabled
```

（注） それぞれの出力表示フィールドの説明については、 *Routing Command Reference for Cisco ASR 9000 Series Routers*の「*OSPF Commands on Cisco ASR 9000* シリーズ ルータ 」モジュールで、**show ospf** コマンドを参照してください。

Cisco for OSPF Version 2 固有のノンストップ フォワーディングの設定

このタスクでは、NSF 対応ルータの Cisco に専用の OSPF NSF を設定する方法を説明します。このタスクはオプションです。

始める前に

OSPF NSF では、すべてのネイバー ネットワーキング デバイスが NSF 対応である必要があります。ルータにCisco IOS XR ソフトウェア イメージをインストールすると自動的に NSF 対応になります。NSF 対応ルータが特定のネットワーク セグメントで NSF 非認識ネイバーを検出すると、そのセグメントで NSF 機能をディセーブルにします。NSF 対応または NSF 認識ルータで完全に構成された他のネットワーク セグメントに対しては、継続して NSF 機能を提供します。

（注） ノンストップ フォワーディングの設定では、次の制約事項が適用されます。

- 仮想リンク用 Cisco OSPF NSF はサポートされません。
- ネイバーは NSF 対応である必要があります。

手順の概要

1. **configure**
2. **router ospf** *process-name*
3. **router-id** { *router-id* }
4. 次のいずれかを実行します。
   - **nsf cisco**
   - **nsf cisco enforce global**
5. **nsf interval** *seconds*
6. **nsfflush-delay-time***seconds*
7. **nsflifetime***seconds*
8. **nsfietf**
9. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router ospf** *process-name* 例： `RP/0/RSP0/cpu 0: router(config)# router ospf 1`	指定したルーティング プロセスに OSPF ルーティングを有効にし、ルータ コンフィギュレーション モードでルータを配置します。 （注） *process-name* 引数は、40 文字未満の英数字です。
ステップ 3	**router-id** { *router-id* } 例： `RP/0/RSP0/cpu 0: router(config-ospf)# router-id 192.168.4.3`	OSPF プロセスのルータ ID を設定します。 （注） 固定 IPv4 アドレスをルータ ID として使用することを推奨します。
ステップ 4	次のいずれかを実行します。 • **nsf cisco** • **nsf cisco enforce global** 例：	OSPF プロセスの Cisco NSF 操作をイネーブルにします。 • オプションの **enforce** および **global** キーワードを指定せずに **nsf cisco** コマンドを使用して、検出された NSF 以外のネイバーのインターフェ

	コマンドまたはアクション	目的
	`RP/0/RSP0/cpu 0: router(config-ospf)# nsf cisco enforce global`	イスでNSF再起動メカニズムを中断し、NSFネイバーが適切に機能できるようにします。 • 再起動中にルータがNSFを実行するようにする場合は、オプションの **enforce** および **global** キーワードを指定して **nsf cisco** コマンドを使用します。ただし、NSF以外のネイバーが検出されると、OSPFプロセス全体でNSF再起動はキャンセルされます。
ステップ 5	**nsf interval** *seconds* 例： `RP/0/RSP0/cpu 0: router(config-ospf)# nsf interval 120`	NSF 再起動の試行間隔の最小時間を設定します。 （注） このコマンドを使用する場合、OSPF が NSF 再起動実行を試みる前の OSPF プロセスを、最小でも 90 秒に設定する必要があります。
ステップ 6	**nsfflush-delay-time***seconds* 例： `RP/0/RSP0/cpu 0: router(config-ospf)#nsf flush-delay-time 1000`	外部ルートの学習に許可される最大時間を秒単位で設定します。
ステップ 7	**nsflifetime***seconds* 例： `RP/0/RSP0/cpu 0: router(config-ospf)#nsf lifetime 90`	再起動に続く NSF のルートの最大有効期間を秒単位で設定します。
ステップ 8	**nsfietf** 例： `RP/0/RSP0/cpu 0: router(config-ospf)#nsf ietf`	ietfグレースフルリスタートをイネーブルにします。
ステップ 9	**commit**	

MPLS トラフィック エンジニアリングの OSPF Version 2 の設定

このタスクでは、MPLS TE の OSPF を設定する手順について説明します。このタスクはオプションです。

MPLS TE タスクおよびトンネルをサポートするルータを設定できるコマンド、OSPF が使用できる MPLS トンネルを設定できるコマンド、およびMPLS TE のトラブルシューティングの説明については、 *MPLS Configuration Guide for Cisco ASR 9000 Series RoutersMPLS Configuration Guide for Cisco NCS 560 Series Routers*の「*Implementing MPLS Traffic Engineering* on *Cisco ASR 9000 Series Router* 」のモジュールを参照してください。

始める前に

ルータで OSPF の MPLS TE を有効にするには、ネットワークで次の機能がサポートされている必要があります。

- MPLS
- IP シスコ エクスプレス フォワーディング（CEF）

（注） ネットワークのトラフィック エンジニアリング部分にあるすべてのOSPFルータ上で、次のタスクのコマンドを入力する必要があります。

手順の概要

1. **configure**
2. **router ospf** *process-name*
3. **router-id** { *router-id* }
4. **mpls traffic-eng router-id** *interface-type interface-instance*
5. **area** *area-id*
6. **mpls traffic-eng**
7. **interface** *type interface-path-id*
8. **commit**
9. **show ospf** [*process-name*] [*area-id*] **mpls traffic-eng** { **link** | **fragment** }

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router ospf** *process-name* 例： `RP/0/RSP0/cpu 0: router(config)# router ospf 1`	指定したルーティング プロセスにOSPFルーティングを有効にし、ルータ コンフィギュレーション モードでルータを配置します。 （注） *process-name* 引数は、40 文字未満の英数字です。
ステップ 3	**router-id** { *router-id* } 例： `RP/0/RSP0/cpu 0: router(config-ospf)# router-id 192.168.4.3`	OSPF プロセスのルータ ID を設定します。 （注） 固定IPv4アドレスをルータ IDとして使用することを推奨します。
ステップ 4	**mpls traffic-eng router-id** *interface-type interface-instance* 例：	（任意）ノードのトラフィック エンジニアリング ルータ識別子が、指定されたインターフェイスに関連付けられているIPアドレスになるように指定します。

	コマンドまたはアクション	目的
	`RP/0/RSP0/cpu 0: router(config-ospf)# mpls traffic-eng router-id loopback 0`	• この IP アドレスは TE LSA 内のすべてのノードにフラッディングされます。 • 他のノードから始まり、このノードで終了するすべてのトラフィック エンジニアリング トンネルに対して、トンネル宛先を宛先ノードのトラフィック エンジニアリング ルータ ID に設定する必要があります。これは、そのアドレスが、トンネル ヘッドのトラフィック エンジニアリング トポロジ データベースがそのパス計算に使用するアドレスであるためです。 • ループバック インターフェイスは物理インターフェイスより安定しているため、ループバック インターフェイスを MPLS TE ルータ ID に使用することを推奨します。
ステップ 5	**area** *area-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf)# area 0`	エリア コンフィギュレーション モードを開始し、OSPF プロセスのエリアを設定します。 • *area-id* 引数は、area 1000 や area 0.0.3.232 など、ドット付き 10 進表記または IPv4 アドレス形式で入力できます。ただし、1 つのエリアでは同じ形式を選択する必要があります。
ステップ 6	**mpls traffic-eng** 例： `RP/0/RSP0/cpu 0: router(config-ospf)# mpls traffic-eng`	OSPF エリアで MPLS TE を設定します。
ステップ 7	**interface** *type interface-path-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf-ar)# interface interface loopback0`	インターフェイス コンフィギュレーション モードを開始して、1 つ以上のインターフェイスをエリアに関連付けます。
ステップ 8	**commit**	
ステップ 9	**show ospf** [*process-name*] [*area-id*] **mpls traffic-eng** { **link** | **fragment** } 例： `RP/0/RSP0/cpu 0: router# show ospf 1 0 mpls traffic-eng link`	（任意）MPLS TE のローカル ルータで利用可能なリンクとフラグメントに関する情報を表示します。

例

ここでは、次の出力例について説明します。

MPLS TE を設定する前の show ospf コマンドのサンプル出力

次ニ、**show route ospf** EXEC コンフィギュレーション コマンドは、ギガビットイーサネット インターフェイス 0/3/0/0 が存在することおよび MPLS TE が設定されていないことを検証する例を示します。

```
show route ospf 1

 O    11.0.0.0/24 [110/15] via 0.0.0.0, 3d19h, tunnel-te1
 O    192.168.0.12/32 [110/11] via 11.1.0.2, 3d19h, GigabitEthernet0/3/0/0
 O    192.168.0.13/32 [110/6] via 0.0.0.0, 3d19h, tunnel-te1
```

show ospf mpls traffic-eng コマンドの出力例

次に、MPLS TE フラグメントが正しく設定されていることを、**show ospf mpls traffic-eng** EXEC コンフィギュレーション コマンドで検証する例を示します。

```
  show ospf 1 mpls traffic-eng fragment

OSPF Router with ID (192.168.4.3) (Process ID 1)

  Area 0 has 1  MPLS TE fragment. Area instance is 3.
  MPLS router address is 192.168.4.2
  Next fragment ID is 1

  Fragment 0 has 1 link. Fragment instance is 3.
  Fragment has 0 link the same as last update.
  Fragment advertise MPLS router address
    Link is associated with fragment 0. Link instance is 3
      Link connected to Point-to-Point network
      Link ID :55.55.55.55
      Interface Address :192.168.50.21
      Neighbor Address :192.168.4.1
      Admin Metric :0
      Maximum bandwidth :19440000
      Maximum global pool reservable bandwidth :25000000
      Maximum sub pool reservable bandwidth    :3125000
      Number of Priority :8
      Global pool unreserved BW
      Priority 0 :  25000000  Priority 1 :  25000000
      Priority 2 :  25000000  Priority 3 :  25000000
      Priority 4 :  25000000  Priority 5 :  25000000
      Priority 6 :  25000000  Priority 7 :  25000000
      Sub pool unreserved BW
      Priority 0 :   3125000  Priority 1 :   3125000
      Priority 2 :   3125000  Priority 3 :   3125000
      Priority 4 :   3125000  Priority 5 :   3125000
      Priority 6 :   3125000  Priority 7 :   3125000
      Affinity Bit :0
```

次に、エリアインスタンス 3 の MPLS TE リンクが正しく設定されていることを、**show ospf mpls traffic-eng** EXEC コンフィギュレーション コマンドで検証する例を示します。

```
show ospf mpls traffic-eng link

            OSPF Router with ID (192.168.4.1) (Process ID 1)

   Area 0 has 1  MPLS TE links. Area instance is 3.

   Links in hash bucket 53.
     Link is associated with fragment 0. Link instance is 3
       Link connected to Point-to-Point network
       Link ID :192.168.50.20
       Interface Address :192.168.20.50
       Neighbor Address :192.168.4.1
       Admin Metric :0
       Maximum bandwidth :19440000
       Maximum global pool reservable bandwidth :25000000
       Maximum sub pool reservable bandwidth    :3125000
       Number of Priority :8
       Global pool unreserved BW
       Priority 0 :  25000000  Priority 1 :  25000000
       Priority 2 :  25000000  Priority 3 :  25000000
       Priority 4 :  25000000  Priority 5 :  25000000
       Priority 6 :  25000000  Priority 7 :  25000000
       Sub pool unreserved BW
       Priority 0 :   3125000  Priority 1 :   3125000
       Priority 2 :   3125000  Priority 3 :   3125000
       Priority 4 :   3125000  Priority 5 :   3125000
       Priority 6 :   3125000  Priority 7 :   3125000
       Affinity Bit :0
```

show ospf コマンドの MPLS TE を設定した後のコマンドのサンプル出力

次に、**show route ospf** EXEC コンフィギュレーション コマンドで、MPLS TE トンネルがギガビットイーサネット インターフェイス 0/3/0/0 を置き換え、設定が正しく実行されたことを検証する例を示します。

```
show route ospf 1

 O E2 192.168.10.0/24 [110/20] via 0.0.0.0, 00:00:15, tunnel2
 O E2 192.168.11.0/24 [110/20] via 0.0.0.0, 00:00:15, tunnel2
 O E2 192.168.1244.0/24 [110/20] via 0.0.0.0, 00:00:15, tunnel2
 O    192.168.12.0/24 [110/2] via 0.0.0.0, 00:00:15, tunnel2
```

OSPFv3 グレースフル リスタートの設定

このタスクでは、OSPFv3 プロセスのグレースフル リスタートを設定する方法を説明します。このタスクはオプションです。

手順の概要

1. **configure**
2. **router ospfv3** *process-name*
3. **graceful-restart**
4. **graceful-restart lifetime**
5. **graceful-restart interval** *seconds*

6. **graceful-restart helper disable**
7. **commit**
8. **show ospfv3** [*process-name* [*area-id*]] **database grace**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router ospfv3** *process-name* 例： `RP/0/RSP0/cpu 0: router(config)# router ospfv3 test`	OSPFv3 のルータ コンフィギュレーション モードを開始します。プロセス名は OSPF ルーティング プロセスを一意に識別する 1 つの単語です。プロセス名はスペースを含まない 40 文字以内の任意の英数字ストリングです。
ステップ 3	**graceful-restart** 例： `RP/0/RSP0/cpu 0: router(config-ospfv3)#graceful-restart`	現行ルータでグレースフル リスタートをイネーブルにします。
ステップ 4	**graceful-restart lifetime** 例： `RP/0/RSP0/cpu 0: router(config-ospfv3)# graceful-restart lifetime 120`	グレースフル リスタートの最大時間を指定します。 • デフォルトは 95 秒です。 • 値の範囲は 90 ～ 3600 秒です。
ステップ 5	**graceful-restart interval** *seconds* 例： `RP/0/RSP0/cpu 0: router(config-ospfv3)# graceful-restart interval 120`	現行ルータのグレースフル リスタートの間隔（最小時間）を指定します。 • 間隔のデフォルト値は 90 秒です。 • 値の範囲は 90 ～ 3600 秒です。
ステップ 6	**graceful-restart helper disable** 例： `RP/0/RSP0/cpu 0: router(config-ospfv3)# graceful-restart helper disable`	ヘルパー機能をディセーブルにします。
ステップ 7	**commit**	
ステップ 8	**show ospfv3** [*process-name* [*area-id*]] **database grace** 例： `RP/0/RSP0/cpu 0: router# show ospfv3 1 database grace`	グレースフル リスタート リンクのステートを表示します。

グレースフル リスタートに関する情報の表示

ここでは、グレースフル リスタートに関する情報を表示するために使用できるタスクについて説明します。

- 機能がイネーブルかどうかや、グレースフル リスタートが最後に実行された時間を確認するには、show ospf コマンドを使用します。OSPFv3 インスタンスの詳細を参照するには、**show ospfv3** *process-name* [*area-id*] **database grace** コマンドを使用します。

グレースフル リスタート機能のステータスの表示

次の画面出力は、ローカル ルータのグレースフル リスタート機能の状態を示しています。

```
RP/0/RSP0/cpu 0: router# show ospfv3 1

 Routing Process "ospfv3 1" with ID 2.2.2.2
 Initial SPF schedule delay 5000 msecs
 Minimum hold time between two consecutive SPFs 10000 msecs
 Maximum wait time between two consecutive SPFs 10000 msecs
 Initial LSA throttle delay 0 msecs
 Minimum hold time for LSA throttle 5000 msecs
 Maximum wait time for LSA throttle 5000 msecs
 Minimum LSA arrival 1000 msecs
 LSA group pacing timer 240 secs
 Interface flood pacing timer 33 msecs
 Retransmission pacing timer 66 msecs
 Maximum number of configured interfaces 255
 Number of external LSA 0. Checksum Sum 00000000
 Number of areas in this router is 1. 1 normal 0 stub 0 nssa
 Graceful Restart enabled, last GR 11:12:26 ago (took 6 secs)
    Area BACKBONE(0)
        Number of interfaces in this area is 1
        SPF algorithm executed 1 times
        Number of LSA 6. Checksum Sum 0x0268a7
        Number of DCbitless LSA 0
        Number of indication LSA 0
        Number of DoNotAge LSA 0
        Flood list length 0
```

OSPFv3 インスタンスのグレースフル リスタート情報の表示

次の画面出力では、OSPFv3 のリンク ステート インスタンスのリンク ステータスを示します。

```
RP/0/RSP0/cpu 0: router# show ospfv3 1 database grace

            OSPFv3 Router with ID (1.1.1.1) (Process ID 1)

            Grace (Type-11) Link States (Area 0)

LS age: 2
LS Type: Grace Links
Link State ID: 34
Advertising Router: 1.1.1.1
LS Seq Number: 80000001
Checksum: 0x7a4a
Length: 36
```

```
Grace Period : 90
Graceful Restart Reason : Software reload/upgrade
```

OSPFv2 模造リンクの設定

このタスクでは、プロバイダーエッジ（PE）ルータを設定し、VPNバックボーン全体でOSPFv2模造リンク接続を確立する方法について説明します。このタスクはオプションです。

始める前に

マルチプロトコル ラベル スイッチング（MPLS）VPN 内の模造リンクを

プロバイダーエッジ（PE）ルータでは、OSPF を次のように有効にする必要があります。

- OSPF ルーティングプロセスを作成する。
- VRF に属するループバック インターフェイスを設定し、ホスト マスクを使用して IPv4 アドレスを割り当てます。
- エリア サブモードで模造リンクを設定します。

これらのOSPF設定の前提条件の詳細については、OSPF のイネーブル化（425 ページ）を参照してください。

手順の概要

1. **configure**
2. **interface** *type interface-path-id*
3. **vrf** *vrf-name*
4. **ipv4 address** *ip-address mask*
5. end
6. **router ospf** *instance-id*
7. **vrf** *vrf-name*
8. **router-id** { *router-id* }
9. **redistribute bgp** *process-id*
10. **area** *area-id*
11. **sham-link** *source-address destination-address*
12. **cost** *cost*
13. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**interface** *type interface-path-id* 例：	インターフェイス コンフィギュレーション モードを開始します。

	コマンドまたはアクション	目的
	`RP/0/RSP0/cpu 0: router(config)# interface loopback 3`	
ステップ 3	**vrf** *vrf-name* 例： `RP/0/RSP0/cpu 0: router(config-if)# vrf vrf1`	VPN ルーティングおよび転送（VRF）インスタンスをインターフェイスに割り当てます。
ステップ 4	**ipv4 address** *ip-address mask* 例： `RP/0/RSP0/cpu 0: router(config-if)# ipv4 address 172.18.189.38 255.255.255.225`	IP アドレスとサブネット マスクをインターフェイスに割り当てます。
ステップ 5	end 例： `RP/0/RSP0/cpu 0: router(config-if)# end`	設定変更を保存します。 **end** コマンドを実行すると、変更をコミットするように要求されます。 `Uncommitted changes found, commit them before exiting(yes/no/cancel)?[cancel]:` • **yes** と入力すると、実行コンフィギュレーション ファイルに変更が保存され、コンフィギュレーション セッションが終了して、ルータが EXEC モードに戻ります。 • **no** と入力すると、コンフィギュレーション セッションが終了して、ルータが EXEC モードに戻ります。設定の変更はコミットされません。 • **cancel** と入力すると、現在のコンフィギュレーションセッションが継続します。コンフィギュレーション セッションは終了せず、設定変更もコミットされません。
ステップ 6	**router ospf** *instance-id* 例： `RP/0/RSP0/cpu 0: router(config)# router ospf isp`	指定したルーティング プロセスに OSPF ルーティングをイネーブルにし、ルータ コンフィギュレーションモードでルータを配置します。この例では、OSPF インスタンスは isp と呼ばれます。
ステップ 7	**vrf** *vrf-name* 例： `RP/0/RSP0/cpu 0: router(config-ospf)# vrf vrf1`	VRF インスタンスを作成し、VRF コンフィギュレーション モードを開始します。

	コマンドまたはアクション	目的

ステップ 8	**router-id** { *router-id* }
	例：
	`RP/0/RSP0/cpu 0:`
	`router(config-ospf-vrf)# router-id 192.168.4.3`
		OSPF プロセスのルータ ID を設定します。
		（注） 固定 IPv4 アドレスをルータ ID として使用することを推奨します。

ステップ 9	**redistribute bgp** *process-id*
	例：
	`RP/0/RSP0/cpu 0:`
	`router(config-ospf-vrf)# redistribute bgp 1`
		1 つのルーティング ドメインから別のルーティング ドメインに OSPF ルートを再配布します。
		• このコマンドを実行すると、定義上ルータが ASBR になります。
		• OSPF は再配布によって学習したすべてのルートを external とタグ付けします。
		• プロトコルとそのプロセス ID（設定されている場合）は、OSPF に再配布されるプロトコルを示します。
		• BGP VPN ルートが OSPF に再配布される場合、BGP の MED 値は LSA メトリック フィールドにコピーされます。

ステップ 10	**area** *area-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf-vrf)# area 0`	エリア コンフィギュレーション モードを開始し、OSPF プロセスのエリアを設定します。 • *area-id* 引数は、area 1000 や area 0.0.3.232 など、ドット付き 10 進表記または IPv4 アドレス形式で入力できます。ただし、1 つのエリアでは同じ形式を選択する必要があります。
ステップ 11	**sham-link** *source-address destination-address* 例： `RP/0/RSP0/cpu 0: router(config-ospf-vrf-ar)# sham-link 10.0.0.1 10.0.0.3`	2 つの VPN サイト間のポイントツーポイント アンナンバード インターフェイスを設定します。
ステップ 12	**cost** *cost* 例： `RP/0/RSP0/cpu 0: router(config-ospf-vrf-ar-sl)# cost 76`	OSPF インターフェイスでパケットを送信するコストを明示的に指定します。指定したコストは、インターフェイスの自動コスト計算のデフォルト値よりも優先されます。
ステップ 13	**commit**	

OSPF SPF プレフィックスの優先順位付けの設定

このタスクを実行して、OSPF SPF（Shortest Path First）プレフィックス プライオリティを設定します。

手順の概要

1. **configure**
2. **prefix-set** *prefix-set name*
3. **route-policy** *route-policy name* **if destination in** *prefix-set name* **then set spf-priority** {**critical** | **high** | **medium**} **endif**
4. 次のいずれかのコマンドを使用します。
   - **router ospf** *ospf-name*
   - **router ospfv3** *ospfv3-name*
5. **spf prefix-priority route-policy** *route-policy name*
6. **commit**
7. **show rpl route-policy** *route-policy name* **detail**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	

ステップ 2	**prefix-set** *prefix-set name*
	例：
	`RP/0/RSP0/cpu 0: router(config)#prefix-set ospf-critical-prefixes`
	`RP/0/RSP0/cpu 0: router(config-pfx)#66.0.0.0/16`
	`RP/0/RSP0/cpu 0: router(config-pfx)#end-set`
		プレフィックス セットを設定します。

ステップ 3	**route-policy** *route-policy name* **if destination in** *prefix-set name* **then set spf-priority** {**critical** | **high** | **medium**} **endif**
	例：
	`RP/0/RSP0/cpu 0: router#route-policy ospf-spf-priority`
	`RP/0/RSP0/cpu 0: router(config-rpl)#if destination in ospf-critical-prefixes then`
	`set spf-priority critical`
	`endif`
	`RP/0/RSP0/cpu 0: router(config-rpl)#end-policy`
		ルート ポリシーと OSPF SPF プライオリティを設定します。

ステップ 4	次のいずれかのコマンドを使用します。 • **router ospf** *ospf-name* • **router ospfv3** *ospfv3-name* 例：	ルータ OSPF コンフィギュレーション モードを開始します。

	コマンドまたはアクション	目的

	`RP/0/RSP0/cpu 0: router# router ospf 1`
	または
	`RP/0/RSP0/cpu 0: router# router ospfv3 1`	

ステップ 5	**spf prefix-priority route-policy** *route-policy name* 例： または `RP/0/RSP0/cpu 0: router(config-ospfv3)#spf prefix-priority route-policy ospf3-spf-priority`	定義されているルート ポリシーの SPF プレフィックス プライオリティを設定します。 （注） OSPF ルータで spf prefix-priority コマンドを設定します。
ステップ 6	**commit**	

ステップ 7	**show rpl route-policy** *route-policy name* **detail**
	例：
	`RP/0/RSP0/cpu 0: router#show rpl route-policy ospf-spf-priority detail`
	`  prefix-set ospf-critical-prefixes`
	`    66.0.0.0/16`
	`  end-set`
	`  !`
	`  route-policy ospf-spf-priority`
	`    if destination in ospf-critical-prefixes then`
	`      set spf-priority critical`
	`    endif`
	`  end-policy`
	`  !`
		SPF のプレフィックス プライオリティのセットを表示します。

OSPFv2 の multicast-intact の有効化

このオプションの手順では、IPv4 アドレスを使用する OSPFv2 ルータの multicast-intact を有効にする方法について説明します。

手順の概要

1. **configure**
2. **router ospf** *instance-id*
3. **mpls traffic-eng multicast-intact**
4. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	

	コマンドまたはアクション	目的
ステップ 2	**router ospf** *instance-id* 例： `RP/0/RSP0/cpu 0: router(config)# router ospf isp`	指定したルーティング プロセスに OSPF ルーティングをイネーブルにし、ルータ コンフィギュレーション モードでルータを配置します。この例では、OSPF インスタンスは isp と呼ばれます。
ステップ 3	**mpls traffic-eng multicast-intact** 例： `RP/0/RSP0/cpu 0: router(config-ospf)# mpls traffic-eng multicast-intact`	multicast-intact を有効にします。
ステップ 4	**commit**	

インターフェイスの VRF への関連付け

このタスクでは、VPN ルーティングおよび転送（VRF）インスタンスにインターフェイスを関連付ける方法について説明します。

手順の概要

1. **configure**
2. **router ospf** *process-name*
3. **vrf** *vrf-name*
4. **area** *area-id*
5. **interface** *type interface-path-id*
6. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router ospf** *process-name* 例： `RP/0/RSP0/cpu 0: router(config)# router ospf 1`	指定したルーティング プロセスに OSPF ルーティングを有効にし、ルータ コンフィギュレーション モードでルータを配置します。 （注） *process-name* 引数は、40 文字未満の英数字です。
ステップ 3	**vrf** *vrf-name* 例： `RP/0/RSP0/cpu 0: router(config-ospf)# vrf vrf1`	VRF インスタンスを作成し、VRF コンフィギュレーション モードを開始します。

	コマンドまたはアクション	目的
ステップ 4	**area** *area-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf-vrf)# area 0`	エリア コンフィギュレーション モードを開始し、OSPF プロセスのエリアを設定します。 • *area-id* 引数は、area 1000 や area 0.0.3.232 など、ドット付き 10 進表記または IPv4 アドレス形式で入力できます。ただし、1 つのエリアでは同じ形式を選択する必要があります。
ステップ 5	**interface** *type interface-path-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf-vrf-ar)# interface GigabitEthernet 0/0/0/0`	インターフェイス コンフィギュレーション モードを開始して、1 つ以上のインターフェイスを VRF に関連付けます。
ステップ 6	**commit**	

プロバイダーエッジからカスタマーエッジ（PE-CE）プロトコルとしての OSPF の設定

手順の概要

1. **configure**
2. **router ospf** *process-name*
3. **vrf** *vrf-name*
4. **router-id** { *router-id* }
5. **redistribute** *protocol* [*process-id*] { **level-1** | **level-1-2** | **level-2** } [**metric** *metric-value*] [**metric-type** *type-value*] [**match** { **external** [**1** | **2**] }] [**tag** *tag-value*] **route-policy** *policy-name*]
6. **area** *area-id*
7. **interface** *type interface-path-id*
8. exit
9. **domain-id** *[* **secondary** *]* **type** *{* **0005** | **0105** | **0205** | **8005** *}* **value** *value*
10. **domain-tag** *tag*
11. disable-dn-bit-check
12. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	

	コマンドまたはアクション	目的
ステップ 2	**router ospf** *process-name* 例： `RP/0/RSP0/cpu 0: router(config)# router ospf 1`	指定したルーティング プロセスに OSPF ルーティングを有効にし、ルータ コンフィギュレーション モードでルータを配置します。 （注） *process-name* 引数は、40 文字未満の英数字です。
ステップ 3	**vrf** *vrf-name* 例： `RP/0/RSP0/cpu 0: router(config-ospf)# vrf vrf1`	VRF インスタンスを作成し、VRF コンフィギュレーション モードを開始します。
ステップ 4	**router-id** { *router-id* } 例： `RP/0/RSP0/cpu 0: router(config-ospf-vrf)# router-id 192.168.4.3`	OSPF プロセスのルータ ID を設定します。 （注） 固定 IPv4 アドレスをルータ ID として使用することを推奨します。
ステップ 5	**redistribute** *protocol* [*process-id*] { **level-1** | **level-1-2** | **level-2** } [**metric** *metric-value*] [**metric-type** *type-value*] [**match** { **external** [**1** | **2**] }] [**tag** *tag-value*] **route-policy** *policy-name*] 例： `RP/0/RSP0/cpu 0: router(config-ospf-vrf)# redistribute bgp 1 level-1`	1 つのルーティング ドメインから別のルーティング ドメインへの OSPF ルートの再配布 • このコマンドを実行すると、定義上ルータが ASBR になります。 • OSPF は再配布によって学習したすべてのルートを external とタグ付けします。 • プロトコルとそのプロセス ID（設定されている場合）は、OSPF に再配布されるプロトコルを示します。 • メトリックは外部ルートに割り当てるコストです。すべてのプロトコルでデフォルトは 20 です。ただし、BGP のデフォルトのメトリックは 1 です。 • この例では、BGP 自律システム 1、レベル 1 のルートを OSPF にタイプ 2 の外部ルートとして再配布します。
ステップ 6	**area** *area-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf-vrf)# area 0`	エリア コンフィギュレーション モードを開始し、OSPF プロセスのエリアを設定します。 • *area-id* 引数は、area 1000 や area 0.0.3.232 など、ドット付き 10 進表記または IPv4 アドレス形式で入力できます。ただし、1 つのエリアでは同じ形式を選択する必要があります。

	コマンドまたはアクション	目的
ステップ 7	**interface** *type interface-path-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf-vrf)# interface GigabitEthernet 0/0/0/0`	インターフェイス コンフィギュレーション モードを開始して、1 つ以上のインターフェイスを VRF に関連付けます。
ステップ 8	exit 例： `RP/0/RSP0/cpu 0: router(config-if)# exit`	インターフェイス コンフィギュレーション モードを終了します。
ステップ 9	**domain-id** *[* **secondary** *]* **type** *{* **0005** | **0105** | **0205** | **8005** *}* **value** *value* 例： `RP/0/RSP0/cpu 0: router(config-ospf-vrf)# domain-id type 0105 value 1AF234`	OSPF VRF のドメイン ID を指定します。 • *value* 引数は 6 オクテットの 16 進数です。
ステップ 10	**domain-tag** *tag* 例： `RP/0/RSP0/cpu 0: router(config-0spf-vrf)# domain-tag 234`	OSPF VRF ドメイン タグを指定します。 • *tag* の有効範囲は、0 ～ 4294967295 です。
ステップ 11	disable-dn-bit-check 例： `RP/0/RSP0/cpu 0: router(config-ospf-vrf)# disable-dn-bit-check`	ダウン ビットを無視するように設定します。
ステップ 12	**commit**	

複数の OSPF インスタンスの作成（OSPF プロセスおよび VRF）

このタスクでは、複数の OSPF インスタンスの作成方法を説明します。この場合、インスタンスは通常の OSPF インスタンスと VRF インスタンスです。

手順の概要

1. **configure**
2. **router ospf** *process-name*
3. **area** *area-id*
4. **interface** *type interface-path-id*
5. **exit**
6. **vrf** *vrf-name*

7. **area** *area-id*
8. **interface** *type interface-path-id*
9. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router ospf** *process-name* 例： `RP/0/RSP0/cpu 0: router(config)# router ospf 1`	指定したルーティング プロセスに OSPF ルーティングを有効にし、ルータ コンフィギュレーション モードでルータを配置します。 （注） *process-name* 引数は、40 文字未満の英数字です。
ステップ 3	**area** *area-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf)#  area 0`	エリア コンフィギュレーション モードを開始し、バックボーン エリアを設定します。 • *area-id* 引数は、area 1000 や area 0.0.3.232 など、ドット付き 10 進表記または IPv4 アドレス形式で入力できます。ただし、1 つのエリアでは同じ形式を選択する必要があります。IPv4 アドレス形式を使用することを推奨します。
ステップ 4	**interface** *type interface-path-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf-ar)# interface GigabitEthernet 0/1/0/3`	インターフェイス コンフィギュレーション モードを開始して、1 つ以上のインターフェイスをエリアに関連付けます。
ステップ 5	**exit** 例： `RP/0/RSP0/cpu 0: router(config-ospf-ar)# exit`	OSPF コンフィギュレーション モードを開始します。
ステップ 6	**vrf** *vrf-name* 例： `RP/0/RSP0/cpu 0: router(config-ospf)# vrf vrf1`	VRF インスタンスを作成し、VRF コンフィギュレーション モードを開始します。
ステップ 7	**area** *area-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf-vrf)# area 0`	エリア コンフィギュレーション モードを開始して、OSPF プロセスで VRF インスタンスのエリアを設定します。 • *area-id* 引数は、area 1000 や area 0.0.3.232 など、ドット付き 10 進表記または IPv4 アドレス形式

	コマンドまたはアクション	目的
		で入力できます。ただし、1 つのエリアでは同じ形式を選択する必要があります。
ステップ 8	**interface** *type interface-path-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf-vrf)# interface GigabitEthernet 0/0/0/0`	インターフェイス コンフィギュレーション モードを開始して、1 つ以上のインターフェイスを VRF に関連付けます。
ステップ 9	**commit**	

マルチエリアの隣接関係の設定

このタスクでは、OSPF のプライマリ インターフェイスで複数のエリアを作成する方法について説明します。

始める前に

（注） OSF スピーカーが 2 つだけアタッチされている任意のインターフェイスでは、マルチエリアの隣接関係を設定できます。ネイティブ ブロードキャスト ネットワークの場合、マルチエリア隣接関係のインターフェイスを有効にする **network point-to-point** コマンドを使用して、インターフェイスを OPSF ポイントツーポイント型で設定する必要があります。

手順の概要

1. **configure**
2. **router ospf** *process-name*
3. **area** *area-id*
4. **interface** *type interface-path-id*
5. **area** *area-id*
6. **multi-area-interface** *type interface-path-id*
7. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router ospf** *process-name* 例： `RP/0/RSP0/cpu 0: router(config)# router ospf 1`	指定したルーティング プロセスに OSPF ルーティングを有効にし、ルータ コンフィギュレーション モードでルータを配置します。

	コマンドまたはアクション	目的
		（注） *process-name* 引数は、40 文字未満の英数字です。
ステップ 3	**area** *area-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf)#  area 0`	エリア コンフィギュレーション モードを開始し、バックボーン エリアを設定します。 • *area-id* 引数は、area 1000 や area 0.0.3.232 など、ドット付き 10 進表記または IPv4 アドレス形式で入力できます。ただし、1 つのエリアでは同じ形式を選択する必要があります。IPv4 アドレス形式を使用することを推奨します。
ステップ 4	**interface** *type interface-path-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf-ar)# interface Serial 0/1/0/3`	インターフェイス コンフィギュレーション モードを開始して、1 つ以上のインターフェイスをエリアに関連付けます。
ステップ 5	**area** *area-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf)#  area 1`	エリア コンフィギュレーション モードを開始し、複数エリア隣接関係に使用されるエリアを設定します。 • *area-id* 引数は、area 1000 や area 0.0.3.232 など、ドット付き 10 進表記または IPv4 アドレス形式で入力できます。ただし、1 つのエリアでは同じ形式を選択する必要があります。IPv4 アドレス形式を使用することを推奨します。
ステップ 6	**multi-area-interface** *type interface-path-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf)# multi-area-interface Serial 0/1/0/3`	異なる OSPF エリアに対して複数の隣接関係を有効にし、マルチエリア インターフェイス コンフィギュレーション モードを開始します
ステップ 7	**commit**	

OSPF のラベル配布プロトコル IGP 自動設定の設定

このタスクでは、OSPF インスタンスに対する LDP 自動設定を設定する方法について説明します。

オプションで、OSPF インスタンスのエリアにこの機能を設定できます。

手順の概要

1. **configure**

2. **router ospf** *process-name*
3. **mpls ldp auto-config**
4. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router ospf** *process-name* 例： `RP/0/RSP0/cpu 0: router(config)# router ospf 1`	指定したルーティング プロセスに OSPF ルーティングを有効にし、ルータ コンフィギュレーション モードでルータを配置します。 （注） *process-name* 引数は、40 文字未満の英数字です。
ステップ 3	**mpls ldp auto-config** 例： `RP/0/RSP0/cpu 0: router(config-ospf)# mpls ldp auto-config`	OSPF インスタンスの LDP IGP インターフェイスの自動設定をイネーブルにします。 • 任意で、このコマンドを OSPF インスタンスのエリアに設定されます。
ステップ 4	**commit**	

LDP IGP 同期の設定：OSPF

このタスクを実行して、OSPF で LDP IGP 同期を設定します。

（注） デフォルトでは、LDP と IGP 間の同期は行われません。

手順の概要

1. **configure**
2. **router ospf** *process-name*
3. 次のいずれかのコマンドを使用します。
   - **mpls ldp sync**
   - **area** *area-id* **mpls ldp sync**
   - **area** *area-id* **interface** *name* **mpls ldp sync**
4. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router ospf** *process-name* 例： `RP/0/RSP0/cpu 0: router(config)# router ospf 100`	OSPF ルーティング プロセスを識別し、OSPF コンフィギュレーション モードを開始します。
ステップ 3	次のいずれかのコマンドを使用します。 • **mpls ldp sync** • **area** *area-id* **mpls ldp sync** • **area** *area-id* **interface** *name* **mpls ldp sync** 例： `RP/0/RSP0/cpu 0: router(config-ospf)# mpls ldp sync`	インターフェイスで LDP IGP 同期をイネーブルにします。
ステップ 4	**commit**	

OSPF の認証メッセージ ダイジェスト管理の設定

このタスクでは、OSPFインターフェイスのキーチェーンの認証を管理する方法を説明します。

始める前に

このタスクを実行するには、有効なキーチェーンを設定する必要があります。

キーチェーンとそれに関連付けられている属性の設定方法については、 *System Security Configuration Guide for Cisco ASR 9000 Series Routers*の「*Implementing Key Chain Management on Cisco ASR 9000* シリーズ ルータ 」のモジュールを参照してください。

手順の概要

1. **configure**
2. **router ospf** *process-name*
3. **router-id** { *router-id* }
4. **area** *area-id*
5. **interface** *type interface-path-id*
6. **authentication** [**message-digest** *keychain* | **null**]
7. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	

	コマンドまたはアクション	目的
ステップ 2	**router ospf** *process-name* 例： `RP/0/RSP0/cpu 0: router(config)# router ospf 1`	指定したルーティング プロセスに OSPF ルーティングを有効にし、ルータ コンフィギュレーション モードでルータを配置します。 （注） *process-name* 引数は、40 文字未満の英数字です。
ステップ 3	**router-id** { *router-id* } 例： `RP/0/RSP0/cpu 0: router(config-ospf)# router id 192.168.4.3`	OSPF プロセスのルータ ID を設定します。 （注） 固定 IPv4 アドレスをルータ ID として使用することを推奨します。
ステップ 4	**area** *area-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf)# area 1`	エリア コンフィギュレーション モードを開始します。 *area-id* 引数は、area 1000 や area 0.0.3.232 など、ドット付き 10 進表記または IPv4 アドレス形式で入力できます。ただし、1 つのエリアでは同じ形式を選択する必要があります。IPv4 アドレス形式を使用することを推奨します。
ステップ 5	**interface** *type interface-path-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf-ar)# interface GigabitEthernet0/4/0/1`	インターフェイス コンフィギュレーション モードを開始して、1 つ以上のインターフェイスをエリアに関連付けます。
ステップ 6	**authentication** [**message-digest** *keychain* | **null**] 例： `RP/0/RSP0/cpu 0: router(config-ospf-ar-if)# authentication message-digest keychain ospf_int1`	MD5 キーチェーンを設定します。 （注） この例では、この手順を実行する前に *ospf_intl* キーチェーンを設定する必要があります。
ステップ 7	**commit**	

例

次の例は、5 つのキー ID を持つキーチェーン *ospf_intf_1* の設定方法を示します。各キー ID は異なる **send-lifetime** 値で設定されます。ただし、すべてのキー ID はキーに同じテキスト文字列を指定します。

```
key chain ospf_intf_1
key 1
send-lifetime 11:30:30 May 1 2007 duration 600
cryptographic-algorithm MD5T
key-string clear ospf_intf_1
key 2
```

```
send-lifetime 11:40:30 May 1 2007 duration 600
cryptographic-algorithm MD5
key-string clear ospf_intf_1
key 3
send-lifetime 11:50:30 May 1 2007 duration 600
cryptographic-algorithm MD5
key-string clear ospf_intf_1
key 4
send-lifetime 12:00:30 May 1 2007 duration 600
cryptographic-algorithm MD5
key-string clear ospf_intf_1
key 5
send-lifetime 12:10:30 May 1 2007 duration 600
cryptographic-algorithm MD5
key-string clear ospf_intf_1
```

次の例は、ギガビット イーサネット インターフェイス 0/4/0/1 でキーチェーン認証が有効になっていることを示します。

```
show ospf 1 interface GigabitEthernet0/4/0/1

GigabitEthernet0/4/0/1 is up, line protocol is up
  Internet Address 100.10.10.2/24, Area 0
  Process ID 1, Router ID 2.2.2.1, Network Type BROADCAST, Cost: 1
  Transmit Delay is 1 sec, State DR, Priority 1
  Designated Router (ID) 2.2.2.1, Interface address 100.10.10.2
  Backup Designated router (ID) 1.1.1.1, Interface address 100.10.10.1
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    Hello due in 00:00:02
  Index 3/3, flood queue length 0
  Next 0(0)/0(0)
  Last flood scan length is 2, maximum is 16
  Last flood scan time is 0 msec, maximum is 0 msec
  Neighbor Count is 1, Adjacent neighbor count is 1
    Adjacent with neighbor 1.1.1.1  (Backup Designated Router)
  Suppress hello for 0 neighbor(s)
  Keychain-based authentication enabled
    Key id used is 3
  Multi-area interface Count is 0
```

次に、アクティブに設定されたキーの出力の例を示します。

```
show key chain ospf_intf_1

Key-chain: ospf_intf_1/ -

Key 1 -- text "0700325C4836100B0314345D"
  cryptographic-algorithm -- MD5
  Send lifetime:   11:30:30, 01 May 2007 - (Duration) 600
  Accept lifetime: Not configured
Key 2 -- text "10411A0903281B051802157A"
  cryptographic-algorithm -- MD5
  Send lifetime:   11:40:30, 01 May 2007 - (Duration) 600
  Accept lifetime: Not configured
Key 3 -- text "06091C314A71001711112D5A"
  cryptographic-algorithm -- MD5
  Send lifetime:   11:50:30, 01 May 2007 - (Duration) 600  [Valid now]
  Accept lifetime: Not configured
Key 4 -- text "151D181C0215222A3C350A73"
  cryptographic-algorithm -- MD5
```

```
  Send lifetime:   12:00:30, 01 May 2007 - (Duration) 600
  Accept lifetime: Not configured
Key 5 -- text "151D181C0215222A3C350A73"
  cryptographic-algorithm -- MD5
  Send lifetime:   12:10:30, 01 May 2007 - (Duration) 600
  Accept lifetime: Not configured
```

OSPF の一般 TTL セキュリティ メカニズム（GTSM）の設定

このタスクでは、GTSMのインターフェイスにおけるセキュリティの存続可能時間メカニズムの設定方法を説明します。

手順の概要

1. **configure**
2. **router ospf** *process-name*
3. **router-id** { *router-id* }
4. **log adjacency changes** [**detail** | **disable**]
5. **nsf** { **cisco** [**enforce global**] | **ietf** [**helper disable**]}
6. **timers throttle spf** *spf-start spf-hold spf-max-wait*
7. **area** *area-id*
8. **interface** *type interface-path-id*
9. **security ttl** [**disable** | **hops** *hop-count*]
10. **commit**
11. **show ospf** [*process-name*] [**vrf** *vrf-name*] [*area-id*] **interface** [*type interface-path-id*]

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router ospf** *process-name* 例： `RP/0/RSP0/cpu 0: router(config)# router ospf 1`	指定したルーティング プロセスに OSPF ルーティングを有効にし、ルータ コンフィギュレーション モードでルータを配置します。 （注） *process-name* 引数は、40 文字未満の英数字です。
ステップ 3	**router-id** { *router-id* } 例： `RP/0/RSP0/cpu 0: router(config-ospf)# router id 10.10.10.100`	OSPF プロセスのルータ ID を設定します。 （注） 固定 IPv4 アドレスをルータ ID として使用することを推奨します。
ステップ 4	**log adjacency changes** [**detail**	**disable**] 例：	（任意）ネイバー変更の通知を要求します。 • デフォルトでは、この機能はイネーブルです。

	コマンドまたはアクション	目的
	`RP/0/RSP0/cpu 0: router(config-ospf-ar-if)# log adjacency changes detail`	• ネイバー変更によって生成されたメッセージは通知と見なされます。このメッセージはlogging console コマンドで重大度レベル 5 に分類されます。logging console コマンドではどの重大度レベルのメッセージをコンソールに送信するかを制御します。デフォルトでは、すべての重大度レベルのメッセージが送信されます。
ステップ 5	**nsf** { **cisco** [**enforce global**] | **ietf** [**helper disable**]} 例： `RP/0/RSP0/cpu 0: router(config-ospf)# nsf ietf`	（任意）NSF OSPF プロトコルを設定します。 この例ではグレースフル リスタートをイネーブルにします。
ステップ 6	**timers throttle spf** *spf-start spf-hold spf-max-wait* 例： `RP/0/RSP0/cpu 0: router(config-ospf)# timers throttle spf 500 500 10000`	（任意）SPF スロットリング タイマーを設定します。
ステップ 7	**area** *area-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf)# area 1`	エリア コンフィギュレーション モードを開始します。 *area-id* 引数は、area 1000 や area 0.0.3.232 など、ドット付き 10 進表記または IPv4 アドレス形式で入力できます。ただし、1 つのエリアでは同じ形式を選択する必要があります。IPv4 アドレス形式を使用することを推奨します。
ステップ 8	**interface** *type interface-path-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf-ar)# interface GigabitEternet0/5/0/0`	インターフェイス コンフィギュレーション モードを開始して、1 つ以上のインターフェイスをエリアに関連付けます。
ステップ 9	**security ttl** [**disable** | **hops** *hop-count*] 例： `RP/0/RSP0/cpu 0: router(config-ospf-ar-if)# security ttl hopes 2`	OSPF パケットの IP ヘッダーのセキュリティ TTL 値を設定します。
ステップ 10	**commit**	
ステップ 11	**show ospf** [*process-name*][**vrf** *vrf-name*][*area-id*] **interface** [*type interface-path-id*] 例：	OSPF インターフェイス情報を表示します。

	コマンドまたはアクション	目的
	`RP/0/RSP0/cpu 0: router# show ospf 1 interface GigabitEthernet0/5/0/0`	

例

OSPF インターフェイスに設定されている GTSM セキュリティ TTL 値を表示する出力例を次に示します。

```
show ospf 1 interface GigabitEthernet0/5/0/0

 GigabitEthernet0/5/0/0 is up, line protocol is up
   Internet Address 120.10.10.1/24, Area 0
   Process ID 1, Router ID 100.100.100.100, Network Type BROADCAST, Cost: 1
   Transmit Delay is 1 sec, State BDR, Priority 1
   TTL security enabled, hop count 2
   Designated Router (ID) 102.102.102.102, Interface address 120.10.10.3
   Backup Designated router (ID) 100.100.100.100, Interface address 120.10.10.1
   Flush timer for old DR LSA due in 00:02:36
   Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
     Hello due in 00:00:05
   Index 1/1, flood queue length 0
   Next 0(0)/0(0)
   Last flood scan length is 1, maximum is 4
   Last flood scan time is 0 msec, maximum is 0 msec
   Neighbor Count is 1, Adjacent neighbor count is 1
     Adjacent with neighbor 102.102.102.102  (Designated Router)
   Suppress hello for 0 neighbor(s)
   Multi-area interface Count is 0
```

OSPF の設定と動作の確認

このタスクでは、OSPF の設定および操作を確認する方法について説明します。

手順の概要

1. **show** { **ospf** | **ospfv3** } [*process-name*]
2. **show** { **ospf** | **ospfv3** } [*process-name*] **border-routers** [*router-id*]
3. **show** { **ospf** | **ospfv3** } [*process-name*] **database**
4. **show** { **ospf** | **ospfv3** } [*process-name*] [*area-id*] **flood-list interface** *type interface-path-id*
5. **show** { **ospf** | **ospfv3** } [*process-name*] [**vrf** *vrf-name*] [*area-id*] **interface** [*type interface-path-id*]
6. **show** { **ospf** | **ospfv3** }[*process-name*] [*area-id*] **neighbor** [*t ype interface- path-id*] [*neighbor-id*] [**detail**]
7. **clear** { **ospf** | **ospfv3** }[*process-name*] **process**
8. **clear**{**ospf**|**ospfv3**[*process-name*] **redistribution**
9. **clear**{**ospf**|**ospfv3**[*process-name*] **routes**
10. **clear**{**ospf**|**ospfv3**[*process-name*] **vrf** [*vrf-name*|**all**] {**process** |**redistribution**|**routes**|**statistics** [**interface** *type interface-path-id*|**message-queue**|**neighbor**]}

11. **clear** { **ospf** | **ospfv3** }[*process-name*] **statistics** [**neighbor** [*type interface-path-id*] [*ip-address*]]

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**show** { **ospf** | **ospfv3** } [*process-name*] 例： `RP/0/RSP0/cpu 0: router# show ospf group1`	（任意）OSPF ルーティング プロセスに関する一般情報を表示します。
ステップ 2	**show** { **ospf** | **ospfv3** } [*process-name*] **border-routers** [*router-id*] 例： `RP/0/RSP0/cpu 0: router# show ospf group1 border-routers`	（任意）ABR および ASBR への内部 OSPF ルーティング テーブル エントリを表示します。
ステップ 3	**show** { **ospf** | **ospfv3** } [*process-name*] **database** 例： `RP/0/RSP0/cpu 0: router# show ospf group2 database`	（任意）特定のルータの OSPF データベースに関係する情報の一覧を表示します。 • このコマンドは、さまざまな形式で、異なる OSPF LSA に関する情報を提供します。
ステップ 4	**show** { **ospf** | **ospfv3** } [*process-name*] [*area-id*] **flood-list interface** *type interface-path-id* 例： `RP/0/RSP0/cpu 0: router# show ospf 100 flood-list interface GigabitEthernet 0/3/0/0`	（任意）インターフェイス上でのフラッディングを待機している OSPF LSA のリストを表示します。
ステップ 5	**show** { **ospf** | **ospfv3** } [*process-name*] [**vrf** *vrf-name*] [*area-id*] **interface** [*type interface-path-id*] 例： `RP/0/RSP0/cpu 0: router# show ospf 100 interface GigabitEthernet 0/3/0/0`	（任意）OSPF インターフェイス情報を表示します。
ステップ 6	**show** { **ospf** | **ospfv3** }[*process-name*] [*area-id*] **neighbor** [*t ype interface- path-id*] [*neighbor-id*] [**detail**] 例： `RP/0/RSP0/cpu 0: router# show ospf 100 neighbor`	（任意）個々のインターフェイスに基づいた OSPF ネイバー情報を表示します。
ステップ 7	**clear** { **ospf** | **ospfv3** }[*process-name*] **process** 例：	（任意）OSPF ルータ プロセスを停止および再起動せずにリセットします。

	コマンドまたはアクション	目的
	RP/0/RSP0 /CPU0:router# clear ospf 100 process	
ステップ 8	**clear**{**ospf**|**ospfv3**[*process-name*] **redistribution** 例： RP/0/RSP0/cpu 0: router#clear ospf 100 redistribution	OSPF ルート再配布をクリアします。
ステップ 9	**clear**{**ospf**|**ospfv3**[*process-name*] **routes** 例： RP/0/RSP0/cpu 0: router#clear ospf 100 routes	OSPF ルート テーブルをクリアします。
ステップ 10	**clear**{**ospf**|**ospfv3**[*process-name*] **vrf** [*vrf-name*|**all**] {**process** |**redistribution**|**routes**|**statistics** [**interface** *type interface-path-id*|**message-queue**|**neighbor**]} 例： RP/0/RSP0/cpu 0: router#clear ospf 100 vrf vrf_1 process	OSPF ルート テーブルをクリアします。
ステップ 11	**clear** { **ospf** | **ospfv3** }[*process-name*] **statistics** [**neighbor** [*type interface-path-id*] [*ip-address*]] 例： RP/0/RSP0/cpu 0: router# clear ospf 100 statistics	（任意）ネイバー状態遷移の OSPF 統計情報をクリアします。

IP 高速再ルーティング ループフリー代替の設定

この手順では、リンク障害に関連するトラフィックフローを収束させるために IP 高速再ルーティング（IPFRR）リンク単位のループフリー代替（LFA）の計算を有効にする方法について説明します。

ブロードキャストリンクで保護を有効にするには、OSPF 下のインターフェイスで IPFRR および双方向フォワーディング検出（BFD）を有効にする必要があります。

IPFRR LFA の有効化

手順の概要

1. **configure**
2. **router ospf** *process-name*
3. **area** *area-id*

4. **interface** *type interface-path-id*
5. **fast-reroute per-link** { **enable** | **disable** }
6. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router ospf** *process-name* 例： `RP/0/RSP0/cpu 0: router(config)# router ospf`	指定したルーティング プロセスに OSPF ルーティングを有効にし、ルータ コンフィギュレーション モードでルータを配置します。
ステップ 3	**area** *area-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf)#area 1`	エリア コンフィギュレーション モードを開始します。
ステップ 4	**interface** *type interface-path-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf-ar)# interface GigabitEternet0/5/0/0`	インターフェイス コンフィギュレーション モードを開始して、1 つ以上のインターフェイスをエリアに関連付けます。
ステップ 5	**fast-reroute per-link** { **enable** | **disable** } 例： `RP/0/RSP0/cpu 0: router(config-ospf-ar)#fast-reroute per-link enable`	インターフェイスのリンク単位の LFA 計算を有効または無効にします。
ステップ 6	**commit**	

リンク単位のIP高速再ルーティングの計算からのインターフェイスの除外

手順の概要

1. **configure**
2. **router ospf** *process-name*
3. **area** *area-id*
4. **interface** *type interface-path-id*
5. **fast-reroute per-link exclude interface** *type interface-path-id*
6. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router ospf** *process-name* 例： `RP/0/RSP0/cpu 0: router(config)# router ospf`	指定したルーティングプロセスに OSPF ルーティングを有効にし、ルータ コンフィギュレーション モードでルータを配置します。
ステップ 3	**area** *area-id* 例： `RP/0/RSP0/cpu 0: router(config)#area area-id`	エリア コンフィギュレーション モードを開始します。
ステップ 4	**interface** *type interface-path-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf)#interface type interface-path-id`	インターフェイス コンフィギュレーション モードを開始して、1 つ以上のインターフェイスをエリアに関連付けます。
ステップ 5	**fast-reroute per-link exclude interface** *type interface-path-id* 例： `RP/0/RSP0/cpu 0: router(config-ospf-ar)# fast-reroute per-link exclude interface GigabitEternet0/5/0/1`	リンク単位の IP 高速再ルーティングの計算からインターフェイスを除外します。
ステップ 6	**commit**	

SRMS サーバとの OSPF 連携動作の有効化

SRMS サーバとの OSPF 連携動作を有効にするには、次の手順を実行します。

手順の概要

1. **configure**
2. **router ospf** *instance-id*
3. **segment-routing mpls**
4. **segment-routing forwarding mpls**
5. **segment-routing prefix-sid-mapadvertise-local**
6. **segment-routing sr-preferprefix-list***[acl-name]*

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	

	コマンドまたはアクション	目的
ステップ 2	**router ospf** *instance-id* 例： `RP/0/RSP0/cpu 0: router(config)# router ospf isp`	指定したルーティング インスタンスに OSPF ルーティングを有効にし、ルータ コンフィギュレーション モードでルータを配置します。
ステップ 3	**segment-routing mpls** 例： `RP/0/RSP0/cpu 0: router(config-ospf)# segment-routing mpls`	
ステップ 4	**segment-routing forwarding mpls** 例： `RP/0/RSP0/cpu 0: router(config-ospf)# segment-routing forwarding mpls`	このインスタンス OSPF が有効になっているすべてのインターフェイスで SR 転送を有効にします。
ステップ 5	**segment-routing prefix-sid-mapadvertise-local** 例： `RP/0/RSP0/cpu 0: router(config-ospf)# segment-routing prefix-sid-map advertise local`	サーバ機能を有効にし、OSPFがエリア範囲のフラッディングを使用してローカル マッピング エントリをアドバタイズできるようにします。フラッディングは、セグメントルーティングが有効になっているエリアに限定されます。デフォルトでは、無効になっています。
ステップ 6	**segment-routing sr-preferprefix-list***[acl-name]* 例： `RP/0/RSP0/cpu 0: router(config-ospf)# segment-routing sr-prefer prefix-list foo`	SR ラベルが LDP ラベルよりも優先されるルーティング情報ベース（RIB）と通信するように、OSPF を有効にします。ACL が使用されている場合、OSPF は、ACL に一致するプレフィックスの LDP ラベルを介した SR ラベルの優先順位を通知します。ACL を使用しない場合は、OSPF がすべてのプレフィックスの SR ラベルの優先順位を通知します。

例

次に、エリアフラッディング範囲を使用して、OSPFがローカルマッピングエントリをアドバタイズする例を示します。

```
ipv4 prefix-list foo
permit 2.2.2.2/32
!
router ospf 1
router-id 1.1.1.1
segment-routing mpls
segment-routing forwarding mpls
segment-routing prefix-sid-map receive
segment-routing prefix-sid-map advertise-local
segment-routing sr-prefer prefix-list foo
area 0
```

```
interface Loopback0
prefix-sid index 1
!
interface GigabitEthernet0/0/0/0
!
interface GigabitEthernet0/2/0/0
!
interface GigabitEthernet0/2/0/3
!
!
area 1
interface GigabitEthernet0/2/0/7
!
```

OSPF の実装の設定例

ここでは、次の設定例について説明します。

Cisco IOS XR ソフトウェア OSPF Version 2 の設定：例

次の例では、のエリアのCisco IOS XR ソフトウェアOSPF インターフェイスの設定方法を示します。

エリア 0 は **area** コマンドで明示的に設定する必要があり、10.1.2.0 から 10.1.2.255 の範囲にあるすべてのインターフェイスはエリア 0 にバインドされます。インターフェイスは（エリア コンフィギュレーション モードにルータがある間に）**interface** コマンドで設定され、インターフェイス ステートメントには **area** キーワードを含めません。

Cisco IOS XR ソフトウェア の設定

```
interface GigabitEthernet 0/3/0/0
 ip address 10.1.2.1 255.255.255.255
 negotiation auto
!
router ospf 1
router-id 10.2.3.4
 area 0
  interface GigabitEthernet 0/3/0/0
!
!
```

次の例では、Cisco IOS XR ソフトウェアのエリアでの OSPF インターフェイス パラメータの設定方法を示します。

Cisco IOS XR ソフトウェアでは、OSPF インターフェイスの固有のパラメータがインターフェイス コンフィギュレーション モードで設定され、エリア 0 に明示的に定義されます。さらに、**ip ospf** キーワードは必要なくなりました。

Cisco IOS XR ソフトウェアの設定

```
interface GigabitEthernet 0/3/0/0
 ip address 10.1.2.1 255.255.255.0
```

```
 negotiation auto
!
router ospf 1
 router-id 10.2.3.4
area 0
 interface GigabitEthernet 0/3/0/0
  cost 77
  mtu-ignore
  authentication message-digest
  message-digest-key 1 md5 0 test
!
!
```

次の例では Cisco IOS XR ソフトウェアの階層 CLI 構造を示します。

Cisco IOS XR ソフトウェアでは、OSPF エリアは明示的に設定する必要があり、エリア コンフィギュレーション モードで設定されたインターフェイスは、そのエリアに明示的にバインドされています。この例では、インターフェイス 10.1.2.0/24 がエリア 0 に、インターフェイス 10.1.3.0/24 がエリア 1 にバインドされています。

Cisco IOS XR ソフトウェアの設定

```
interface GigabitEthernet 0/3/0/0
 ip address 10.1.2.1 255.255.255.0
 negotiation auto
!
interface GigabitEthernet 0/3/0/1
 ip address 10.1.3.1 255.255.255.0
 negotiation auto
!
router ospf 1
 router-id 10.2.3.4
area 0
 interface GigabitEthernet 0/3/0/0
!
area 1
 interface GigabitEthernet 0/3/0/1
!
!
```

OSPF Version 2 の CLI 継承および優先：例

次の例では、OSPF トポロジの異なる階層レベルでコスト パラメータを設定します。また、パラメータが継承される方法と 1 つの設定だけが優先される方法について説明します。優先ルールに従って、最も明示的な設定が使用されます。

コスト パラメータは、OSPF プロセスのルータ コンフィギュレーション モードで 5 に設定されます。エリア 1 はコストを 15 に、エリア 6 はコストを 30 に設定します。エリア 0 またはそのインターフェイスではコストが設定されていないため、エリア 0 のすべてのインターフェイスは OSPF プロセスの 5 のコストを継承します。

エリア 1 のすべてのインターフェイスには 15 のコストがあります。これは、エリア 1 でコストが設定され、ルータ コンフィギュレーション モードで設定された 5 が 15 で上書きされるためです。

エリア 4 ではコストを設定しませんが、ギガビットイーサネット インターフェイス 01/0/2 ではコストを 20 に設定します。エリア 4 の残りのインターフェイスには、OSPF プロセスから継承された 5 のコストがあります。

エリア 6 はコストを 30 に設定し、ギガビットイーサネット インターフェイス 0/1/0/3 および 0/2/0/3 によって継承されます。ギガビットイーサネット インターフェイス 0/3/0/3 は 1 のコストを使用します。これはインターフェイス コンフィギュレーション モードで設定されます。

```
router ospf 1
 router-id 10.5.4.3
 cost 5
 area 0
  interface GigabitEthernet 0/1/0/0
  !
  interface GigabitEthernet 0/2/0/0
  !
  interface GigabitEthernet 0/3/0/0
  !
 !
 area 1
  cost 15
  interface GigabitEthernet 0/1/0/1
  !
  interface GigabitEthernet 0/2/0/1
  !
  interface GigabitEthernet 0/3/0/1
  !
 !
 area 4
  interface GigabitEthernet 0/1/0/2
   cost 20
  !
  interface GigabitEthernet 0/2/0/2
  !
  interface GigabitEthernet 0/3/0/2
  !
 !
 area 6
  cost 30
  interface GigabitEthernet 0/1/0/3
  !
  interface GigabitEthernet 0/2/0/3
  !
  interface GigabitEthernet 0/3/0/3
   cost 1
  !
 !
```

OSPF Version 2 の MPLS TE：例

次の例では、MPLS TE の OSPF 部分の設定方法を示します。ただし、引き続き MPLS TE トポロジを構築して、MPLS TE トンネルを作成する必要があります。詳細については、 *MPLS Configuration Guide for Cisco ASR 9000 Series RoutersMPLS Configuration Guide for Cisco NCS 560 Series Routers*を参照してください。

この例では、ループバック インターフェイス 0 がエリア 0 に関連付けられ、MPLS TE がエリア 0 内で設定されています。

```
interface Loopback 0
 address 10.10.10.10 255.255.255.0
!
interface GigabitEthernet 0/2/0/0
 address 10.1.2.2 255.255.255.0
!
router ospf 1
 router-id 10.10.10.10
 nsf
 auto-cost reference-bandwidth 10000
 mpls traffic-eng router-id Loopback 0
 area 0
  mpls traffic-eng
  interface GigabitEthernet 0/2/0/0
  interface Loopback 0
```

OSPFv3 の集約を持つ ABR：例

次の例では、エリア 1 からバックボーンに集約されたプレフィックス範囲 2300::/16 を示します。

```
router ospfv3 1
 router-id 192.168.0.217
 area 0
  interface GigabitEthernet 0/2/0/1
 area 1
  range 2300::/16
  interface GigabitEthernet 0/2/0/0
```

OSPFv3 の ABR スタブ エリア：例

エリア 1 がスタブ エリアとして設定される例を次に示します。

```
 router ospfv3 1
  router-id 10.0.0.217
  area 0
   interface GigabitEthernet 0/2/0/1
  area 1
   stub
   interface GigabitEthernet 0/2/0/0
```

OSPFv3 の ABR 完全スタブ エリア：例

エリア 1 が完全スタブ エリアとして設定される例を次に示します。

```
router ospfv3 1
 router-id 10.0.0.217
 area 0
  interface GigabitEthernet 0/2/0/1
 area 1
  stub no-summary
  interface GigabitEthernet 0/2/0/0
```

OSPF SPF プレフィックスの優先順位付けの設定：例

この例では、/32 プレフィックスを一般的に medium プライオリティに設定し、一部の /32 および /24 プレフィックスを critical プライオリティおよび high プライオリティ キューに設定する方法を示します。

```
prefix-set ospf-critical-prefixes
  192.41.5.41/32,
  11.1.3.0/24,
  192.168.0.44/32
  end-set
  !
  prefix-set ospf-high-prefixes
  44.4.10.0/24,
  192.41.4.41/32,
  41.4.41.41/32
  end-set
  !
  prefix-set ospf-medium-prefixes
  0.0.0.0/0 ge 32
  end-set
  !

  route-policy ospf-priority
    if destination in ospf-high-prefixes then
      set spf-priority high
    else
      if destination in ospf-critical-prefixes then
        set spf-priority critical
      else
        if destination in ospf-medium-prefixes then
          set spf-priority medium
        endif
      endif
    endif
  end-policy
```

OSPFv2

```
router ospf 1
   spf prefix-priority route-policy ospf-priority
   area 0
    interface GigabitEthernet0/3/0/0
    !
   !
   area 3
    interface GigabitEthernet0/2/0/0
    !
   !
   area 8
    interface GigabitEthernet0/2/0/0.590
```

OSPFv3

```
router ospfv3 1
   spf prefix-priority route-policy ospf-priority
   area 0
    interface GigabitEthernet0/3/0/0
    !
```

```
 !
 area 3
  interface GigabitEthernet0/2/0/0
  !
 !
 area 8
  interface GigabitEthernet0/2/0/0.590
```

OSPFv3 のルート再配布：例

次の例では、プレフィックスリストを使用して、他のプロトコルから再配布されるルートを制限します。

上位 32 ビットの 9898:1000 および 32 から 64 のプレフィックス長を持つルートだけが BGP 42 から再配布されます。このパターンに一致しないルートだけが BGP 1956 から再配布されます。

```
ipv6 prefix-list list1
 seq 10 permit 9898:1000::/32 ge 32 le 64
ipv6 prefix-list list2
 seq 10 deny 9898:1000::/32 ge 32 le 64
 seq 20 permit ::/0 le 128
router ospfv3 1
 router-id 10.0.0.217
 redistribute bgp 42
 redistribute bgp 1956
 distribute-list prefix-list list1 out bgp 42
 distribute-list prefix-list list2 out bgp 1956
 area 1
  interface GigabitEthernet 0/2/0/0
```

OSPFv3 のエリア 1 から設定された仮想リンク：例

この例では、エリア 0 と 1 および仮想リンク 10.0.0.217 と 10.0.0.212 で構成される OSPFv3 トポロジのエリア 1 からバックボーンを接続するように仮想リンクを設定する方法を説明します。

ABR 1 の設定

```
router ospfv3 1
 router-id 10.0.0.217
 area 0
  interface GigabitEthernet 0/2/0/1
 area 1
  virtual-link 10.0.0.212
  interface GigabitEthernet 0/2/0/0
```

ABR 2 の設定

```
router ospfv3 1
 router-id 10.0.0.212
 area 0
  interface GigabitEthernet 0/3/0/1
 area 1
```

```
  virtual-link 10.0.0.217
  interface GigabitEthernet 0/2/0/0
```

次のように、仮想リンクを確認します。

```
show ospfv3 virtual-links
Mon Dec 17 11:18:29.249 EST

Virtual Link OSPF_VL0 to router 192.168.0.4 is up
Interface ID 1000000, IPv6 address 13:13:13::4
Run as demand circuit
DoNotAge LSA allowed.
Transit area 1, via interface GigabitEthernet0/0/0/0, Cost of using 2
Transmit Delay is 1 sec, State POINT_TO_POINT,
Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
Hello due in 00:00:06
Adjacency State INIT (Hello suppressed)
Index 0/0/0, retransmission queue length 0, number of retransmission 0
First 0(0)/0(0)/0(0) Next 0(0)/0(0)/0(0)
Last retransmission scan length is 0, maximum is 0
Last retransmission scan time is 0 msec, maximum is 0 msec
```

OSPF Version 2 の MD5 認証を使用して設定された仮想リンク：例

次の例では、バックボーンへの仮想リンクを設定して、MD5 認証を適用する方法を示します。説明されている手順は、仮想リンクの各端にある両方の ABR で実行する必要があります。

ABR を明示的に設定したら、そのインターフェイスの値を上書きし、明示的に設定しないかぎり、エリアにバインドされているすべてのインターフェイスにコンフィギュレーションが継承されます。

仮想リンクを理解するには、OSPF の仮想リンクおよび中継エリア （406 ページ）を参照してください。

この例では、ルータ ABR1 のすべてのインターフェイスは MD5 認証を使用します。

```
router ospf ABR1
 router-id 10.10.10.10
 authentication message-digest
 message-digest-key 100 md5 0 cisco
 area 0
  interface GigabitEthernet 0/2/0/1
  interface GigabitEthernet 0/3/0/0
 area 1
  interface GigabitEthernet 0/3/0/1
  virtual-link 10.10.5.5
!
!
```

この例では、ルータ ABR3 のエリア 1 インターフェイスだけが MD5 認証を使用します。

```
router ospf ABR2
 router-id 10.10.5.5
 area 0
 area 1
  authentication message-digest
  message-digest-key 100 md5 0 cisco
  interface GigabitEthernet 0/9/0/1
  virtual-link 10.10.10.10
```

```
 area 3
  interface Loopback 0
  interface GigabitEthernet 0/9/0/0
!
```

OSPF Version 2 に設定された VPN バックボーンと模造リンク：例

次の例では、VPN バックボーンと模造リンクの接続を確立するようにプロバイダーエッジ（PE）ルータを設定する方法を示します。

```
logging console debugging
 vrf vrf_1
  address-family ipv4 unicast
  import route-target
   100:1
   !
  export route-target
   100:1
   !
 !
!
  interface Loopback0
   ipv4 address 2.2.2.1 255.255.255.255
   !
  interface Loopback1
   vrf vrf_1
   ipv4 address 10.0.1.3 255.255.255.255
   !
  interface GigabitEthernet0/2/0/2
   vrf vrf_1
   ipv4 address 100.10.10.2 255.255.255.0
   !
  interface GigabitEthernet0/2/0/3
   ipv4 address 100.20.10.2 255.255.255.0
   !
  !
route-policy pass-all
 pass
 end-policy
 !
router ospf 1
 log adjacency changes
 router-id 2.2.2.2
 vrf vrf_1
  router-id 22.22.22.2
  domain-id type 0005 value 111122223333
  domain-tag 140
  nsf ietf
  redistribute bgp 10
  area 0
   sham-link 10.0.1.3 10.0.0.101
    !
   interface GigabitEthernet0/2/0/2
    !
  !
 !
!
router ospf 2
 router-id 2.22.2.22
 area 0
  interface Loopback0
```

```
  !
 interface GigabitEthernet0/2/0/3
  !
  !
!
router bgp 10
 bgp router-id 2.2.2.1
 bgp graceful-restart restart-time 300
 bgp graceful-restart
 address-family ipv4 unicast
  redistribute connected
  !
 address-family vpnv4 unicast
  !
 neighbor 2.2.2.2
 remote-as 10
 update-source Loopback0
 address-family ipv4 unicast
  !
 address-family vpnv4 unicast
  !
 !
 vrf vrf_1
  rd 100:1
   address-family ipv4 unicast
    redistribute connected route-policy pass-all
    redistribute ospf 1 match internal external
    !
   !
  !
 mpls ldp
  router-id 2.2.2.1
  interface GigabitEthernet0/2/0/3
  !
 !
```

設定用の show コマンドは、次のようになります。

```
show ospf vrf all-inclusive sham-links
Mon Dec 17 10:27:41.815 EST


Sham Links for OSPF 1, VRF vrf1

Sham Link OSPF_SL0 to address 3.3.3.3 is up
Area 1, source address 1.1.1.1
IfIndex = 3
Run as demand circuit
DoNotAge LSA allowed., Cost of using 2
Transmit Delay is 1 sec, State POINT_TO_POINT,
Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
Hello due in 00:00:08:911
Adjacency State FULL (Hello suppressed)
Number of DBD retrans during last exchange 0
Index 2/2, retransmission queue length 0, number of retransmission 1
First 0(0)/0(0) Next 0(0)/0(0)
Last retransmission scan length is 1, maximum is 1
Last retransmission scan time is 0 msec, maximum is 0 msec
Keychain-based authentication enabled
Keychain name key1
Key id used is 1
Cryptographic algorithm MD5_16
```

OSPF v3 模造リンク

次に、OSPFv3 模造リンクの設定例を示します。

```
router ospfv3 1
vrf vrf1
auto-cost reference-bandwidth 1000
router-id 1.1.1.1
redistribute bgp 100 route-policy vrf1_rpl
area 1
sham-link 1111::1111 3333::3333
cost 2
!
```

設定用の show コマンドは、次のようになります。

```
show ospfv3 vrf all-inclusive sham-links
Mon Dec 17 11:06:05.192 EST

Sham Links for OSPFv3 1, VRF vrf1

Sham Link OSPF_SL0 to address 3333::3333 is up
Area 1, source address 1111::1111
IfIndex = 3
Run as demand circuit
DoNotAge LSA allowed., Cost of using 2
Transmit Delay is 1 sec, State POINT_TO_POINT,
Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
Hello due in 00:00:08
Adjacency State FULL (Hello suppressed)
Number of DBD retrans during last exchange 0
Index 2/2, retransmission queue length 0, number of retransmission 1
First 0(0)/0(0) Next 0(0)/0(0)
Last retransmission scan length is 1, maximum is 1
Last retransmission scan time is 0 msec, maximum is 0 msec
```

次の作業

OSPF バージョン 2 の RPL からルートマップを設定するには、「 *Cisco ASR 9000* シリーズ ルータ でのルーティングポリシーの実装」のモジュールを参照してください。

MPLS TE トポロジを構築するには、トンネルを作成し、OSPF バージョン 2 のトンネルを介して転送を設定します。 *MPLS Configuration Guide for Cisco ASR 9000 Series RoutersMPLS Configuration Guide for Cisco NCS 560 Series Routers*を参照してください。

その他の参考資料

ここでは、OSPF の実装に関する関連資料について説明します。

関連資料

関連項目	マニュアル タイトル
OSPF コマンドと OSPFv3 のマンド：すべてのコマンドシンタックス、コマンドモード、コマンド履歴、デフォルト値、使用上の注意事項、例	*Routing Command Reference for Cisco ASR 9000 Series Routers*
MPLS TE 機能情報	*MPLS Configuration Guide for Cisco ASR 9000 Series RoutersMPLS Configuration Guide for Cisco NCS 560 Series Routers*の「*Implementing MPLS Traffic Engineering on* Cisco ASR 9000 シリーズ ルータ 」のモジュール
MIB リファレンス	『*Cisco ASR 9000 Series Aggregation Services Router MIB Specification Guide*』

標準

標準	タイトル
draft-ietf-ospf-multi-area-adj-07.txt	『OSPF Multi-Area Adjacency』
draft-ietf-pce-disco-proto-ospf-08.txt	『OSPF Protocol Extensions for Path Computation Element（PCE）』
draft-ietf-mpls-igp-sync-00.txt	『LDP IGP Synchronization』
draft-ietf-ospf-ospfv3-graceful-restart-07.txt	OSPFv3 グレースフル リスタート

MIB

MIB	MIB のリンク
—	選択したプラットフォーム、Cisco IOS リリース、およびフィーチャ セットに関する MIB を探してダウンロードするには、次の URL にある Cisco MIB Locator を使用します。 http://www.cisco.com/go/mibs

RFC

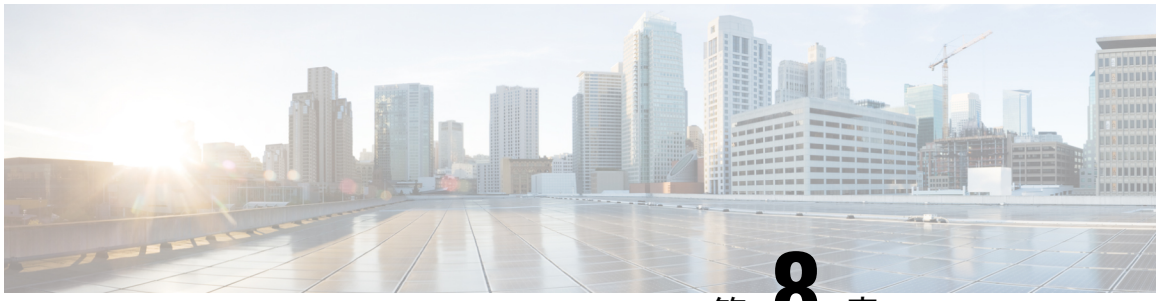
RFC	タイトル
RFC 1587	『The OSPF NSSA Option』
RFC 1793	『Extending OSPF to Support Demand Circuits』

RFC	タイトル
RFC 2328	『OSPF Version 2』
RFC 2370	『The OSPF Opaque LSA Option』
RFC 2740	『OSPF for IPv6』
RFC 3101	『The OSPF Not-So-Stubby Area (NSSA) Option』
RFC 3137	『OSPF Stub Router Advertisement』
RFC 3509	『Alternative Implementations of OSPF Area Border Routers』
RFC 3623	『Graceful OSPF Restart』
RFC 3630	OSPF バージョン 2 へのトラフィック エンジニアリング（TE）の拡張
RFC 3682	『The Generalized TTL Security Mechanism (GTSM)』
RFC 3906	『Calculating Interior Gateway Protocol (IGP) Routes Over Traffic Engineering Tunnels』
RFC 4136	『OSPF Refresh and Flooding Reduction in Stable Topologies』
RFC 4206	『Label Switched Paths (LSP) Hierarchy with Generalized Multi-Protocol Label Switching (GMPLS) Traffic Engineering (TE)』
RFC 4124	『Protocol Extensions for Support of Diffserv-aware MPLS Traffic Engineering』
RFC 4576	『Using a Link State Advertisement (LSA) Options Bit to Prevent Looping in BGP/MPLS IP Virtual Private Networks (VPNs) ownbit Extension for L3VPN』
RFC 4577	『OSPF as the Provider/Customer Edge Protocol for BGP/MPLS IP Virtual Private Networks (VPNs)』
RFC 4750	『OSPF Version 2 Management Information Base』
RFC 4811	『OSPF Out-of-Band Link State Database (LSDB) Resynchronization』

RFC	タイトル
RFC 4812	『OSPF Restart Signaling』
RFC 4813	『OSPF Link-Local Signaling』
RFC 4970	『Extensions to OSPF for Advertising Optional Router Capabilities』
RFC 5643	OSPFv3 の管理情報ベース（MIB）

シスコのテクニカル サポート

説明	リンク
シスコのテクニカル サポート Web サイトでは、製品、テクノロジー、ソリューション、技術的なヒント、およびツールへのリンクなどの、数千ページに及ぶ技術情報が検索可能です。Cisco.com に登録済みのユーザは、このページから詳細情報にアクセスできます。	http://www.cisco.com/techsupport



第 8 章

IP Fast Reroute ループフリー代替の実装

IP Fast Reroute ループフリー代替機能により、障害のあるリンクを含むパケットを、リモートループフリー代替（数ホップ離れている）までトンネリングすることができます。

IPv4/IPv6 ループフリー代替高速再ルーティングのための前提条件

- ループフリー代替（LFA）高速再ルーティング（FRR）は、インターフェイスがポイントツーポイントインターフェイスである場合だけ、インターフェイスを介して到達可能なパスを保護できます。
- LAN インターフェイスが 1 つのネイバーに物理的に接続されている場合、LFA FRR で保護するために、LAN インターフェイスをポイントツーポイント インターフェイスとして設定する必要があります。

ループフリー代替高速再ルーティングの制約事項

- ロード バランス サポートは、FRR で保護されたプレフィックスで利用可能ですが、50 ミリ秒のカットオーバーの時間は保証されません。

- 最大 8 個の FRR 保護のインターフェイスで同時にカットオーバーを実行することができます。
- レイヤ 3 VPN だけがサポートされます。
- MPLS トラフィックのリモート LFA バックアップパスは、LDP を使用してのみ設定できます。
- LFA 計算は、同じレベルまたは領域に属するインターフェイスまたはリンクに制限されます。したがって、バックアップ LFA の計算時に同じ LAN 上のすべてのネイバーを除外すると、トポロジのサブセットで修復を使用できなくなる可能性があります。
- 物理インターフェイスおよび物理ポートチャネルインターフェイスのみ保護されます。サブインターフェイス、トンネル、および仮想インターフェイスは保護されません。
- ボーダー ゲートウェイ プロトコル（BGP）プレフィックス独立コンバージェンス（PIC）と IP FRR は、同じプレフィックスに使用されない限り、同じインターフェイス上に設定できます。
- TE トンネルを介した IPv6 LFA FRR はサポートされていません。

IS-IS および IP FRR

ローカル リンクがネットワークで失敗した場合、IS-IS は、影響を受けるすべてのプレフィックスの新しいプライマリ ネクストホップルートを再計算します。これらのプレフィックスは、RIBおよび転送情報ベース（FIB）で更新されます。プライマリ プレフィックスがフォワーディング プレーンで更新されるまで、影響を受けるプレフィックス宛てのトラフィックは廃棄されます。このプロセスには数百ミリ秒かかることがあります。

IP FRR で、IS-IS はプライマリ パスで障害が発生した場合に使用するために、フォワーディング プレーンに対する LFA ネクストホップ ルートを計算します。LFA はプレフィックスごとに計算されます。

特定のプライマリ パスに複数の LFA がある場合、IS-IS はプライマリ パスの単一 LFA を選ぶために、タイブレーク ルールを使用します。複数 LFA パスを持つプライマリ パスの場合、プレフィックスは LFA パス間で均等に分散されます。

修復パス

修復パスでは、ルーティングの遷移時にトラフィックが転送されます。リンクまたはルータに障害が発生すると、物理層の信号が失われるため、当初は隣接ルータしかこの障害を認識できません。ネットワーク内のその他すべてのルータは、この障害に関する情報がルーティング プロトコルによって伝播されるまで（これには数百ミリ秒かかる可能性があります）、この障害の性質と場所を認識しません。したがって、このネットワーク障害の影響を受けたパケットがそれぞれの宛先に到達するように準備する必要があります。

障害が発生したリンクに隣接するルータは、障害が発生したリンクを使用していた可能性のあるパケットに対して、一連の修復パスを使用します。これらの修復パスは、ルータが障害を検出してから、ルーティングの遷移が完了するまで使用されます。ルーティングの遷移が完了するまでに、ネットワーク内のすべてのルータは転送データを変更し、障害が発生したリンクはルーティングの計算から除外されます。

修復パスは、障害が検出されるとすぐにアクティブになるようにするために、障害を予測して事前計算されます。

LFA FRR 機能では次の修復パスを使用します。

- 等コストマルチパス（ECMP）は、宛先の等コストパス分割セットのメンバーとしてリンクを使用します。セットの他のメンバーは、リンクに障害が発生したときに代替パスを提供できます。
- LFA は、ループバックしないで宛先にパケットを送るネクストホップ ルートです。ダウンストリーム パスは LFA のサブセットです。

LFA の概要

LFA はプライマリ ネイバー以外のノードです。トラフィックは、ネットワーク障害発生後に LFA にリダイレクトされます。LFA は、失敗について認識せずに転送を決定します。

LFA は、トラフィックの転送に障害のある要素を使用したり、保護ノードを使用することはできません。LFA はループを発生させてはなりません。LFA は、インターフェイスがプライマリ パスとして使用できる限り、デフォルトでサポートされるすべてのインターフェイスでイネーブルになります。

プレフィックスごとの LFA を使用する利点は次のとおりです。

- プライマリ パスでリンクがダウンした場合、修復パスが移行中にトラフィックを転送します。
- プレフィックスごとの LFA を持つすべての宛先が保護されます。これにより、サブセット（障害の遠端のノード）のみが保護されない状態で残ります。

LFA の計算

プレフィックスごとに LFA を計算する汎用アルゴリズムについては、RFC 5286 を参照してください。IS-IS は、メモリ使用量を減らすための少量の変更とともに RFC 5286 を実装します。保護のプレフィックスを検証する前にすべてのネイバーの最短パス優先（SPF）計算を実行する代わりに、IS-IS は SPF 計算がネイバーごとに実行された後でプレフィックスを検査します。IS-IS は SPF 計算の実行後にプレフィックスを検査するため、IS-IS は SPF 計算がネイバーごとに実行された後も最適な修復パスを保持します。IS-IS では、すべてのネイバーに対する SPF の結果を保存する必要はありません。

RIB とルーティング プロトコル間の連携

ルーティング プロトコルは、タイブレーク アルゴリズムを実装して、プレフィックスの修復パスを計算します。計算の結果は、プライマリ パス付きの一連のプレフィックスになり、いくつかのプライマリ パスが修復パスに関連付けられます。

タイブレーク アルゴリズムは特定の条件を満たすか、または特定の属性を持つ LFA を考慮します。複数の LFA がある場合は、 **tie-break** キーワードを使用して **fast-reroute per-prefix** コマンドを設定します。ルールによってすべての候補LFAが除外される場合、そのルールはスキップされます。

プライマリ パスには、複数の LFA を設定できます。デフォルトのタイブレーク ルールを実装し、ユーザがこれらのルールを変更できるようにするには、ルーティングプロトコルが必要です。タイブレーク アルゴリズムの目的は、複数の候補 LFA を除外し、プレフィックス単位のプライマリ パスごとに 1 つの LFA を選択し、プライマリ パスが失敗したときに複数の候補 LFA でトラフィックを分散させることです。

タイブレーク ルールでは、すべての候補を除外することはできません。

タイブレークには、次の属性が使用されます。

- ダウンストリーム：保護された宛先へのメトリックが宛先へのノードを保護しているメトリックよりも低い候補を除外します。
- ラインカード分離：保護されたパスと同じラインカードを共有している候補を除外します。
- 共有リスク リンク グループ（SRLG）：保護されたパス SRLG のいずれかに属する候補を除外します。
- 負荷分散：保護されたパスを共有するプレフィックスで残りの候補を分散させます。
- 最低修復パス メトリック：保護されたプレフィックスへのメトリックが高い候補を除外します。
- ノードの保護：保護されたノードではない候補を除外します。
- プライマリ パス：ECMP ではない候補を除外します。
- セカンダリ パス：ECMP の候補を除外します。

高速再ルーティングのサポートの設定

（注） LFA 計算はすべてのルータに対して有効になり、FRR はサポートされているすべてのインターフェイスに有効になります。

手順の概要

1. **configure**
2. **router isis** *process-id*
3. **is-type**{ **level-1** | **level-1-2** | **level-2-only** }
4. **net** *net*
5. **address-family** {**ipv4** | **ipv6**} [**unicast** | **multicast**]
6. **metric-style wide**
7. **exit**
8. **interface** *bundle bundle-id*
9. **address-family** {**ipv4** | **ipv6**} [**unicast** | **multicast**]
10. **fast-reroute per-prefix**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure** 例： `RP/0/RP0/CPU0:router# configure`	グローバル コンフィギュレーション モードを開始します。
ステップ 2	**router isis** *process-id* 例： `RP/0/RP0/CPU0:router(config-if)# router isis core`	指定したルーティング インスタンスの IS-IS ルーティングを有効にし、ルータをルータ コンフィギュレーション モードにします。デフォルトでは、すべての IS-IS インスタンスが自動的にレベル 1 とレベル 2 になります。is-type ルータ コンフィギュレーション コマンドを使用して、特定のルーティング インスタンス別にこのレベルを変更できます。
ステップ 3	**is-type**{ **level-1**	**level-1-2**	**level-2-only** } 例： `RP/0/RP0/CPU0:router(config-isis)#is-type level-2-only`	（任意）システムタイプ（エリアまたはバックボーン ルータ）を設定します。 デフォルトでは、すべての IS-IS インスタンスは level-1-2 ルータとして動作します。 • level-1 キーワードは、レベル 1（エリア内）ルーティングのみを実行するようにソフトウェアを設定します。レベル 1 の隣接関係のみが確立されます。ソフトウェアは、そのエリア内の宛先のみを検出します。エリア外の宛先がある場合にそれらを含むパケットが、エリア内の直近の level-1-2 ルータに送信されます。 • level-2-only キーワードは、レベル 2（バックボーン）ルーティングのみを実行するようにソフトウェアを設定します。ルータはレベル 2 の隣接関係のみを確立します。これは、他の

	コマンドまたはアクション	目的
		level-2-only ルータまたは level-1-2 ルータのいずれかで確立されます。 • level-1-2 キーワードは、レベル 1 とレベル 2 の両方のルーティングを実行するようにソフトウェアを設定します。レベル 1 とレベル 2 の両方の隣接関係が確立されます。ルータはレベル 2 バックボーンとレベル 1 エリアの間の境界ルータとして動作します。
ステップ 4	**net** *net* 例： `RP/0/RP0/CPU0:router(config-isis)# net 47.0001.0000.0000.8888.00`	ルーティングプロセスの IS-IS Network Entity（NET）を設定します。
ステップ 5	**address-family** {**ipv4 | ipv6**} [**unicast | multicast**] 例： `RP/0/RP0/CPU0:router(config-isis)# address-family ipv4 unicast`	IPv4 または IPv6 アドレスファミリを指定して、インターフェイス アドレス ファミリ コンフィギュレーション モードを開始します。
ステップ 6	**metric-style wide** 例： `RP/0/RP0/CPU0:router(config-isis-af)# metric-style wide`	ワイドリンクメトリックのみを生成して受け入れるようにルータを設定します。
ステップ 7	**exit** 例： `RP/0/RP0/CPU0:router(config-isis-af)# exit`	ルータ アドレス ファミリ コンフィギュレーション モードを終了し、ルータをルータ コンフィギュレーション モードにリセットします。
ステップ 8	**interface** *bundle bundle-id* 例： `RP/0/RP0/CPU0:router(config-isis)# interface Bundle-Ether 9`	新しいイーサネット リンク バンドルを作成し名前を付与します。
ステップ 9	**address-family** {**ipv4 | ipv6**} [**unicast | multicast**] 例： `RP/0/RP0/CPU0:router(config-isis-if)# address-family ipv4 unicast`	IPv4 または IPv6 アドレスファミリを指定して、インターフェイス アドレス ファミリ コンフィギュレーション モードを開始します。
ステップ 10	**fast-reroute per-prefix** 例： `RP/0/RP0/CPU0:router(config-isis-if-af)# fast-reroute per-prefix`	プレフィックス単位の FRR を有効にします。

IPv4 ループフリー代替高速再ルーティングのサポートの設定：例

次に、IPv4 LFA FRR を設定する例を示します。

```
router isis core
 is-type level-2-only
 net 47.0001.0000.0000.8888.00
 address-family ipv4 unicast
  metric-style wide
  exit
 !
 interface Bundle-Ether 9
  point-to-point
  address-family ipv4 unicast
   fast-reroute per-prefix
  !
 !
```

その他の参考資料

以降の項では、IPv4/IPv6 ループフリー代替高速再ルーティングの実装に関連する参考資料について説明します。

関連資料

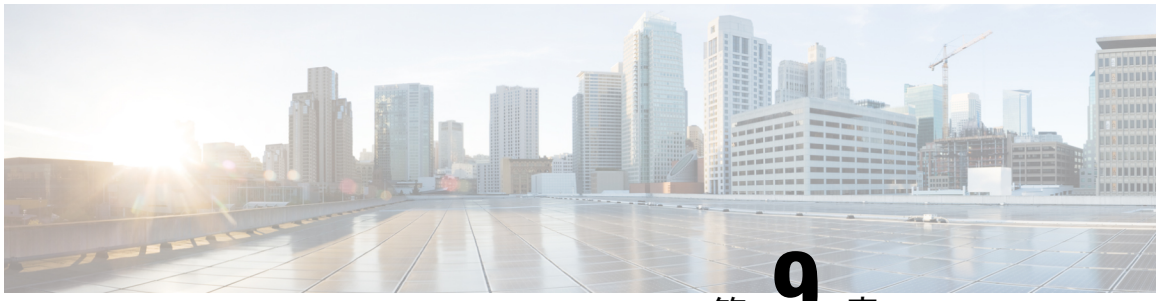
関連項目	マニュアル タイトル
IS-IS コマンド	*Routing Command Reference for Cisco ASR 9000 Series Routers*
MPLS コマンド	*Routing Command Reference for Cisco ASR 9000 Series Routers*

MIB

MIB	MIB のリンク
—	Cisco IOS XR ソフトウェアを使用して MIB の場所を特定してダウンロードするには、次の URL にある Cisco MIB Locator を使用して、[Cisco Access Products] メニューからプラットフォームを選択します。 https://mibs.cloudapps.cisco.com/ITDIT/MIBS/servlet/index

シスコのテクニカル サポート

説明	リンク
シスコのテクニカル サポート Web サイトでは、製品、テクノロジー、ソリューション、技術的なヒント、およびツールへのリンクなどの、数千ページに及ぶ技術情報が検索可能です。Cisco.com に登録済みのユーザは、このページから詳細情報にアクセスできます。	http://www.cisco.com/techsupport



第 9 章

RIB の実装とモニタリング

ルーティング情報ベース（RIB）は、ネットワークのすべてのノード間のルーティングの接続に関する情報を収集して配布したものです。各ルータには、そのルータのルーティング情報を含む RIB を維持します。RIB は、システムで実行されているすべてのルーティング プロトコルでの最良ルートを保存します。

このモジュールでは、Cisco IOS XR ネットワークで RIB を実装およびモニタリングする方法を説明します。

（注） Cisco IOS XR ソフトウェアの RIB に関する情報と、このモジュールに一覧で示されている RIB コマンドに関しては、このモジュールの「その他の参考資料」の項を参照してください。

設定作業を実行中に表示されることのある他のコマンドのドキュメントを検索するには、オンラインで *Cisco ASR 9000 Series Aggregation Services Router Commands Master List* を検索してください。

RIB の実装とモニタリングの機能履歴

リリース	変更内容
リリース 3.7.2	この機能が導入されました。
リリース 4.2.0	次の機能が追加されました。 • ルートとラベルの整合性チェッカ（RCC および LCC）
リリース 4.2.1	RIB および FIB の BGP Prefix Independent Convergence のサポートが追加されました。
リリース 4.3.0	アップデート生成のための BGP-RIB のフィードバック メカニズム機能が追加されました。

RIB の実装の前提条件

- 適切なタスク ID を含むタスク グループに関連付けられているユーザ グループに属している必要があります。このコマンド リファレンスには、各コマンドに必要なタスク ID が含まれます。ユーザ グループの割り当てが原因でコマンドを使用できないと考えられる場合、AAA 管理者に連絡してください。
- RIB はベースのCisco IOS XR ソフトウェアによって分散されます。インストールの特別な要件はありません。ベース ソフトウェア インストールの要件は次のとおりです。
  - ルータ
  - Cisco IOS XR ソフトウェア
  - ベース パッケージ

RIB の設定情報

Cisco RIB 機能を実装するには、次の概念を理解しておく必要があります。

RIB の概要

各ルーティング プロトコルは独自の最適ルートのセットを選択し、これらのルートとその属性を RIB に取り込みます。RIB はこれらのルートを格納し、すべてのルーティング プロトコルの中から最適ルートを選択します。これらのルートは転送パケットで使用するために、ラインカードにダウンロードされます。頭字語の RIB は、RIB プロセスと、RIB 内に含まれるルートデータの集合を表すために使用されます。

プロトコル内で、ルートはそのプロトコルによって使用されているメトリックに基づいて選択されます。プロトコルは最適なルート（最も低いメトリックまたは結び付けられたメトリック）を RIB にダウンロードします。RIB は、関連付けられているプロトコルのアドミニストレーティブ ディスタンスを比較して、全体的に最適なルートを選択します。

BGP およびその他のプロトコルでの RIB データ構造

RIB はプロセスを使用し、ボーダー ゲートウェイ プロトコル（BGP）などの他のルーティング アプリケーション、および他のユニキャスト ルーティング プロトコル、または Protocol Independent Multicast（PIM）、Multicast Source Discovery Protocol（MSDP）などのマルチキャスト プロトコルとは異なるデータ構造を維持します。ただし、これらのルーティング プロトコルは、RIB が使用するものと似た内部データ構造を使用し、RIB としてそのデータ構造を内部的に参照することがあります。たとえば、BGP ルートは BGP RIB（BRIB）に保存され、PIM および MSDP などのマルチキャスト ルーティング プロトコルによって計算されたマルチキャスト ルートはマルチキャスト RIB（MRIB）に保存されます。RIB プロセスは BRIB および MRIB に対処しません。これらは、BGP によって およびマルチキャスト プロセスによってそれぞれ処理されます。

パケットを転送するためにラインカードおよび RP によって使用されるテーブルは、転送情報ベース（FIB）と呼ばれます。RIB プロセスは FIB を構築しません。代わりに、RIB はバルク コンテンツ ダウンローダ（BCDL）プロセスによって、FIBプロセスに最適な、選択されたルートのセットをバルク各ラインカードにダウンロードします。続いて、FIB が構築されます。

RIB アドミニストレーティブ ディスタンス

転送は最長プレフィックス照合に基づいて行われます。10.0.2.1 宛てのパケットを転送する場合、マスク /24 は /16 よりも長い（より具体的である）ため、10.0.2.0/24 は 10.0.0.0/16 よりも優先されます。

同じプレフィックスと同じ長さを持つ、異なるプロトコルからのルートは、アドミニストレーティブ ディスタンスに基づいて選択されます。たとえば、Open Shortest Path First（OSPF）プロトコルのアドミニストレーティブ ディスタンスは 110、Intermediate System-to-Intermediate System（IS-IS）プロトコルのアドミニストレーティブ ディスタンスは 115 です。IS-IS および OSPF の両方が RIB に 10.0.1.0/24 をダウンロードすると、OSPF のアドミニストレーティブ ディスタンスの方が小さいため、RIB は OSPF ルートを優先します。同じ長さの複数のルート間で選択するためだけにアドミニストレーティブ ディスタンスが使用されます。

次の表に、一般的なプロトコルのデフォルトのアドミニストレーティブディスタンスを示します。

表 *5:* デフォルトのアドミニストレーティブ ディスタンス

プロトコル	アドミニストレーティブディスタンスのデフォルト
接続されているルートまたはローカル ルート	0
スタティック ルート	1
外部 BGP ルート	20
OSPF ルート	110

プロトコル	アドミニストレーティブディスタンスのデフォルト
IS-IS ルート	115
内部 BGP ルート	200

一部のルーティング プロトコル（たとえば、IS-IS、OSPF、BGP など）のアドミニストレーティブ ディスタンスは変更できます。プロトコルのアドミニストレーティブ ディスタンスを変更する適切な方法については、そのプロトコル固有のマニュアルを参照してください。

（注） すべてではなく一部のルータで、プロトコルのアドミニストレーティブディスタンスを変更すると、ルーティングループなどの予想外の動作が発生することがあります。したがって、これは推奨されません。

IPv4 および IPv6 の RIB サポート

Cisco IOS XR ソフトウェアでは、RIB テーブルはマルチキャスト ルーティングおよびユニキャストルーティングをサポートしています。

Cisco IOS XR ソフトウェア RIB のデフォルトのルーティングテーブルは、IPv4 ルーティングのユニキャスト RIB テーブルおよび IPv6 ルーティングのマルチキャスト/ユニキャスト RIB テーブルです。マルチキャスト ルーティングの場合、ルーティング プロトコルは、マルチキャスト ユニキャスト RIB テーブルにユニキャスト ルートを挿入します。マルチキャスト プロトコルは、その情報を使用してマルチキャスト ルートを構築します（次にそのマルチキャスト ルートが MRIB に保存されます）。マルチキャストの使用と設定に関する詳細については、マルチキャストのマニュアルを参照してください。

RIB プロセスの ipv4_rib および ipv6_rib は、RP カードで実行されます。プロセス配置機能がルータの複数の RP で使用可能でサポートされている場合、RIB プロセスは任意の使用可能なノードに配置できます。

RIB 統計情報

RIB は、RIB とクライアントとの間でやり取りされるメッセージ（要求）の統計情報をサポートします。プロトコル クライアントは、メッセージを RIB に送信します（たとえば、ルート追加、ルート削除、ネクスト ホップの登録など）。RIB もメッセージを送信します（たとえば、ルート、アドバタイズメント、ネクスト ホップ通知などの再配布）。これらの統計情報は、どのようなメッセージが送信されたかに関して、また送信されたメッセージ数に関する情報を収集するために使用されます。これらの統計情報には、RIB サーバとそのクライアント間で転送される各種メッセージのカウンタが含まれています。統計情報は、show rib statistics コマンドを使用して表示します。

RIB は、次に挙げるような、クライアントから送信されるすべての要求のカウンタを保持します。

- ルートの動作
- テーブルの登録
- ネクストホップの登録
- 再配布の登録
- 属性の登録
- 同期の完了

RIB は、RIB によって送信されるすべての要求のカウンタも保持します。設定は RIB ネクスト ホップ ダンプニング機能をディセーブルにします。この結果、クライアントが登録したネクスト ホップが解決された、または解決されなかった場合に RIB はクライアントにすぐに通知します。

RIB は、要求の結果に関する情報も保持します。

IPv6 プロバイダーエッジ IPv6 および MPLS を介する IPv6 VPN プロバイダーエッジ転送

IPv6 プロバイダーエッジ（6PE）および IPv6 VPN プロバイダーエッジ（6VPE）では、IPv6 転送に既存のマルチプロトコル ラベル スイッチング（MPLS）の IPv4 コア インフラストラクチャを活用します。6PE および 6VPE を使うと、MPLS ラベル スイッチド パス（LSP）を使用して MPLS IPv4 コア ネットワークを介して IPv6 サイトが相互に通信できるようになります。

RIB は、6VPE ネクスト ホップを提供することにより、6PE および 6VPE をサポートしています。ネクストホップ情報は、RIB の隠されたデータベースに格納されています。これには、プロトコル クライアントによって転送情報ベース（FIB）に送信されるデータが読み込まれます。

MPLS を介する 6PE および 6VPE の設定については、 *MPLS Configuration Guide for Cisco ASR 9000 Series RoutersMPLS Configuration Guide for Cisco NCS 560 Series Routers*を参照してください。

RIB 隔離

RIB 検疫は、ルーティング プロトコルと RIB 間の相互作用における問題を解決します。問題は、ルートが継続的に挿入され、RIB から取り消される場合に発生する、RIB とルーティング プロトコルの間の持続振動です。問題が解決されるまで、CPU使用率にスパイクが生じます。振動に減衰がない場合、プロトコル プロセスおよび RIB プロセスが CPU を多く使用するため、システムのその他の部分に影響を与え、さらにプロトコルおよびRIBのその他の動作の障害となります。この問題は、RIBにルートの特定の組み合わせが受信されて取り込まれた場合に発生します。この問題は、通常、ネットワークの設定が間違っている場合に発生します。ただし、設定ミスはネットワーク全体であるため、単一のルータの設定時に問題を検出できません。

隔離メカニズムでは相互に再帰的なルートが検出されますが、ここで隔離されるのは相互の再帰が完了した最終ルートです。検疫ルートは、相互の再帰が解消したか確認するために定期的に評価されます。再帰が引き続き存在する場合は、ルートは検疫対象のままとなります。再帰が解消した場合は、ルートは検疫対象から外れます。

次の手順を使用して、ルートを隔離します。

1. RIB は問題がある特定のパスがインストールされている場合に検出します。
2. RIB は、そのパスを取り込んだプロトコルに通知を送信します。
3. プロトコルは問題のルートに関する隔離通知を受信すると、そのルートを「隔離中」とマークします。これが BGP ルートである場合、BGP はそのネイバーにルートへの到達可能性をアドバタイズしません。
4. RIB は、すべての検疫対象パスに対して、安全に取り込む（検疫対象から「使用 OK」状態に移行）ことができるようになったかどうかを定期的にテストします。パスが安全に使用できるようになったことを示す通知がプロトコルに送信されます。

ルートとラベルの整合性チェッカ

ルート整合性チェッカおよびラベル整合性チェッカ（RCC/LCC）はコマンドライン ツールです。これは、コントロール プレーンとデータ プレーン ルート間および IOS XR ソフトウェアのラベル プログラミングの整合性を検証するために使用できます。

運用中ネットワークのルータは、転送情報がコントロール プレーン情報と一致しない状態になった可能性があります。この原因は、ルート プロセッサ（RP）とラインカード（LC）間でのファブリック障害または転送障害、または転送情報ベース（FIB）に関する問題である可能性があります。RCC/LCC を使用すると、結果として生じたコントロール プレーンとデータ プレーン間の不整合を識別して詳細情報を出力できます。この情報は、転送問題とトラフィック損失の原因をさらに調査して診断するために使用できます。

RCC/LCC は、2 つのモードで実行できます。RCC/LCC は、EXEC モードからオンデマンドの 1 回かぎりのスキャンとしてトリガーする（オンデマンドスキャン）か、通常のルータ動作中にバックグラウンドで定義した間隔で実行するように設定（バックグラウンドスキャン）できます。RCC は、ルーティング情報ベース（RIB）を転送情報ベース（FIB）と比較します。一方、LCC は、ラベル スイッチング データベース（LSD）を FIB と比較します。不整合が検出されると、RCC/LCC 出力では、特定のルートまたはラベルを識別し、検出された不整合のタイプを識別して、さらなるトラブルシューティングに役立つ追加のデータも提供します。

RCC はルート プロセッサで動作します。FIB は、ライン カード上のエラーについてチェックし、最初の 20 のエラー レポートを RCC に送信します。RCC はすべてのノードからエラー レポートを受信し、それらを要約し（完全一致についてチェックし）、2 つのキュー（ソフトまたはハード）に追加します。各キューのエラー レポート数の制限は 1000 で、キューに優先度はありません。RCC/LCC は、異なるノードからの同じエラー（完全一致）を 1 つのエラーとして記録します。RCC/LCC は、エラーのプレフィックス/ラベル、バージョン番号、タイプなどに基づいてエラーを比較します。

オンデマンド スキャン

オンデマンドスキャンでは、ユーザは、特定のテーブルの特定のプレフィックスまたはテーブル内のすべてのプレフィックス関するコマンドラインインターフェイス全体のスキャンを要求します。スキャンはただちに実行され、結果がすぐに発行されます。LCC は LSD でオンデマンド スキャンを実行するのに対し、RCC は VRF 単位で実行します。

バックグラウンド スキャン

バックグラウンドスキャンでは、ユーザはバックグラウンドで実行されるスキャンを設定します。設定は、定期的なスキャンの間隔で構成されます。このスキャンは、単一または複数のテーブルに設定できます。LCC は LSD でバックグラウンド スキャンを実行するのに対し、RCC はデフォルトの VRF または他の VRF に対し実行します。

RIB の導入およびモニタリング方法

RIB を導入およびモニタするには、次の概念を理解しておく必要があります。

ルーティングテーブルを使用した RIB 設定の検証

ルーティングテーブルの概要と詳細の情報をチェックすることで、RIB が RP 上で実行され、正常に機能していることを確認するために、RIBの設定を確認するには、次のタスクを実行します。

手順の概要

1. **show route** [**vrf** { *vrf-name* | **all** }] [**afi-all** | **ipv4** | **ipv6**] [**unicast** | **multicast** | **safi-all**] **summary** [**detail**] [**standby**]
2. **show route** [**vrf** { *vrf-name* | **all** }] [**afi-all** | **ipv4** | **ipv6**] [**unicast** | **multicast** | **safi-all**] [*protocol* [*instance*] | *ip-address mask*] [**standby**] [**detail**]

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**show route** [**vrf** { *vrf-name*	**all** }] [**afi-all**	**ipv4**	**ipv6**] [**unicast**	**multicast**	**safi-all**] **summary** [**detail**] [**standby**] 例： `RP/0/RSP0/cpu 0: router# show route summary`	指定したルーティング テーブルに関するルート サマリー情報を表示します。 • 要約されたデフォルト テーブルは、IPv4 ユニキャスト ルーティング テーブルです。
ステップ 2	**show route** [**vrf** { *vrf-name*	**all** }] [**afi-all**	**ipv4**	**ipv6**] [**unicast**	**multicast**	**safi-all**] [*protocol* [*instance*]	*ip-address mask*] [**standby**] [**detail**] 例：	指定したルーティングテーブルに関する詳細なルート情報を表示します。 • このコマンドは、表示を制限するために通常は IPアドレスまたは他のオプションフィルタを使

	コマンドまたはアクション	目的
	`RP/0/RSP0/cpu 0: router# show route ipv4 unicast`	用して発行します。それ以外の場合は、デフォルトの IPv4 ユニキャスト ルーティング テーブルからすべてのルートを表示します。ネットワークの設定に応じて大規模なリストになる可能性があります。

ネットワーキングとルーティングの問題の検証

ノード間のルートの動作を検証するには、次のタスクを実行します。

手順の概要

1. **show route** [**vrf** { *vrf-name* | **all** }] [**afi-all** | **ipv4** | **ipv6**] [**unicast** | **multicast** | **safi-all**] [*protocol* [*instance*] | *ip-address mask*] [**standby**] [**detail**]
2. **show route** [**vrf** { *vrf-name* | **all** }] [**afi-all** | **ipv4** | **ipv6**] [**unicast** | **multicast** | **safi-all**] **backup** [*ip-address*] [**standby**]
3. **show route** [**vrf** { *vrf-name* | **all** }] [**ipv4** | **ipv6**] [**unicast** | **multicast** | **safi-all**] **best-local** *ip-address* [**standby**]
4. **show route** [**vrf** { *vrf-name* | **all** }] [**afi-all** | **ipv4** | **ipv6**] [**unicast** | **multicast** | **safi-all**] **connected** [**standby**]
5. **show route** [**vrf** { *vrf-name* | **all** }] [**afi-all** | **ipv4** | **ipv6**] [**unicast** | **multicast** | **safi-all**] **local** [*interface*] [**standby**]
6. **show route** [**vrf** { *vrf-name* | **all** }] [**ipv4** | **ipv6**] [**unicast** | **multicast** | **safi-all**] **longer-prefixes** { *ip-address mask* | *ip-address* / *prefix-length* } [**standby**]
7. **show route** [**vrf** { *vrf-name* | **all** }] [**ipv4** | **ipv6**] [**unicast** | **multicast** | **safi-all**] **next-hop** *ip-address* [**standby**]

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**show route** [**vrf** { *vrf-name*	**all** }] [**afi-all**	**ipv4**	**ipv6**] [**unicast**	**multicast**	**safi-all**] [*protocol* [*instance*]	*ip-address mask*] [**standby**] [**detail**] 例： `RP/0/RSP0/cpu 0: router# show route ipv4 unicast 192.168.1.11/8`	RIB の現在のルートを表示します。
ステップ 2	**show route** [**vrf** { *vrf-name*	**all** }] [**afi-all**	**ipv4**	**ipv6**] [**unicast**	**multicast**	**safi-all**] **backup** [*ip-address*] [**standby**] 例： `RP/0/RSP0/cpu 0: router# show route ipv4 unicast backup 192.168.1.11/8`	RIB のバックアップ ルートを表示します。

	コマンドまたはアクション	目的
ステップ 3	**show route** [**vrf** { *vrf-name* | **all** }] [**ipv4** | **ipv6**] [**unicast** | **multicast** | **safi-all**] **best-local** *ip-address* [**standby**] 例： `RP/0/RSP0/cpu 0: router# show route ipv4 unicast best-local 192.168.1.11/8`	特定の宛先からの応答パケットに使用する場合に最善のローカル アドレスが表示されます。
ステップ 4	**show route** [**vrf** { *vrf-name* | **all** }] [**afi-all** | **ipv4** | **ipv6**] [**unicast** | **multicast** | **safi-all**] **connected** [**standby**] 例： `RP/0/RSP0/cpu 0: router# show route ipv4 unicast connected`	ルーティングテーブルの現在の接続ルートを表示します。
ステップ 5	**show route** [**vrf** { *vrf-name* | **all** }] [**afi-all** | **ipv4** | **ipv6**] [**unicast** | **multicast** | **safi-all**] **local** [*interface*] [**standby**] 例： `RP/0/RSP0/cpu 0: router# show route ipv4 unicast local`	ルーティングテーブルの受信エントリのローカル ルートを表示します。
ステップ 6	**show route** [**vrf** { *vrf-name* | **all** }] [**ipv4** | **ipv6**] [**unicast** | **multicast** | **safi-all**] **longer-prefixes** { *ip-address mask* | *ip-address* / *prefix-length* } [**standby**] 例： `RP/0/RSP0/cpu 0: router# show route ipv4 unicast longer-prefixes 192.168.1.11/8`	指定のネットワークと指定の数のビットを共有する RIB の現在のルートを表示します。
ステップ 7	**show route** [**vrf** { *vrf-name* | **all** }] [**ipv4** | **ipv6**] [**unicast** | **multicast** | **safi-all**] **next-hop** *ip-address* [**standby**] 例： `RP/0/RSP0/cpu 0: router# show route ipv4 unicast next-hop 192.168.1.34`	宛先アドレスまでのネクスト ホップ ゲートウェイまたはホストを表示します。

RIB ネクストホップ ダンプニングの無効化

RIB ネクストホップ ダンプニングを無効にするには、次のタスクを実行します。

手順の概要

1. **router rib**
2. **address-family** { **ipv4** | **ipv6** } **next-hop dampening disable**
3. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**router rib** 例： `RP/0/RSP0/cpu 0: router# route rib`	RIB コンフィギュレーション モードを開始します。
ステップ 2	**address-family** { **ipv4**	**ipv6** } **next-hop dampening disable** 例： `RP/0/RSP0/cpu 0: router(config-rib)# address family ipv4 next-hop dampening disable`	IPv4 アドレス ファミリのネクストホップ ダンプニングを無効にします。
ステップ 3	**commit**	

RCC および LCC の設定

RCC および LCC オンデマンド スキャンの有効化

ルート整合性チェッカ（RCC）、およびラベル整合性チェッカ（LCC）オンデマンドスキャンをトリガーするには、次の作業を実行します。オンデマンドスキャンは、特定のアドレス ファミリ（AFI）で、サブアドレス ファミリ（SAFI）、テーブル、および、プレフィックス、VRF、またはテーブルのすべてのプレフィックスに関して実行できます。

手順の概要

1. 次のいずれかのコマンドを使用します。
   - **show rcc** {**ipv4** | **ipv6**} **unicast** [**all**] [*prefix*/*mask*] [**vrf** *vrf-name*]
   - **show lcc** {**ipv4** | **ipv6**} **unicast** [**all**] [*prefix*/*mask*] [**vrf** *vrf-name*]
2. 次のいずれかのコマンドを使用します。
   - **clear rcc** {**ipv4** | **ipv6**} **unicast** [**all**] [*prefix*/*mask*] [**vrf** *vrf-name*] **log**
   - **clear lcc** {**ipv4** | **ipv6**} **unicast** [**all**] [*prefix*/*mask*] [**vrf** *vrf-name*] **log**

手順の詳細

	コマンドまたはアクション	目的

ステップ 1	次のいずれかのコマンドを使用します。
	• **show rcc** {**ipv4** | **ipv6**} **unicast** [**all**] [*prefix*/*mask*] [**vrf** *vrf-name*]
	• **show lcc** {**ipv4** | **ipv6**} **unicast** [**all**] [*prefix*/*mask*] [**vrf** *vrf-name*]
	例：
	`RP/0/RSP0/cpu 0: router#show rcc ipv6 unicast 2001:DB8::/32 vrf vrf_1`
	または
	`RP/0/RSP0/cpu 0: router#show lcc ipv6 unicast 2001:DB8::/32 vrf vrf_1`
		ルート整合性チェッカ（RCC）またはラベル整合性チェッカ（LCC）オンデマンドで実行します。

ステップ 2	次のいずれかのコマンドを使用します。
	• **clear rcc** {**ipv4** | **ipv6**} **unicast** [**all**] [*prefix*/*mask*] [**vrf** *vrf-name*] **log**
	• **clear lcc** {**ipv4** | **ipv6**} **unicast** [**all**] [*prefix*/*mask*] [**vrf** *vrf-name*] **log**
	例：
	`RP/0/RSP0/cpu 0: router#clear rcc ipv6 unicast log`
	または
	`RP/0/RSP0/cpu 0: router#show lcc ipv6 unicast log`
		以前のスキャンのログをクリアします。

RCC および LCC バックグラウンド スキャンの有効化

ルート整合性チェッカ（RCC）およびラベル整合性チェッカ（LCC）のバックグラウンド スキャンを実行するには、次のタスクを実行します。

手順の概要

1. **configure**
2. 次のいずれかのコマンドを使用します。
   • **rcc** {**ipv4** | **ipv6**} **unicast** {**enable** | **period** *milliseconds*}
   • **lcc** {**ipv4** | **ipv6**} **unicast** {**enable** | **period** *milliseconds*}
3. **commit**
4. 次のいずれかのコマンドを使用します。

- **show rcc** {**ipv4**| **ipv6**} **unicast** [**summary** | **scan-id** *scan-id-value*]
- **show lcc** {**ipv4**| **ipv6**} **unicast** [**summary** | **scan-id** *scan-id-value*]

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	

ステップ 2	次のいずれかのコマンドを使用します。
	• **rcc** {**ipv4** | **ipv6**} **unicast** {**enable** | **period** *milliseconds*}
	• **lcc** {**ipv4** | **ipv6**} **unicast** {**enable** | **period** *milliseconds*}
	例：
	`RP/0/RSP0/cpu 0: router(config)#rcc ipv6 unicast enable`
	`RP/0/RSP0/cpu 0: router(config)#rcc ipv6 unicast period 500`
	または
	`RP/0/RSP0/cpu 0: router(config)#lcc ipv6 unicast enable`
	`RP/0/RSP0/cpu 0: router(config)#lcc ipv6 unicast period 500`
		RCC または LCC バックグラウンド スキャンをトリガーします。検証のトリガー頻度を制御するには、**period** オプションを使用します。スキャンをトリガーするたびに、転送情報ベース（FIB）に送信されたルートまたはラベルの残りの 1 バッファ分の場所から検証が再開されます。

ステップ 3	**commit**	

ステップ 4	次のいずれかのコマンドを使用します。
	• **show rcc** {**ipv4**| **ipv6**} **unicast** [**summary** | **scan-id** *scan-id-value*]
	• **show lcc** {**ipv4**| **ipv6**} **unicast** [**summary** | **scan-id** *scan-id-value*]
	例：
	`RP/0/RSP0/cpu 0: router#show rcc ipv6 unicast statistics scan-id 120`
	または
	`RP/0/RSP0/cpu 0: router#show lcc ipv6 unicast statistics scan-id 120`
		バックグラウンド スキャンに関する統計情報を表示します。
		• **summary**：現在進行中のスキャン ID および以前の少数のスキャンの要約を表示します。
		• **scan-id** *scan-id-value*：特定のスキャンに関する詳細情報を表示します。

アップデート生成のための BGP-RIB のフィードバック メカニズム

アップデート生成機能のためのボーダー ゲートウェイ プロトコル ルーティング情報ベース（BGP-RIB）のフィードバック メカニズムによって、ネットワークで不完全なルート アドバタイズメントが行われて、それによってパケット損失が発生するのを防ぐことができます。このメカニズムによって、ルートがネイバーにアドバタイズされる前にローカルに組み込まれるようになります。

BGP は RIB からのフィードバックを待ちます。このフィードバックには、BGP によって RIB に組み込まれたルートが、BGP がネイバーにアップデートを送信する前に転送情報ベース（FIB）に組み込まれたことが示されています。RIB は BCDL のフィードバック メカニズムを使用して、そのバージョンのルートが FIB によって使用されたかを判断し、BGP をそのバージョンで更新します。BGP がアップデートを送信するのは、FIB が組み込んだバージョン以下のバージョンのルートだけです。この選択的な更新によって、BGP が不完全なアップデートを送信しないようになり、ルータのリロード、LC OIR、または代替パスが使用可能になるリンク フラップ後にデータ プレーンがプログラミングされる前であっても、トラフィックの引き込みが行われるようになります。

BGP が RIB に組み込んだルートが FIB に組み込まれたことを示す RIB からのフィードバックを BGP が待機し、その後で BGP がネイバーにアップデートを送信するように設定するには、ルータ アドレスファミリ IPv4 またはルータ アドレスファミリ VPNv4 コンフィギュレーション モードで **update wait-install** コマンドを使用します。**show bgp**、**show bgp neighbors**、および **show bgp process performance-statistics** コマンドを実行すると、update wait-install 設定の情報が表示されます。

RIB モニタリングの設定例

RIB は、Cisco IOS XR システム用に別に設定されていません。RIB は、ルーティング プロトコルからの入力に基づいて、ネットワークのルータとその他のノードの接続を計算します。RIB は、RIB とそのクライアント間の接続のモニタおよびトラブルシューティングに使用できますが、ネットワークのノード間のルーティング接続のモニタに主に使用します。ここでは、そのアクティビティをモニタするために使用する **show** コマンドによる表示について説明します。

show route コマンドの出力：例

次に、アドレスを指定せずに入力した show route コマンドの出力例を示します。

```
show route

  Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
         D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
         N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
         E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
```

```
       i - ISIS, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, su - IS-IS summary null, * - candidate default
       U - per-user static route, o - ODR, L - local

Gateway of last resort is 172.23.54.1 to network 0.0.0.0

C    10.2.210.0/24 is directly connected, 1d21h, Ethernet0/1/0/0
L    10.2.210.221/32 is directly connected, 1d21h, Ethernet0/1/1/0
C    172.20.16.0/24 is directly connected, 1d21h, ATM4/0.1
L    172.20.16.1/32 is directly connected, 1d21h, ATM4/0.1
C    10.6.100.0/24 is directly connected, 1d21h, Loopback1
L    10.6.200.21/32 is directly connected, 1d21h, Loopback0
S    192.168.40.0/24 [1/0] via 172.20.16.6, 1d21h
```

show route backup コマンドの出力：例

次に、**show route backup** コマンドの出力例を示します。

```
show route backup

Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - ISIS, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, su - IS-IS summary null, * - candidate default
       U - per-user static route, o - ODR, L - local
S    172.73.51.0/24 is directly connected, 2d20h, GigabitEthernet 4/0/0/1
                    Backup  O E2 [110/1] via 10.12.12.2, GigabitEthernet 3/0/0/1
```

show route best-local コマンドの出力：例

次に、**show route best-local** コマンドの出力例を示します。

```
show route best-local 10.12.12.1

Routing entry for 10.12.12.1/32
  Known via "local", distance 0, metric 0 (connected)
  Routing Descriptor Blocks
    10.12.12.1 directly connected, via GigabitEthernet3/0
      Route metric is 0
```

show route connected コマンドの出力：例

次に、**show route connected** コマンドの出力例を示します。

```
show route connected

C    10.2.210.0/24 is directly connected, 1d21h, Ethernet0
C    172.20.16.0/24 is directly connected, 1d21h, ATM4/0.1
C    10.6.100.0/24 is directly connected, 1d21h, Loopback1
```

show route local コマンドの出力：例

次に、**show route local** コマンドの出力例を示します。

```
show route local

L    10.10.10.1/32 is directly connected, 00:14:36, Loopback0
L    10.91.36.98/32 is directly connected, 00:14:32, Ethernet0/0
L    172.22.12.1/32 is directly connected, 00:13:35, GigabitEthernet3/0
L    192.168.20.2/32 is directly connected, 00:13:27, GigabitEthernet2/0
L    10.254.254.1/32 is directly connected, 00:13:26, GigabitEthernet2/2
```

show route longer-prefixes コマンドの出力：例

次に、**show route longer-prefixes** コマンドの出力例を示します。

```
 show route ipv4 longer-prefixes 172.16.0.0/8


Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       O - OSPF, IA - OSPF inter area, N1 - OSPF NSSA external type 1
       N2 - OSPF NSSA external type 2, E1 - OSPF external type 1
       E2 - OSPF external type 2, E - EGP, i - ISIS, L1 - IS-IS level-1
       L2 - IS-IS level-2, ia - IS-IS inter area
       su - IS-IS summary null, * - candidate default
       U - per-user static route, o - ODR, L - local

Gateway of last resort is 172.23.54.1 to network 0.0.0.0
S    172.16.2.0/32 is directly connected, 00:00:24, Loopback0
S    172.16.3.0/32 is directly connected, 00:00:24, Loopback0
S    172.16.4.0/32 is directly connected, 00:00:24, Loopback0
S    172.16.5.0/32 is directly connected, 00:00:24, Loopback0
S    172.16.6.0/32 is directly connected, 00:00:24, Loopback0
S    172.16.7.0/32 is directly connected, 00:00:24, Loopback0
S    172.16.8.0/32 is directly connected, 00:00:24, Loopback0
S    172.16.9.0/32 is directly connected, 00:00:24, Loopback0
```

show route next-hop コマンドの出力：例

次に、**show route resolving-next-hop** コマンドの出力例を示します。

```
show route resolving-next-hop 10.0.0.1

Nexthop matches 0.0.0.0/0
  Known via "static", distance 200, metric 0, candidate default path
  Installed Aug 18 00:59:04.448
  Directly connected nexthops
    172.29.52.1, via MgmtEth0/RSP0

/CPU0/0
       Route metric is 0
     172.29.52.1, via MgmtEth0/RP1/CPU0/0
       Route metric is 0
```

RCC および LCC の有効化：例

RCC および LCC バックグラウンド スキャンの有効化：例

次に、ルート整合性チェッカ（RCC）バックグラウンド スキャンを IPv6 ユニキャスト テーブルのスキャンのバッファ間 500 ミリ秒の時間で有効にする例を示します。

```
rcc ipv6 unicast period 500
```

次に、ラベル整合性チェッカ（LCC）バックグラウンド スキャンを IPv6 ユニキャスト テーブルのスキャンのバッファ間 500 ミリ秒の時間でイネーブルにする例を示します。

```
lcc ipv6 unicast period 500
```

RCC および LCC オンデマンド スキャンの有効化：例

次に、vrf1 のサブネット 10.10.0.0/16 のルート整合性チェッカ（RCC）オンデマンド スキャンを行う例を示します。

```
show rcc ipv4 unicast 10.10.0.0/16 vrf vrf 1
```

次に、ラベル整合性チェッカ（LCC）オンデマンド スキャンを IPv6 プレフィックスのすべてのラベルで実行する例を示します。

```
show lcc ipv6 unicast all
```

次の作業

RIB と対話するプロトコルの詳細については、次のマニュアルを参照してください。

- *MPLS Configuration Guide for Cisco ASR 9000 Series RoutersMPLS Configuration Guide for Cisco NCS 560 Series Routers*の「*Implementing MPLS Layer 3 VPNs*」
- *Routing Configuration Guide for Cisco ASR 9000 Series Routers*の「*Implementing BGP*」
- *Routing Configuration Guide for Cisco ASR 9000 Series Routers*の「*Implementing EIGRP*」
- *Routing Configuration Guide for Cisco ASR 9000 Series Routers*の「*Implementing IS-IS*」
- *Routing Configuration Guide for Cisco ASR 9000 Series Routers*の「*Implementing OSPF*」
- *Routing Configuration Guide for Cisco ASR 9000 Series Routers*の「*Implementing RIP*」
- *Routing Command Reference for Cisco ASR 9000 Series Routers*の「*RIB Commands*」

その他の参考資料

関連資料

関連項目	マニュアル タイトル
ルーティング情報ベース コマンド：コマンド構文の詳細、コマンド モード、コマンド履歴、デフォルト設定、使用に関する注意事項、および例	*Routing Command Reference for Cisco ASR 9000 Series Routers*の「*RIB Commands on Cisco IOS XR Software*」

標準および RFC

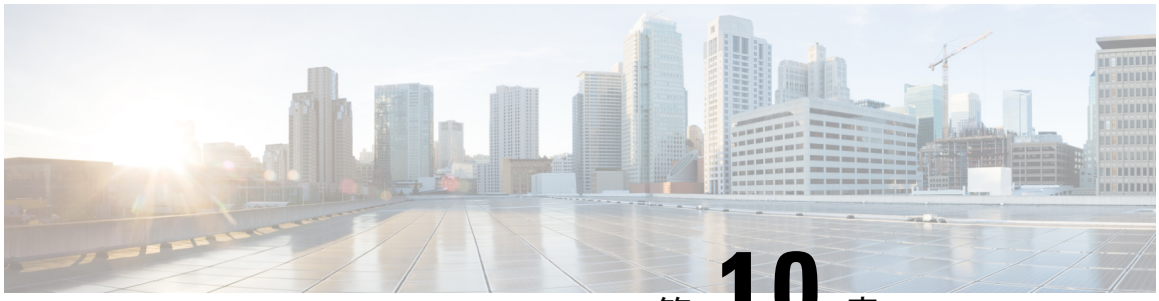
標準/RFC	タイトル
Draft-ietf-rtgwg-ipfrr-framework-06.txt	『*IP Fast Reroute Framework*』（M. Shand、S. Bryant）
Draft-ietf-rtgwg-lf-conv-frmwk-00.txt	『*A Framework for Loop-free Convergence*』（M. Shand、S. Bryant）
この機能によりサポートされた新規RFCまたは改訂RFCはありません。またこの機能による既存RFCのサポートに変更はありません。	—

MIB

MIB	MIB のリンク
—	選択したプラットフォーム、Cisco IOS リリース、およびフィーチャ セットに関する MIB を探してダウンロードするには、次の URL にある Cisco MIB Locator を使用します。 http://www.cisco.com/go/mibs

シスコのテクニカル サポート

説明	リンク
シスコのサポート Web サイトでは、シスコの製品やテクノロジーに関するトラブルシューティングにお役立ていただけるように、マニュアルやツールをはじめとする豊富なオンライン リソースを提供しています。 お使いの製品のセキュリティ情報や技術情報を入手するために、Cisco Notification Service（Field Notice からアクセス）、Cisco Technical Services Newsletter、Really Simple Syndication（RSS）フィードなどの各種サービスに加入できます。 シスコのサポート Web サイトのツールにアクセスする際は、Cisco.com のユーザ ID およびパスワードが必要です。	http://www.cisco.com/support



第 10 章

RIP の実装

ルーティング情報プロトコル（RIP）は、小規模ネットワークの自律システム（AS）内で情報を交換するために設計された従来のディスタンスベクトル内部ゲートウェイプロトコル（IGP）です。

このモジュールでは、基本的なRIPルーティングを実装するための概念とタスクについて説明します。Cisco IOS XR ソフトウェアは、RFC 2453 に記載されているとおり RIP バージョン 1（RIPv1）との下位互換性をサポートする RIP バージョン 2（RIPv2）の標準実装をサポートします。

次の機能に関連する RIP 設定情報については、このモジュールの関連資料 （542 ページ）の項を参照してください。

- マルチプロトコル ラベル スイッチング（MPLS）レイヤ 3 バーチャル プライベート ネットワーク（VPN）
- Site of Origin（SoO）のサポート

（注） Cisco IOS XR ソフトウェアでの RIP の詳細、およびこのモジュールに記載されている RIP コマンドの詳細については、このモジュールの関連資料 （542 ページ）の項を参照してください。設定タスクを実行中に表示される他のコマンドのマニュアルを見つけるには、オンラインでを検索してください。 *Cisco ASR 9000 Series Aggregation Services Router Commands Master List*

RIP の実装の機能履歴

リリース	変更内容
リリース 3.7.2	この機能が導入されました。
リリース 4.0.0	キーチェーンを使用した MD5 認証機能が追加されました。

- RIP の実装の前提条件 （522 ページ）

RIP の実装の前提条件

適切なタスク ID を含むタスク グループに関連付けられているユーザ グループに属している必要があります。このコマンド リファレンスには、各コマンドに必要なタスク ID が含まれます。ユーザ グループの割り当てが原因でコマンドを使用できないと考えられる場合、AAA 管理者に連絡してください。

RIP の実装に関する情報

RIP 機能の概要

RIP バージョン 1（RIP v1）は、実装が最も容易なルーティング プロトコルと見なされるクラスフル ディスタンスベクトル プロトコルです。OSPF とは異なり、RIP はユーザ データグラム プロトコル（UDP）データ パケットをブロードキャストして、階層型ではなくフラットなインターネットワークのルーティング情報を交換します。ネットワークの複雑さ、およびネットワーク管理に要する時間が軽減されます。ただし、クラスフル ルーティング プロトコルとして RIP v1 では、1 つのルートで表されるホスト、サブネットまたはネットワークの連続ブロックだけが許可されるため、有用性が厳しく制限されます。

RIP v2 では、RIP アップデート パケットで伝送される情報も許可され、次の項目がサポートされます。

- ルート集約
- クラスレス ドメイン間ルーティング（CIDR）
- 可変長サブネット マスク（VLSM）
- 自律システムと再配布の使用
- RIP アドバタイズメント用のマルチキャスト アドレス 224.0.0.9

異なるルートの値を評価するときに RIP が使用するメトリックは、ホップカウントです。ホップ カウントは、ルート内で経由されるルータ数です。直接接続しているネットワークのメトリックはゼロです。到達不能のネットワークのメトリックは 16 です。RIP はこのようにメトリックの範囲が小さいので、大規模なネットワークに適したルーティング プロトコルではありません。

ルーティング情報のアップデートはデフォルトでは 30 秒ごとにアドバタイズされ、隣接ルータで検出された新しいアップデートはルーティングテーブルに格納されます。

RFC 2453 に記載のとおり、RIP バージョン 2（RIP v2）のみが Cisco IOS XR ソフトウェアでサポートされていて、デフォルトでは、このソフトウェアは RIP v2 パケットのみを送受信します。一方で、バージョン 1 パケットとバージョン 2 パケットのバージョン タイプのパケットの両方、またはいずれか一方のみを送受信、または送信と受信のいずれかを実行するようにソフトウェアをインターフェイスごとに設定できます。

RIP を使用する利点は、次のとおりです。

- さまざまなネットワーク デバイスとの互換性
- 使用される帯域幅、設定、および管理時間の観点からして、オーバーヘッドがわずかなため小規模ネットワークに最適
- レガシー ホスト システムのサポート

RIP は使用が容易なため、世界中のネットワークに実装されています。

（注） VRF では、グループの設定をルータ RIP の直下に適用することはできません。グループの設定は、グローバルに適用するか、または VRF 下に適用します。

RIP のスプリット ホライズン

通常、ブロードキャスト型の IP ネットワークに接続し、ディスタンスベクトル ルーティング プロトコルを使用しているルータは、スプリット ホライズン メカニズムを使用して、ルーティングがループする可能性を軽減します。スプリット ホライズンでは、情報が発生したインターフェイス外部のルータによって、ルートに関する情報がアドバタイズされることが防止されます。通常、この動作は、複数のルータ間の（特にリンクが破損した場合の）通信を最適化します。

セカンダリ IP アドレスを使用してインターフェイスを設定し、スプリット ホライズンが有効になっている場合、すべてのセカンダリ アドレスからアップデートを送信できないことがあります。スプリット ホライズンを無効にしない場合、1 つのルーティング アップデートは、1 つのネットワーク番号ごとに送信されます。

（注） スプリット ホライズン機能は、デフォルトで有効になっています。一般的に、適切にルートをアドバタイズするために動作で変更が必要なことが確実である場合を除き、スプリット ホライズンのデフォルト状態を変更しないことを推奨します。

RIP のルート タイマー

RIP では、ルーティング アップデートの頻度、ルートが無効になるまでの時間、および他のパラメータなどの変数を決めるいくつかのタイマーを使用します。次のタイマーを調整すること

により、インターネットワークのニーズに合うように、ルーティング プロトコルのパフォーマンスを調整できます。

- ルーティング アップデートを送信する頻度（アップデートの秒単位の間隔）
- ルートが無効と宣言された後の間隔（秒単位）
- より適切なパスに関するルーティング情報が抑制されている間隔（秒単位）
- RIP トポロジ テーブルからルートが削除する前に経過する必要がある時間（秒単位）
- RIP アップデート パケット間の遅延時間

最初の 4 個のタイマーの調整は **timers basic** コマンドによって設定できます。**output-delay** コマンドは、RIP アップデート パケット間の遅延時間を変更します。設定の詳細については、RIP のカスタマイズ （530 ページ）を参照してください。

また、ソフトウェアの IP ルーティングのサポートを調整して、多様な IP ルーティング アルゴリズムのコンバージェンスを高速化でき、必要に応じて冗長ルータへのドロップバックが迅速にできます。総体的な結果として、迅速なリカバリが重要な状況で、ネットワークのエンドユーザの作業が中断する問題が最小限に抑えられます。

RIP のルート再配布

再配布とは、異なるルーティング ドメインでルーティング情報を交換できる機能のことです。異なるルーティングドメイン間をルーティングするネットワーキングデバイスは、境界ルータと呼ばれています。これらのデバイスが 1 つのルーティングプロトコルから別のルーティング プロトコルにルートを挿入します。ルーティング ドメイン内のルータは、境界ルータに再配布が実装されていないかぎり、ドメイン内部のルートのみを認識します。

ルーティング ドメインで RIP を実行しているとき、インターネットワーク内で複数のルーティング プロトコルを使用してそれらの間でルートを再配布する必要がある場合があります。次に、一般的な理由をいくつか示します。

- 他のプロトコルから RIP にルートをアドバタイズするため（スタティック、接続済み、OSPF、および BGP など）。
- RIP から EIGRP などの新しい内部ゲートウェイ プロトコル（IGP）に移行するため。
- ホスト システムをサポートするために、いくつかのルータでルーティング プロトコルを保持する一方で、他の部門グループのルータをアップグレードするため。
- 多様なルータ ベンダー環境間で通信するため。基本的にはネットワークの一部ではシスコに固有のプロトコルを使用して、シスコ以外のデバイスと通信するために RIP を使用する場合があります。

また、ルート再配布を使用すると、企業は異なるルーティング プロトコルをそれぞれのプロトコルが特に効果的な作業グループまたは領域で実行できます。Cisco IOS XR ルート再配布は、ユーザに対して単一ルーティング プロトコルのみを使用するように制限を加えないことによ

り、ダイバーシティによって技術的利点を最大化する一方でコストを最小化する優れた機能です。

インターネットワークへのルート再配布の実装に関しては、非常に単純にもでき、また非常に複雑にもできます。単純な単一方向再配布の例は、RIP が有効になっているルータにログインして、**redistribute static** コマンドを使用して、スタティック接続のみをバックボーンネットワークにアドバタイズして RIP ネットワークをパススルーさせることです。複雑な例では、ルーティングループ、互換性のないルーティング情報、および不整合なコンバージェンス時間を考慮する必要があり、複数のルーティング プロトコルが管理上のコストを実行しているときにシスコ ルータが最適なパスを選択する方法を調査して、これらの問題が発生する理由を判断する必要があります。

RIP のデフォルトのアドミニストレーティブ ディスタンス

アドミニストレーティブ ディスタンスは、IP ルーティング情報源の信頼性の尺度として使用されます。RIP などのダイナミック ルーティング プロトコルが設定されているときに、ルーティング情報の交換に再配布機能を使用する場合、適切なディスタンスの重みを設定できるように他のルート送信元のデフォルトのアドミニストレーティブディスタンスを認識していることが大切です。

次の表に、ルーティング プロトコルのデフォルトのアドミニストレーティブ ディスタンスを示します。

表 *6:* ルーティング プロトコルのデフォルト アドミニストレーティブ ディスタンス

ルーティング プロトコル	アドミニストレーティブ ディスタンス値
接続されているインターフェイス	0
インターフェイス外部のスタティック ルート	0
ネクスト ホップへのスタティック ルート	1
EIGRP サマリー ルート	5
外部 BGP	20
内部 EIGRP	90
OSPF	110
IS-IS	115
RIP バージョン 1 および 2	120
外部 EIGRP	170
内部 BGP	200

ルーティング プロトコル	アドミニストレーティブ ディスタンス値
不明	255

アドミニストレーティブ ディスタンスは、0 ～ 255 の整数です。通常は、値が大きいほど、信頼性の格付けが下がります。アドミニストレーティブディスタンスが 255 の場合はルーティング情報の送信元をまったく信頼できないため、無視する必要があります。アドミニストレーティブ ディスタンス値は主観的なものです。ルートを選択するための定量的方法はありません。

RIP のルーティング ポリシーのオプション

ルートポリシーは、**route-policy** キーワードと **end-policy** キーワードで囲まれた一連のステートメントと式によって構成されます。個別のコマンド（1 行に 1 つのコマンド）の集合ではなく、ルートポリシー内のステートメントには相互に関連するコンテキストがあります。そのため、個別のコマンドを各行に記すのではなく、各ポリシーまたはセットは独立した設定オブジェクトとして、1 つのユニットとして使用、入力、操作できます。

ポリシー設定の各行は論理サブユニットです。 **then** 、 **else** 、 **end-policy** キーワードの後ろには、少なくとも 1 つの新しい行を続ける必要があります。AS パス セット、コミュニティ セット、拡張コミュニティ セット、またはプレフィックス セットを参照するパラメータ リストと名前ストリングを閉じる括弧の後には改行が必要です。ルート ポリシー、AS パス セット、コミュニティセット、拡張コミュニティセット、またはプレフィックスセットの定義の前には、少なくとも新しい行が 1 行必要です。アクション ステートメントの後ろには 1 行以上の新しい行を続けることができます。名前付き AS パス セット、コミュニティ セット、拡張コミュニティ セット、プレフィックス セットのカンマ区切りの後ろには 1 行以上の新しい行を続けることができます。新しい行はポリシー式の論理ユニットの最後に記される必要があります。他の場所に記すことはできません。

RIP でのキーチェーンを使用した認証

Cisco IOS XR ルーティング情報プロトコル（RIP）でキーチェーンを使用する認証は、キーチェーン認証に基づいて、RIP インターフェイスのすべての RIP プロトコル トラフィックを認証するためのメカニズムを提供します。このメカニズムは、Cisco IOS XR セキュリティ キーチェーンのインフラストラクチャを使用して秘密キーを保存、取得し、それを使用してインターフェイス単位で受信および送信トラフィックを認証します。

キーチェーン管理は、相互に信頼を確立する前にキーなどの秘密を交換するすべてのエンティティに共有秘密を設定する認証の一般的な方式です。Cisco IOS XR ソフトウェアのルーティング プロトコルおよびネットワーク管理アプリケーションは、多くの場合、ピアと通信中のセキュリティを強化するために認証を使用します。

ヒント Cisco IOS XR ソフトウェア システムのセキュリティ コンポーネントは、キーチェーン管理など、さまざまなシステム セキュリティ機能を実装します。キーチェーン管理の概念、設定作業、例、およびキーチェーン管理を設定するために使用するコマンドの詳細については、次のマニュアルを参照してください。

- *System Security Configuration Guide for Cisco ASR 9000 Series Routers*の『*Implementing Keychain Management*』のモジュール
- *System Security Command Reference for Cisco ASR 9000 Series Routers*の『*Keychain Management Commands*』のモジュール

（注） キーチェーン自体には関連性がないため、キーチェーンはピアとキーを（認証のために）使用して通信する必要のあるアプリケーションで使用する必要があります。キーチェーンには、存続期間に基づいてキーおよびロールオーバーを処理するセキュアなメカニズムが備えられています。Cisco IOS XR キーチェーン インフラストラクチャは、キーチェーンの秘密キーのヒットレス ロール オーバーを処理します。

IOS XR キーチェーン データベースにキーチェーンを設定すると、特定の RIP インターフェイスに同じキーチェーンが設定されている場合は、そのインターフェイス上のすべての着信および発信 RIP トラフィックの認証に使用されます。認証キーチェーンが（デフォルト VRF または非デフォルト VRF の） RIP インターフェイスで設定されていないかぎり、すべての RIP のトラフィックは信頼性が高いと見なされ、受信 RIP トラフィックおよび発信 RIP トラフィックに対してセキュリティを保護する認証メカニズムは使用されません。

RIP は、キー付きメッセージ ダイジェスト モードとクリアテキストモードの 2 種類の認証のモードを使用します。キーチェーンの機能を使用して認証を設定するには、**authentication keychain** *keychain-name* **mode** {**md5** | **text**} コマンドを使用します。

キーチェーンがRIPインターフェイスに設定されている一方で、キーチェーンがキーチェーンデータベースで実際に設定されていないか、またはキーチェーンが MD5 暗号化アルゴリズムで設定されていない場合、インターフェイスのすべての着信 RIP パケットはドロップされます。発信パケットは認証データなしで送信されます。

インターフェイスの着信 RIP トラフィック

次は、インターフェイスにキーチェーンが設定されている場合のRIPインターフェイス上のすべての受信 RIP パケットの検証基準です。

問題	結果
RIPインターフェイスで設定されたキーチェーンが、キーチェーンのデータベースに存在しません。	パケットはドロップされます。コンポーネント レベルの RIP デバッグ メッセージは、認証失敗の特定の詳細を確認できるようにログに記録されます。

問題	結果
キーチェーンはMD5暗号化アルゴリズムを使用して設定されていません。	パケットはドロップされます。コンポーネントレベルの RIP デバッグ メッセージは、認証失敗の特定の詳細を確認できるようにログに記録されます。
メッセージの最初の（かつ最初だけの）エントリのアドレス ファミリ識別子が 0xFFFF ではない場合、認証は使用されません。	パケットはドロップされます。コンポーネントレベルの RIP デバッグ メッセージは、認証失敗の特定の詳細を確認できるようにログに記録されます。
「認証データ」のMD5ダイジェストが無効であることが検出されました。	パケットはドロップされます。コンポーネントレベルの RIP デバッグ メッセージは、認証失敗の特定の詳細を確認できるようにログに記録されます。
それ以外の場合、パケットは残りの処理のために転送されます。	

インターフェイスの発信 RIP トラフィック

次は、インターフェイスにキーチェーンが設定されている場合のRIPインターフェイス上のすべての発信 RIP パケットの検証基準です。

問題	次を実行します。
RIPインターフェイスで設定されたキーチェーンが、キーチェーンのデータベースに存在します。	同じキーチェーンを使用してパケットを認証するようにもリモート ルータが設定されている場合、RIP パケットはリモート/ピア エンドの認証チェックをパスします。
キーチェーンはMD5暗号化アルゴリズムを使用して設定されています。	同じキーチェーンを使用してパケットを認証するようにもリモート ルータが設定されている場合、RIP パケットはリモート/ピア エンドの認証チェックをパスします。
それ以外の場合、RIP パケットは認証チェックに失敗します。	

RIP の実装方法

ここでは、次のタスクの手順を示します。

（注） 設定の変更を保存するには、システムでプロンプトが表示されたら、変更を確定する必要があります。

RIP のイネーブル化

この作業では、RIP ルーティングを有効にし、RIP ルーティングプロセスを確立します。

始める前に

IP アドレスを設定する前に RIP を設定することはできますが、IP アドレスが設定されるまで、RIP はルーティングされません。

手順の概要

1. **configure**
2. **router rip**
3. **neighbor** *ip-address*
4. **broadcast-for-v2**
5. **interface** *type interface-path-id*
6. **receive version** { **1** | **2** | **1 2** }
7. **send version** { **1** | **2** | **1 2** }
8. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router rip** 例： `RP/0/RSP0/cpu 0: router(config)# router rip`	RIP ルーティングプロセスを設定します。
ステップ 3	**neighbor** *ip-address* 例： `RP/0/RSP0/cpu 0: router(config-rip)# neighbor 172.160.1.2`	（任意）RIP プロトコル情報を交換する隣接ルータを定義します。
ステップ 4	**broadcast-for-v2** 例： `RP/0/RSP0/cpu 0: router(config-rip)# broadcast-for-v2`	（任意）RIP v2 のマルチキャストアドレス（224.0.0.9）ではなくブロードキャスト IP アドレスにバージョン 2 パケットのみを送信するように RIP を設定します。このコマンドは、インターフェイスまたはグローバル コンフィギュレーション レベルで適用されます。
ステップ 5	**interface** *type interface-path-id* 例： `RP/0/RSP0/cpu 0: router(config-rip)# interface GigabitEthernet 0/1/0/0`	（任意）RIP ルーティング プロトコルを実行するインターフェイスを定義します。

	コマンドまたはアクション	目的
ステップ 6	**receive version** { **1** | **2** | **1 2** } 例： `RP/0/RSP0/cpu 0: router(config-rip-if)# receive version 1 2`	（任意）次のパケットを受信するようにインターフェイスを設定します。 • RIP v1 のみ • RIP v2 のみ • RIP v1 および RIP v2 の両方
ステップ 7	**send version** { **1** | **2** | **1 2** } 例： `RP/0/RSP0/cpu 0: router(config-rip-if)# send version 1 2`	（任意）次のパケットを送信するようにインターフェイスを設定します。 • RIP v1 のみ • RIP v2 のみ • RIP v1 および RIP v2 の両方
ステップ 8	**commit**	

RIP のカスタマイズ

このタスクでは、ネットワーク タイミングおよびルート エントリの受け入れのために RIP をカスタマイズする方法について説明します。

手順の概要

1. **configure**
2. **router rip**
3. **auto-summary**
4. **timers basic** *update invalid holddown flush*
5. **output-delay** *delay*
6. **nsf**
7. **interface** *type interface-path-id*
8. **metric-zero-accept**
9. **split-horizon disable**
10. **poison-reverse**
11. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router rip** 例：	RIP ルーティングプロセスを設定します。

	コマンドまたはアクション	目的
	`RP/0/RSP0/cpu 0: router(config)# router rip`	
ステップ 3	**auto-summary** 例： `RP/0/RSP0/cpu 0: router(config-rip)# auto-summary`	（任意）ネットワーク レベル ルートへのサブネット ルートの自動ルート サマライズを有効にします。 • デフォルトでは、自動集約は無効になっています。 （注） サブネットを切断した場合は **no** キーワードを使用して、自動ルート集約を無効にし、ソフトウェアによるサブネットとホストのルーティング情報のクラスフルネットワーク境界を越えた送信を許可します。
ステップ 4	**timers basic** *update invalid holddown flush* 例： `RP/0/RSP0/cpu 0: router(config-rip)# timers basic 5 15 15 30`	（任意）RIP ネットワーク タイマーを調整します。 （注） 現在のタイマー値とデフォルトのタイマー値を表示するには、**show rip** コマンドの出力を表示します。
ステップ 5	**output-delay** *delay* 例： `RP/0/RSP0/cpu 0: router(config-rip)# output-delay 10`	（任意）送信する RIP アップデートのパケット間遅延を変更します。 （注） 高速のレートで受信できない可能性がある低速のルータに向けて高速で送信しているハイエンド ルータがある場合、このコマンドを使用します。
ステップ 6	**nsf** 例： `RP/0/RSP0/cpu 0: router(config-rip)# nsf`	（任意）RIP プロセスのシャットダウンまたはリスタート後に RIP ルートに NSF を設定します。
ステップ 7	**interface** *type interface-path-id* 例： `RP/0/RSP0/cpu 0: router(config-rip)# interface GigabitEthernet 0/1/0/0`	（任意）RIP ルーティング プロトコルを実行するインターフェイスを定義します。
ステップ 8	**metric-zero-accept** 例： `RP/0/RSP0/cpu 0: router(config-rip-if)# metro-zero-accept`	（任意）ネットワーキング デバイスがメトリック ゼロ（0）のアップデート パケットで受信したルート エントリを受け入れられるようにします。受信したルート エントリがメトリック 1（1）に設定されます。

	コマンドまたはアクション	目的
ステップ 9	**split-horizon disable** 例： `RP/0/RSP0/cpu 0: router(config-rip-if)# split-horizon disable`	（任意）スプリット ホライズン メカニズムを無効にします。 • デフォルトでは、スプリット ホライズンは有効です。 • 一般に、アプリケーションで正しくルートをアドバタイズするために変更が必要なことが分かっている場合を除き、**split-horizon** コマンドのデフォルト状態を変更しないことを推奨します。シリアル インターフェイスでスプリット ホライズンが無効になっており、そのインターフェイスがパケットスイッチド ネットワークに接続されている場合、そのネットワークの関連マルチキャスト グループ内にあるすべてのネットワーキング デバイスに対し、スプリット ホライズンを無効にする必要があります。
ステップ 10	**poison-reverse** 例： `RP/0/RSP0/cpu 0: router(config-rip-if)# poison-reverse`	RIP ルータ アップデートのポイズン リバース処理を有効にします。
ステップ 11	**commit**	

ルーティング情報の制御

このタスクでは、ルーティング アップデートの交換および伝搬を制御または防止する方法について説明します。

次に、ルーティング アップデートを制御または防止するいくつかの理由を示します。

- WAN リンクのアップデート トラフィックを遅くするか停止するため。オンデマンド WAN リンクのアップデート トラフィックを制御しないと、リンクは常にアップ状態のままです。デフォルトでは、RIP ルーティング アップデートは 30 秒おきに発生します。
- ルーティング ループを防止するため。冗長パスがある場合、または別のルーティング ドメインにルートを再配布している場合、いずれかのパスの伝搬をフィルタします。
- アップデートで受信されるネットワークをフィルタリングするため。特定のデバイスの 1 つ以上のルートの解釈を他のルータに学習させない必要がある場合、その情報を抑制できます。
- 他のルータによるダイナミックなルート処理を防止するため。インターフェイスに入るルーティング アップデートを処理したくない場合、その情報を抑制できます。

- 帯域幅を節約するため。必要のないルーティング アップデート トラフィックを削減することによって、データ トラフィックに使用可能な帯域幅を最大化できます。

手順の概要

1. **configure**
2. **router rip**
3. **neighbor** *ip-address*
4. **interface** *type interface-path-id*
5. **passive-interface**
6. exit
7. **interface** *type interface-path-id*
8. **route-policy** { **in** | **out** }
9. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router rip** 例： `RP/0/RSP0/cpu 0: router(config)# router rip`	RIP ルーティングプロセスを設定します。
ステップ 3	**neighbor** *ip-address* 例： `RP/0/RSP0/cpu 0: router(config-rip)# neighbor 172.160.1.2`	（任意）RIP プロトコル情報を交換する隣接ルータを定義します。
ステップ 4	**interface** *type interface-path-id* 例： `RP/0/RSP0/cpu 0: router(config-rip)# interface GigabitEthernet 0/1/0/0`	（任意）RIP ルーティング プロトコルを実行するインターフェイスを定義します。
ステップ 5	**passive-interface** 例： `RP/0/RSP0/cpu 0: router(config-rip-if)# passive-interface`	（任意）明示的に設定されたネイバー宛てを除き、インターフェイスの RIP アップデートの送信を抑制します。
ステップ 6	exit 例： `RP/0/RSP0 /CPU0:router(config-rip-if)# exit`	（任意）ルータを次に高いコンフィギュレーション モードへ戻します。

	コマンドまたはアクション	目的
ステップ 7	**interface** *type interface-path-id* 例： `RP/0/RSP0/cpu 0: router(config-rip)# interface GigabitEthernet 0/2/0/0`	（任意）RIP ルーティング プロトコルを実行するインターフェイスを定義します。
ステップ 8	**route-policy** { **in** | **out** } 例： `RP/0/RSP0/cpu 0: router(config-rip-if)# route-policy out`	（任意）RIP ネイバーにアドバタイズするアップデートや、RIP ネイバーから受信するアップデートに、ルーティング ポリシーを適用します。
ステップ 9	**commit**	

RIP のルート ポリシーの作成

このタスクでは、ルートポリシーを定義して、RIP プロセスのインスタンスに付加する方法を示します。ルート ポリシーは、次の目的で使用できます。

- 送受信されるルートの制御
- 再配信されるルートの制御
- デフォルト ルートの発生の制御

ルートポリシーの定義は、**route-policy** コマンドと *name* 引数、その後に続く一連のオプションのポリシーステートメントで構成され、**end-policy** コマンドで閉じられます。

ルート ポリシーはルーティング プロトコルのルートに適用されてはじめて役に立ちます。

手順の概要

1. **configure**
2. **route-policy** *name*
3. **set rip-metric** *number*
4. **end-policy**
5. **commit**
6. **configure**
7. **router rip**
8. **route-policy** *route-policy-name* { **in** | **out** }
9. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	

	コマンドまたはアクション	目的
ステップ 2	**route-policy** *name* 例： `RP/0/RSP0/cpu 0: router(config)# route-policy IN-IPv4`	ルート ポリシーを定義して、ルート ポリシー コンフィギュレーション モードを開始します。
ステップ 3	**set rip-metric** *number* 例： `RP/0/RSP0/cpu 0: router(config-rpl)# set rip metric 42`	（任意）RIP メトリック属性を設定します。
ステップ 4	**end-policy** 例： `RP/0/RSP0/cpu 0: router(config-rpl)# end-policy`	ルートポリシーの定義を終了して、ルートポリシーコンフィギュレーション モードを終了します。
ステップ 5	**commit**	
ステップ 6	**configure**	
ステップ 7	**router rip** 例： `RP/0/RSP0/cpu 0: router(config)# router rip`	RIP ルーティングプロセスを設定します。
ステップ 8	**route-policy** *route-policy-name* { **in** | **out** } 例： `RP/0/RSP0/cpu 0: router(config-rip)# route-policy rp1 in`	RIP ネイバーにアドバタイズされる更新または RIP ネイバーから受信する更新にルーティングポリシーを適用します。
ステップ 9	**commit**	

RIP 認証キーチェーンの設定

デフォルト以外の VRF の IPv4 インターフェイスの RIP 認証キーチェーンの設定

RIP 認証キーチェーンを非デフォルト VRF の IPv4 インターフェイスに設定するには、次のタスクを実行します。

始める前に

キーチェーンを RIP インターフェイス/VRF に適用するには、*System Security Configuration Guide for Cisco ASR 9000 Series Routers*の「*Implementing Keychain Management*」のモジュールで説明されているコンフィギュレーション コマンドを使用して、Cisco IOS XR キーチェーンデータベース内にすべてのキーチェーンを設定する必要があります。

authentication keychain *keychain-name* と **mode md5** コンフィギュレーションは、IOS XR キーチェーンデータベースにまだ設定されていないキーチェーン、または MD5 暗号アルゴリズムを使用せずに IOS XR キーチェーンデータベースに設定されているキーチェーンの名前を受け入れます。ただし、両方の場合ですべての着信パケットはインターフェイスでドロップされ、送信パケットは認証データなしで送信されます。

手順の概要

1. **configure**
2. **router rip**
3. **vrf** *vrf_name*
4. **interface** *type interface-path-id*
5. 次のいずれかのコマンドを使用します。
   - **authentication keychain** *keychain-name* **mode md5**
   - **authentication keychain** *keychain-name* **mode text**
6. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router rip** 例： `RP/0/RSP0/cpu 0: router(config)#router rip`	RIP ルーティングプロセスを設定します。
ステップ 3	**vrf** *vrf_name* 例： `RP/0/RSP0/cpu 0: router(config-rip)#vrf vrf_rip_auth`	非デフォルトの VRF を設定します。
ステップ 4	**interface** *type interface-path-id* 例： `RP/0/RSP0/cpu 0: router(config-rip-vrf)#interface POS 0/6/0/0`	RIP ルーティング プロトコルを実行するインターフェイスを定義します。
ステップ 5	次のいずれかのコマンドを使用します。 • **authentication keychain** *keychain-name* **mode md5** • **authentication keychain** *keychain-name* **mode text** 例： `RP/0/RSP0/cpu 0: router(config-rip-if)#authentication keychain key1 mode md5` または	RIP の認証キーチェーン モードを設定します。 • **md5**：キーメッセージダイジェスト（md5）認証モード • **text**：クリアテキストの認証モード

	コマンドまたはアクション	目的

	`RP/0/RSP0/cpu 0:`
	`router(config-rip-if)#authentication keychain key1`
	` mode text`	

ステップ 6	**commit**	

デフォルトの VRF の IPv4 インターフェイスの RIP 認証キーチェーンの設定

RIP 認証キーチェーンをデフォルト VRF の IPv4 インターフェイスに設定するには、次のタスクを実行します。

始める前に

キーチェーンを RIP インターフェイス/VRF に適用するには、 *System Security Configuration Guide for Cisco ASR 9000 Series Routers*の「*Implementing Keychain Management*」のモジュールで説明されているコンフィギュレーション コマンドを使用して、Cisco IOS XR キーチェーンデータベース内にすべてのキーチェーンを設定する必要があります。

authentication keychain *keychain-name* と **mode md5** コンフィギュレーションは、IOS XR キーチェーンデータベースにまだ設定されていないキーチェーン、または MD5 暗号アルゴリズムを使用せずに IOS XR キーチェーンデータベースに設定されているキーチェーンの名前を受け入れます。ただし、両方の場合ですべての着信パケットはインターフェイスでドロップされ、送信パケットは認証データなしで送信されます。

手順の概要

1. **configure**
2. **router rip**
3. **interface** *type interface-path-id*
4. 次のいずれかのコマンドを使用します。
   - **authentication keychain** *keychain-name* **mode md5**
   - **authentication keychain** *keychain-name* **mode text**
5. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router rip** 例： `RP/0/RSP0/cpu 0: router(config)#router rip`	RIP ルーティングプロセスを設定します。
ステップ 3	**interface** *type interface-path-id* 例：	RIP ルーティング プロトコルを実行するインターフェイスを定義します。

	コマンドまたはアクション	目的
	`RP/0/RSP0/cpu 0: router(config-rip)#interface POS 0/6/0/0`	

ステップ 4	次のいずれかのコマンドを使用します。
	• **authentication keychain** *keychain-name* **mode md5**
	• **authentication keychain** *keychain-name* **mode text**
	例：
	`RP/0/RSP0/cpu 0: router(config-rip-if)#authentication keychain key1 mode md5`
	または
	`RP/0/RSP0/cpu 0: router(config-rip-if)#authentication keychain key1 mode text`
		RIP の認証キーチェーン モードを設定します。
		• **md5**：キーメッセージダイジェスト（md5）認証モード
		• **text**：クリアテキストの認証モード

ステップ 5	**commit**	

RIP の実装の設定例

ここでは、次の設定例について説明します。

基本的な RIP の設定：例

次に、2 つのギガビット イーサネット インターフェイスを RIP を使用して設定する例を示します。

```
interface GigabitEthernet0/6/0/0
 ipv4 address 172.16.0.1 255.255.255.0
!

interface GigabitEthernet0/6/0/2
 ipv4 address 172.16.2.12 255.255.255.0
!

router rip
 interface GigabitEthernet0/6/0/0
 !
 interface GigabitEthernet0/6/0/2
 !
!
```

プロバイダーエッジでの RIP の設定：例

次に、2 つの VPN ルーティングおよび転送（VRF）インスタンスを使用して PE に基本的な RIP を設定する例を示します。

```
router rip
 interface GigabitEthernet0/6/0/0
 !
 vrf vpn0
  interface GigabitEthernet0/6/0/2
  !
 !
 vrf vpn1
  interface GigabitEthernet0/6/0/3
  !
 !
!
```

各 VRF インスタンスの RIP タイマーの調整：例

次に、各 VPN ルーティングおよび転送（VRF）インスタンスの RIP タイマーを調整する例を示します。

VRF インスタンス vpn0 の場合、**timers basic** コマンドは更新を 10 秒ごとにブロードキャストするように設定します。ルータから 30 秒間送信がないと、そのルートは使用不能と宣言されます。以降の情報はさらに 30 秒間抑止されます。フラッシュ期間（45 秒）の終了時に、ルーティングテーブルからルートがフラッシュされます。

VRF インスタンス vpn1 の場合、タイマーは 20、60、60、および 70 秒と異なる調整が行われます。

output-delay コマンドは、vpn1 の RIP 更新パケット間の遅延を 10 ミリ秒に変更します。デフォルトでは、パケット間遅延はオフになっています。

```
router rip
 interface GigabitEthernet0/6/0/0
 !
 vrf vpn0
  interface GigabitEthernet0/6/0/2
  !
  timers basic 10 30 30 45
 !
 vrf vpn1
  interface GigabitEthernet0/6/0/3
  !
  timers basic 20 60 60 70
  output-delay 10
 !
!
```

RIP の再配布の設定：例

次に、ボーダー ゲートウェイ プロトコル（BGP）およびスタティック ルートを RIP に再配布する例を示します。

再配布されるルートで使用される RIP メトリックは、ルート ポリシーによって決まります。ルート ポリシーが設定されていないか、ルート ポリシーで RIP メトリックが設定されていな

い場合は、再配布されるプロトコルに基づいてメトリックが決定されます。BGPによって再配布されるVPNv4ルートの場合、リモートPEルータで設定されたRIPメトリックが有効であれば、それが使用されます。

その他すべての場合（BGP、IS-IS、OSPF、EIGRP、接続済み、静的）、**default-metric** コマンドで設定されたメトリックが使用されます。有効なメトリックが決定できない場合、再配布は起こりません。

```
route-policy ripred
  set rip-metric 5
end-policy
!

router rip
 vrf vpn0
  interface GigabitEthernet0/6/0/2
  !
  redistribute connected
  default-metric 3
 !
 vrf vpn1
  interface GigabitEthernet0/6/0/3
  !
  redistribute bgp 100 route-policy ripred
  redistribute static
  default-metric 3
 !
!
```

RIP のルート ポリシーの設定：例

次に、RIPインターフェイスによって受信されるまたはRIPインターフェイスから送信されるルート アップデートを制御するために使用される、着信および発信ルート ポリシーを設定する例を示します。

```
prefix-set pf1
  10.1.0.0/24
end-set
!

prefix-set pf2
  150.10.1.0/24
end-set
!

route-policy policy_in
  if destination in pf1 then
    pass
  endif
end-policy
!

route-policy pass-all
  pass
end-policy
!
```

```
route-policy infil
  if destination in pf2 then
    add rip-metric 2
    pass
  endif
end-policy
!

router rip
 interface GigabitEthernet0/6/0/0
  route-policy policy_in in
 !
 interface GigabitEthernet0/6/0/2
 !
 route-policy infil in
 route-policy pass-all out
```

RIP のパッシブ インターフェイスおよび明示的なネイバーの設定：例

次に、パッシブインターフェイスおよび明示的なネイバーを設定する例を示します。インターフェイスがパッシブな場合は、ルーティングアップデートを受信するのみです。つまり、明示的に設定されたネイバー宛てを除き、インターフェイスからアップデートは送信されません。

```
router rip
 interface GigabitEthernet0/6/0/0
  passive-interface
 !
 interface GigabitEthernet0/6/0/2
 !
 neighbor 172.17.0.1
 neighbor 172.18.0.5
!
```

RIP ルートの制御：例

次に、**distance** コマンドを使用して、RIP ルートをルーティング情報ベース（RIB）にインストールする例を示します。**maximum-paths** コマンドは、RIP ルートごとに許可される最大パス数を制御します。

```
router rip
 interface GigabitEthernet0/6/0/0
  route-policy polin in
 !
 distance 110
 maximum-paths 8
!
```

RIP 認証キーチェーンの設定：例

次に、RIP のデフォルト VRF インターフェイスに認証キーチェーンを適用する例を示します。

```
router rip
 interface POS0/6/0/0
  authentication keychain key1 mode md5
 !
!
end
```

次に、RIP の非デフォルト インターフェイスに認証キーチェーンを適用する例を示します。

```
router rip
 vrf rip_keychain_vrf
  interface POS0/6/0/0
   authentication keychain key1 mode md5
  !
 !
!
end
```

その他の参考資料

次の各項では、RIP の実装に関連するその他の資料について説明します。

関連資料

関連項目	マニュアル タイトル
RIP コマンド：コマンド構文の詳細、コマンド モード、コマンド履歴、デフォルト設定、使用に関する注意事項、および例	*Routing Command Reference for Cisco ASR 9000 Series Routers*
RIP の MPLS VPN サポートの機能情報	*MPLS Configuration Guide for Cisco ASR 9000 Series RoutersMPLS Configuration Guide for Cisco NCS 560 Series Routers*の「*Implementing MPLS Traffic Engineering on* Cisco ASR 9000 シリーズ ルータ 」のモジュール
RIP の Site of Origin（SoO）サポートの機能情報	*MPLS Configuration Guide for Cisco ASR 9000 Series RoutersMPLS Configuration Guide for Cisco NCS 560 Series Routers*の「*Implementing MPLS Traffic Engineering on* Cisco ASR 9000 シリーズ ルータ 」のモジュール
Cisco IOS XR スタートアップ ガイド	*Cisco ASR 9000 Series Aggregation Services Router Getting Started Guide*
ユーザ グループとタスク ID に関する情報	*System Security Configuration Guide for Cisco ASR 9000 Series Routers*の「*Configuring AAA Services on* Cisco ASR 9000 シリーズ ルータ 」のモジュール

標準

標準	タイトル
この機能でサポートされる新規の標準または変更された標準はありません。また、既存の標準のサポートは変更されていません。	—

MIB

MIB	MIB のリンク
—	Cisco IOS XR ソフトウェアを使用して MIB の場所を特定してダウンロードするには、次の URL にある Cisco MIB Locator を使用して、[Cisco Access Products] メニューからプラットフォームを選択します。 https://mibs.cloudapps.cisco.com/ITDIT/MIBS/servlet/index

RFC

RFC	タイトル
RFC 2453	『RIP Version 2』

シスコのテクニカル サポート

説明	リンク
シスコのテクニカル サポート Web サイトでは、製品、テクノロジー、ソリューション、技術的なヒント、およびツールへのリンクなどの、数千ページに及ぶ技術情報が検索可能です。Cisco.com に登録済みのユーザは、このページから詳細情報にアクセスできます。	http://www.cisco.com/techsupport

第 11 章

ルーティング ポリシーの実装

ピアから受け入れるか、ピアにアドバタイズされる、または1個のルーティング プロトコルから別のプロトコルへ再配布されるときに、ルートを検査し、フィルタリングして、属性を変更するように、ルーティング ポリシーがルータに指示します。

このモジュールでは、ルーティング プロトコルが設定済みのルーティング ポリシーに基づいて、ルートのアドバタイズ、集約、廃棄、配布、エクスポート、保留、インポート、再配布、変更を決定する方法について説明します。

ルーティング ポリシー言語（RPL）では、すべてのルーティング ポリシーのニーズを表現できる、単一の直接的な言語です。RPLは、大規模なルーティング設定をサポートするように設計されました。以前のルーティング ポリシー コンフィギュレーション方式に固有の冗長性が大幅に削減されました。RPL では、ルーティング ポリシーの設定が簡素化され、これらの設定の保存および処理に必要なシステム リソースが削減され、トラブルシューティングが容易になりました。

（注） Cisco IOS XR ソフトウェアのルーティング ポリシーの詳細情報とこのモジュールに掲げられたルーティング ポリシー コマンドの詳細については、このモジュールの関連資料 （645 ページ）の項を参照してください。設定タスクを実行中に表示される他のコマンドのマニュアルを見つけるには、オンラインで を検索してください。 *Cisco ASR 9000 Series Aggregation Services Router Commands Master List*

ルーティング ポリシーの実装の機能履歴

リリース	変更内容
リリース 3.7.2	この機能が導入されました。
リリース 3.9.0	すべての接続点で、パラメータ化がサポートされました。
リリース 4.2.0	次の機能が追加されました。 • 階層的な条件 • 条件ポリシーの適用

リリース	変更内容
リリース 4.2.1	次の機能が導入されました。 • 拡張プレフィックス長操作。 • ネストされたワイルドカード適用ポリシー。 • XML を使用したルーティング ポリシー言語セット要素の編集。 • bgp export および bgp import の接続点の「med」属性の有効な演算子として「set」をサポートします。
リリース 4.3.1	次の機能が導入されました。 • VRF RPL ベースのインポート ポリシー • フレキシブル L3VPN ラベル割り当て

ルーティング ポリシー実装の前提条件

次に、Cisco IOS XR ソフトウェアでルーティング ポリシーを実装するための前提条件を示します。

- 適切なタスク ID を含むタスク グループに関連付けられているユーザ グループに属している必要があります。このコマンド リファレンスには、各コマンドに必要なタスク ID が含まれます。ユーザ グループの割り当てが原因でコマンドを使用できないと考えられる場合、AAA 管理者に連絡してください。
- ボーダー ゲートウェイ プロトコル（BGP）、Integrated Intermediate System-to-Intermediate（IS-IS）または Open Shortest Path First（OSPF）がネットワーク内で設定されている必要があります。

ルーティング ポリシー実装の制約事項

次の制約事項は、Cisco IOS XR ソフトウェアでルーティング ポリシー言語実装を使用する場合に適用されます。

- 最大1000のステートメントのポリシー定義がサポートされています。ポリシー内のステートメントの総数は、階層型ポリシー構造を使用して4000ステートメントに拡張できます。ただし、この制限は、**apply** ステートメントの使用に限定されます。
- 接続点で直接または間接的に付加されたポリシーを変更する必要がある場合、単一の **commit** 操作は次の場合に実行できません。
  - 接続点に直接または間接的に付加された別のポリシーによって参照されているセットまたはポリシーを削除する。
  - 削除するものと同じセットまたはポリシーへの参照を削除するためにポリシーを変更する。

  commit は、次の 2 つの手順で実行する必要があります。
  1. ポリシーまたはセットへの参照を削除するためにポリシーを変更してから、**commit** を実行します。
  2. ポリシーまたはセットを削除してから、**commit** を実行します。
- 内部および外部の BGP マルチパスが設定されている Carrier Supporting Carrier（CSC）ネットワークでは、vrf 単位のラベル モードはサポートされていません。
- IPv4 ピアから始まるルートについては、RPL ポリシーを介してネクストホップアドレスを IPv6 アドレスに変更することはできません。

ルーティング ポリシーの実装に関する情報

RPL を実装するには、次の概念を理解する必要があります。

ルーティング ポリシー言語

ここでは、次の内容について説明します。

ルーティング ポリシー言語の概要

RPLは、大規模なルーティング設定をサポートするように開発されました。RPLには、従来のルート マップ、アクセス リスト、プレフィックス リスト向けの設定の機能とは異なる、重要な機能がいくつか備えられています。これらの機能の1つめは、モジュラ形式でポリシーを構築する機能です。ポリシーの共通ブロックは、個別に定義および維持できます。次に、ポリシーのこれらの共通ブロックをポリシーの他のブロックから適用して完全なポリシーを構築できます。この機能により、維持する必要のある設定情報の量が減ります。また、ポリシーのこれらの共通ブロックはパラメータ化できます。パラメータ化により、同じ構造を共有するが、設定または一致された特定の値が異なるポリシーをポリシーの独立したブロックとして維持できます。たとえば、ローカルプリファレンス値以外はすべて同一である3つのポリシーは、ポ

リシーのパラメータとして異なるローカルプリファレンス値を持つ1つの共通のパラメータ化ポリシーとして表せます。

このポリシー言語では、セットという概念が導入されました。セットとは、ルート属性の一致および設定演算で使用できる類似したデータのコンテナです。セットタイプには、prefix-sets、community-sets、as-path-sets、およびextcommunity-setsの4つがあります。これらのセットはそれぞれ、IPv4またはIPv6プレフィックス、コミュニティ値、ASパス正規表現、および拡張コミュニティ値のグループ化を保持します。セットは、データの単なるコンテナです。セットの大半はインライン変数も保持します。インラインセットでは、名前付きセットを参照せずに、値の短い列挙をポリシーで直接使用できます。プレフィックスリスト、コミュニティリスト、およびASパスリストは、リストに項目が1つか2つしかない場合でも維持が必要です。RPLのインラインセットを使用すると、名前付きセットを参照することなく、値の小さいセットをポリシー本体に直接配置できます。

許可や拒否などの決定は、ポリシー定義自体で明示的に制御されます。RPLは、セットデータを使用する可能性のある一致演算子を、従来のブール論理演算子AND、OR、およびNOTと組み合わせて複雑な条件式にします。すべての一致演算はtrueまたはfalseの結果を返します。その後、これらの条件式の実行および関連するアクションは、*if then*、*elseif*、および*else*の単純な構造を使用して制御できます。これにより、ポリシーを通じて評価パスをすべて指定できます。

ルーティング ポリシー言語の構造

ここでは、RPLの基本構造について説明します。

名前

ポリシー言語には、セットとポリシーの2種類の持続的で名前を付けられるオブジェクトがあります。これらのオブジェクトの定義は、開始および終了のコマンドラインとして括弧で囲まれます。たとえば、testという名前のポリシーを定義する設定構文は、次のようになります。

```
route-policy test
[ . . . policy statements . . . ]
end-policy
```

ポリシーオブジェクトの正規名は、任意の連続する英数字（大文字および小文字）、数字の0～9、および句読文字のピリオド、ハイフン、およびアンダースコアで指定できます。名前は、文字または数字ではじまる必要があります。

セット

このコンテキストでは、セットという用語を、順序付けのない固有のエレメントの集合を意味する数学的な概念で使用されます。ポリシー言語は、セットをマッチング用の値のグループに対するコンテナとして提供します。セットは、条件式で使用されます。セットの要素はカンマで区切ります。ヌル（空）のセットは許可されます。

次の例で、

```
prefix-set backup-routes
```

```
  # currently no backup routes are defined
end-set
```

次の条件は、

```
if destination in backup-routes then
```

すべてのルートに対して FALSE として評価されます。これは、プレフィックス セットに一致する一致条件がないためです。

次の 5 種類のセットがあります。as-path-set （550 ページ）、community-set （550 ページ）、extcommunity-set （551 ページ）、prefix-set （556 ページ）、および rd-set （558 ページ）。たとえば、2 つまたは 3 つのコミュニティ値のように少ない数の要素に対して比較を実行する場合があるとします。これらの比較を実現するため、ユーザはこれらの値を直接列挙できます。これらの列挙はインラインセットと呼ばれます。インラインセットは、機能的には名前付きセットと同じですが、単純なテストをインラインにできるようにします。つまり、1 つや 2 つだけの要素を比較する際には、別の名前付きセットの維持を比較では必要としません。構文については、次の項で説明するセット型を参照してください。通常、インラインセットの構文は、(element-entry, element-entry, element-entry, ...element-entry) のように括弧で囲まれたカンマ区切りのリストです。ここで element-entry は、プレフィックスやコミュニティ値などの使用タイプに適した項目のエントリです。

次に、インライン コミュニティ セットを使用する例を示します。

```
route-policy sample-inline
if community matches-any ([10..15]:100) then
set local-preference 100
endif
end-policy
```

次に、test-communities という名前付きセットを使用する同等の例を示します。

```
community-set test-communities
10:100,
11:100,
12:100,
13:100,
14:100,
15:100
end-set

route-policy sample
if community matches-any test-communities then
set local-preference 100
endif
end-policy
```

これらの両方のポリシーは機能的に同等ですが、インライン形式では、6 つの値を格納するだけのためにコミュニティ セットの設定を必要としません。設定コンテキストに適切な形式を選

択できます。次の各項では、名前付きセットバージョンとインライン形式の両方の例が必要に応じて示されています。

as-path-set

AS パス セットは、AS パス属性と一致させるための演算で構成されます。一致演算は、正規表現一致だけです。

名前付きセット形式

名前付きセット形式は、**ios-regex** キーワードを使用して正規表現のタイプを示します。また、正規表現を単一引用符で囲む必要があります。

次の例では、名前付き AS パス セットの定義を示します。

```
as-path-set aset1
ios-regex '_42$',
ios-regex '_127$'
end-set
```

この AS パス セットは 2 つの要素から構成されます。一致演算で使用する場合は、この AS パスセットは、AS パスが自律システム（AS）番号 42 または 127 のいずれかで終わる任意のルートと一致します。

名前付き AS パス セットを削除するには、**no as-path-set aset1** コマンドライン インターフェイス（CLI）コマンドを使用します。

（注） 正規表現一致により、CPU の負荷が高くなります。ポリシー パフォーマンスを大幅に向上するには、次のいずれかを実行します。正規表現パターンをまとめて正規表現の合計呼び出し数を減らす、または「as-path neighbor-is」、「as-path originates-from」、「as-path passes-through」などの同等のネイティブ as-path 一致演算を使用します。

インライン セット形式

インライン セットは、次のようにカンマ区切りの式のリストを括弧で囲んだ形式です。

```
(ios-regex '_42$', ios-regex '_127$')
```

このセットは、前の名前付きセットと同じ AS パスと一致させますが、ポリシーが使用する名前付きセットとは別に名前付きセットを作成する余計な作業を必要としません。

community-set

コミュニティ セットは、BGP コミュニティ属性との一致のためにコミュニティ値を保持しています。コミュニティは、32 ビット量です。整数のコミュニティ値は半分に分けて、コロンで

区切った、0 ～ 65535 の範囲内の 2 つの符号なし 10 進整数で表す必要があります。単一の 32 ビットコミュニティ値は指定できません。次に、名前付きセット形式を示します。

名前付きセット形式

```
community-set cset1
12:34,
12:56,
12:78,
internet
end-set
```

インライン セット形式

```
(12:34, 12:56, 12:78)
($as:34, $as:$tag1, 12:78, internet)
```

コミュニティセットのインライン形式では、パラメータ化もサポートされます。コミュニティの 16 ビット部分のそれぞれをパラメータ化できます。詳細については、パラメータ化 （563 ページ）を参照してください。

RPL では、標準の well-known コミュニティ値のシンボル名が提供されます。internet は 0:0、no-export は 65535:65281、no-advertise は 65535:65282、local-as は 65535is-empty:65283 です。

RPL では、コミュニティの指定でワイルドカードを使用するためのファシリティも用意されています。ワイルドカードを指定するには、コミュニティ指定の 16 ビット部分の 1 つの代わりに、アスタリスク（*）を挿入します。ワイルドカードはコミュニティのその部分の任意の値が一致することを示します。つまり、次のポリシーが一致するすべてのコミュニティの中で、自律システムが属するコミュニティは 123 です。

```
community-set cset3
   123:*
end-set
```

コミュニティセットは空にするか、または 1 つ以上のコミュニティ値を含めることができます。空のコミュニティセットとともに使用すると、**is-empty** 演算子は TRUE と評価され、**matches-any** 演算子と **matches-every** 演算子は FALSE と評価されます。

extcommunity-set

拡張コミュニティセットは、通常のコミュニティ値の代わりに拡張コミュニティ値が含まれている点を除き、コミュニティ セットと似ています。拡張コミュニティ セットは、名前付き形式およびインライン形式もサポートします。拡張コミュニティ セットには、cost、soo、rt の 3 つのタイプがあります。

コミュニティセットと同様に、インライン形式では、パラメータ化ポリシー内のパラメータ化がサポートされます。拡張コミュニティ値のいずれかの部分をパラメータ化できます。

ワイルドカード（*）および正規表現は、拡張コミュニティ セット要素で使用できます。

すべての拡張コミュニティセットに、少なくとも1つの拡張コミュニティ値が含まれている必要があります。空の拡張コミュニティ セットは無効なため拒否されます。

次に、構文例を示します。

extcommunity-set cost の名前付き形式

cost セットは、コスト EIGRP コスト コミュニティ タイプの拡張コミュニティ タイプ コミュニティを格納するために使用される extcommunity セットです。

```
extcommunity-set cost a_cost_set
   IGP:1:10
end-set
```

次のオプションが拡張コミュニティ セット Cost でサポートされています。

```
RP/0/RSP0/cpu 0: router(config)#extcommunity-set cost cost_set
RP/0/RSP0/cpu 0: router(config-ext)#?
  #-remark       Remark beginning with '#'
  <0-255>        decimal number
  abort          Discard RPL definition and return to top level config
  end-set        End of set definition
  exit           Exit from this submode
  igp:           Cost Community with IGP as point of insertion
  pre-bestpath:  Cost Community with Pre-Bestpath as point of insertion
  show           Show partial RPL configuration
```

オプション	説明
#-remark	「#」ではじまる注記
<0-255>	10 進数
abort	RPL 定義を廃棄して、top level config に戻る
end-set	セット定義の終了
exit	このサブモードを終了
igp:	IGP を挿入ポイントとして使用するコスト コミュニティ
pre-bestpath:	Pre-Bestpath を挿入ポイントとして使用するコスト コミュニティ
show	一部の RPL 設定を表示

extcommunity-set rt の名前付き形式

rt セットは BGP ルート ターゲット（RT）拡張コミュニティ タイプのコミュニティを格納するために使用される extcommunity セットです。

```
extcommunity-set rt a_rt_set
   1.2.3.4:666
```

```
    1234:666,
    1.2.3.4:777,
    4567:777
  end-set

  Inline Set Form for Extcommunity-set RT

  (1.2.3.4:666, 1234:666, 1.2.3.4:777, 4567:777)
  ($ipadrr:666, 1234:$tag, 1.2.3.4:777, $tag2:777)
```

次のオプションが拡張コミュニティ セット RT でサポートされています。

```
RP/0/RSP0/cpu 0: router(config)#extcommunity-set rt rt_set
RP/0/RSP0/cpu 0: router(config-ext)#?
  #-remark        Remark beginning with '#'
  *               Wildcard (any community or part thereof)
  <1-4294967295>  32-bit decimal number
  <1-65535>       16-bit decimal number
  A.B.C.D/M:N     Extended community - IPv4 prefix format
  A.B.C.D:N       Extended community - IPv4 format
  ASN:N           Extended community - ASPLAIN format
  X.Y:N           Extended community - ASDOT format
  abort           Discard RPL definition and return to top level config
  dfa-regex       DFA style regular expression
  end-set         End of set definition
  exit            Exit from this submode
  ios-regex       Traditional IOS style regular expression
  show            Show partial RPL configuration
```

オプション	説明
#-remark	「#」ではじまる注記
*	ワイルドカード（任意のコミュニティまたはその一部）
<1-4294967295>	32 ビットの 10 進数
<1-65535>	16 ビットの 10 進数
A.B.C.D/M:N	拡張コミュニティ：IPv4 プレフィックス形式
A.B.C.D:N	拡張コミュニティ：IPv4 形式
ASN:N	拡張コミュニティ：AS プレーン形式
X.Y:N	拡張コミュニティ：ASDOT 形式
abort	RPL 定義を廃棄して、top level config に戻る
dfa-regex	DFA スタイルの正規表現
end-set	セット定義の終了
exit	このサブモードを終了
ios-regex	従来の IOS スタイルの正規表現
show	一部の RPL 設定を表示

extcommunity-set soo の名前付き形式

soo セットは、BGP Site-of-Origin（SoO）拡張コミュニティ タイプ コミュニティを格納するために使用される extcommunity セットです。

```
  extcommunity-set soo a_soo_set
1.1.1:100,
     100:200
  end-set
```

次のオプションが拡張コミュニティ セット soo でサポートされています。

```
RP/0/RSP0/cpu 0: router(config)#extcommunity-set soo soo_set
RP/0/RSP0/cpu 0: router(config-ext)#?
  #-remark         Remark beginning with '#'
  *                Wildcard (any community or part thereof)
  <1-4294967295>   32-bit decimal number
  <1-65535>        16-bit decimal number
  A.B.C.D/M:N      Extended community - IPv4 prefix format
  A.B.C.D:N        Extended community - IPv4 format
  ASN:N            Extended community - ASPLAIN format
  X.Y:N            Extended community - ASDOT format
  abort            Discard RPL definition and return to top level config
  dfa-regex        DFA style regular expression
  end-set          End of set definition
  exit             Exit from this submode
  ios-regex        Traditional IOS style regular expression
  show             Show partial RPL configuration
```

オプション	説明
#-remark	「#」ではじまる注記
*	ワイルドカード（任意のコミュニティまたはその一部）
<1-4294967295>	32 ビットの 10 進数
<1-65535>	16 ビットの 10 進数
A.B.C.D/M:N	拡張コミュニティ：IPv4 プレフィックス形式
A.B.C.D:N	拡張コミュニティ：IPv4 形式
ASN:N	拡張コミュニティ：AS プレーン形式
X.Y:N	拡張コミュニティ：ASDOT 形式
abort	RPL 定義を廃棄して、top level config に戻る
dfa-regex	DFA スタイルの正規表現
end-set	セット定義の終了
exit	このサブモードを終了
ios-regex	従来の IOS スタイルの正規表現

オプション	説明
show	一部の RPL 設定を表示

Extcommunity-set 帯域幅の名前付き形式

帯域幅設定では、マルチホーム サイトの不等ロード バランシングを実行するための、リンク帯域幅の拡張コミュニティ属性のサポートが提供されます。

```
extcommunity-set bandwidth extcomm-bw
 100:25000
  end-set
```

Demilitarized Zone（DMZ; 緩衝地帯）リンク帯域幅の値は、ルーティング テーブルを使用するか、または *additive* キーワードを追加した、outbound route-policy を使用して設定されます。また、これによって、ピアの受信端で、routes-not-imported 状態になります。

```
extcommunity-set bandwidth dmz_ext
   1:8000
 end-set
 !
 route-policy dmz_rp_vpn
   set extcommunity bandwidth dmz_ext additive     <<< 'additive' keyword.
   pass
 end-policy
```

次のオプションが拡張コミュニティ セットの帯域幅でサポートされています。

```
RP/0/RSP0/cpu 0: router(config)# extcommunity-set bandwidth extcomm-bw
RP/0/RSP0/cpu 0: router(config-ext)# ?

  #-remark   Remark beginning with '#'
  <1-65534>  16-bit decimal number
  AS-TRANS   ASN23456
  ASN:N      extended community - ASPLAIN format
  abort      Discard RPL definition and return to top level config
  end-set    End of set definition
  exit       Exit from this submode
  show       Show partial RPL configuration
```

オプション	説明
#-remark	「#」ではじまる注記
*	ワイルドカード（任意のコミュニティまたはその一部）
<1-65535>	16 ビットの 10 進数
AS-TRANS	ASN23456（予約済み ASN）
ASN:N	拡張コミュニティ：ASPLAIN 形式。ここで、ASN は AS の番号で、N は帯域幅の値です。
abort	RPL 定義を廃棄して、top level config に戻る

オプション	説明
end-set	セット定義の終了
exit	このサブモードを終了
show	一部の RPL 設定を表示

prefix-set

prefix-set は、それぞれ 4 つの部分（アドレス、マスク長、最小一致長、最大一致長）がある IPv4 または IPv6 プレフィックス一致指定を保持しています。アドレスは必須ですが、他の 3 つの部分は任意です。アドレスは、標準のドット付き IPv4 またはコロンで区切られた 16 進数の IPv6 アドレスです。マスク長（存在する場合）は、0 ～ 32（IPv6 の場合は 0 ～ 128）の範囲内の負以外の 10 進整数で、その前のアドレスはスラッシュで区切ります。アドレスと任意のマスク長の後には、任意の最小一致長が続き、これはキーワード **ge**（以上（**g**reater than or **e**qual to）のニーモニック）で表され、その後に 0 ～ 32（IPv6 の場合は 0 ～ 128）の範囲内の負以外の 10 進整数が続きます。最後には、任意の最大一致長が続き、これはキーワード **le**（以下（**l**ess than or **e**qual to）のニーモニック）で表され、その後に 0 ～ 32（IPv6 の場合は 0 ～ 128）の範囲内の負以外の別の 10 進整数が続きます。一致させるプレフィックスの正確な長さを指定するための構文ショートカットは、**eq** キーワード（等しい（**eq**ual to）のニーモニック）です。

プレフィックス一致指定にマスク長がない場合は、デフォルトのマスク長は、IPv4 では 32、IPv6 では 128 です。デフォルトの最小マッチング長はマスク長です。最小一致長が指定されている場合、デフォルトの最大一致長は、IPv4 では 32、IPv6 では 128 です。最小と最大のいずれも指定しない場合は、デフォルトの最大長はマスク長になります。

prefix-set 自体は、プレフィックス一致指定のカンマ区切りのリストです。次に例を示します。

```
prefix-set legal-ipv4-prefix-examples
   10.0.1.1,
   10.0.2.0/24,
   10.0.3.0/24 ge 28,
   10.0.4.0/24 le 28,
   10.0.5.0/24 ge 26 le 30,
   10.0.6.0/24 eq 28,
   10.0.7.2/32 ge 16 le 24,
   10.0.8.0/26 ge 8 le 16
end-set

prefix-set legal-ipv6-prefix-examples
  2001:0:0:1::/64,
  2001:0:0:2::/64 ge 96,
  2001:0:0:2::/64 ge 96 le 100,
  2001:0:0:2::/64 eq 100
end-set
```

prefix-set の最初の要素は、唯一の有効値 10.0.1.1/32 またはホスト アドレス 10.0.1.1 と一致します。2 番めの要素は、唯一の有効値 10.0.2.0/24 と一致します。3 番めの要素は、10.0.3.0/28 ～ 10.0.3.255/32 の範囲のプレフィックス値と一致します。4 番めの要素は、10.0.4.0/24 ～ 10.0.4.240/28 の範囲の値と一致します。5 番めの要素は、10.0.5.0/26 ～ 10.0.5.252/30 の範囲内

のプレフィックスと一致します。6 番めの要素は、10.0.6.0/28 ～ 10.0.6.240/28 の範囲内にある長さ 28 の任意のプレフィックスと一致します。7 番目の要素は、10.0.[0..255].2/32（10.0.0.2/32 ～ 10.0.255.2）の範囲内にある長さ 32 の任意のプレフィックスと一致します。8 番目の要素は、10.[0..255].8.0/26（10.0.8.0/26 ～ 10.255.8.0/26）の範囲内にある長さ 26 の任意のプレフィックスと一致します。

次の prefix-set はすべて、無効なプレフィックス一致指定からなります。

```
prefix-set ILLEGAL-PREFIX-EXAMPLES
   10.1.1.1 ge 16,
   10.1.2.1 le 16,
   10.1.3.0/24 le 23,
   10.1.4.0/24 ge 33,
   10.1.5.0/25 ge 29 le 28
end-set
```

最小長と最大長のいずれも、マスク長がないと無効です。IPv4 の場合、最小長は、IPv4 プレフィックスの最大長である 32 未満でなければなりません。IPv6 の場合、最小長は、IPv6 プレフィックスの最大長である 128 未満でなければなりません。最大長は、最小長以上でなければなりません。

拡張プレフィックス長操作

prefix-set における拡張プレフィックス長操作のサポートは、プレフィックス一致指定での **ge** セマンティックを使用する prefix-range を拡張します。これは、プレフィックス 0.0.0.0/0, 0.0.0.0/1, 0.0.0.0/2, ..., 0.0.0.0/32 と一致する単一のエントリを保持する要求に応えます。prefix-length は、**ge** セマンティックを使用して操作できます。たとえば、prefix-set（0.0.0.0/30 ge 0 le 32）は、0.0.0.0/0 ～ 0.0.0.3/32 の範囲内のすべてのプレフィックスと一致します。これにより、単一の prefix-set エントリ 0.0.0.0/32 ge 0 le 32 は、プレフィックス 0.0.0.0/0, 0.0.0.0/1, 0.0.0.0/2, ..., 0.0.0.0/32 と一致します。

次に、IPv4 プレフィックス構文と対応するマスク長範囲を含むプレフィックス範囲を示します。

- <A.B.C.D>/<len> ge <G> le <L>
  - <A.B.C.D>/[<len>..<G>] (if <len> is lesser than <G>)
  - <A.B.C.D>/[<G>..<len>] (if <len> is greater than <G>)
- <A.B.C.D>/<len> ge <G>
  - <A.B.C.D>/[<len>..<G>] (if <len> is lesser than <G>)
  - <A.B.C.D>/[<G>..<len>] (if <len> is greater than <G>)
- <A.B.C.D>/<len> eq <E>
  - <A.B.C.D>/[<len>..<E>] (if <len> is lesser than <E>)
  - <A.B.C.D>/[<E>..<len>] (if <len> is greater than <E>)

RPL プレフィックス セットでの *ACL* サポート

アクセス コントロール リスト（ACL）タイプのプレフィックス セット エントリは、IPv4 または IPv6 のプレフィックス一致指定を保持し、それぞれにアドレスとワイルドカード マスクがあります。アドレスおよびワイルドカードマスクは、標準のドット付き 10 進数表記の IPv4 アドレスまたはコロンで区切られた 16 進数の IPv6 アドレスです。照合するビット セットはワイルドカードの形式（反転マスクとも呼ばれる）で提供され、バイナリ 0 は必須一致を、バイナリ 1 は一致しない条件をそれぞれ表します。プレフィックス セットを使用して、任意のルートで一致する必要がある連続したビット セットと非連続のビット セットを指定できます。

rd-set

rd-set は、ルート識別子（RD）要素を含むセットを作成するために使用します。RD セットは、固有のボーダー ゲートウェイ プロトコル（BGP）VPN IPv4 アドレスをグローバルに作成するために、IPv4 アドレスが前に付いた 64 ビット値です。

RD 値は、次のコマンドを使用して定義できます。

- *a.b.c.d:m:**：IPv4 形式の BGP VPN RD とワイルドカード文字。たとえば、10.0.0.2:255.255.0.0:* です。
- *a.b.c.d/m:n*：IPv4 形式の BGP VPN RD とマスク。たとえば、10.0.0.2:255.255.0.0:666 です。
- *a.b.c.d:***：IPv4 形式の BGP VPN RD とワイルドカード文字。たとえば、10.0.0.2:255.255.0.0 です。
- *a.b.c.d:n*：IPv4 形式の BGP VPN RD。たとえば、10.0.0.2:666 です。
- *asn:**：ASN 形式の BGP VPN RD とワイルドカード文字。たとえば、10002:255.255.0.0 です。
- *asn:n*：ASN 形式の BGP VPN RD。たとえば、10002:666 です。

次に、rd-set の例を示します。

```
rd-set rdset1
   10.0.0.0/8:*,
   10.0.0.0/8:777,
   10.0.0.0:*,
   10.0.0.0:777,
   65000:*,
   65000:777
end-set
```

ルーティング ポリシー言語コンポーネント

ルーティング ポリシー言語の 4 種類の主要コンポーネント、設定フロント エンド、ポリシー リポジトリ、実行エンジン、およびポリシー クライアントそのものは、ポリシーの定義、変更、および使用に関係します。

設定フロントエンド（CLI）は、ポリシーを定義し、変更する機能です。この設定は、通常の保存方法を使用してルータに保存され、通常の設定の **show** コマンドを使用して表示できます。

ポリシー インフラストラクチャの 2 番目のコンポーネントである、ポリシー リポジトリには複数の役割があります。最初に、ユーザ入力設定を実行エンジンが認識可能な形式にコンパイルします。2 番目に、ポリシー検証の多くを実行し、定義されたポリシーが実際に適切に実行できることを確認します。3 番目に、いずれの接続点がいずれのポリシーを使用しているかを追跡して、ポリシーが変更された場合に適切なクライアントが適切な新しいポリシーで正常にアップデートされるようにします。

3 番目のコンポーネントは実行エンジンです。このコンポーネントは、クライアントの要求に応じて実際にポリシーを実行する部分です。このプロセスは、ポリシークライアントのいずれかからルートを受信して、特定のルートデータに対して実際のポリシーを実行するまでと見なせます。

4 つめのコンポーネントはポリシークライアント（ルーティングプロトコル）です。このコンポーネントは、適切なタイミングで実行エンジンをコールし、特定のポリシーを特定のルートに適用し、次にいくつかのアクションを実行します。これらのアクションには、ルートをドロップする必要があるとポリシーに示されている場合にルートを削除する、最適なルートの候補としてプロトコル決定ツリーにルートを渡す、またはポリシーに変更されたルートを必要に応じてネイバーまたはピアにアドバタイズすることが含まれます。

ルーティング ポリシー言語使用方法

ここでは、基本的なルーティングポリシー言語の使用方法の例について説明します。ルーティングポリシー言語の実装方法の詳細については、ルーティングポリシーの実装方法（633 ページ）を参照してください。

パス ポリシー パス ポリシー

次に、ルートを変更せずにポリシーがすべての渡されたルートを受け入れる例を示します。

```
route-policy quickstart-pass
pass
end-policy
```

すべてをドロップするポリシー

次に、渡されたすべてのルートをポリシーが明示的に拒否する例を示します。このタイプのポリシーは、特定のピアから送信されるすべてを無視するために使用されます。

```
route-policy quickstart-drop
drop
end-policy
```

パスの特定の AS 番号を使用するルートを無視する

次に、3 個の部分からなるポリシー定義の例を示します。まず、as-path-set コマンドは、AS パスとマッチングするための 3 つの正規表現を定義します。2 番目に、route-policy コマンドは、ルートに AS パス セットを適用します。ルートの AS パス属性が as-path-set コマンドで定義された正規表現と一致する場合、プロトコルはこのルートを拒否します。3 番めに、ルート ポリ

シーは、BGP ネイバー 10.0.1.2 に付加されます。BGP は、ネイバー 10.0.1.2 から（インポート）受信したルートの ignore_path_as という名前のポリシーを参照します。

```
as-path-set ignore_path
ios-regex '_11_',
ios-regex '_22_',
ios-regex '_33_'
end-set

route-policy ignore_path_as
if as-path in ignore_path then
drop
else
pass
endif
end-policy

router bgp 2
neighbor 10.0.1.2 address-family ipv4 unicast policy ignore_path_as in
```

MED に基づくセット コミュニティ

次に、ポリシーがルートのMEDをテストし、MEDの値に基づいてルートのコミュニティ属性を変更する例を示します。MED 値が 127 の場合、ポリシーはルートにコミュニティ 123:456 を追加します。MED 値が 63 の場合、ポリシーはルートのコミュニティ属性に値 123:789 を追加します。それ以外の場合、ポリシーはコミュニティ 123:123 をルートから削除します。いかなる場合でも、ルートを受け入れるようにポリシーからプロトコルに指示します。

```
route-policy quickstart-med
if med eq 127 then
set community (123:456) additive
elseif med eq 63 then
set community (123:789) additive
else
delete community in (123:123)
endif
pass
end-policy
```

コミュニティに基づくローカル プリファレンスの設定

次に、quickstart-communities という名前のコミュニティ セットがコミュニティ値を定義する例を示します。quickstart-localpref という名前のルート ポリシーは、quickstart-communities コミュニティ セットに指定されたコミュニティの存在に対してルートをテストします。ルートにいずれかのコミュニティ値が存在する場合、ルート ポリシーはルートのローカル プリファレンス属性を 31 に設定します。いかなる場合でも、ルートを受け入れるようにポリシーからプロトコルに指示します。

```
community-set quickstart-communities
987:654,
987:543,
987:321,
987:210
end-set
```

```
route-policy quickstart-localpref
if community matches-any quickstart-communities then
set local-preference 31
endif
pass
end-policy
```

持続的な注記

次に、ポリシーのセットおよびステートメントのエントリの意味を明確にするために、ポリシーにコメントを配置する例を示します。注記は持続的なため、ポリシーに付加されたままです。たとえば、show running-config コマンドの出力には注記が表示されます。ポリシーに注記を追加すると、ポリシーの理解、また後日の変更が容易になるとともに、予期しない動作が発生した場合にトラブルシューティングが容易になります。

```
prefix-set rfc1918
# These are the networks defined as private in RFC1918 (including
# all subnets thereof)
10.0.0.0/8 ge 8,
172.16.0.0/12 ge 12,
192.168.0.0/16 ge 16
end-set

route-policy quickstart-remarks
# Handle routes to RFC1918 networks
if destination in rfc1918 then
# Set the community such that we do not export the route
set community (no-export) additive

endif
end-policy
```

ルーティング ポリシーの設定の基本

ルートポリシーは、**route-policy** キーワードと **end-policy** キーワードで囲まれた一連のステートメントと式によって構成されます。個別のコマンド（1 行に 1 つのコマンド）の集合ではなく、ルートポリシー内のステートメントには相互に関連するコンテキストがあります。そのため、個別のコマンドを各行に記すのではなく、各ポリシーまたはセットは独立した設定オブジェクトとして、1 つのユニットとして使用、入力、操作できます。

ポリシー設定の各行は論理サブユニットです。 **then** 、 **else** 、 **end-policy** キーワードの後ろには、少なくとも 1 つの新しい行を続ける必要があります。AS パス セット、コミュニティ セット、拡張コミュニティ セット、またはプレフィックス セットを参照するパラメータ リストと名前ストリングを閉じる括弧の後には改行が必要です。ルート ポリシー、AS パス セット、コミュニティ セット、拡張コミュニティ セット、またはプレフィックス セットの定義の前には、少なくとも新しい行が 1 行必要です。アクション ステートメントの後ろには 1 行以上の新しい行を続けることができます。名前付き AS パス セット、コミュニティ セット、拡張コミュニティ セット、プレフィックス セットのカンマ区切りの後ろには 1 行以上の新しい行を続けることができます。新しい行はポリシー式の論理ユニットの最後に記される必要があります。他の場所に記すことはできません。

ポリシー定義

ポリシー定義によって、ポリシー ステートメントの名前付きシーケンスが作成されます。ポリシー定義は、CLI の **route-policy** キーワードと、その後に続く名前、ポリシー ステートメントのシーケンス、および **end-policy** キーワードで構成されます。たとえば、次のポリシーは検出されたルートをすべてドロップします。

```
route-policy drop-everything
drop
end-policy
```

名前は、ポリシーをプロトコルにバインドするためのハンドルとして機能します。ポリシー定義を削除するには、no route-policy name コマンドを発行します。

ポリシーの共通ブロックを再利用できるように、ポリシーは他のポリシーを参照していることもあります。他のポリシーの参照は、**apply** ステートメントを使用して、次の例のように実行されます。

```
route-policy check-as-1234
if as-path passes-through ‘1234.5’ then
apply drop-everything
else
pass
endif
end-policy
```

apply ステートメントは、検討中のルートが受信前に自律システム 1234.5 を介してパススルーされた場合に、ポリシー drop-everything を実行する必要があることを示しています。AS パスに自律システム 1234.5 があるルートが受信された場合、ルートはドロップされます。それ以外の場合、ルートは変更なしで受け入れられます。このポリシーは、階層型ポリシーの例です。つまり、**apply** ステートメントのセマンティックは、適用されるポリシーを切り取って適用するポリシーに貼り付けた場合と同様です。

```
route-policy check-as-1234-prime
 if as-path passes-through '1234.5' then
    drop
 else
    pass
 endif
end-policy
```

必要に応じて階層レベルはいくつでも使用できます。しかし、レベルを多くすると、維持や理解が困難になることがあります。

パラメータ化

apply ステートメントを使用したポリシーの再利用サポートに加えて、属性の一部のパラメータ化ができるようにポリシーを定義できます。次に、param-example という名前のパラメータ化ポリシーを定義する例を示します。この場合、ポリシーは $mytag という 1 つのパラメータを取ります。パラメータは、常にドル記号ではじまり、任意の英数字で構成されます。パラメータは、パラメータを取る属性に置き換えられます。

次の例では、16 ビット コミュニティ タグがパラメータとして使用されています。

```
route-policy param-example ($mytag)
set community (1234:$mytag) additive
end-policy
```

その後、このパラメータ化ポリシーは、次の例に示すように、別のパラメータ化で再利用できます。この方法で、共通の構造を共有するが、一部の個別のステートメントで別の値を使用するポリシーをモジュール化できます。パラメータ化できる属性の詳細については、個別の属性についての項を参照してください。

```
route-policy origin-10
if as-path originates-from ‘10.5’ then
apply param-example(10.5)
else
pass
endif
end-policy

route-policy origin-20
if as-path originates-from ‘20.5’ then
apply param-example(20.5)
else
pass
endif
end-policy
```

パラメータ化ポリシー param-example は、apply ステートメントのパラメータとして提供されている値で拡張されたポリシー定義を提供します。ポリシー階層が常に維持されるため、param-example の定義が変更されると、origin_10 および origin_20 の動作が一致するように変更されることに注意してください。

origin-10 ポリシーの効果として、このポリシーをパススルーし、自律システム 10 から発信されたルートを示す AS パスを持つすべてのルートにコミュニティ 1234:10 を追加します。origin-20 ポリシーは同様ですが、自律システム 20 から発信されたルートに対してコミュニティ 1234:20 を追加します。

接続点でのパラメータ化

パラメータ化 （563 ページ）で説明したように apply ステートメントを使用したパラメータ化のサポートに加えて、ポリシーは接続点での属性のパラメータ化ができるようにも定義できます。すべての接続点で、パラメータ化がサポートされます。

次の例では、パラメータ化ポリシー「param-example」を定義します。この例では、ポリシーは、「$mymed」と「$prefixset」の 2 つのパラメータを取ります。パラメータは、常にドル記号ではじまり、任意の英数字で構成されます。パラメータは、パラメータを取る属性に置き換えられます。この例では、MED 値およびプレフィックス セット名をパラメータとして渡しています。

```
route-policy param-example ($mymed, $prefixset)
  if destination in $prefixset then
  set med $mymed
  endif
end-policy
```

その後、このパラメータ化ポリシーは、次の例に示すように、別のパラメータ化で再利用できます。この方法で、共通の構造を共有するが、一部の個別のステートメントで別の値を使用するポリシーをモジュール化できます。パラメータ化できる属性の詳細については、各プロトコルの個別の属性を参照してください。

```
router bgp 2
  neighbor 10.1.1.1
    remote-as 3
    address-family ipv4 unicast
      route-policy param-example(10, prefix_set1)
      route-policy param-example(20, prefix_set2)
```

パラメータ化ポリシー param-example は、neighbor route-policy in and out ステートメントのパラメータとして提供されている値で拡張されたポリシー定義を提供します。

グローバル パラメータ化

RPL では、ポリシー定義内で使用できるシステム全体のグローバル パラメータの定義がサポートされます。グローバル パラメータは、次のように設定できます。

```
Policy-global
  glbpathtype ‘ebgp’
  glbtag ‘100’
end-global
```

グローバル パラメータ値は、パラメータ化ポリシーのローカル パラメータと類似したポリシー定義内で直接使用できます。次の例では、*globalparam* 引数は、グローバル パラメータ gbpathtype と glbtag を使用し、非パラメータ化ポリシーに対して定義されます。

```
route-policy globalparam
  if path-type is $glbpathtype then
    set tag $glbtag
  endif
end-policy
```

パラメータ化ポリシーのパラメータ名に、グローバル パラメータ名との「衝突」がある場合は、ポリシー定義に対してローカルなパラメータが優先され、グローバルパラメータが効果的に隠されます。さらに、特定のグローバルパラメータが任意のポリシーによって参照されている場合は、削除されないように、検証メカニズムが実施されます。

ポリシー適用のセマンティック

ここでは、ルーティングポリシーの評価および適用方法について説明します。説明する概念は次のとおりです。

ブール演算子優先

ブール式は、演算子優先順位に従って、左から右に評価されます。最も優先される演算子は NOT であり、その後に AND、次に OR と続きます。次に、式の例を示します。

```
med eq 10 and not destination in (10.1.3.0/24) or community matches-any ([10..25]:35)
```

評価の順を表示するために、すべてが括弧で囲まれる場合は、次のようになります。

```
(med eq 10 and (not destination in (10.1.3.0/24))) or community matches-any ([10..25]:35)
```

内部の NOT は、宛先のテストのみに適用されます。AND は、NOT 式の結果を Multi Exit Discriminator（MED）テストと組み合わせます。さらに、OR は、その結果をコミュニティ テストと組み合わせます。演算の順は配列し直される場合があります。

```
not med eq 10 and destination in (10.1.3.0/24) or community matches-any ([10..25]:35)
```

その場合、すべてが括弧で囲まれた式は、次のようになります。

```
((not med eq 10) and destination in (10.1.3.0/24)) or community matches-any ([10..25]:35)
```

同じ属性の複数の変更

ポリシーが属性の値を複数回置換する場合、すべてのアクションが実行されるため、最後の割り当てが優先されます。BGP の MED 属性は、1 つの固有値であるため、それに設定された最後の値が優先されます。つまり、次のポリシーでは、ルートの MED 値は 12 になります。

```
set med 9
set med 10
set med 11
set med 12
```

この例は単純ですが、機能の場合は複雑です。属性の値を効率的に変更するポリシーを記述することも可能です。次に例を示します。

```
set med 8
if community matches-any cs1 then
set local-preference 122
if community matches-any cs2 then
set med 12
endif
endif
```

結果として、ルートの MED は 8 になります。ただし、ルートのコミュニティ リストが cs1 と cs2 の両方と一致する場合は、結果として、ルートの MED は 12 になります。

変更している属性に1つの値だけが含められる場合は、最後のステートメントが優先されるため、この例を理解するのは簡単です。一方、複数の値を含められる属性もいくつかあり、このような属性における複数のアクションの結果は、置換ではなく累積します。最初の例として、コミュニティおよび拡張コミュニティ評価で **additive** キーワードを使用する場合があります。形式のポリシーを検討します。

```
route-policy community-add
set community (10:23)
set community (10:24) additive
set community (10:25) additive
end-policy
```

このポリシーは、10:23、10:24、および 10:25 の 3 つのコミュニティ値すべてを含めるためにルートにコミュニティ ストリングを設定します。

2番目の例として、ASパスの先頭に追加する場合があります。形式のポリシーを検討します。

```
route-policy prepend-example
prepend as-path 2.5 3
prepend as-path 666.5 2
end-policy
```

このポリシーは、AS パスの先頭に 666.5 666.5 2.5 2.5 2.5 を追加します。この先頭に追加する動作は、実行するすべてのアクションの結果であり、単一のスカラー値ではなく、値の配列を含む属性となる AS パスに対して実行されます。

属性を変更するとき

ポリシーは、すべてのテストが完了するまでルート属性値を変更しません。つまり、比較演算子はルートの初期データで常に実行されます。ルート属性の中間での変更は、ポリシーの評価にカスケード効果を与えません。次に、例を示します。

```
ifmed eq 12 then
set med 42
if med eq 42 then
```

```
drop
endif
endif
```

このポリシーは、drop ステートメントを一切実行しません。これは、2 番目のテスト（med eq 42）がルートの元の変更されていない MED 値を見るためです。2 番目のテストに到達するには、MED が 12 である必要があるため、2 番目のテストは常に false を返します。

デフォルトのドロップ処理

すべてのルート ポリシーには、評価中のルートをドロップするデフォルト アクションがあります。ただし、ルートがポリシーアクションによって変更された場合と明示的に渡された場合は除きます。適用（ネスト）されるポリシーは、適用されるポイントに適用されるポリシーを貼り付けることによってこの処理を実装します。

ネットワーク 10 のすべてのルートを許可し、そのローカルプリファレンスを 200 に設定して、他のルートすべてをドロップするポリシーについて検討します。ポリシーは次のように記述できます。

```
route-policy two
if destination in (10.0.0.0/8 ge 8 le 32) then
set local-preference 200
endif
end-policy

route-policy one
apply two
end-policy
```

明示的な **pass** ステートメントが含まれていなく、ルート属性も変更しないため、ポリシー one はすべてのルートをドロップするように見える可能性があります。しかし、適用されるポリシーは実際は一部のルートに属性を設定し、この処理はポリシー one に渡されます。結果として、ポリシー one はネットワーク 10 の宛先を含むルートを渡して、その他すべてをドロップします。

制御フロー

ポリシー ステートメントは、設定に表示される順に従って順番に処理されます。他のポリシー ブロックを階層的に参照するポリシーは、参照されるポリシー ブロックが直接インラインに置換されたように処理されます。たとえば、次のポリシーが定義されているとします。

```
route-policy one
set weight 100
end-policy

route-policy two
set med 200
end-policy

route-policy three
apply two
```

```
set community (2:666) additive
end-policy

route-policy four
apply one
apply three
pass
end-policy
```

ポリシー four は次と同様の方法で書き換えられます。

```
route-policy four-equivalent
set weight 100
set med 200
set community (2:666) additive
pass
end-policy
```

（注） **pass** ステートメントは必要ないため削除して、別の方法で同様のポリシーを表せます。

ポリシー検証

ポリシーが定義されて使用される際には、いくつかの異なるタイプの検証が発生します。

範囲チェック

ポリシーが定義されるとき、値の範囲チェックなどのいくつかの簡単な検証が行われます。たとえば、設定される MED が MED 属性の適切な範囲内にあることを検証するためにチェックされます。しかし、この範囲チェックはパラメータ指定を対象にできません。これは、パラメータ指定が値をまだ定義していない可能性があるためです。パラメータ指定は、ポリシーが接続点に付加されたときに検証されます。ポリシーリポジトリでも、ポリシーの再帰定義がなく、パラメータの数値が正しいことが検証されます。付加時には、すべてのポリシーが正しく形成されている必要があります。参照されるすべてのセットおよびポリシーが定義されていて、有効値を持っている必要があります。同様に、パラメータ値もすべて適切な範囲内にある必要があります。

不完全なポリシーとセットの参照

指定のポリシーが接続点に付加されていないかぎり、ポリシーは存在しないセットおよびポリシーを参照できます。これにより、ワークフローが自由になります。まだ定義されていないセットまたはポリシーブロックを参照する設定を構築して、後からこれらの未定義のポリシーおよびセットに入力できます。これにより、ポリシー定義でのより高い柔軟性を実現します。参照するポリシーの各部分は、ポリシーを定義しているときに設定に存在する必要はありません。つまり、ユーザはポリシーバーが存在しない場合でも、**apply** ステートメントを使用してポリシーバーを参照するポリシー例を定義できます。同様に、ユーザは、存在しないセットを参照するポリシー ステートメントを入力できます。

ただし、参照されているすべてのポリシーとセットの存在は、ポリシーが付加されたときに適用されます。neighbor 1.2.3.4 address-family ipv4 unicast policy sample in コマンドを使用してインバウンド BGP ポリシーで未定義ポリシー バーを参照するポリシー サンプルをアタッチしようとすると、ポリシー バーが存在しないために設定が拒否されます。

同様に、接続点で現在使用中のルートポリシーまたはセットは削除できません。これは、削除の結果、未定義参照が発生するためです。現在使用中のルートポリシーまたはセットを削除しようとすると、ユーザに対してエラー メッセージが表示されます。

ポリシーバーが存在しているものの、ステートメント、アクション、ディスポジションが存在しないヌル ポリシーと呼ばれる条件が存在します。つまり、ポリシー バーは次の場合に存在します。

```
route-policy bar
end-policy
```

これは、有効なポリシーブロックです。これは、ルートを一切変更せず、pass ステートメントも含まれていないポリシーブロックであるため、すべてのルートのドロップを実質的に強制します。したがって、ポリシー ブロックのドロップのデフォルト アクションが実行されます。

アタッチされたポリシーの変更

使用中のポリシーを変更する必要が生じる場合もあります。従来、設定変更は完全に関連する設定を削除し、設定を再入力して行われます。しかし、これによりポリシーが付加されていないためにデフォルト アクションが実行される期間ができます。RPL では、ポリシーが再び宣言された場合、またはテキストエディタを使用して編集された場合、新しい設定がただちに適応される、アトミック変更のメカニズムがあります。これによって、ポリシーが特定の接続点に適用されない期間が発生することなく、使用中のポリシーを変更できます。

属性比較とアクションの検証

ポリシー リポジトリは、各接続点で有効な属性、アクションおよび比較を認識しています。ポリシーが付加されたとき、これらのアクションおよび比較はその特定の接続点の機能に対して検証されます。例として、次のポリシー定義があります。

```
route-policy bad
set med 100
set level level-1-2
set ospf-metric 200
end-policy
```

このポリシーは、BGP 属性 med、IS-IS 属性レベル、および OSPF 属性コストを設定するアクションの実行を試行します。システムは、このようなポリシーの定義を許可しますが、このようなポリシーの付加は許可しません。このポリシー bad を定義し、BGP コンフィギュレーション ステートメント **neighbor 1.2.3.4 address-family ipv4 unicast route-policy bad in** を使用してインバウンド BGP ポリシーとして接続しようとする場合、システムはこの設定の試行を拒否します。この拒否は、ポリシーをチェックする検証プロセスの結果であり、BGP は MED を設定できるが、レベルおよびコストがそれぞれ IS-IS および OSPF 属性であるため、レベルまたは

コストを設定する方法がありません。実行できないアクションを黙示的に省く代わりに、システムはユーザに対してエラーを生成します。同様に、存在しない属性を変更、または存在しない属性との比較の試行を発生させる場合のように、接続点で使用中の有効なポリシーは変更できません。ベリファイアは、存在しない属性があるかをテストして、このような設定の試行を拒否します。

ポリシー ステートメント

ポリシー ステートメントには、注記、処理（drop および pass）、アクション（set）、および if（比較演算子）の 4 つのタイプがあります。

注記

注記は、ポリシー設定に添付されているテキストですが、それ以外の点ではポリシー言語パーサーによって無視されるテキストです。注記は、備考はポリシーの一部を記述するのに役立ちます。注記の構文は、各行の先頭にポンド記号（#）があるテキストです。

```
# This is a simple one-line remark.

# This
# is a remark
# comprising multiple
# lines.
```

通常、注記はセットの完全なステートメントまたは要素の間で使用されます。ステートメントの途中、またはインライン セット定義内での注記はサポートされていません。

CLI における従来の！コメントとは異なり、RPL の注記は、リブートしても持続し、また設定がディスクまたは TFTP サーバに保存されてからルータにロードされても持続します。

処理

ポリシーがルートを変更した場合、デフォルトではポリシーは、そのルートを受け入れます。RPL には、その反対を強制する **drop** ステートメントがあります。ポリシーがルートに一致してドロップを実行する場合、ポリシーはルートを受け入れません。ポリシーがルートを変更しない場合、デフォルトでそのルートはドロップされます。ルートのドロップを防止するには、**pass** ステートメントを使用します。

drop ステートメントは、ルートを破棄するアクションを実行するように指示します。ルートがドロップされると、ポリシーはこれ以上実行されません。たとえば、ポリシーの最初の2つのステートメントを実行した後で、**drop** ステートメントが検出されると、ポリシーは停止してルートが破棄されます。

（注） すべてのポリシーにおいて、実行の最後にはデフォルトの **drop** アクションがあります。

pass ステートメントを使用すると、ルートが変更されなかった場合でもポリシーは実行を継続できます。ポリシーの実行が終了すると、ポリシーで変更されたか、ポリシーで pass 処理を受信したルートはすべてポリシーを渡し、実行は完了します。ルート ポリシー B_rp がルート ポリシー A_rp 内に適用されたとき、プレフィックスがポリシー B_rp によってドロップされない場合は、実行はポリシー A_rp からポリシー B_rp へと続けられてから、ポリシー A_rp に戻ります。

```
route-policy A_rp
  set community (10:10)
  apply B_rp
end-policy
!

route-policy B_rp
  if destination in (121.23.0.0/16 le 32, 155.12.0.0/16 le 32) then
  set community (121:155) additive
  endif
end-policy
!
```

ポリシーがルート属性を**変更する**か、ポリシーが明示的な **pass** ステートメントによってルートを渡さないかぎり、デフォルトでルートはポリシー処理の最後に**ドロップされます**。たとえば、ルート ポリシー B がルート ポリシー A 内に適用されたとき、プレフィックスがポリシー B によってドロップされない場合は、実行はポリシー A からポリシー B へと続けられてから、ポリシー A に戻ります。

```
route-policy A
   if as-path neighbor-is '123' then
   apply B
   policy statement N
end-policy
```

一方で、次のポリシーは評価するすべてのルートを渡します。

```
route-policy PASS-ALL
pass
end-policy


route-policy SET-LPREF
set local-preference 200
end-policy
```

黙示的にドロップされることに加えて、ルートは**明示的な drop** ステートメントによってドロップされることもあります。**drop** ステートメントによってルートがすぐにドロップされるため、ポリシー処理はこれ以上実行されません。また、**drop** ステートメントは、それ以前に処理された **pass** ステートメントまたは属性変更をすべて上書きすることにも注意してください。たとえば、次のポリシーはすべてのルートをドロップします。最初の **pass** ステートメントは実行されますが、すぐに **drop** ステートメントによって上書きされます。2 番目の **pass** ステートメントが実行されることは決してありません。

```
route-policy DROP-EXAMPLE
pass
drop
pass
end-policy
```

1 つのポリシーが別のポリシーを適用する場合、適用されるポリシーは適用するポリシーの正しい位置にコピーされたようになり、次に、同じ drop-and-pass セマンティックが施行されます。たとえば、次のポリシー ONE および TWO は、ポリシー ONE-PRIME と同等です。

```
route-policy ONE
apply two
if as-path neighbor-is '123' then
pass
endif
end-policy

route-policy TWO
if destination in (10.0.0.0/16 le 32) then
drop
endif
end-policy

route-policy ONE-PRIME
if destination in (10.0.0.0/16 le 32) then
drop
endif
if as-path neighbor-is '123' then
pass
endif
end-policy
```

明示的な drop ステートメントの効果が即時であるため、10.0.0.0/16 le 32 内のルートは、これ以上のポリシー処理なしでドロップされます。次に、その他のルートが自律システム 123 によってアドバタイズされているかどうかを確認するために検討されます。アドバタイズされている場合、それらのルートは渡されます。それ以外の場合は、すべてのポリシー処理の最後に黙示的にドロップされます。

done ステートメントは、ポリシーの実行を停止してルートを受け入れるアクションを実行するように指示します。**done** ステートメントを検出すると、ルートが渡され、ポリシー ステートメントはこれ以上実行されません。**done** ステートメントの前にルートに対して行った変更はすべて、有効なままです。

アクション

アクションとは、ルートを変更する基本操作のシーケンスです。すべてではありませんが、大部分のアクションは **set** キーワードによって区別されます。ルート ポリシーでは、アクションはグループ化できます。次に、3 つのアクションから構成される 1 つのルート ポリシーの例を示します。

```
route-policy actions
set med 217
```

```
set community (12:34) additive
delete community in (12:56)
end-policy
```

If

最も単純な形式では、**if** ステートメントは、条件式を使用して、指定のルートで行う必要があるアクションまたは処理を決定します。次に例を示します。

```
if as-path in as-path-set-1 then
drop
endif
```

この例では、AS パスがセット as-path-set-1 にあるすべてのルートがドロップされることを示しています。**then** 句の内容には、ポリシー ステートメントの任意のシーケンスが指定できます。

次の例では、2 つのアクション ステートメントが含まれています。

```
if origin is igp then
set med 42
prepend as-path 73.5 5
endif
```

CLI では、endif コマンドに代わる、exit コマンドがサポートされています。

if ステートメントでは、if 条件が false の場合に実行される **else** 句も使用できます。

```
if med eq 8 then
set community (12:34) additive
else
set community (12:56) additive
endif
```

ポリシー言語では、テストのシーケンスをつなぎ合わせるために、**elseif** キーワードを使用した構文も用意されています。

```
if med eq 150 then
set local-preference 10
elseif med eq 200 then
set local-preference 60
elseif med eq 250 then
set local-preference 110
else
set local-preference 0
endif
```

次の例に示すように、**if** ステートメント内のステートメント自体が **if** ステートメントになることがあります。

```
if community matches-any (12:34,56:78) then
if med eq 150 then
drop
endif
set local-preference 100
endif
```

このポリシーの例では、コミュニティ値 12:34 または 56:78 が関連付けられたすべてのルートで、ローカルプリファレンス属性の値が 100 に設定されます。ただし、これらのルートのいずれかで MED 値が 150 になっている場合は、コミュニティ値 12:34 または 56:78 のいずれかおよび MED 150 が指定されたこれらのルートはドロップされます。

ブール条件

if ステートメントについて説明した前の項では、すべての例で true または false を評価する単純なブール条件を使用しています。RPL には、ブール演算子を使用して、単純条件から複合条件を構築する方法もあります。

ブール演算子には、否定（**not**）、論理積（**and**）、および論理和（**or**）の 3 つがあります。ポリシー言語では、否定が最も優先され、その後に論理積、次に論理和と続きます。優先を上書きする、または可読性を高める目的で複合条件をグループ化するために括弧を使用できます。

次に、単純条件の例を示します。

```
med eq 42
```

ルートの MED の値が 42 の場合は true、それ以外の場合は false です。

単純条件は、**not** 演算子を使用しても否定できます。

```
not next-hop in (10.0.2.2)
```

括弧で囲まれているブール条件は、それ自体がブール条件です。

```
(destination in prefix-list-1)
```

複合条件は次の 2 つの形式のいずれかになります。複合条件は、単純式の後に **and** 演算子が続くか、それ自体の後に単純条件が続きます。

```
med eq 42 and next-hop in (10.0.2.2)
```

複合条件は、単純式の後に **or** 演算子、またその後に別の単純条件が続くこともできます。

```
origin is igp or origin is incomplete
```

複合条件全体を括弧で囲むこともできます。

```
(med eq 42 and next-hop in (10.0.2.2))
```

括弧は、サブ条件のグループの可読性を高めるために使用される場合、またはサブ条件を1つのユニットとして評価するように強制する場合があります。

次の例では、最優先の **not** 演算子は、宛先のテストのみに適用されます。**and** 演算子は、**not** 式の結果をコミュニティテストと組み合わせます。また、**or** 演算子は、その結果をMEDテストと組み合わせます。

```
med eq 10 or not destination in (10.1.3.0/24) and community matches-any
([12..34]:[56..78])
```

優先を表すために一連の括弧を使用すると、次のようになります。

```
med eq 10 or ((not destination in (10.1.3.0/24)) and community matches-any
([12..34]:[56..78])
```

次に、複雑な式の別の例を示します。

```
(origin is igp or origin is incomplete or not med eq 42) and next-hop in (10.0.2.2)
```

左の論理積は、括弧で囲まれた複合条件です。複合条件内部の最初の単純条件は、送信元属性の値をテストします。値が内部ゲートウェイ プロトコル（IGP）の場合、複合条件内部は true です。それ以外の場合、評価は再び送信元属性の値のテストに進み、これが不完全な場合は複合条件内部はtrueです。それ以外の場合、評価は次の成分条件（単純条件の否定）のチェックに進みます。

apply

ポリシー定義およびポリシーのパラメータ化の項で説明したように、**apply** コマンドは別のポリシー（パラメータ化または未パラメータ化のいずれか）を別のポリシー内から実行します。これにより、ポリシーの共通ブロックの再利用が可能になります。ポリシーの共通ブロックのパラメータ化機能と併用した場合、**apply** コマンドは、設定の繰り返しを軽減するための強力なツールになります。

接続点

ポリシーはルートに適用されるまで有用になりません。ルーティングプロトコルがルートに適用されるポリシーを知る必要があります。たとえば、BGPではポリシーが使用されるいくつかの状況がありますが、最も一般的なのはインポートおよびエクスポートポリシーの定義です。

ポリシー接続点は、特定のプロトコル エンティティ（この場合 BGP ネイバー）と特定の名前付きポリシーとのアソシエーションが形成されるポイントです。検証手順がこのポイントで発生することに留意してください。あるポリシーがアタッチされるたびに、そのポリシーとそれが適用される可能性があるすべてのポリシーについて、そのポリシーを接続点で有効に使用できるか確かめられます。たとえば、ユーザが IS-IS レベル属性を設定するポリシーを定義し、このポリシーをインバウンド BGP ポリシーとしてアタッチしようとする場合、BGP ルートは IS-IS 属性を持たないため、この操作は拒否されます。同様に、使用中のポリシーが変更される場合、そのポリシーの現在の使用すべてについて、変更内容が現在の使用と互換性があるかどうかが検証されます。

各プロトコルは、それぞれルートを構成する属性（コマンド）のセットの定義を持っています。たとえば、BGP ルートに OSPF で未定義のコミュニティ属性が存在する場合があります。IS-IS 内のルートには、BGP には未知のレベル属性があります。RIB で内部的に保持されるルートは、タグ属性を持つ場合があります。

あるプロトコルにポリシーがアタッチされると、プロトコルはそのポリシーがプロトコルにとって既知のルート属性を使用して動作するものであるか確認します。プロトコルが未知の属性を使用している場合、プロトコルはアタッチを拒否します。たとえば、OSPF は BGP コミュニティの値をテストするポリシーのアタッチを拒否します。

各プロトコルは最低 2 つの異なるルート タイプにアクセスできるため、状況はさらに複雑になります。ネイティブ プロトコル ルートに加え、BGP または IS-IS を例にすると、一部のプロトコル ポリシー接続点は RIB ルート上で動作し、これは共通の中心的表現です。BGP を例にすると、プロトコルは RIB から BGP に再配布されたルートへポリシーを適用する接続点を提供します。2 つの異なる種類のルートを処理する接続点では、マッチングには RIB 属性の動作、設定には BGP 属性の動作というように、動作の混在が許可されます。

（注） プロトコル コンフィギュレーションは、サポートされていない動作を実行するポリシーの付加を拒否します。

続く項では、プロトコル接続点について、各接続点で有効な属性（コマンド）および動作に関する情報を含めて説明します。

属性および動作の詳細については、 *Routing Command Reference for Cisco ASR 9000 Series Routers* を参照してください。

テスト用の新しいパラメータ

BGP ポリシー接続点

この項では、それぞれの BGP ポリシー接続点について説明し、BGP 属性と演算子の概要を示します。

Additional-Path

additional-path 接続点を使用して、さまざまな属性のマッチング演算に基づいた、いっそう詳細な制御が行えます。この接続点は、BGP スピーカーがプレフィックスに対して複数のパスを

送信できるように追加パスを選択するために、ルートポリシーを使用するかどうかを決定するために使用されます。

追加パスにより、エッジルータでのBGPプレフィックス独立コンバージェンス（PIC）が可能になります。

次に、追加パス選択のイネーブル化に使用するルート ポリシー「add-path-policy」を設定する例を示します。

```
router bgp 100
 address-family ipv4 unicast
 additional-paths selection route-policy add-path-policy
```

ダンプニング

dampening 接続点は BGP 内でデフォルトのルート ダンプニングの動作を制御します。関連するピアの特定のポリシーによって上書きされないかぎり、BGPのすべてのルートが関連するポリシーを適用して、ダンプニング属性を設定します。

次のポリシーは、BGP の IPv4 ユニキャスト ルートのダンプニング値を設定します。/25 より具体的なこれらのルートは、/25 より具体性の低いルートに比べて、ダンプ後の回復に長い時間がかかります。

```
route-policy sample_damp
  if destination in (0.0.0.0/0 ge 25) then
    set dampening halflife 30 others default
  else
    set dampening halflife 20 others default
  endif
end-policy

router bgp 2
  address-family ipv4 unicast
    bgp dampening route-policy sample_damp
    .
    .
    .
```

Default Originate

default originate接続点により、デフォルトルート（0.0.0.0/0）を条件に応じて生成し、他のルートの存在に基づいてピアにアドバタイズできます。このコンフィギュレーションは、Routing Information Base（RIB）に対して関連付けられたポリシーの評価によって実行されます。ポリシーをパスするルートがある場合、デフォルト ルートが生成され、関連ピアに送られます。

次のポリシーでは、RIB 内に 10.0.0.0/8 ge 8 le 32 にマッチするルートが存在する場合に、デフォルト ルートを生成して BGP ネイバー 10.0.0.1 に送ります。

```
route-policy sample-originate
  if rib-has-route in (10.0.0.0/8 ge 8 le 32) then
     pass
  endif
end-policy

router bgp 2
```

```
  neighbor 10.0.0.1
    remote-as 3
    address-family ipv4 unicast
    default-originate route-policy sample-originate
    .
    .
    .
```

Neighbor Export

neighbor export 接続点は、特定のピアまたはピアのグループに送信する BGP ルートを選択します。このルートは、関連するポリシー全体に考えられる BGP ルートのセットを実行することによって選択されます。ポリシーをパスするすべてのルートが、ピアまたはピア グループにアップデートとして送信されます。送信されるルートは、適用されているポリシーによってその BGP 属性が変更されている場合があります。

次のポリシーは、すべての BGP ルートをネイバー 10.0.0.5 に送ります。2:100 から 2:200 までの範囲内の任意のコミュニティ タグが付けられたルートは、MED の値が 100、コミュニティの値が 2:666 で送信されます。その他のルートは、MED の値が 200、コミュニティの値が 2:200 で送信されます。

```
route-policy sample-export
  if community matches-any (2:[100-200]) then
    set med 100
    set community (2:666)
  else
    set med 200
    set community (2:200)
  endif
end-policy

router bgp 2
  neighbor 10.0.0.5
    remote-as 3
  address-family ipv4 unicast
    route-policy sample-export out
    .
    .
    .
```

Neighbor Import

neighbor import 接続点は、指定ピアからのルートの受け入れを制御します。ピアから受け取るすべてのルートは、アタッチされたポリシーについて実行されます。付加されたポリシーを渡すルートは、最適なパス ルート選択の候補として BGP Routing Information Base（BRIB）に渡されます。

BGP インポート ポリシーが変更されると、そのピアから受け取ったルートすべてを新しいポリシーに対して再実行することが必要になります。変更されたポリシーは、それまで全体に許可されていたルートを廃棄し、それまで廃棄されていたルートを全体に許可します。または、ルートが修正された方法を変更します。BGP の新しいコンフィギュレーション オプション（**bgp auto-policy-soft-reset**）によって、ソフト再設定が実行されるか BGP ルート リフレッシュ機能がネゴシエートされた場合に、この変更を自動的に発生させられます。

次の例は、ネイバー 10.0.0.1 からルートを受け取る方法を示しています。コミュニティ 3:100 で受け取ったルートは、ローカル プリファレンスが 100 に設定され、コミュニティ タグは 2:666 に設定されます。このピアからのその他のルートはすべて、ローカル プリファレンスが 200 に、コミュニティ タグが 2:200 に設定されます。

```
route-policy sample_import
  if community matches-any (3:100) then
    set local-preference 100
    set community (2:666)
  else
    set local-preference 200
    set community (2:200)
  endif
end-policy

router bgp 2
  neighbor 10.0.0.1
    remote-as 3
    address-family ipv4 unicast
      route-policy sample_import in
      .
      .
      .
```

ネットワーク

network 接続点は RIB から BGP へのルートの挿入を制御します。このポイントでアタッチされるルート ポリシーは、挿入されるルートのどの有効な BGP 属性でも設定できます。

次の例では、/24 よりも具体的なすべてのルートに対して well-known コミュニティ no-export を設定するネットワーク接続点にアタッチされたルート ポリシーを表示します。

```
route-policy NetworkControl
  if destination in (0.0.0.0/0 ge 25) then
    set community (no-export) additive
  endif
end-policy

router bgp 2
  address-family ipv4 unicast
    network 172.16.0.5/27 route-policy NetworkControl
```

Redistribute

OSPF 内の redistribute 接続点は、他のルーティング プロトコル ソースから OSPF リンクステートデータベースにルートを挿入します。各プロトコルからインポートするルートを選択することで実行されます。その後、コストとメトリック タイプの OSPF パラメータを設定します。ポリシーは、set metric-type または set ospf-metric コマンドを使用して OSPF へのルート挿入を制御できます。

次の例に、ポリシー OSPF-redist を使用して IS-IS instance_10 から OSPF インスタンス 1 にルートを再配布する方法を示します。ポリシーは再配布されたすべてのルートでメトリック タイプを type-2 に設定します。タグ 10 を持つ IS-IS ルートはコストが 100 に設定され、タグ 20 を持

つ IS-IS ルートは OSPF コストが 200 に設定されます。タグが 10 と 20 のいずれでもない IS-IS ルートは OSPF リンクステート データベースに再配布されません。

```
route-policy OSPF-redist
  set metric-type type-2
  if tag eq 10 then
    set ospf cost 100
  elseif tag eq 20 then
    set ospf cost 200
  else
    drop
  endif
end-policy
router ospf 1
  redistribute isis instance_10 policy OSPF-redist
  .
  .
  .
```

Show BGP

show bgp 接続点を使用して、指定のポリシーを渡す、選択した BGP ルートを表示できます。アタッチされたポリシーによってドロップされていないルートが、show bgp コマンドによる出力に似た形式で表示されます。

次の例では、show bgp route-policy コマンドを使用して、MED の値が 5 の BGP ルートを表示します。

```
route-policy sample-display
  if med eq 5 then
    pass
  endif
end-policy
!
show bgp route-policy sample-display
```

show bgp policy route-policy コマンドも存在しており、RIB がアウトバウンド BGP ポリシーであるかのようにして、名前付きポリシーを通過した RIB 内のすべてのルートを実行します。このコマンドは、次の例に示すように、各ルートが変更の前と後にどのようになったかを表示します。

```
show rpl route-policy test2

route-policy test2
  if (destination in  (10.0.0.0/8 ge 8 le 32)) then
    set med 333
  endif
end-policy
!

show bgp

BGP router identifier 10.0.0.1, local AS number 2
BGP main routing table version 11
BGP scan interval 60 secs
```

```
Status codes:s suppressed, d damped, h history, * valid, > best
             i - internal, S stale
Origin codes:i - IGP, e - EGP, ? - incomplete
   Network            Next Hop            Metric LocPrf Weight Path
*> 10.0.0.0           10.0.1.2                10             0 3 ?
*> 10.0.0.0/9         10.0.1.2                10             0 3 ?
*> 10.0.0.0/10        10.0.1.2                10             0 3 ?
*> 10.0.0.0/11        10.0.1.2                10             0 3 ?
*> 10.1.0.0/16        10.0.1.2                10             0 3 ?
*> 10.3.30.0/24       10.0.1.2                10             0 3 ?
*> 10.3.30.128/25     10.0.1.2                10             0 3 ?
*> 10.128.0.0/9       10.0.1.2                10             0 3 ?
*> 10.255.0.0/24      10.0.101.2            1000    555      0 100 e
*> 10.255.64.0/24     10.0.101.2            1000    555      0 100 e
....
```

show bgp policy route-policy test2

```
10.0.0.0/8 is advertised to 10.0.101.2

  Path info:
    neighbor:10.0.1.2        neighbor router id:10.0.1.2
    valid  external  best
  Attributes after inbound policy was applied:
    next hop:10.0.1.2
    MET ORG AS
    origin:incomplete  neighbor as:3  metric:10
    aspath:3
  Attributes after outbound policy was applied:
    next hop:10.0.1.2
    MET ORG AS
    origin:incomplete  neighbor as:3  metric:333
    aspath:2 3
...
```

テーブル ポリシー

BGPのテーブルポリシー機能を使用すると、ルートのトラフィック索引の値をグローバルルーティング テーブルにインストールされるときに設定できます。この機能を有効にするには **table-policy** コマンドを使用します。また BGP ポリシーアカウンティング機能もサポートされています。

BGP ポリシー アカウンティングでは、BGP ルートに設定されたトラフィック索引を使用してさまざまなカウンタをトラックします。テーブルポリシーの使用法の詳細については、『*Routing Configuration Guide for Cisco ASR 9000 Series Routers*』の「 *Implementing Routing Policy on Cisco ASR 9000 Series Router* 」のモジュールを参照してください。BGP ポリシーアカウンティングの詳細については、『*IP Addresses and Services Command Reference for Cisco ASR 9000 Series Routers*』の「 *Cisco Express Forwarding Commands on Cisco ASR 9000 Series Router* 」のモジュールを参照してください。

テーブル ポリシーを使用すると、一致基準に基づいて RIB からのルートをドロップすることもできます。この機能は特定のアプリケーションにおいて有用ですが、BGPがグローバルルーティングおよびフォワーディングテーブルにインストールしていないネイバーに対して、BGPがルートをアドバタイズするところに、簡単にルーティング「ブラック ホール」が作成されてしまうため、注意して使用する必要があります。

Import

import 接続点を使用して、グローバル VPN IPv4 テーブルから特定の VPN ルーティングおよび転送（VRF）インスタンスへのルートのインポートを制御できます。

レイヤ 3 VPN ネットワークの場合、Provider Edge（PE）ルータは他の PE ルータから Multiprotocol Internal Border Gateway Protocol（MP-iBGP）を通じて VPN IPv4 ルートを学習し、その VRF のインポート ルート ターゲットにマッチするルート ターゲットを含まないルート通知を自動的にフィルタリングして除外します。

この自動ルート フィルタリングは RPL コンフィギュレーションなしで発生します。ただし、VRF でのルートのインポートをより詳細に制御するために、VRF インポート ポリシーを設定できます。

次の例に、ルート ターゲット拡張コミュニティに基づいたマッチングおよびそれに続くネクスト ホップ設定の実行方法を示します。ルートのルート ターゲット値が 10:91 が設定されている場合、ネクスト ホップは 172.16.0.1 に設定されます。ルートのルート ターゲット値が 11:92 が設定されている場合、ネクスト ホップは 172.16.0.2 に設定されます。ルートの Site of Origin（SoO）値が 10:111111 または 10:111222 の場合、ルートはドロップされます。一致しないすべてのルートはドロップされます。

```
route-policy bgpvrf_import
  if extcommunity rt matches-any (10:91) then
    set next-hop 172.16.0.1
  elseif extcommunity rt matches-every (11:92) then
    set next-hop 172.16.0.2
  elseif extcommunity soo matches-any (10:111111, 10:111222) then
    pass
  endif
end-policy

vrf vrf_import
  address-family ipv4 unicast
    import route-policy bgpvrf_import
    .
    .
    .
```

（注） bgp import 接続点では、「Set」は「med」属性の有効な演算子です。

Export

export 接続点は、特定の VRF からグローバル VPN IPv4 テーブルへのルートのエクスポート全体を制御します。

レイヤ 3 VPN ネットワークでは、VRF IPv4 ルートが VPN IPv4 ルートに変換され MP-iBGP 経由で他の PE ルータへアドバタイズされる場合（またはある VRF から PE ルータ内の他の VRF へ流れる場合）、エクスポート ルート ターゲットは VPN IPv4 ルートに追加されます。

エクスポート ルート ターゲットのセットは RPL コンフィギュレーションなしで VRF に設定されます。ただし、条件に応じてルート ターゲットを設定するために、VRF エクスポート ポリシーを設定できます。

エクスポート ルート ポリシー用にサポートされているマッチングと設定の操作例を次に示します。あるルートが 172.16.1.0/24 にマッチした場合、ルート ターゲット拡張コミュニティは 10:101 に、重みは 211 に設定されます。ルートが 172.16.1.0/24 にマッチしないもののその始点が egp である場合、ローカル プリファレンスが 212 に、ルート ターゲット拡張コミュニティは 10:101 に設定されます。指定された条件にマッチしないルートの場合、ルート ターゲット拡張コミュニティ 10:111222 がルートに追加されます。さらに、前記の条件のいずれかにマッチするルートにも RT 10:111222 が追加されます。

```
route-policy bgpvrf_export
  if destination in (172.16.1.0/24) then
    set extcommunity rt (10:101)
    set weight 211
  elseif origin is egp then
    set local-preference 212
    set extcommunity rt (10:101)
  endif
  set extcommunity rt (10:111222) additive
end-policy

vrf vrf-export
  address-family ipv4 unicast
    export route-policy bgpvrf-export
    .
    .
    .
```

（注） bgp export 接続点では、「Set」は「med」属性の有効な演算子です。

Allocate-Label

allocate-label 接続点を使用して、さまざまな属性のマッチング操作に基づいた、いっそう詳細な制御が行えます。この接続点は、IPv4 ラベル付きユニキャスト アドレス ファミリ用にアップデートをネイバーに送信するときに、inter-AS オプション C 内のラベル割り当てが必要かどうか判断するために使用されます。サポートされている属性設定アクションは、パスとドロップです。

次の例に、プレフィックス 0.0.0.0 をプレフィックス長 0 でパスするルート ポリシーの設定方法を示します。ラベル割り当てが発生するのはプレフィックス 0.0.0.0 が存在する場合だけです。

```
route-policy label_policy
  if destination in (0.0.0.0/0) then
    pass
  endif
end-policy

router bgp 2
```

```
vrf vrf1
  rd auto
address-family ipv4 unicast
  allocate-label route-policy label-policy
  .
  .
  .
```

Retain Route-Target

BGP 内の retain route target 接続点を使用して、ルート ターゲット拡張コミュニティだけに基づいたマッチング条件を指定できます。この接続点は、Route Reflector（RR）または Autonomous System Boundary Router（ASBR）で役立ちます。

通常、RR はその PE ルータとのピアのためにすべての IPv4 VPN ルートを保持しておく必要があります。これら PE は、異なるルート ターゲット IPv4 VPN ルートでタグ付けされたルータを要求する場合があり、結果として RR が拡張不能になります。ルート ターゲット拡張コミュニティの定義済みセット、およびサービスする VPN の特定のセットを持つルートを RR が保持するよう設定することで、拡張性が得られます。

この接続点を使用する別の理由は、ASBR のためです。ASBR では VRF を設定する必要はありませんが、このコンフィギュレーションによって IPv4 VPN プレフィックス情報を保持する必要があります。

次の例に、ルート ポリシー リテイナーを設定し、retain route target 接続点に適用する方法を示します。ルート ターゲット拡張コミュニティ 10:615、10:6150、15.15.15.15.15:15 を含むルートが受け入れられます。一致しないすべてのルートはドロップされます。

```
extcommunity-set rt rtset1
  0:615,
  10:6150,
  15.15.15.15.15:15
  end-set

route-policy retainer
  if extcommunity rt matches-any rtset1 then
    pass
  endif
end-policy

router bgp 2
  address-family vpnv4 unicast
    retain route-target route-policy retainer
    .
    .
    .
```

Label-Mode

ラベルモード接続点では、プレフィックス値、コミュニティなどの任意の一致基準に基づいてラベル モードを選択する機能が提供されます。この接続点は、通常、ラベル モードのタイプを、展開環境設定に基づいて per-ce または per-vrf または per-prefix に設定するために使用されます。サポートされている属性設定アクションは、パスとドロップです。

次に、VPNv4 AF（アドレス ファミリ）レベルおよびVRF IPv4 AF レベルでのラベル モードの選択例を示します。

```
route-policy set_label_mode
  set label-mode per-prefix
end-policy
!
router bgp 100
 address-family vpnv4 unicast
  vrf all
   label mode route-policy pass-all
  !
 !
 vrf abc
  rd 1:1
  address-family ipv4 unicast
   label mode route-policy set_label_mode
  !
 !
!
end
```

Neighbor-ORF

neighbor-orf 接続点によって、プレフィックス ベースのマッチングだけを使用した着信 BGP ルート アップデートのフィルタリングが行えます。インバウンド フィルタとしての使用に加え、フィルタリングを実行できるよう、Outbound Route Filter（ORF）としてプレフィックスと処理内容（ドロップまたはパス）が上流ネイバーに送られます。

次の例に、ルート ポリシー orf-preset を設定し、ネイバー ORF 接続点に適用する方法を示します。orf-preset で指定されたプレフィックス（172.16.1.0/24、172.16.5.0/24、172.16.11.0/24）にマッチした場合、ルートのプレフィックスはドロップされます。ネイバーが宛先に送信する前にフィルタ更新を行うことができるように、BGP も、このインバウンド フィルタリングのほかに、許可または拒否とともにこれらのプレフィックス エントリをアップストリーム ネイバーに送信します。

```
  prefix-set orf-preset
    172.16.1.0/24,
    172.16.5.0/24,
    172.16.11.0/24
  end-set

  route-policy policy-orf
    if orf prefix in orf-preset then
      drop
    endif
    if orf prefix in (172.16.3.0/24, 172.16.7.0/24, 172.16.13.0/24) then
      pass
    endif

  router bgp 2
    neighbor 1.1.1.1
      remote-as 3
      address-family ipv4 unicast
        orf route-policy policy-orf
      .
      .
```

.

Next-hop

next-hop 接続点を使用して、より詳細なプロトコル ベースの制御とプレフィックス ベースのマッチング操作が行えます。この接続点は通常、ネクストホップ通知（アップまたはダウン）イベントで実行するかどうか決定するために使用されます。

ネクストホップのトラッキングにより、BGPはBGPプレフィックスに直接影響を与えるRouting Information Base（RIB）内のルートの到達可能性をモニタできます。BGP ネクストホップ接続点でのルート ポリシーは、特定のプレフィックス向けに BGP に配信される通知を抑制するのに役立ちます。ルート ポリシーは RIB ルートに割り当てられます。通常、非 BGP ルート監視のため、ルート ポリシーはネクスト ホップ トラッキングとともに使用されます。

次の例に、プレフィックス 10.0.0.0 およびプレフィックス長 8 を持つスタティック ルートまたは接続ルートを監視するための、ルート ポリシーを使用した BGP ネクストホップ トラッキング機能の設定方法を示します。

```
route-policy nxthp_policy_A
  if destination in (10.0.0.0/8) and protocol in (static, connected) then
    pass
  endif
end-policy

router bgp 2
  address-family ipv4 unicast
    nexthop route-policy nxthp_policy_A
      .
      .
      .
```

Clear-Policy

clear-policy 接続点によって、clear bgp コマンドを使用するときに、さまざまな AS パス マッチング操作に基づいたより詳細な制御が行えます。この接続点は、通常、AS パス ベースのマッチング操作に基づいて BGP フラップ統計情報をクリアするかどうか判断するときに使用します。

次の例に、セット my-as-set 内でマッチングを行う 1 つ以上の正規表現がルートに関連付けられたASパスにマッチした場合にin演算子が真となるルートポリシーの設定方法を示します。マッチした場合、clear コマンドは関連付けられたフラップ統計情報をクリアします。

```
as-path-set my-as-set
  ios-regex '_12$',
  ios-regex '_13$'
end-set

route-policy policy_a
  if as-path in my-as-set then
    pass
  else
    drop
  endif
end-policy
```

```
clear bgp ipv4 unicast flap-statistics route-policy policy_a
```

Debug

debug 接続点を使用して、より詳細なプレフィックス ベースのマッチング演算が行えます。この接続点、通常、ルートのプレフィックスに基づいてさまざまな BGP コマンドのデバッグ出力をフィルタリングするために使用されます。

次に、プレフィックス長が 8 のプレフィックス 20.0.0.0 だけをパスするルート ポリシーを設定する例を示します。デバッグ出力はそのプレフィックスに対してだけに表示されます。

```
route-policy policy_b
  if destination in (10.0.0.0/8) then
    pass
  else
    drop

  endif
end-policy

debug bgp update policy_b
```

BGP 属性と演算子

このテーブルでは、接続点ごとの BGP 属性と演算子をまとめます。

表 *7: BGP* 属性と演算子

接続点	属性	一致	セット
additional-paths	path-selection	—	set
	コミュニティ	matches-every is-empty matches-any	—

接続点	属性	一致	セット
aggregation	as-path	in is-local length neighbor-is originates-from passes-through unique-length	—
	as-path-length	is、ge、le、eq	—
	as-path-unique-length	is、ge、le、eq	—
	community	is-empty matches-any matches-every	set set additive delete in delete not in delete all
	destination	in	—
	extcommunity cost	—	set set additive
	local-preference	is、ge、le、eq	set
	med	is、eg、ge、le	setset +set -
	next-hop	in	set
	origin	is	set
	source	in	—
	suppress-route	—	suppress-route
	weight	—	set

接続点	属性	一致	セット
allocate-label	as-path	in is-local length neighbor-is originates-from passes-through unique-length	—
	as-path-length	is、ge、le、eq	—
	as-path-unique-length	is、ge、le、eq	—
	community	is-empty matches-any matches-every	—
	destination	in	—
	label	—	set
	local-preference	is、ge、le、eq	—
	med	is、eg、ge、le	—
	next-hop	in	—
	origin	is	—
	source	in	—

接続点	属性	一致	セット
clear-policy	as-path	in is-local length neighbor-is originates-from passes-through unique-length	—
	as-path-length	is、ge、le、eq	—
	as-path-unique-length	is、ge、le、eq	—

接続点	属性	一致	セット
dampening	as-path	in is-local length neighbor-is originates-from passes-through unique-length	—
	as-path-length	is、ge、le、eq	—
	as-path-unique-length	is、ge、le、eq	—
	community	is-empty matches-any matches-every	—
	dampening	—/	set dampening （ダンプニングを制御する値の設定については、ダンプニング （577 ページ）を参照してください）
	destination	in	—
	local-preference	is、ge、le、eq	—
	med	is、eg、ge、le	—
	next-hop	in	—
	origin	is	—
	source	in	—
debug	destination	in	—

接続点	属性	一致	セット
default-originate	as-path	該当なし	prepend
	community community with `peeras'	該当なし	set set additive
	extcommunity cost	該当なし	set set additive
	extcommunity rt	該当なし	set
	extcommunity soo	該当なし	set
	local-preference	該当なし	set
	med	該当なし	set set + set-assign igp
	next-hop	該当なし	set set-to-peer-address set-to-self
	origin	該当なし	set
	rib-has-route	in	該当なし

接続点	属性	一致	セット
export（VRF）	as-path	in is-local length neighbor-is originates-from passes-through unique-length	該当なし
	as-path-length	is、ge、le、eq	—
	as-path-unique-length	is、ge、le、eq	—
	community	is-empty matches-any matches-every	set set additive delete in delete not in delete all
	destination	in	—
	extcommunity rt	is-empty matches-any matches-every matches-within	set additive delete-in delete-not-in delete-all
	extcommunity soo	is-empty matches-any matches-every matches-within	—
	local-preference	is、ge、le、eq	set
	med	is、eg、ge、le	set
	next-hop	in	—
	origin	is	—

接続点	属性	一致	セット
	source	in	—
	weight	—	set

接続点	属性	一致	セット
import（VRF）	as-path	in is-local length neighbor-is originates-from passes-through unique-length	—
	as-path-length	is、ge、le、eq	—
	as-path-unique-length	is、ge、le、eq	—
	community	is-empty matches-any matches-every	—
	destination	in	—
	extcommunity rt	is-empty matches-any matches-every matches-within	—
	extcommunity soo	is-empty matches-any matches-every matches-within	—
	local-preference	is、ge、le、eq	set
	med	is、eg、ge、le	set
	next-hop	in	set set peer address set destination vrf
	origin	is	—
	source	in	—

接続点	属性	一致	セット
label-mode	as-path	in is-local length neighbor-is originates-from passes-through unique-length	—
	as-path-length	is、ge、le、eq	—
	as-path-unique-length	is、ge、le、eq	—
	community	is-empty matches-any matches-every	—
	destination	in	—
	label	—	set
	local-preference	is、ge、le、eq	—
	med	is、eg、ge、le	—
	next-hop	in	—
	origin	is	—
	source	in	—

接続点	属性	一致	セット
neighbor-in	as-path	in is-local length neighbor-is originates-from passes-through unique-length	prepend prepend most-recent replace
	as-path-length	is、ge、le、eq	—
	as-path-unique-length	is、ge、le、eq	—
	communitycommunity with ‘peeras’	is-empty matches-any matches-every	set set additive delete-in delete-not-in delete-all
	destination	in	—
	rd	in	—
	evpn-route-type	is	—
	esi	in	はい
	etag	in	はい
	mac	in	はい
	evpn-originator	in	—
	evpn-gateway	in	—
	extcommunity cost	—	set set additive
	extcommunity rt	is-empty matches-any matches-every matches-within	set additive delete-in delete-not-in delete-all

接続点	属性	一致	セット	
	extcommunity soo	is-empty matches-any matches-every matches-within	—	
	local-preference	is、ge、le、eq	set	
	med	is、eg、ge、le	set set + set -	
	next-hop	in	set set peer address	
	origin	is	set	
	source	in	—	
	weight	—	set	
	neighbor-out	as-path	in is-local length neighbor-is originates-from passes-through unique-length	prepend prepend most-recent replace
as-path-length		is、ge、le、eq	—	
as-path-unique-length		is、ge、le、eq	—	
communitycommunity with ‘peeras’		is-empty matches-any matches-every	set set additive delete-in delete-not-in delete-all	
destination		in	—	

接続点	属性	一致	セット
rd		in	—
evpn-route-type		is	—
esi		in	あり
etag		in	あり
mac		in	あり
evpn-originator		in	—
evpn-gateway		in	—
extcommunity cost		—	set set additive
extcommunity rt		is-empty matches-any matches-every matches-within	set additive delete-in delete-not-in delete-all
extcommunity soo		is-empty matches-any matches-every matches-within	—
local-preference		is、ge、le、eq	set
med		is、eg、ge、le	set set + set - set max-unreachable set igp-cost
next-hop		in	set set self
origin		is	set
path-type		is	—
rd		in	—

接続点	属性	一致	セット	
source		in	—	
unsuppress-route		—	unsuppress-route	
vpn-distinguisher		—	set	
neighbor-orf		orf-prefix	in	n/a
network	as-path	—	prepend	
	community	—	set set additive delete-in delete-not-in delete-all	
	destination	in	—	
	extcommunity cost	—	set set additive	
	mpls-label	route-has-label	—	
	local-preference	—	set	
	med	—	set set+ set-	
	next-hop	in	set	
	origin	—	set	
	route-type	is	—	
	tag	is、ge、le、eq	—	
	weight	—	set	
next-hop	destination	in	—	
	protocol	is、in	—	
	source	in	—	

接続点	属性	一致	セット
redistribute	as-path	—	prepend
	community	—	set set additive delete in delete not in delete all
	destination	in	—
	extcommunity cost	—	setset additive
	local-preference	—	set
	med	—	set set+ set-
	next-hop	in	set
	origin	—	set
	mpls-label	route-has-label	—
	route-type	is	—
	tag	is、eq、ge、le	—
	weight	—	set
retain-rt	extcommunity rt	is-empty matches-any matches-every matches-within	—

接続点	属性	一致	セット
show	as-path	in is-local length neighbor-is originates-from passes-through unique-length	—
	as-path-length	is、ge、le、eq	—
	as-path-unique-length	is、ge、le、eq	—
	community	is-empty matches-any matches-every	—
	destination	in	—
	extcommunity rt	is-empty matches-any matches-every matches-within	—
	extcommunity soo	is-empty matches-any matches-every matches-within	—
	med	is、eg、ge、le	—
	next-hop	in	—
	origin	is	—
	source	in	—

接続点	属性	一致	セット
table-policy	as-path	in is-local length neighbor-is originates-from passes-through unique-length	—
	as-path-unique-length	is、ge、le、eq	—
	community	is-empty matches-any matches-every	—
	local-preference	is、ge、le、eq	—
	destination	in	—
	med	is、eg、ge、le	—
	next-hop	in	—
	origin	is	—
	rib-metric	—	set
	source	in	—
	tag	—	set
	traffic-index	—	set

一部の BGP ルート属性は、さまざまな理由のため一部の BGP 接続点からアクセスできません。たとえば、set med igp-cost only コマンドは、設定された IGP コストがあり、ソース値を示す場合に意味があります。

Default-Information Originate

default-information originate 接続点を使用して、条件に応じてデフォルトのルート 0.0.0.0/0 を OSPF リンクステート データベースに挿入できます。これはアタッチされたポリシーの評価によって実行されます。ローカル RIB の任意のルートがポリシーをパスすると、デフォルトのルートがリンクステート データベースに挿入されます。

次の例では、10.0.0.0/8 ge 8 le 25 に一致するルートが RIB に存在する場合に、デフォルトのルートを生成する方法を示します。

```
route-policy ospf-originate
  if rib-has-route in (10.0.0.0/8 ge 8 le 25) then
    pass
  endif
end-policy

router ospf 1
  default-information originate policy ospf-originate
  .
  .
  .
```

RPL：プレフィックスが is-best-path/is-best-multipath の場合

ボーダー ゲートウェイ プロトコル（BGP）ルータは、同じ宛先への複数のパスを受信します。標準として、デフォルトでは、BGP ベストパスアルゴリズムが IP ルーティングテーブルにインストールする最適なパスを決定します。これはトラフィックの転送に使用されます。

BGP は、最初の有効なパスを現在のベストパスとして割り当てます。次に、BGP は、ベストパスとリスト内の次のパスを比較します。このプロセスは、BGP が有効なパスのリストの最後に到達するまで継続されます。これには、ベストパスの決定に使用されるすべてのルールが含まれます。指定されたアドレスプレフィックスに複数のパスがある場合、BGP は次のように処理します。

- ベストパス選択ルールに従って、パスの 1 つをベストパスとして選択します。
- 転送テーブルにベストパスをインストールします。各 BGP スピーカーは、ピアへのベストパスのみをアドバタイズします。

（注） ベストパスのみを送信するアドバタイズメントルールは、そのピアに対して BGP スピーカ上に存在する宛先の完全なルーティング状態を伝達しません。

BGP スピーカがピアのいずれかからパスを受信した後、ピアがそのパスをパケットの転送に使用します。他のすべてのピアは、このピアから同じパスを受信します。これにより、BGP ネットワークでの一貫したルーティングが実現します。リンク帯域幅使用率を向上させるには、ほとんどの BGP 実装では、特定の条件を満たす追加パスをマルチパスとして選択し、それらを転送テーブルにインストールします。このような着信パケットは、ベストパスとマルチパス上でロードバランシングされます。ピアにアドバタイズされていない転送テーブルにパスをインストールできます。RR ルートリフレクタは、ベストパスとマルチパスを検出します。このようにして、ルートリフレクタはベストパスとマルチパスに異なるコミュニティを使用します。この機能を使用すると、RR または境界ルータによって実行されるローカルの決定を BGP で通知できます。この新機能を使用した場合は、コミュニティストリングを使用して RR によって選択されました（たとえば、is-best-path の場合は community 100:100）。コントローラは、どのベストパスがすべての R に送信されるかを確認します。ボーダー ゲートウェイ プロトコル ルータは、同じ宛先への複数のパスを受信します。ベストパスの計算を実行している間は、1

つのベストパスが存在し、場合によっては同等のパスおよび同等でない若干数のパスが存在します。したがって、best-path と is-equal-best-path の要件です。

BGP のベストパスアルゴリズムは、IP ルーティングテーブル内でベストパスを決定し、トラフィックの転送に使用します。RPL 内のこの機能拡張により、決定を行うためのポリシーを作成できます。ベストパスのローカル選択のためのコミュニティストリングの追加。BGP 追加パス（Add Path）の導入により、BGP はベストパスよりも多くを通知するようになりました。BGP はベストパスと、ベストパスと同等のパス全体を通知できます。これは、BGP マルチパスルールとすべてのバックアップパスに従っています。

OSPF ポリシー接続点

この項では、それぞれの OSPF ポリシー接続点について説明し、OSPF 属性と演算子の概要を示します。

Default-Information Originate

default-information originate 接続点を使用して、条件に応じてデフォルトのルート 0.0.0.0/0 を OSPF リンクステート データベースに挿入できます。これはアタッチされたポリシーの評価によって実行されます。ローカル RIB の任意のルートがポリシーをパスすると、デフォルトのルートがリンクステート データベースに挿入されます。

次の例では、10.0.0.0/8 ge 8 le 25 に一致するルートが RIB に存在する場合に、デフォルトのルートを生成する方法を示します。

```
route-policy ospf-originate
  if rib-has-route in (10.0.0.0/8 ge 8 le 25) then
    pass
  endif
end-policy

router ospf 1
  default-information originate policy ospf-originate
  .
  .
  .
```

Redistribute

OSPF 内の redistribute 接続点は、他のルーティング プロトコル ソースから OSPF リンクステートデータベースにルートを挿入します。各プロトコルからインポートするルートを選択することで実行されます。その後、コストとメトリック タイプの OSPF パラメータを設定します。ポリシーは、set metric-type または set ospf-metric コマンドを使用して OSPF へのルート挿入を制御できます。

次の例に、ポリシー OSPF-redist を使用して IS-IS instance_10 から OSPF インスタンス 1 にルートを再配布する方法を示します。ポリシーは再配布されたすべてのルートでメトリック タイプを type-2 に設定します。タグ 10 を持つ IS-IS ルートはコストが 100 に設定され、タグ 20 を持つ IS-IS ルートは OSPF コストが 200 に設定されます。タグが 10 と 20 のいずれでもない IS-IS ルートは OSPF リンクステート データベースに再配布されません。

```
route-policy OSPF-redist
  set metric-type type-2
  if tag eq 10 then
    set ospf cost 100
  elseif tag eq 20 then
    set ospf cost 200
  else
    drop
  endif
end-policy
router ospf 1
  redistribute isis instance_10 policy OSPF-redist
  .
  .
  .
```

Area-in

OSPF 内の area-in 接続点では、受信 OSPF タイプ 3 サマリー リンク ステート アドバタイズメント（LSA）をフィルタリングできます。この接続点ではプレフィックス ベースのマッチングが行えるため、より詳細なタイプ 3 サマリー LSA のフィルタリングが実現します。

次の例では、OSPF サマリー LSA のプレフィックスの設定方法を示します。プレフィックスが 10 .105.3.0/24、10 .105.7.0/24、10 .105.13.0/24 のいずれかに一致する場合は受け入れられます。プレフィックスが 10 .106.3.0/24、10 .106.7.0/24、10 .106.13.0/24 のいずれかに一致する場合はドロップされます。

```
  route-policy OSPF-area-in
    if destination in (10
.105.3.0/24, 10
.105.7.0/24, 10
.105.13.0/24) then
      drop
    endif
    if destination in (10
.106.3.0/24, 10
.106.7.0/24, 10
.106.13.0/24) then
      pass
    endif
  end-policy

  router ospf 1
    area 1
      route-policy OSPF-area-in in
```

Area-out

OSPF 内の area-in 接続点では、発信 OSPF タイプ 3 サマリー LSA をフィルタリングできます。この接続点ではプレフィックス ベースのマッチングが行えるため、より詳細なタイプ 3 サマリー LSA のフィルタリングが実現します。

次の例では、OSPF サマリー LSA のプレフィックスの設定方法を示します。プレフィックスが 10 .105.3.0/24、10 .105.7.0/24、10 .105.13.0/24 のいずれかに一致する場合は通知されます。プレ

フィックスが 10.105.3.0/24、10 .105.7.0/24、10 .105.13.0/24 のいずれかに一致する場合はドロップされ、通知されません。

```
  route-policy OSPF-area-out
    if destination in (10
.105.3.0/24, 10
.105.7.0/24, 10
.105.13.0/24) then
      drop
    endif
    if destination in (10
.105.3.0/24, 10
.105.7.0/24, 10
.105.13.0/24) then
      pass
    endif
  end-policy

  router ospf 1
    area 1
      route-policy OSPF-area-out out
```

SPF Prefix-priority

OSPF 内の spf-prefix-priority 接続点を使用して、OSPFv2 プレフィックスのプライオリティ付けに適用するルート ポリシーを定義できます。

OSPF 属性と演算子

この表では接続点ごとの OSPF 属性と演算子をまとめます。

表 *8 : OSPF* 属性と演算子

接続点	属性	一致	セット
distribute-list-in-area	destination rib-metric tag	in in eq、ge、is、le	該当なし 該当なし 該当なし
distribute-list-in-instance	destination rib-metric tag	in in eq、ge、is、le	該当なし 該当なし 該当なし

接続点	属性	一致	セット
distribute-list-in-interface	destination rib-metric tag	in in eq、ge、is、le	該当なし 該当なし 該当なし
default-information originate	ospf-metric	—	set
	metric-type	—	set
	tag	—	set
	rib-has-route	in	—
redistribute	destination	in	—
	metric-type	—	set
	ospf-metric	—	set
	next-hop	in	—
	mpls-label	route-has-label	—
	rib-metric	is、le、ge、eq	n/a
	route-type	is	—
	tag	is、eq、ge、le	set
area-in	destination	in	—
area-out	destination	in	—
spf-prefix-priority	destination	in	n/a
	spf-priority	n/a	set
	tag	is、le、ge、eq	n/a

Distribute-list in

OSPF 内の distribute-list in 接続点では、ルート ポリシーを使用して、OSPF プレフィックスをフィルタリングできます。distribute-list in ルート ポリシーは、OSPF インスタンス、エリア、およびインターフェイス レベルで設定できます。distribute-list in コマンドで使用されるルート

ポリシーは、match ステートメント、「destination」および「rib-metric」をサポートします。「set」コマンドはルート ポリシーではサポートされません。

次は、「distribute-list in」の有効なルート ポリシーの例です。

```
route-policy DEST
  if destination in (10.10.10.10/32) then
    drop
  else
    pass
  endif
end-policy


route-policy METRIC
  if rib-metric ge 10 and rib-metric le 19 then
    drop
  else
    pass
  endif
end-policy


prefix-set R-PFX
  10.10.10.30
end-set

route-policy R-SET
  if destination in R-PFX and rib-metric le 20 then
    pass
  else
    drop
  endif
end-policy
```

OSPFv3 ポリシー接続点

この項では、それぞれの OSPFv3 ポリシー接続点について説明し、OSPFv3 属性と演算子の概要を示します。

Default-Information Originate

default-information originate 接続点を使用して、条件に応じてデフォルトのルート 0::/0 を OSPFv3 リンクステートデータベースに挿入できます。これはアタッチされたポリシーの評価によって実行されます。ローカル RIB の任意のルートがポリシーをパスすると、デフォルトのルートがリンクステート データベースに挿入されます。

次の例では、2001::/96 に一致するルートが RIB に存在する場合に、デフォルトのルートを生成する方法を示します。

```
  route-policy ospfv3-originate
    if rib-has-route in (2001::/96) then
      pass
    endif
  end-policy
```

```
router ospfv3 1
  default-information originate policy ospfv3-originate
  .
  .
```

Redistribute

OSPFv3 内の redistribute 接続点は、他のルーティング プロトコル ソースから OSPFv3 リンクステート データベースにルートを挿入します。各プロトコルからインポートするルート タイプを選択することで実行されます。その後、コストとメトリック タイプの OSPFv3 パラメータを設定します。ポリシーは、metric type コマンドを使用して OSPFv3 へのルート挿入を制御できます。

次の例に、ポリシー OSPFv3-redist を使用して BGP インスタンス 15 から OSPF インスタンス 1 にルートを再配布する方法を示します。ポリシーは再配布されたすべてのルートでメトリック タイプを type-2 に設定します。タグ 10 を持つ BGP ルートはコストが 100 に設定され、タグ 20 を持つ BGP ルートは OSPFv3 コストが 200 に設定されます。タグが 10 と 20 のいずれでもない BGP ルートは OSPFv3 リンクステート データベースに再配布されません。

```
route-policy OSPFv3-redist
  set metric-type type-2
  if tag eq 10 then
    set extcommunity cost 100
  elseif tag eq 20 then
    set extcommunity cost 200
  else
    drop
  endif
end-policy

router ospfv3 1
  redistribute bgp 15 policy OSPFv3-redist
  .
  .
  .
```

OSPFv3 属性と演算子

この表では、接続点ごとの OSPFv3 属性と演算子をまとめます。

表 *9 : OSPFv3* 属性と演算子

接続点	属性	一致	セット
default-information originate	ospf-metric	—	set
	metric-type	—	set
	tag	—	set
	rib-has-route	in	—

接続点	属性	一致	セット
redistribute	destination	in	—
	ospf-metric	—	set
	metric-type	—	set
	route-type	is	—
	tag	is、eq、ge、le	—

IS-IS ポリシー接続点

この項では、それぞれの IS-IS ポリシー接続点について説明し、IS-IS 属性と演算子の概要を示します。

Redistribute

IS-IS 内の redistribute 接続点を使用して、他のプロトコルからのルートを IS-IS によって再アドバタイズできます。ポリシーは IS-IS に再配布するルートの種類を選択するための制御構造のセットです。ポリシーでは、ルートを挿入する IS-IS レベルおよびそのときのメトリック値も制御可能です。

次に例を示します。この例では、ポリシー ISIS-redist を使用して IS-IS インスタンス 1 からのルートが IS-IS インスタンス instance_10 に再配布されます。このポリシーは、再配布されるルートすべてのレベルを level-1-2 に設定します。タグ 10 を持つ IS-IS ルートはメトリックが 100 に設定され、タグ 20 を持つ IS-IS ルートは IS-IS メトリックが 200 に設定されます。タグが 10 と 20 のいずれでもない IS-IS ルートは IS-IS データベースに再配布されません。

```
route-policy ISIS-redist
  set level level-1-2
  if tag eq 10 then
    set isis-metric 100
  elseif tag eq 20 then
    set isis-metric 200
  else
    drop
  endif
end-policy

router isis instance_10
  address-family ipv4 unicast
    redistribute isis 1 policy ISIS-redist
    .
    .
    .
```

Default-Information Originate

IS-IS 内の default-information originate 接続点を使用して、デフォルトのルート 0.0.0.0/0 を条件に応じて IS-IS ルート データベースに挿入できます。

次の例では、10.0.0.0/8 ge 8 le 25 に一致するルートが RIB に存在する場合に、IPv4 ユニキャストのデフォルト ルートを生成する方法を示します。IS-IS データベースに挿入されるデフォルト ルート上で、IS-IS ルートのコストは 100 に、レベルは level-1-2 に設定されます。

```
route-policy isis-originate
  if rib-has-route in (10.0.0.0/8 ge 8 le 25) then
    set metric 100
    set level level-1-2
  endif
end-policy

router isis instance_10
  address-family ipv4 unicast
    default-information originate policy isis_originate
    .
```

Inter-area-propagate

IS-IS 内の inter-area-propagate 接続点を使用して、プレフィックスをあるレベルから同じ IS-IS インスタンス内の別のレベルへと条件に応じてプロパゲートできます。

次の例に、プレフィックスのいずれかが 10.0.0.0/8 ge 8 le 25 に一致した場合に、プレフィックスをレベル 1 LSP からレベル 2 LSP へリークさせる方法を示します。

```
route-policy isis-propagate
  if destination in (10.0.0.0/8 ge 8 le 25) then
    pass
  endif
end-policy


router isis instance_10
  address-family ipv4 unicast
    propagate level 1 into level 2 policy isis-propagate
    .
```

IS-IS 属性と演算子

この表では、接続点ごとの IS-IS 属性と演算子をまとめます。

表 *10 : IS-IS* 属性と演算子

接続点	属性	一致	セット
redistribution	tag	is、eq、ge、le	set
	route-type	is （注） ルートタイプ、*ospf-nssa-type-1* と *ospf-nssa-type-2* を一致させることはできません。	—
	destination	in	—
	next-hop	in	—
	mpls-label	route-has-label	—
	level	—	set
	isis-metric	—	set
	metric-type	—	set
default-information originate	rib-has-route	in	—
	level	—	set
	isis-metric	—	set
	tag	—	set
inter-area-propagate	destination	in	—

EIGRP ポリシー接続点

この項では、それぞれの EIGRP ポリシー接続点について説明し、EIGRP 属性と演算子の概要を示します。

Default-Accept-In

default-accept-in 接続点を使用すると、付加されたポリシーの評価によって EIGRP ルートの条件付きデフォルト フラグの設定とリセットが行えます。

次の例に、10.0.0.0/8 にマッチするルートと最大 10.0.0.0/25 までの長いプレフィックスすべてに条件付きデフォルト フラグを設定するポリシーを示します。

```
route-policy eigrp-cd-policy-in
```

```
    if destination in (10.0.0.0/8 ge 8 le 25) then
      pass
    endif
  end-policy
  !
  router eigrp 100
    address-family ipv4
      default-information allowed in route-policy eigrp-cd-policy-in
      .
      .
      .
```

Default-Accept-Out

default-accept-out 接続点を使用すると、付加されたポリシーの評価によって EIGRP ルートの条件付きデフォルト フラグの設定とリセットが行えます。

次に、10.10.0.0/16 に一致するすべてのルートに条件付きデフォルトフラグを設定するポリシーの例を示します。

```
  route-policy eigrp-cd-policy-out
    if destination in (10
.10.0.0/16) then
      pass
    endif
  end-policy
  !
  router eigrp 100
    address-family ipv4
      default-information allowed out route-policy eigrp-cd-policy-out
      .
      .
      .
```

Policy-In

policy-in 接続点を使用して、インバウンド EIGRP ルートのフィルタリングと修正が行えます。このポリシーは、インターフェイス インバウンド ルート ポリシーを持たないすべてのインターフェイスに適用されます。

次の例に、EIGRP でのコマンドを示します。

```
  router eigrp 100
    address-family ipv4
      route-policy global-policy-in in
      .
      .
      .
```

Policy-Out

policy-out 接続点を使用して、アウトバウンド EIGRP ルートのフィルタリングと修正が行えます。このポリシーは、インターフェイス アウトバウンド ルート ポリシーを持たないすべてのインターフェイスに適用されます。

次の例に、EIGRP でのコマンドを示します。

```
router eigrp 100
  address-family ipv4
    route-policy global-policy-out out
    .
    .
    .
```

If-Policy-In

if-policy-in接続点を使用して、特定のEIGRPインターフェイス上で受け取るルートのフィルタリングが行えます。次に、GigabitEthernetインターフェイス0/2/0/3のインバウンドポリシーの例を示します。

```
router eigrp 100
  address-family ipv4
    interface GigabitEthernet0/2/0/3
      route-policy if-filter-policy-in in
    .
    .
    .
```

If-Policy-Out

if-policy-out 接続点を使用して、特定の EIGRP インターフェイス上で送出するルートのフィルタリングが行えます。次に、GigabitEthernet インターフェイス 0/2/0/3 のアウトバウンドポリシーの例を示します。

```
router eigrp 100
  address-family ipv4
    interface GigabitEthernet0/2/0/3
      route-policy if-filter-policy-out out
    .
    .
    .
```

Redistribute

EIGRP 内の redistribute 接続点によって、他のルーティング プロトコルから再配布されたルートのフィルタリングや、ルートをEIGRPデータベースにインストールする前に一部のルーティング パラメータを変更することが可能です。次の例に、RIPルートのEIGRPへの再配布をフィルタするポリシーを示します。

```
router-policy redistribute-rip
  if destination in (100.1.1.0/24) then
    set eigrp-metric 5000000 4000 150 30 2000
  else
    set tag 200
  endif
end-policy

router eigrp 100
  address-family ipv4
    redistribute rip route-policy redistribute-rip
    .
```

.
.

EIGRP 属性と演算子

この表では、接続点ごとの EIGRP 属性と演算子をまとめます。

表 *11 : EIGRP* 属性と演算子

接続点	属性	一致	セット
default-accept-in	destination	in	—
default-accept-out	destination	in	—
if-policy-in	destination	in	—
	next-hop	in	—
	eigrp-metric	—	add、set
	tag	is、eq、ge、le	set
if-policy-out	destination	in	—
	next-hop	in	—
	protocol	is、in	—
	eigrp-metric	—	add、set
	tag	is、eq、ge、le	set
policy-in	destination	in	—
	next-hop	in	—
	eigrp-metric	—	add、set
	tag	is、eq、ge、le	set

接続点	属性	一致	セット
policy-out	destination	in	—
	next-hop	in	—
	protocol	is、in	—
	eigrp-metric	—	add、set
	tag	is、eq、ge、le	set
redistribute	destination	in	—
	next-hop	in	—
	mpls-label	route-has-label	—
	eigrp-metric	—	add、set
	route-type	is	—
	tag	is、eq、ge、le	set

RIP ポリシー接続点

この項では、それぞれの RIP ポリシー接続点について説明し、RIP 属性と演算子の概要を示します。

Default-Information Originate

default-information originate 接続点を使用して、条件に応じてデフォルトのルート 0.0.0.0/0 を RIP アップデートに挿入できます。これは付加されたポリシーの評価によって実行されます。ローカル RIB の任意のルートがポリシーをパスすると、デフォルトのルートが挿入されます。

次の例では、10.0.0.0/8 ge 8 le 25 に一致するルートが RIB に存在する場合に、デフォルトのルートを生成する方法を示します。

```
route-policy rip-originate
  if rib-has-route in (10.0.0.0/8 ge 8 le 25) then
    pass
  endif
end-policy

router rip
  default-information originate route-policy rip-originate
```

Redistribute

RIP 内の redistribute 接続点を使用して、他のルーティング プロトコル ソースから RIP データベースにルートを挿入できます。

次の例は、RIP に OSPF ルートを挿入する方法を示しています。

```
route-policy redist-ospf
  set rip-metric 5
end-policy

router rip
  redistribute ospf 1 route-policy redist-ospf
```

Global-Inbound

RIP 用 global-inbound 接続点を使用して、ルート ポリシーに一致するインバウンド RIP ルートのフィルタリングや更新が行えます。

次の例に、rip-in という名前のルート ポリシーに一致するインバウンド RIP ルートをフィルタする方法を示します。

```
router rip
  route-policy rip-in in
```

Global-Outbound

RIP 用 global-outbound 接続点を使用して、ルート ポリシーに一致するアウトバウンド RIP ルートのフィルタリングや更新が行えます。

次の例に、rip-out という名前のルート ポリシーに一致するアウトバウンド RIP ルートをフィルタする方法を示します。

```
router rip
  route-policy rip-out out
```

Interface-Inbound

interface-inbound 接続点を使用して、特定のインターフェイス向けルート ポリシーに一致するインバウンド RIP ルートのフィルタリングや更新が行えます。

次の例に、インターフェイス 0/1/0/1 向けルート ポリシーに一致するインバウンド RIP ルートをフィルタする方法を示します。

```
router rip
  interface GigabitEthernet0/1/0/1
    route-policy rip-in in
```

Interface-Outbound

interface-outbound 接続点を使用して、特定のインターフェイス向けルート ポリシーに一致するアウトバウンド RIP ルートのフィルタリングや更新が行えます。

次の例に、インターフェイス 0/2/0/1 向けルート ポリシーに一致するアウトバウンド RIP ルートをフィルタする方法を示します。

```
router rip
  interface GigabitEthernet0/2/0/1
    route-policy rip-out out
```

RIP 属性と演算子

この表では、接続点ごとの RIP 属性と演算子をまとめます。

表 *12 : RIP* 属性と演算子

接続点	属性	一致	セット
default-information originate	next-hop	na	set
	rip-metric	na	set
	rip-tag	na	set
	rib-has-route	in	na
global-inbound	destination	in	na
	next-hop	in	na
	rip-metric	na	add
global-outbound	destination	in	na
	protocol	is、in	na
	rip-metric	na	add
interface-inbound	destination	in	na
	next-hop	in	na
	rip-metric	na	add
interface-outbound	destination	in	na
	protocol	is、in	na
	rip-metric	na	add

接続点	属性	一致	セット
redistribute	destination	in	na
	next-hop	in	set
	rip-metric	na	set
	rip-tag	na	set
	mpls-label	route-has-label	na
	route-type	is	na
	tag	is、eq、ge、le	set

PIM ポリシー接続点

この項では、PIM ポリシー **rpf-topology** 接続点について説明し、PIM 属性と演算子の概要を示します。

ルーティング ポリシーの非破壊編集

ルーティング ポリシーの非破壊編集によって、ルーティング ポリシー コンフィギュレーション モードでのデフォルトの終了動作が設定を中断するように変更されます。

デフォルトの **exit** コマンドは、end-policy、end-set、または end-if として機能します。**exit** コマンドがルート ポリシー コンフィギュレーション モードで実行される場合は、変更が適用され、設定が更新されます。これによって、既存のポリシーが破壊されます。**rpl set-exit-as-abort** コマンドを使用すると、ルート ポリシー コンフィギュレーション モードで **exit** コマンドのデフォルトの動作を上書きできます。

アタッチされたポリシーの変更

使用中のポリシーを変更する必要が生じる場合もあります。従来のコンフィギュレーション モデルでのポリシーの変更では、いったんポリシーを完全に削除してから再入力していました。しかし、このモデルではポリシーがアタッチされずデフォルトのアクションが使用される時間帯が生じてしまうため、不一致が発生する可能性がありました。この時間帯をなくすためには、接続点で使用中のポリシーを再指定することで変更を行います。使用中の変更対象ポリシーを、接続点にどのポリシーも適用されない時間帯をつくらずに変更できます。

（注） 接続点で使用中のルート ポリシーまたはセットは削除できません。削除によって未定義の参照が生じるからです。接続点で使用中のルート ポリシーまたはセットを削除しようとした場合、ユーザにはエラーメッセージが表示されます。

アタッチされないポリシーの変更

ポリシーは、接続点にアタッチされていないのであれば、存在していないセットやポリシーを参照することが許可されます。まだ定義されていない参照セットまたはポリシーブロックの設定を構築でき、その後、それらの定義されていないポリシーおよびセットに入力できます。この設定を構築する方法によって、ポリシー定義の柔軟性がより高まります。参照するポリシーの各部分は、ポリシーを定義しているときに設定に存在する必要はありません。このため、ポリシー sample2 が存在しない場合でも、apply ステートメントよってポリシー sample2 を参照するポリシー sample1 を定義できます。同様に、存在しないセットを参照するポリシー ステートメントを入力できます。

ただし、参照されているすべてのポリシーとセットの存在は、ポリシーが付加されたときに適用されます。このため、ステートメント **neighbor 1.2.3.4 address-family ipv4 unicast policy sample1 in** を使用してインバウンド BGP ポリシーで未定義ポリシー sample2 を参照するポリシー sample1 をアタッチしようとすると、ポリシー sample2 が存在しないために設定が拒否されます。

ルーティング ポリシー設定要素の編集

RPL は、行ではなくステートメントをベースにしています。つまり、CLI からのポリシー ステートメントをはさむ begin と end のペアの内部では、改行はセパレータでしかなく、スペース文字も同様です。

CLI によってルート ポリシー ステートメントの入力や削除が行えます。RPL では、begin と end の間にはさまれたポリシーの内容をテキスト エディタで編集する手順が準備されています。Cisco IOS XR では、RPL ポリシーの編集に次のテキスト エディタを利用できます。

- Nano（デフォルト）
- Emacs
- Vim

Nano エディタを使用したルーティング ポリシー設定要素の編集

Nano エディタを使用してルーティング ポリシーの内容を編集するには、EXEC モードで次の CLI コマンドを使用します。

```
edit route-policy
  name
  nano
```

ルート ポリシーのコピーが一時ファイルにコピーされ、エディタが起動します。編集後、Ctrl-X を入力してファイルを保存し、エディタを終了します。利用可能なエディタのコマンドは画面上に表示されます。

Nano エディタの使用について詳しくは、次の URL を参照してください。
http://www.nano-editor.org/

Cisco IOS XR ソフトウェアでは、Nano エディタの機能の一部がサポートされていません。

Emacs エディタを使用したルーティング ポリシー設定要素の編集

Emacs エディタを使用してルーティング ポリシーの内容を編集するには、EXEC モードで次の CLI コマンドを使用します。

```
edit
  route-policy
  name
  emacs
```

ルート ポリシーのコピーが一時ファイルにコピーされ、エディタが起動します。編集後に、Ctrl+X および Ctrl+S のキーストロークを使用して編集バッファを保存します。エディタを保存して終了するには、Ctrl+X および Ctrl+C のキーストロークを使用します。エディタを終了すると、バッファがコミットされます。解析エラーがなければ、コンフィギュレーションがコミットされます。

```
RP/0/RSP0/cpu 0: router# edit route-policy policy_A
----------------------------------------
== MicroEMACS 3.8b () == rpl_edit.139281 ==
  if destination in (2001::/8) then
    drop
  endif
end-policy
!

== MicroEMACS 3.8b () == rpl_edit.139281 ==
Parsing.
83 bytes parsed in 1 sec (82)bytes/sec
Committing.
1 items committed in 1 sec (0)items/sec
Updating.
Updated Commit database in 1 sec
```

解析エラーがある場合は、編集を続行するかどうかを尋ねられます。

```
RP/0/RSP0/cpu 0: router#edit route-policy policy_B
```

```
== MicroEMACS 3.8b () == rpl_edit.141738
route-policy policy_B
 set metric-type type_1
 if destination in (2001::/8) then
    drop
  endif
end-policy
!
== MicroEMACS 3.8b () == rpl_edit.141738 ==
Parsing.
105 bytes parsed in 1 sec (103)bytes/sec

% Syntax/Authorization errors in one or more commands.!! CONFIGURATION
FAILED DUE TO SYNTAX/AUTHORIZATION ERRORS
 set metric-type type_1
 if destination in (2001::/8) then
    drop
  endif
end-policy
!

Continue editing? [no]:
```

yes と答えると、エディタは、中断した場所からテキストバッファを続行します。**no** と答えると、実行コンフィギュレーションは変更されず、編集セッションは終了します。

Vim エディタを使用したルーティング ポリシー設定要素の編集

Vim（Vi IMproved）を使用したルーティング ポリシーの要素の編集は、保存や終了のキーストロークといった一部機能の違いを除き、Emacs での編集に似ています。現在のファイルに書き込んで終了するには、**:wq**、**:x**、または **ZZ** のキーストロークを使用します。終了して確認するには、**:q** キーストロークを使用します。終了して変更を廃棄するには、**:q!** キーストロークを使用します。

Vim の詳細なオンライン マニュアルは、次の URL から参照できます。http://www.vim.org/

CLI を使用したルーティング ポリシー設定要素の編集

CLI を使用して、ルート ポリシー ステートメントの入力や削除が行えます。ポリシー コンフィギュレーション ブロックは、**end-policy** や **end-set** といった適用可能なコマンドの入力によって完了できます。または、CLI インタープリタでは exit コマンドでポリシー コンフィギュレーション ブロックを完了させることも可能です。現在のポリシー コンフィギュレーションを廃棄して global configuration モードに戻るには、**abort** コマンドを使用します。

XML を使用したルーティング ポリシー設定要素の編集

RPL は、XML を使用した設定要素の編集をサポートしています。XML 経由で、既存のセットを置き換えることなくエントリの後方追加、前方追加、削除が行えます。

階層型ポリシー条件

階層型ポリシー条件機能は、他のルートポリシーの「if」ステートメント内のルートポリシーを指定する機能をイネーブルにします。この機能によって、ルートポリシーを階層的ポリシーに基づいたコンフィギュレーションに適用できます。

階層型ポリシー条件機能によって、Cisco IOS-XR RPL ではさまざまなマッチング文に加え、さまざまな種類の Boolean 演算子とともに使用できる条件ポリシーをサポートしています。

条件ポリシーの適用

Cisco IOS XR RPL がサポートする条件ポリシーでは、他のルート ポリシーの「if」ステートメント内のルート ポリシーを使用できます。

Parent、*Child A*、および *Child B* ルート ポリシー コンフィギュレーションを検討します。

```
route-policy Child A
 if destination in (10.10.0.0/16) then
  set local-pref 111
 endif
end-policy
!

route-policy Child B
 if as-path originates-from '222' then
  set community (333:222) additive
 endif
end-policy
!

route-policy Parent
 if apply Child A and apply Child B then
  set community (333:333) additive
 else
  set community (333:444) additive
 endif
end-policy
!
```

上記のシナリオでは、*Parent* ポリシーが実行されるたびに、ポリシー *Child A* および *Child B* の結果に基づいて、ポリシーの中の「if」条件の判断が選択されます。ポリシー *Parent* は、次に示すように、ポリシー *merged* に相当します。

```
route-policy merged
 if destination in (10.10.0.0/16) and as-path originates-from '222' then
  set local-pref 111
  set community (333:222, 333:333) additive
 elseif destination in (10.10.0.0/16) then /*Only Policy Child A is pass */
  set local-pref 111
  set community (333:444) additive /*From else block */
 elseif as-path originates-from '222' then /*Only Policy Child B is pass */
  set community (333:222, 333:444) additive /*From else block */
 else
  set community (333:444) additive /*From else block */
 endif
end-policy
```

条件の適用はパラメータとともに使用され、すべての接続点およびすべてのクライアントでサポートされます。条件の階層的適用は、カスケード レベル上で制約なく使用できます。

既存のルート ポリシー セマンティックは、この条件適用を含むように拡張できます。

```
Route-policy policy_name
 If apply policyA and apply policyB then
      Set med 100
 Else if not apply policyD then
      Set med 200
 Else
    Set med 300
 Endif
End-policy
```

単純な階層型ポリシーの pass/drop/done RPL ステートメントの動作

次の表は、単純な階層型ポリシーの **pass**/**drop**/**done** RPL ステートメントの動作、および単純な階層型ポリシーの考えられる **done** ステートメントの実行シーケンスを説明します。

単純な階層型ポリシーのあるルート ポリシー	考えられる **done** ステートメントの実行シーケンス	動作
pass	**pass** Continue_list	プレフィックスに「acceptable」としてマークを付け、continue_list ステートメントの実行を続けます。
drop	Stmts_list **drop**	**drop** ステートメントにヒットするとただちにルートを拒否し、ポリシー実行を停止します。
done	Stmts_list **done**	**done** ステートメントにヒットするとただちにルートを受け入れ、ポリシー実行を停止します。
pass それから **done**	**pass** Statement_list **done**	「accept route」のある **done** ステートメントでは、ただちに終了します。
drop それから **done**	**drop** Statement list **done**	これは、実行時点では無効なシナリオです。ポリシーはステートメント リストまたは **done** ステートメントには進まずに、**drop** ステートメント自体で実行を終了します。プレフィックスは、拒否またはドロップされます。

階層型ポリシー条件の pass/drop/done RPL ステートメントの動作

次の項では、階層型ポリシー条件の **pass**/**drop**/**done** RPL ステートメントの動作、および階層型ポリシー条件の考えられる **done** ステートメントの実行シーケンスを説明します。

ポリシー実行の用語：「true-path」と、「false-path」、および「continue-path」。

```
Route-policy parent
  If apply hierarchical_policy_condition then
     TRUE-PATH         : if hierarchical_policy_condition  returns TRUE then this path
will be executed.
  Else
     FALSE-PATH       : if hierarchical_policy_condition  returns FALSE then this path
will be executed.
  End-if
  CONTINUE-PATH            : Irrespective of the TRUE/FALSE this path will be executed.

 End-policy
```

階層型ポリシー条件	考えられる **done** ステートメントの実行シーケンス	動作
pass	**pass** Continue_list	戻り値を「true」とし、同じポリシー条件内で実行を続けます。 「**pass**」の後にステートメントがない場合は、「true」を返します。
pass それから **done**	**pass** または **set** アクション ステートメント Stmt_list **done**	戻り値を「true」とし、**done** ステートメントまで実行を続けます。「true-path」になる適用ポリシー条件に「true」を返します。
done	**pass** または **set** 操作のない Stmt_list DONE	「false」を返します。条件は「false-path」になります。
drop	Stmt_list **drop** Stmt_list	プレフィックスはドロップされるか拒否されます。

ネストされたワイルドカード適用ポリシー

ルーティングポリシー言語（RPL）の階層構造では、ポリシーが異なるポリシーを参照できます。参照されている、または呼び出されているポリシーは、子ポリシーと呼ばれます。別のポリシーを参照するまたは呼び出すポリシーは、親ポリシーと呼ばれます。呼び出すポリシーまたは親ポリシーは、BGP ネイバーの共通セットとの接続用の複数の子ポリシーをネストできま

す。ネストされたワイルドカード適用ポリシーでは、適用のネスティングに基づくワイルドカード（*）が可能です。ワイルドカード操作では、ルータ上に定義される、特定の定義済み英数字セットを含むポリシーすべてを呼び出す一般的な apply ステートメントを宣言することが許されています。

ワイルドカードは、apply ステートメントにポリシー名の最後にアスタリスク（*）を配置することによって指定されます。ワイルドカードポリシーに、パラメータを渡すことはサポートされていません。ワイルドカードは、適用ポリシーのその部分が任意の値と一致することを示します。

ネストされたワイルドカード適用ポリシーの例として、次のようなポリシー階層を考えてみます。

```
route-policy Nested_Wilcard
apply service_policy_customer*
end-policy

route-policy service_policy_customer_a
if destination in prfx_set_customer_a  then
set extcommunity rt (1:1) additive
endif
end-policy

route-policy service_policy_customer_b
if destination in prfx_set_customer_b then
set extcommunity rt (1:1) additive
endif
end-policy

route-policy service_policy_customer_c
if destination in prfx_set_customer_c then
set extcommunity rt (1:1) additive
endif
end-policy
```

ここで、単一の親 apply ステートメント（apply service_policy_customer*）が、指定された文字列「service_policy_customer」を含むすべての子ポリシーを呼び出して（継承して）います。それぞれの子ポリシーをグローバルに定義されると、親ポリシーは動的にポリシー名に基づいて子ポリシーをネストします。親ポリシーを設定すると、各子ポリシーをオンデマンドで継承します。親ポリシーと子ポリシーの間に、ワイルドカード一致ステートメント以上の直接的な関連付けはありません。

ルートポリシーセットのワイルドカード

ルートポリシーは、モジュール型の形式で定義され、比較ステートメントのセットで構成されます。ワイルドカードを使用してセットの範囲を定義すると、ポリシーの複雑さが大幅に軽減されます。

ワイルドカードは、さまざまなプレフィックスセット、コミュニティセット、ASパスセット、または拡張コミュニティセットを定義するために使用できます。ポリシーセットでワイルドカードの使用する方法については、ルーティングポリシーセットに対するワイルドカードの使用 （628 ページ）を参照してください。

ルーティングポリシーセットに対するワイルドカードの使用

この項では、ワイルドカードを使用してルーティングポリシーセットを設定する例について説明します。

プレフィックスセットに対するワイルドカードの使用

次の例に示すように、プレフィックスセットにワイルドカードを使用してルーティングポリシーを設定します。

1. グローバルコンフィギュレーションモードで必要なプレフィックスセットを設定します。

```
RP/0/RSP0/cpu 0: router(config)# prefix-set pfx_set1
RP/0/RSP0/cpu 0: router(config-pfx)# 1.2.3.4/32
RP/0/RSP0/cpu 0: router(config-pfx)# end-set
RP/0/RSP0/cpu 0: router(config)# prefix-set pfx_set2
RP/0/RSP0/cpu 0: router(config-pfx)# 2.2.2.2/32
RP/0/RSP0/cpu 0: router(config-pfx)# end-set
```

2. プレフィックスセットを参照するように、ワイルドカードを使用してルートポリシーを設定します。

```
RP/0/RSP0/cpu 0: router(config)# route-policy WILDCARD_PREFIX_SET
RP/0/RSP0/cpu 0: router(config-rpl)# if destination in prefix-set* then pass else
drop endif
RP/0/RSP0/cpu 0: router(config-rpl)# end-policy
```

   このルートポリシー設定は、2 つのプレフィックスセットに記載されているプレフィックスを持つルートを受け入れ、一致しない他のすべてのルートをドロップします。

3. 設定をコミットします。

```
RP/0/RSP0/cpu 0: router(config)# commit
```

これで、プレフィックスセットにワイルドカードを使用したルーティングポリシーの設定が完了します。プレフィックスセットの詳細については、prefix-set （556 ページ）を参照してください。

AS パスセットに対するワイルドカードの使用

次の例に示すように、AS パスセットにワイルドカードを使用してルーティングポリシーを設定します。

1. グローバルコン フィギュレーション モードで必要な AS パスセットを設定します。

```
RP/0/RSP0/cpu 0: router(config)# as-path-set AS_SET1
RP/0/RSP0/cpu 0: router(config-as)# ios-regex '_22$',
RP/0/RSP0/cpu 0: router(config-as)# ios-regex '_25$'
RP/0/RSP0/cpu 0: router(config-as)# end-set
RP/0/RSP0/cpu 0: router(config)# as-path-set AS_SET2
RP/0/RSP0/cpu 0: router(config-as)# ios-regex '_42$',
RP/0/RSP0/cpu 0: router(config-as)# ios-regex '_47$'
RP/0/RSP0/cpu 0: router(config-as)# end-set
```

2. AS パスセットを参照するように、ワイルドカードを使用してルートポリシーを設定します。

```
RP/0/RSP0/cpu 0: router(config)# route-policy WILDCARD_AS_SET
RP/0/RSP0/cpu 0: router(config-rpl)# if as-path in as-path-set* then pass else drop
 endif
RP/0/RSP0/cpu 0: router(config-rpl)# end-policy
```

このルートポリシー設定では、2 つのパスセットで説明したように、AS パス 属性を持つルートを受け入れ、一致しないその他すべてのルートをドロップします。

3. 設定をコミットします。

```
RP/0/RSP0/cpu 0: router(config)# commit
```

これで、AS パスセットにワイルドカードを使用したルーティングポリシーの設定が完了します。AS パスセットの詳細については、as-path-set （550 ページ）を参照してください。

コミュニティセットに対するワイルドカードの使用

次の例に示すように、コミュニティセットにワイルドカードを使用してルーティングポリシーを設定します。

1. グローバルコン フィギュレーション モードで必要なコミュニティセットを設定します。

```
RP/0/RSP0/cpu 0: router(config)# community-set CSET1
RP/0/RSP0/cpu 0: router(config-comm)# 12:24,
RP/0/RSP0/cpu 0: router(config-comm)# 12:36,
RP/0/RSP0/cpu 0: router(config-comm)# 12:72
RP/0/RSP0/cpu 0: router(config-comm)# end-set
RP/0/RSP0/cpu 0: router(config)# community-set CSET2
RP/0/RSP0/cpu 0: router(config-comm)# 24:12,
RP/0/RSP0/cpu 0: router(config-comm)# 24:42,
RP/0/RSP0/cpu 0: router(config-comm)# 24:64
RP/0/RSP0/cpu 0: router(config-comm)# end-set
```

2. コミュニティセットを参照するように、ワイルドカードを使用してルートポリシーを設定します。

```
RP/0/RSP0/cpu 0: router(config)# route-policy WILDCARD_COMMUNITY_SET
RP/0/RSP0/cpu 0: router(config-rpl)# if community matches-any community-set* then
pass else drop endif
RP/0/RSP0/cpu 0: router(config-rpl)# end-policy
```

このルートポリシー設定は、コミュニティセットに記載されているコミュニティセット値を持つルートを受け入れ、一致しない他のすべてのルートをドロップします。

3. 設定をコミットします。

```
RP/0/RSP0/cpu 0: router(config)# commit
```

これで、コミュニティセットにワイルドカードを使用したルーティングポリシーの設定が完了します。コミュニティセットの詳細については、community-set （550 ページ）を参照してください。

拡張コミュニティセットに対するワイルドカードの使用

次の例に示すように、拡張コミュニティセットにワイルドカードを使用してルーティングポリシーを設定します。

1. グローバルコン フィギュレーション モードで必要な拡張コミュニティセットを設定します。

```
RP/0/RSP0/cpu 0: router(config)# extcommunity-set rt RT_SET1
RP/0/RSP0/cpu 0: router(config-ext)# 1.2.3.4:555,
RP/0/RSP0/cpu 0: router(config-ext)# 1234:555
RP/0/RSP0/cpu 0: router(config-ext)# end-set
RP/0/RSP0/cpu 0: router(config)# extcommunity-set rt RT_SET2
RP/0/RSP0/cpu 0: router(config-ext)# 1.1.1.1:777,
RP/0/RSP0/cpu 0: router(config-ext)# 1111:777
RP/0/RSP0/cpu 0: router(config-ext)# end-set
```

2. 拡張コミュニティセットを参照するように、ワイルドカードを使用してルートポリシーを設定します。

```
RP/0/RSP0/cpu 0: router(config)# route-policy WILDCARD_EXT_COMMUNITY_SET
RP/0/RSP0/cpu 0: router(config-rpl)# if extcommunity rt matches-any extcommunity-set*
 then pass else drop endif
RP/0/RSP0/cpu 0: router(config-rpl)# end-policy
```

   このルートポリシー設定は、拡張コミュニティセットに記載されている拡張コミュニティセット値を持つルートを受け入れ、一致しない他のすべてのルートをドロップします。

3. 設定をコミットします。

```
RP/0/RSP0/cpu 0: router(config)# commit
```

これで、拡張コミュニティセットにワイルドカードを使用したルーティングポリシーの設定が完了します。拡張コミュニティセットの詳細については、extcommunity-set（551 ページ）を参照してください。

ルート識別子セットに対するワイルドカードの使用

次の例に示すように、ルート識別子セットにワイルドカードを使用してルーティングポリシーを設定します。

1. グローバルコン フィギュレーション モードで必要なルート識別子セットを設定します。

```
RP/0/RSP0/cpu 0: router(config)# rd-set rd_set_demo
RP/0/RSP0/cpu 0: router(config-rd)# 10.0.0.1/8:77,
RP/0/RSP0/cpu 0: router(config-rd)# 10.0.0.2:888,
RP/0/RSP0/cpu 0: router(config-rd)# 65000:777
RP/0/RSP0/cpu 0: router(config-rd)# end-set
RP/0/RSP0/cpu 0: router(config)# rd-set rd_set_demo2
RP/0/RSP0/cpu 0: router(config-rd)# 20.0.0.1/7:99,
RP/0/RSP0/cpu 0: router(config-rd)# 4784:199
RP/0/RSP0/cpu 0: router(config-rd)# end-set
```

2. ルート識別子セットを参照するように、ワイルドカードを使用してルートポリシーを設定します。

```
RP/0/RSP0/cpu 0: router(config)# route-policy use_rd_set
RP/0/RSP0/cpu 0: router(config-rpl)# if rd in rd-set* then set local-preference 100
RP/0/RSP0/cpu 0: router(config-rpl-if)# elseif rd in(10.0.0.2:888, 10.0.0.2:999)then
 set local-preference 300
RP/0/RSP0/cpu 0: router(config-rpl-elseif)# endif
RP/0/RSP0/cpu 0: router(config-rpl)# end-policy
```

3. 設定をコミットします。

```
RP/0/RSP0/cpu 0: router(config)# commit
```

4. （任意）設定を確認します。

```
RP/0/RSP0/cpu 0: router(config)# show configuration
...
Building configuration...
!! IOS XR Configuration 0.0.0
!
rd-set rd_set_demo
  10.0.0.1/8:77,
  10.0.0.2:888,
  65000:777
end-set
!
!
rd-set rd_set_demo2
  20.0.0.1/7:99,
  4784:199
end-set
!

route-policy use_rd_set
  if rd in rd-set* then
    set local-preference 100
  elseif rd in (10.0.0.2:888, 10.0.0.2:999) then
    set local-preference 300
  endif
end-policy
!
end
```

これで、ルート識別子セットにワイルドカードを使用したルーティングポリシーの設定が完了します。ルート識別子セットの詳細についてはrd-set（558 ページ）、を参照してください。

OSPF エリアセットに対するワイルドカードの使用

次の例に示すように、OSPF エリアセットにワイルドカードを使用してルーティングポリシーを設定します。

1. グローバルコン フィギュレーション モードで必要な OSPF エリアセットを設定します。

```
RP/0/RSP0/cpu 0: router(config)# ospf-area-set ospf_area_set_demo1
RP/0/RSP0/cpu 0: router(config-ospf-area)# 10.0.0.1,
RP/0/RSP0/cpu 0: router(config-ospf-area)# 3553
RP/0/RSP0/cpu 0: router(config-ospf-area)# end-set

RP/0/RSP0/cpu 0: router(config)# ospf-area-set ospf_area_set_demo2
RP/0/RSP0/cpu 0: router(config-ospf-area)# 20.0.0.2,
```

```
RP/0/RSP0/cpu 0: router(config-ospf-area)# 3673
RP/0/RSP0/cpu 0: router(config-ospf-area)# end-set
```

2. OSPF エリアセットを参照するように、ワイルドカードを使用してルートポリシーを設定します。

```
RP/0/RSP0/cpu 0: router(config)# route-policy use_ospf_area_set
RP/0/RSP0/cpu 0: router(config-rpl)# if ospf-area in ospf-area-set* then set
ospf-metric 200
RP/0/RSP0/cpu 0: router(config-rpl-if)# elseif ospf-area in( 10.0.0.1, 10.0.0.2 )then
 set ospf-metric 300
RP/0/RSP0/cpu 0: router(config-rpl-elseif)# endif
RP/0/RSP0/cpu 0: router(config-rpl)# end-policy
```

3. 設定をコミットします。

```
RP/0/RSP0/cpu 0: router(config)# commit
```

4. （任意）設定を確認します。

```
RP/0/RSP0/cpu 0: router(config)# show configuration
Building configuration...
!! IOS XR Configuration 0.0.0
!
ospf-area-set ospf_area_set_demo1
  10.0.0.1,
  3553
end-set
!
!
ospf-area-set ospf_area_set_demo2
  20.0.0.2,
  3673
end-set
!

route-policy use_ospf_area_set
  if ospf-area in ospf-area-set* then
    set ospf-metric 200
  elseif ospf-area in (10.0.0.1, 10.0.0.2) then
    set ospf-metric 300
  endif
end-policy
!
end
```

これで、OSPF エリアセットにワイルドカードを使用したルーティングポリシーの設定が完了します。

VRF インポート ポリシーの強化

VRF RPL ベースのインポート ポリシー機能では、ルートターゲット（RT）およびポリシー内で指定された他の基準を照合することによって、インポートルートポリシーだけに基づいてインポート操作を実行する機能が提供されます。グローバル VRF アドレス ファミリ コンフィギュレーション モードでインポート RT を明示的に設定する必要はありません。インポート RT およびインポート ルートポリシーがすでに定義されている場合、ルートはインポート RT

で設定された RT からインポートされ、インポート ルートポリシーで接続されたルートポリシーに従います。

この機能を有効にするには、VPN アドレスファミリ コンフィギュレーション モードの VRF サブモード下で **source rt import-policy** コマンドを使用します。

フレキシブル L3VPN ラベル割り当てモード

フレキシブル L3VPN ラベル割り当て機能は、ルートポリシーを使用してラベル割り当てモードを設定する機能を提供します。この場合、異なる割り当てモードを異なるプレフィックスのセットに使用できます。したがって、プレフィックス値やコミュニティなどの任意の一致基準に基づいてラベルモードを選択できます。

プレフィックス値に基づいて MPLS/VPN ラベルモードを設定するには、**label mode** コマンドを使用します。ラベルモード接続点を使用すると、任意の基準に基づいてラベルモードを選択できます。

集約されたルートの照合

集約されたルートの照合機能は、集約されていないルートから BGP 集約ルートを照合するのに役立ちます。BGP は、ネイバーに更新を送信する前に、ルートのグループを 1 つのプレフィックスに集約できます。集約されたルートの照合機能を使用して、ルート ポリシーはこの集約されたルートを他のルートから切り離します。

着信ポリシーでのプライベート AS の削除

BGPは、ネイバーにパケットを送信する前に、自身の as-path を付加します。パケットが複数の iBGP ネイバーを通過すると、as-path 構造には、多くのプライベート自律システム（AS）が追加されます。着信ポリシーでのプライベート AS の削除では、RPL route-policy を使用してそれらのプライベート自律システムを削除する機能が与えられます。 **remove as-path private-as** コマンドを使用すると、AS 番号が 64512 ～ 65535 の自律システム（AS）が削除されます。

アドミニストレーティブ ディスタンスの設定

アドミニストレーティブ ディスタンスの設定では、BGP の個別のプレフィックス単位のアドミニストレーティブ ディスタンスを変更します。RIB に同じ宛先への 2 つのルートがある場合、RIB は転送にアドミニストレーティブ ディスタンスが低いルートを選択します。**set-administrative-distance** コマンドは、必要なルートを RIB が選択するように、BGP ルートのアドミニストレーティブ ディスタンスに値を設定します。

ルーティング ポリシーの実装方法

ここでは、次の手順について説明します。

ルート ポリシーの定義

ここでは、ルート ポリシーを定義する方法について説明します。

（注）
- Command-Line Interface（CLI）を使用して既存のルーティング ポリシーを変更する場合、このタスクを実行してポリシーを再定義する必要があります。
- RPL のスケール設定の変更には、時間がかかる場合があります。
- BGP は、大規模な RPL 設定の変更が原因でクラッシュしたり、または連続した RPL の変更時にクラッシュする可能性があります。BGP のクラッシュを回避するには、以降の変更をコミットする前に、BGP In/Out キュー内にメッセージがなくなるまで待機します。

手順の概要

1. **configure**
2. **route-policy** *name* [*parameter1* , *parameter2* , . . . , *parameterN*]
3. **end-policy**
4. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**route-policy** *name* [*parameter1* , *parameter2* , . . . , *parameterN*] 例： `RP/0/RSP0/cpu 0: router(config)# route-policy sample1`	ルートポリシー コンフィギュレーション モードを開始します。 • ルート ポリシーの開始後、ルート ポリシーを定義する一連のコマンドを入力できます。
ステップ 3	**end-policy** 例： `RP/0/RSP0/cpu 0: router(config-rpl)# end-policy`	ルートポリシーの定義を終了して、ルートポリシー コンフィギュレーション モードを終了します。
ステップ 4	**commit**	

ルーティング ポリシーの BGP ネイバーへのアタッチ

このタスクでは、ルーティング ポリシーの BGP ネイバーへのアタッチ方法を説明します。

始める前に

ルーティングポリシーは、接続点に適用される前に、設定を済ませよく定義しておく必要があります。ポリシーが事前に設定されていない場合、ポリシーが定義されていないことを示すエラー メッセージが生成されます。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **neighbor** *ip-address*
4. **address-family** { **ipv4 unicast** | **ipv4 multicast** | **ipv4 labeled-unicast** | **ipv4 tunnel** | **ipv4 mdt** | **ipv6 unicast** | **ipv6 multicast** | **ipv6 labeled-unicast** | **vpnv4 unicast** | **vpnv6 unicast** }
5. **route-policy** *policy-name* { **in** | **out** }
6. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router bgp** *as-number* 例： `RP/0/RSP0/cpu 0: router(config)# router bgp 125`	BGP ルーティング プロセスを設定し、ルータ コンフィギュレーション モードを開始します。 • *as-number* 引数は、ルータが存在する自律システムを識別します。有効値は、0～65535 です。内部ネットワークで使用できるプライベート自律システム番号の範囲は、64512～65535 です。
ステップ 3	**neighbor** *ip-address* 例： `RP/0/RSP0/cpu 0: router(config-bgp)# neighbor 10.0.0.20`	ネイバー IP アドレスを指定します。
ステップ 4	**address-family** { **ipv4 unicast** | **ipv4 multicast** | **ipv4 labeled-unicast** | **ipv4 tunnel** | **ipv4 mdt** | **ipv6 unicast** | **ipv6 multicast** | **ipv6 labeled-unicast** | **vpnv4 unicast** | **vpnv6 unicast** } 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr)# address-family ipv4 unicast`	アドレス ファミリを指定します。
ステップ 5	**route-policy** *policy-name* { **in** | **out** } 例： `RP/0/RSP0/cpu 0: router(config-bgp-nbr-af)# route-policy example1 in`	ルートポリシーをアタッチします。事前に作成と定義を済ませておく必要があります。

	コマンドまたはアクション	目的
ステップ 6	**commit**	

テキスト エディタを使用したルーティング ポリシーの変更

このタスクでは、テキスト エディタを使用して既存のルーティング ポリシーを変更する方法を説明します。テキスト エディタの詳細については、ルーティング ポリシー設定要素の編集（621 ページ）を参照してください。

手順の概要

1. **edit** { **route-policy** | **prefix-set** | **as-path-set** | **community-set** | **extcommunity-set** { **rt** | **soo** } | **policy-global** | **rd-set** } *name* [**nano** | **emacs** | **vim** | **inline** { **add** | **prepend** | **remove** } *set-element*]
2. **show rpl route-policy** [*name* [**detail**] | **states** | **brief**]
3. **show rpl prefix-set** [*name* | **states** | **brief**]

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**edit** { **route-policy** | **prefix-set** | **as-path-set** | **community-set** | **extcommunity-set** { **rt** | **soo** } | **policy-global** | **rd-set** } *name* [**nano** | **emacs** | **vim** | **inline** { **add** | **prepend** | **remove** } *set-element*] 例： `RP/0/RSP0/cpu 0: router# edit route-policy sample1`	変更するルート ポリシー、プレフィックス セット、AS パス セット、コミュニティ セット、または拡張コミュニティ セットの名前を指定します。 • ルート ポリシー、プレフィックス セット、AS パス セット、コミュニティ セット、または拡張コミュニティ セットのコピーが一時ファイルとして作成され、エディタが起動します。 • Nano での編集後に、エディタ バッファを保存し、Ctrl+X キーストロークを使用してエディタを終了します。 • Emacs での編集後に、Ctrl+X および Ctrl+S のキーストロークを使用して編集バッファを保存します。エディタを保存して終了するには、Ctrl+X および Ctrl+C のキーストロークを使用します。 • Vim での編集後に、現在のファイルに書き込んで終了するには、:wq、:x、または ZZ のキーストロークを使用します。終了して確認するには、:q キーストロークを使用します。終了して変更を廃棄するには、:q! キーストロークを使用します。

	コマンドまたはアクション	目的
ステップ 2	**show rpl route-policy** [*name* [**detail**] | **states** | **brief**] 例： `RP/0/RSP0/cpu 0: router# show rpl route-policy sample2`	（任意）特定の名前付きルート ポリシーのコンフィギュレーションを表示します。 • ポリシーが使用するすべてのポリシーとセットを表示するには **detail** キーワードを使用します。 • 未使用、非アクティブ、アクティブ状態のものをすべて表示するには **states** キーワードを使用します。 • すべての拡張コミュニティ セットの名前を設定なしでリストするには、 **brief** キーワードを使用します。
ステップ 3	**show rpl prefix-set** [*name* | **states** | **brief**] 例： `RP/0/RSP0/cpu 0: router# show rpl prefix-set prefixset1`	（任意）名前付きプレフィックス セットの内容を表示します。 • 名前付き AS パス セット、コミュニティ セット、拡張コミュニティ セットの内容を表示するには **prefix-set** キーワードをそれぞれ **as-path-set** 、 **community-set** 、または **extcommunity-set** にそれぞれ置き換えます。

ルーティング ポリシーの実装の設定例

ここでは、次の設定例について説明します。

ルーティング ポリシー定義：例

次の例では、route-policy name コマンドを使用して、sample1 という BGP ルート ポリシーが定義されます。ポリシーはネットワーク層到達可能性情報（NLRI）をプレフィックス セット test 内の要素と比較します。真と評価されると、ポリシーは *then* 句の中の操作を実行します。偽と評価されると、ポリシーは *else* 句の中の操作を実行します。つまり、MED 値を 200 とし、ルートにコミュニティ 2:100 を追加します。例の最終段階では、コンフィギュレーションをルータにコミットし、コンフィギュレーション モードを終了してルート ポリシー sample1 の内容を表示します。

```
configure
 route-policy sample1
  if destination in test then
   drop
  else
   set med 200
   set community (2:100) additive
```

```
  endif
 end-policy
 end
show config running route-policy sample1
Building configuration...
 route-policy sample1
  if destination in test then
   drop
  else
   set med 200
   set community (2:100) additive
  endif
 end-policy
```

シンプル インバウンド ポリシー：例

次のポリシーは、ネットワーク層到達可能性情報（NLRI）が /24 よりも長いプレフィックスを指定するルート、および NLRI が RFC 1918 によって予約されているアドレス空間の宛先を指定するルートを廃棄します。残りのルートすべてに対しては、MED とローカル プリファレンスを設定し、ルートのリストにコミュニティを追加します。

コミュニティ リストに 101:202～106:202 の範囲の値を含み、値 202 を含む 16 ビット タグ部分を持つルートの場合、ポリシーは、自律システム番号 2 を先頭に 2 回追加し、ルートのリストにコミュニティ 2:666 を追加します。これらのルートに対して、MED が 666 または 225 のいずれかである場合、ポリシーはルートの送信元を不完全に設定し、それ以外の場合は IGP に設定します。

コミュニティ リストが範囲 101:202 ～ 106:202 内の値を含まない場合、ポリシーはルート内のリストにコミュニティ 2:999 を追加します。

```
prefix-set too-specific
 0.0.0.0/0 ge 25 le 32
end-set

prefix-set rfc1918
 10.0.0.0/8 le 32,
 172.16.0.0/12 le 32,
 192.168.0.0/16 le 32
end-set

route-policy inbound-tx
 if destination in too-specific or destination in rfc1918 then
  drop
 endif
 set med 1000
 set local-preference 90
 set community (2:1001) additive
 if community matches-any ([101..106]:202) then
  prepend as-path 2.30 2
  set community (2:666) additive
 if med is 666 or med is 225 then
  set origin incomplete
 else
  set origin igp
 endif
else
  set community (2:999) additive
 endif
```

```
end-policy

router bgp 2
 neighbor 10.0.1.2 address-family ipv4 unicast route-policy inbound-tx in
```

モジュール型インバウンド ポリシー：例

次のポリシーの例に、2 つの異なるピア向けに 2 つのインバウンド ポリシー in-100 と in-101 の作成方法を示します。それらのピアに特定のポリシーを作成するとき、ポリシーは複数のピアに共通する可能性があるポリシーの共通ブロックを再利用します。いくつかの基本的な構築ブロックとして、policies common-inbound、filter-bogons、set-lpref-prepend が作成されます。

filter-bogons 構築ブロックは、RFC 1918 アドレス空間からのルートといった不要なルートをフィルタするシンプルなポリシーです。ポリシー set-lpref-prepend は、渡されるパラメータ化された値に応じて、ローカル プリファレンスを設定し、その先頭に AS パスを追加できるユーティリティ ポリシーです。common-inbound ポリシーは、これらの filter-bogons 構築ブロックを使用して、インバウンド ポリシーの共通ブロックを構築します。common-inbound ポリシーは、in-100 と in-101 作成の構築ブロックとして、set-lpref-prepend 構築ブロックとともに使用されます。

これは、ポリシー言語のモジュール化機能を示すシンプルな例だといえます。

```
prefix-set bogon
  10.0.0.0/8 ge 8 le 32,
  0.0.0.0,
  0.0.0.0/0 ge 27 le 32,
  192.168.0.0/16 ge 16 le 32
end-set
!
route-policy in-100
  apply common-inbound
  if community matches-any ([100..120]:135) then
    apply set-lpref-prepend (100,100,2)
    set community (2:1234) additive
  else
    set local-preference 110
  endif
  if community matches-any ([100..666]:[100..999]) then
    set med 444
    set local-preference 200
    set community (no-export) additive
  endif
end-policy
!
route-policy in-101
  apply common-inbound
  if community matches-any ([101..200]:201) then
    apply set-lpref-prepend(100,101,2)
    set community (2:1234) additive
  else
    set local-preference 125
  endif
end-policy
!
route-policy filter-bogons
  if destination in bogon then
drop
else
```

```
pass
  endif
end-policy
!
route-policy common-inbound
  apply filter-bogons
  set origin igp
  set community (2:333)
end-policy
!
route-policy set-lpref-prepend($lpref,$as,$prependcnt)
  set local-preference $lpref
  prepend as-path $as $prependcnt
end-policy
```

ルーティングポリシーセットに対するワイルドカードの使用

この項では、ワイルドカードを使用してルーティングポリシーセットを設定する例について説明します。

プレフィックスセットに対するワイルドカードの使用

次の例に示すように、プレフィックスセットにワイルドカードを使用してルーティングポリシーを設定します。

1. グローバルコンフィギュレーションモードで必要なプレフィックスセットを設定します。

```
RP/0/RSP0/cpu 0: router(config)# prefix-set pfx_set1
RP/0/RSP0/cpu 0: router(config-pfx)# 1.2.3.4/32
RP/0/RSP0/cpu 0: router(config-pfx)# end-set
RP/0/RSP0/cpu 0: router(config)# prefix-set pfx_set2
RP/0/RSP0/cpu 0: router(config-pfx)# 2.2.2.2/32
RP/0/RSP0/cpu 0: router(config-pfx)# end-set
```

2. プレフィックスセットを参照するように、ワイルドカードを使用してルートポリシーを設定します。

```
RP/0/RSP0/cpu 0: router(config)# route-policy WILDCARD_PREFIX_SET
RP/0/RSP0/cpu 0: router(config-rpl)# if destination in prefix-set* then pass else
drop endif
RP/0/RSP0/cpu 0: router(config-rpl)# end-policy
```

   このルートポリシー設定は、2 つのプレフィックスセットに記載されているプレフィックスを持つルートを受け入れ、一致しない他のすべてのルートをドロップします。

3. 設定をコミットします。

```
RP/0/RSP0/cpu 0: router(config)# commit
```

これで、プレフィックスセットにワイルドカードを使用したルーティングポリシーの設定が完了します。プレフィックスセットの詳細については、prefix-set （556 ページ）を参照してください。

AS パスセットに対するワイルドカードの使用

次の例に示すように、AS パスセットにワイルドカードを使用してルーティングポリシーを設定します。

1. グローバルコン フィギュレーション モードで必要な AS パスセットを設定します。

```
RP/0/RSP0/cpu 0: router(config)# as-path-set AS_SET1
RP/0/RSP0/cpu 0: router(config-as)# ios-regex '_22$',
RP/0/RSP0/cpu 0: router(config-as)# ios-regex '_25$'
RP/0/RSP0/cpu 0: router(config-as)# end-set
RP/0/RSP0/cpu 0: router(config)# as-path-set AS_SET2
RP/0/RSP0/cpu 0: router(config-as)# ios-regex '_42$',
RP/0/RSP0/cpu 0: router(config-as)# ios-regex '_47$'
RP/0/RSP0/cpu 0: router(config-as)# end-set
```

2. AS パスセットを参照するように、ワイルドカードを使用してルートポリシーを設定します。

```
RP/0/RSP0/cpu 0: router(config)# route-policy WILDCARD_AS_SET
RP/0/RSP0/cpu 0: router(config-rpl)# if as-path in as-path-set* then pass else drop
 endif
RP/0/RSP0/cpu 0: router(config-rpl)# end-policy
```

このルートポリシー設定では、2 つのパスセットで説明したように、AS パス 属性を持つルートを受け入れ、一致しないその他すべてのルートをドロップします。

3. 設定をコミットします。

```
RP/0/RSP0/cpu 0: router(config)# commit
```

これで、AS パスセットにワイルドカードを使用したルーティングポリシーの設定が完了します。AS パスセットの詳細については、as-path-set （550 ページ）を参照してください。

コミュニティセットに対するワイルドカードの使用

次の例に示すように、コミュニティセットにワイルドカードを使用してルーティングポリシーを設定します。

1. グローバルコン フィギュレーション モードで必要なコミュニティセットを設定します。

```
RP/0/RSP0/cpu 0: router(config)# community-set CSET1
RP/0/RSP0/cpu 0: router(config-comm)# 12:24,
RP/0/RSP0/cpu 0: router(config-comm)# 12:36,
RP/0/RSP0/cpu 0: router(config-comm)# 12:72
RP/0/RSP0/cpu 0: router(config-comm)# end-set
RP/0/RSP0/cpu 0: router(config)# community-set CSET2
RP/0/RSP0/cpu 0: router(config-comm)# 24:12,
RP/0/RSP0/cpu 0: router(config-comm)# 24:42,
RP/0/RSP0/cpu 0: router(config-comm)# 24:64
RP/0/RSP0/cpu 0: router(config-comm)# end-set
```

2. コミュニティセットを参照するように、ワイルドカードを使用してルートポリシーを設定します。

```
RP/0/RSP0/cpu 0: router(config)# route-policy WILDCARD_COMMUNITY_SET
```

```
RP/0/RSP0/cpu 0: router(config-rpl)# if community matches-any community-set* then
pass else drop endif
RP/0/RSP0/cpu 0: router(config-rpl)# end-policy
```

このルートポリシー設定は、コミュニティセットに記載されているコミュニティセット値を持つルートを受け入れ、一致しない他のすべてのルートをドロップします。

3. 設定をコミットします。

```
RP/0/RSP0/cpu 0: router(config)# commit
```

これで、コミュニティセットにワイルドカードを使用したルーティングポリシーの設定が完了します。コミュニティセットの詳細については、community-set（550ページ）を参照してください。

拡張コミュニティセットに対するワイルドカードの使用

次の例に示すように、拡張コミュニティセットにワイルドカードを使用してルーティングポリシーを設定します。

1. グローバルコン フィギュレーション モードで必要な拡張コミュニティセットを設定します。

```
RP/0/RSP0/cpu 0: router(config)# extcommunity-set rt RT_SET1
RP/0/RSP0/cpu 0: router(config-ext)# 1.2.3.4:555,
RP/0/RSP0/cpu 0: router(config-ext)# 1234:555
RP/0/RSP0/cpu 0: router(config-ext)# end-set
RP/0/RSP0/cpu 0: router(config)# extcommunity-set rt RT_SET2
RP/0/RSP0/cpu 0: router(config-ext)# 1.1.1.1:777,
RP/0/RSP0/cpu 0: router(config-ext)# 1111:777
RP/0/RSP0/cpu 0: router(config-ext)# end-set
```

2. 拡張コミュニティセットを参照するように、ワイルドカードを使用してルートポリシーを設定します。

```
RP/0/RSP0/cpu 0: router(config)# route-policy WILDCARD_EXT_COMMUNITY_SET
RP/0/RSP0/cpu 0: router(config-rpl)# if extcommunity rt matches-any extcommunity-set*
 then pass else drop endif
RP/0/RSP0/cpu 0: router(config-rpl)# end-policy
```

このルートポリシー設定は、拡張コミュニティセットに記載されている拡張コミュニティセット値を持つルートを受け入れ、一致しない他のすべてのルートをドロップします。

3. 設定をコミットします。

```
RP/0/RSP0/cpu 0: router(config)# commit
```

これで、拡張コミュニティセットにワイルドカードを使用したルーティングポリシーの設定が完了します。拡張コミュニティセットの詳細については、extcommunity-set（551ページ）を参照してください。

ルート識別子セットに対するワイルドカードの使用

次の例に示すように、ルート識別子セットにワイルドカードを使用してルーティングポリシーを設定します。

1. グローバルコン フィギュレーション モードで必要なルート識別子セットを設定します。

```
RP/0/RSP0/cpu 0: router(config)# rd-set rd_set_demo
RP/0/RSP0/cpu 0: router(config-rd)# 10.0.0.1/8:77,
RP/0/RSP0/cpu 0: router(config-rd)# 10.0.0.2:888,
RP/0/RSP0/cpu 0: router(config-rd)# 65000:777
RP/0/RSP0/cpu 0: router(config-rd)# end-set
RP/0/RSP0/cpu 0: router(config)# rd-set rd_set_demo2
RP/0/RSP0/cpu 0: router(config-rd)# 20.0.0.1/7:99,
RP/0/RSP0/cpu 0: router(config-rd)# 4784:199
RP/0/RSP0/cpu 0: router(config-rd)# end-set
```

2. ルート識別子セットを参照するように、ワイルドカードを使用してルートポリシーを設定します。

```
RP/0/RSP0/cpu 0: router(config)# route-policy use_rd_set
RP/0/RSP0/cpu 0: router(config-rpl)# if rd in rd-set* then set local-preference 100
RP/0/RSP0/cpu 0: router(config-rpl-if)# elseif rd in(10.0.0.2:888, 10.0.0.2:999)then
 set local-preference 300
RP/0/RSP0/cpu 0: router(config-rpl-elseif)# endif
RP/0/RSP0/cpu 0: router(config-rpl)# end-policy
```

3. 設定をコミットします。

```
RP/0/RSP0/cpu 0: router(config)# commit
```

4. （任意）設定を確認します。

```
RP/0/RSP0/cpu 0: router(config)# show configuration
...
Building configuration...
!! IOS XR Configuration 0.0.0
!
rd-set rd_set_demo
  10.0.0.1/8:77,
  10.0.0.2:888,
  65000:777
end-set
!
!
rd-set rd_set_demo2
  20.0.0.1/7:99,
  4784:199
end-set
!

route-policy use_rd_set
  if rd in rd-set* then
    set local-preference 100
  elseif rd in (10.0.0.2:888, 10.0.0.2:999) then
    set local-preference 300
  endif
end-policy
!
end
```

これで、ルート識別子セットにワイルドカードを使用したルーティングポリシーの設定が完了します。ルート識別子セットの詳細についてはrd-set（558ページ）、を参照してください。

OSPF エリアセットに対するワイルドカードの使用

次の例に示すように、OSPF エリアセットにワイルドカードを使用してルーティングポリシーを設定します。

1. グローバルコン フィギュレーション モードで必要な OSPF エリアセットを設定します。

```
RP/0/RSP0/cpu 0: router(config)# ospf-area-set ospf_area_set_demo1
RP/0/RSP0/cpu 0: router(config-ospf-area)# 10.0.0.1,
RP/0/RSP0/cpu 0: router(config-ospf-area)# 3553
RP/0/RSP0/cpu 0: router(config-ospf-area)# end-set

RP/0/RSP0/cpu 0: router(config)# ospf-area-set ospf_area_set_demo2
RP/0/RSP0/cpu 0: router(config-ospf-area)# 20.0.0.2,
RP/0/RSP0/cpu 0: router(config-ospf-area)# 3673
RP/0/RSP0/cpu 0: router(config-ospf-area)# end-set
```

2. OSPF エリアセットを参照するように、ワイルドカードを使用してルートポリシーを設定します。

```
RP/0/RSP0/cpu 0: router(config)# route-policy use_ospf_area_set
RP/0/RSP0/cpu 0: router(config-rpl)# if ospf-area in ospf-area-set* then set
ospf-metric 200
RP/0/RSP0/cpu 0: router(config-rpl-if)# elseif ospf-area in( 10.0.0.1, 10.0.0.2 )then
 set ospf-metric 300
RP/0/RSP0/cpu 0: router(config-rpl-elseif)# endif
RP/0/RSP0/cpu 0: router(config-rpl)# end-policy
```

3. 設定をコミットします。

```
RP/0/RSP0/cpu 0: router(config)# commit
```

4. （任意）設定を確認します。

```
RP/0/RSP0/cpu 0: router(config)# show configuration
Building configuration...
!! IOS XR Configuration 0.0.0
!
ospf-area-set ospf_area_set_demo1
  10.0.0.1,
  3553
end-set
!
!
ospf-area-set ospf_area_set_demo2
  20.0.0.2,
  3673
end-set
!

route-policy use_ospf_area_set
  if ospf-area in ospf-area-set* then
    set ospf-metric 200
  elseif ospf-area in (10.0.0.1, 10.0.0.2) then
    set ospf-metric 300
  endif
end-policy
!
end
```

これで、OSPF エリアセットにワイルドカードを使用したルーティングポリシーの設定が完了します。

VRF インポートポリシー設定：例

次に、VRF インポートポリシーの設定例を示します。

```
router bgp 100
address-family vpnv4 unicast
  vrf all
   source rt import-policy
  !
```

その他の参考資料

ここでは、RPL の実装に関する関連資料について説明します。

関連資料

関連項目	マニュアル タイトル
ルーティング ポリシー言語コマンド：コマンド構文の詳細、コマンドモード、コマンド履歴、デフォルト設定、使用に関する注意事項、および例	*Routing Command Reference for Cisco ASR 9000 Series Routers*の「*Routing Policy Language Commands on* Cisco ASR 9000 シリーズ ルータ 」のモジュール
正規表現の構文	*Cisco ASR 9000 Series Aggregation Services Router Getting Started Guide*の付録「*Understanding Regular Expressions, Special Characters and Patterns*」

標準

標準	タイトル
この機能でサポートされる新規の標準または変更された標準はありません。また、既存の標準のサポートは変更されていません。	—

MIB

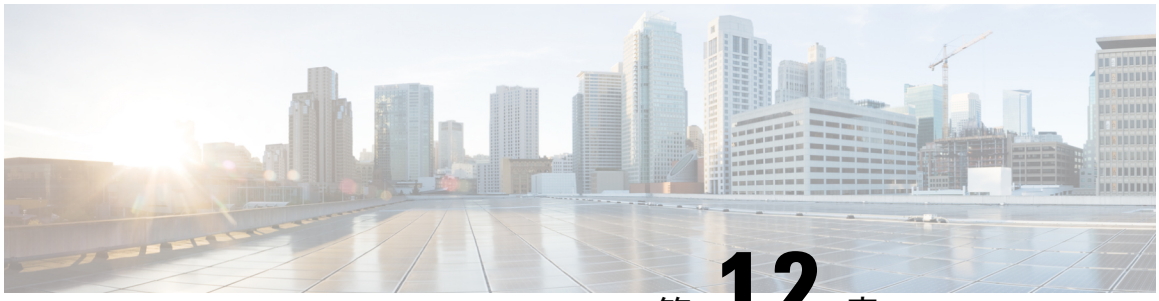
MB	MIB のリンク
—	Cisco IOS XR ソフトウェアを使用して MIB の場所を特定してダウンロードするには、次の URL にある Cisco MIB Locator を使用して、[Cisco Access Products] メニューからプラットフォームを選択します。 https://mibs.cloudapps.cisco.com/ITDIT/MIBS/servlet/index

RFC

RFC	タイトル
RFC 1771	『A Border Gateway Protocol 4 (BGP-4)』
RFC 4360	『BGP Extended Communities Attribute』

シスコのテクニカル サポート

説明	リンク
シスコのテクニカル サポート Web サイトでは、製品、テクノロジー、ソリューション、技術的なヒント、およびツールへのリンクなどの、数千ページに及ぶ技術情報が検索可能です。Cisco.com に登録済みのユーザは、このページから詳細情報にアクセスできます。	http://www.cisco.com/techsupport



第 12 章

スタティック ルートの実装

このモジュールでは、スタティック ルートの実装方法について説明します。

スタティック ルートは、指定のパスを通るように発信元と宛先の間でパケットを移動させるユーザ定義のルートです。スタティック ルートは、Cisco IOS XR ソフトウェアが特定の宛先へのルートを確立できない場合に重要になることがあります。また、ルーティングできないすべてのパケットを送るラスト リゾート ゲートウェイを指定する場合にも役立ちます。

（注） Cisco IOS XR ソフトウェアのスタティック ルートの詳細情報とこのモジュールに掲げられたスタティック ルート コマンドの詳細については、このモジュールの関連資料 （662 ページ）の項を参照してください。設定タスクを実行中に表示される他のコマンドのマニュアルを見つけるには、オンラインで を検索してください。 *Cisco ASR 9000 Series Aggregation Services Router Commands Master List*

スタティック ルート実装の機能履歴

リリース	変更内容
リリース 3.7.2	この機能が導入されました。
リリース 4.0.1	IGP プレフィックス向けダイナミック ECMP サポート機能が追加されました。
リリース 4.2.1	IP スタティック機能の拡張オブジェクト トラッキングが追加されました。

スタティック ルートの実装の前提条件

適切なタスク ID を含むタスク グループに関連付けられているユーザ グループに属している必要があります。このコマンド リファレンスには、各コマンドに必要なタスク ID が含まれます。ユーザ グループの割り当てが原因でコマンドを使用できないと考えられる場合、AAA 管理者に連絡してください。

スタティック ルートの実装に関する制約事項

次の制約事項は、スタティック ルートの実装時に適用されます。

- ローカル サブネットの一部である間接ネクスト ホップへのスタティック ルーティング（RIB によって学習されたプレフィックス。AIB ではより具体的である可能性がある）では、出力インターフェイスを示すグローバル テーブルで、スタティック ルートをネクスト ホップとして設定する必要があります。転送のドロップを避けるには、ネクスト ホップ IP アドレスを示すグローバル テーブルで、スタティック ルートがネクスト ホップになるように設定します。
- 通常、ルートは、グローバル テーブルの AIB から学習され、FIB にインストールされます。ただし、この動作はリークされたプレフィックスには繰り返されません。グローバル テーブルの AIB が VRF にないため、リークされた FIB エントリは、AIB に依存するグローバル テーブルと同じビューではなく、RIB から参照を取得します。これは、転送動作の不整合の原因となることがあります。

スタティック ルートの実装に関する情報

スタティック ルートを実装するには、次の概念を理解しておく必要があります。

スタティック ルート機能の概要

ネットワーキング デバイスでは、手動で設定したルート情報、またはルーティング プロトコルを使用してダイナミックに学習したルート情報を使用して、パケットを転送します。スタティック ルートは、手動で設定され、2 つのネットワーク デバイス間の明示パスを定義します。ダイナミック ルーティング プロトコルとは異なり、スタティック ルートは動的に更新されず、ネットワーク トポロジが変更された場合は手動で再設定する必要があります。スタティック ルートを使用する利点は、セキュリティが高まり、リソースが効率化されることです。スタティック ルートでは、ダイナミック ルーティング プロトコルよりも少ない帯域幅を使用し、ルートの計算および通信に CPU サイクルが使用されません。スタティック ルートを使用する場合の主なデメリットは、ネットワーク トポロジが変更された場合に自動的に再設定されないことです。

スタティック ルートはダイナミック ルーティング プロトコルに再配布できますが、ダイナミック ルーティング プロトコルによって生成されたルートは、スタティック ルーティング テーブルに再配布できません。スタティック ルートを使用するルーティング ループの設定を回避するアルゴリズムはありません。

スタティック ルートは、外部ネットワークへのパスが 1 つしかない小規模ネットワークでは有用です。また、大規模ネットワークの場合は、より厳格な制御が必要な、他のネットワークへの特定のタイプのトラフィックやリンクにセキュリティを提供します。一般に、大半のネットワークでは、ダイナミック ルーティング プロトコルを使用してネットワーキング デバイス間の通信を行いますが、特殊なケース用として 1 つまたは 2 つのスタティック ルートを設定している場合があります。

デフォルトのアドミニストレーティブ ディスタンス

スタティック ルートのデフォルトのアドミニストレーティブ ディスタンスは 1 です。小さい数値は、優先ルートを示します。デフォルトでは、スタティック ルートは、ルーティング プロトコルで学習したルートよりも優先されます。したがって、ダイナミック ルートでスタティック ルートを上書きさせる場合、スタティック ルートとともにアドミニストレーティブ ディスタンスを設定できます。たとえば、Open Shortest Path First（OSPF）プロトコルで追加される、アドミニストレーティブ ディスタンスが 120 のルートを設定できます。OSPF ダイナミック ルートで上書きされるスタティック ルートにするには、120 よりも大きいアドミニストレーティブ ディスタンスを指定します。

直接接続されたルート

ルーティング テーブルは、インターフェイスを指すスタティック ルートを「直接接続されている」と見なします。直接接続されたネットワークは、対応する interface コマンドがそのプロトコルのルータ設定のスタンザに含まれている場合、IGP ルーティング プロトコルによってアドバタイズされます。

直接接続されたスタティック ルートでは、出力インターフェイスだけが指定されます。宛先は、出力インターフェイスに直接接続されていると想定されるため、パケットの宛先はネクスト ホップ アドレスとして使用されます。次の例に、アドレス プレフィックス 2001:0DB8::/32 を持つ宛先すべてをインターフェイス GigabitEthernet 0/5/0/0 経由で直接到達可能と指定する方法を示します。

```
RP/0/RSP0/cpu 0: router(config)# router static
RP/0/RSP0/cpu 0: router(config-static)# address-family ipv6 unicast
RP/0/RSP0/cpu 0: router(config-static-afi)# 2001:0DB8::/32 gigabitethernet 0/5/0/0
```

直接接続されたスタティック ルートは、有効なインターフェイス（つまり、アップ状態にあり、かつ IPv4 または IPv6 がイネーブルになっているインターフェイス）を示している場合にかぎり、ルーティング テーブルに挿入される候補となります。

再帰スタティック ルート

再帰スタティック ルートでは、ネクスト ホップだけが指定されます。出力インターフェイスはネクスト ホップから取得されます。次の例に、アドレス プレフィックス 2001:0DB8::/32 を持つ宛先すべてをアドレス 2001:0DB8:3000::1 のホスト経由で到達可能と指定する方法を示します。

```
RP/0/RSP0/cpu 0: router(config)# router static
RP/0/RSP0/cpu 0: router(config-static)# address-family ipv6 unicast
RP/0/RSP0/cpu 0: router(config-static-afi)# 2001:0DB8::/32 2001:0DB8:3000::1
```

再帰スタティック ルートが有効である（つまり、ルーティング テーブルに挿入される候補である）のは、指定したネクストホップが直接的または間接的に有効な出力インターフェイスに解決され、ルートが自己再帰型ではなく、再帰深度が IPv6 転送の最大再帰深度を超えていない場合だけです。

自身のネクストホップ解決に使用されるのがそのルート自身である場合、ルートは自己再帰します。スタティック ルートが自己再帰型になった場合、RIB は再帰ルートを除外するようスタティック ルートに通知を送ります。

BGP ルート 2001:0DB8:3000::0/16 のネクスト ホップが 2001:0DB8::0104 と仮定すると、次のスタティック ルートは IPv6 RIB に挿入されません。BGP ルート ネクストホップがそのスタティック ルートを介して解決される一方で、そのルートも BGP ルートを介して解決され、自己再帰型になるからです。

```
RP/0/RSP0/cpu 0: router(config)# router static
RP/0/RSP0/cpu 0: router(config-static)# address-family ipv6 unicast
RP/0/RSP0/cpu 0: router(config-static-afi)# 001:0DB8::/32 2001:0DB8:3000::1
```

このスタティック ルートは、自己再帰型であるため、IPv6 ルーティング テーブルには挿入されません。スタティック ルートのネクスト ホップ 2001:0DB8:3000:1 は、自身が再帰ルートである（つまり、ネクスト ホップだけを指定する）BGP ルート 2001:0DB8:3000:0/16 を介して解決されます。BGP ルートのネクスト ホップ 2001:0DB8::0104 は、スタティック ルートを介して解決されます。したがって、スタティック ルートは、スタティック ルート自身のネクスト ホップを解決するために使用されることになります。

一般に、自己再帰型スタティック ルートの手動設定は禁止されていませんが、有用ではありません。ただし、ルーティング テーブルに挿入された再帰スタティック ルートが、ダイナミック ルーティング プロトコルを介して学習された、ネットワークでの何らかの一時的変更の結果として自己再帰になる場合があります。このような状況が発生すると、スタティック ルートが自己再帰になった事実が検出され、そのスタティック ルートはルーティング テーブルから削除されます（設定からは削除されません）。以降のネットワーク変更によって、スタティック ルートが自己再帰でなくなる場合があります。この場合、そのスタティック ルートはルーティング テーブルに再挿入されます。

完全指定のスタティック ルート

完全指定のスタティック ルートでは、出力インターフェイスとネクスト ホップの両方が指定されています。この形式のスタティック ルートは、出力インターフェイスがマルチアクセス

インターフェイスであり、ネクスト ホップを明示的に識別する必要がある場合に使用されます。ネクスト ホップは、指定した出力インターフェイスに直接接続されている必要があります。次の例に、完全指定のスタティック ルートの定義を示します。

```
RP/0/RSP0/cpu 0: router(config)# router static
RP/0/RSP0/cpu 0: router(config-static)# address-family ipv6 unicast
RP/0/RSP0/cpu 0: router(config-static-afi)# 2001:0DB8::/32 Gigethernet0/0/0/0
2001:0DB8:3000::1
```

完全指定のルートが有効である（つまり、ルーティング テーブルに挿入される候補である）のは、指定された IPv4 または IPv6 インターフェイスがイネーブルで、アップ状態の場合です。

フローティング スタティック ルート

フローティング スタティック ルートは、設定されたルーティング プロトコルを介して学習されたダイナミック ルートのバックアップに使用されるスタティック ルートです。フローティング スタティック ルートには、バックアップしているルーティング プロトコルよりも大きなアドミニストレーティブ ディスタンスが設定されています。このため、ルーティング プロトコルを介して学習されたダイナミック ルートは、フローティング スタティック ルートよりも常に優先して使用されます。ルーティング プロトコルを介して学習されたダイナミック ルートが失われると、フローティング スタティック ルートが代わりに使用されます。次の例に、フローティング スタティック ルートの定義方法を示します。

```
RP/0/RSP0/cpu 0: router(config)# router static
RP/0/RSP0/cpu 0: router(config-static)# address-family ipv6 unicast
RP/0/RSP0/cpu 0: router(config-static-afi)# 2001:0DB8::/32 2001:0DB8:3000::1 210
```

3 つのタイプのスタティック ルートのいずれも、フローティング スタティック ルートとして使用できます。フローティング スタティック ルートは、ダイナミック ルーティング プロトコルよりも大きいアドミニストレーティブディスタンスを使用して設定する必要があります。これは、小さいアドミニストレーティブディスタンスが設定されたルートの方が優先されるためです。

（注） デフォルトでは、スタティック ルートはダイナミック ルートよりアドミニストレーティブディスタンスが小さいため、スタティック ルートがダイナミック ルートに優先されます。

デフォルト VRF

スタティック ルートは常に VPN ルーティング/転送（VRF）インスタンスに関連付けられます。VRF には、デフォルト VRF または指定の VRF を設定できます。**vrf** *vrf-name* コマンドを使用して VRF を指定することで、指定の VRF の VRF コンフィギュレーション モードに入り、スタティック ルートを設定できます。VRF が指定されない場合、デフォルトの VRF スタティック ルートが設定されます。

IPv4 および IPv6 スタティック VRF ルート

IPv4 または IPv6 スタティック VRF ルートは、デフォルト VRF 用に設定されたスタティック ルートと同じです。IPv4 および IPV6 アドレス ファミリがそれぞれの VRF でサポートされます。

ダイナミック ECMP

内部ゲートウェイ プロトコル（IGP）プレフィックスのダイナミック Equal-Cost Multi-Path（ECMP）機能は、1 ～ 64 の IGP パスを範囲とする ECMP パスの動的選択をサポートします。非再帰的プレフィックスの ECMP はダイナミックです。ASR 9000 拡張イーサネット ラインカードは、IGP プレフィックスの 64 ECMP パスをサポートしています。

この機能は、出力リンクの間でハードウェアのロードバランシング サポートを有効化します。

（注） BGP 再帰的プレフィックスでは、8 ～ 32 個の ECMP パスが使用できます。ASR 9000 拡張イーサネット ラインカードは、BGP プレフィックスの ECMP パスを 32 個サポートし、ASR 9000 イーサネット ラインカードは BGP プレフィックスの ECMP パスを 8 個サポートします。

スタティック ルートの実装方法

ここでは、次の手順について説明します。

スタティック ルートの設定

スタティック ルートは、すべてユーザが設定であり、ネクスト ホップ インターフェイス、ネクストホップ IP アドレス、またはその両方を指示できます。ソフトウェアでは、インターフェイスが指定された場合、そのインターフェイスが到達可能であれば、スタティック ルートがルーティング情報ベース（RIB）にインストールされます。インターフェイスが指定されていない場合、ネクストホップ アドレスが到達可能であれば、そのルートはインストールされます。このコンフィギュレーションの唯一の例外は、スタティック ルートに permanent 属性が設定されている場合です。このときは到達可能性にかかわらず RIB にインストールされます。

（注） 現在は、デフォルトの VRF のみがサポートされています。VPNv4、VPNv6 および VPN ルーティング/転送（VRF）のアドレス ファミリは、今後のリリースでサポートされる予定です。

ここでは、スタティック ルートを設定する方法について説明します。

手順の概要

1. **configure**

2. **router static**
3. **vrf** *vrf-name*
4. **address-family** { **ipv4** | **ipv6** } { **unicast** | **multicast** }
5. *prefix mask* [**vrf** *vrf-name*] { *ip-address* | *interface-type* *interface-instance* } [*distance*] [**description** *text*] [**tag** *tag*] [**permanent**]
6. **commit**

手順の詳細

ステップ 1 **configure**

ステップ 2 **router static**

例：

```
RP/0/RSP0/cpu 0: router(config)# router static
```

スタティック ルート コンフィギュレーション モードを開始します。

ステップ 3 **vrf** *vrf-name*

例：

```
RP/0/RSP0/cpu 0: router(config-static)# vrf vrf_A
```

（任意）VRF コンフィギュレーション モードを開始します。

VRF が指定されていない場合、スタティック ルートはデフォルトの VRF で設定されます。

ステップ 4 **address-family** { **ipv4** | **ipv6** } { **unicast** | **multicast** }

例：

```
RP/0/RSP0/cpu 0: router(config-static-vrf)# address family ipv4 unicast
```

アドレス ファミリ モードを開始します。

ステップ 5 *prefix mask* [**vrf** *vrf-name*] { *ip-address* | *interface-type* *interface-instance* } [*distance*] [**description** *text*] [**tag** *tag*] [**permanent**]

例：

```
RP/0/RSP0/cpu 0: router(config-static-vrf-afi)# 10.0.0.0/8 172.20.16.6 110
```

アドミニストレーティブ ディスタンス 110 を設定します。

- 次に、アドミニストレーティブ ディスタンスが 110 より小さいダイナミック情報が使用できない場合、172.20.16.6 のネクスト ホップを介してネットワーク 10.0.0.0 のパケットをルーティングする方法の例を示します。

ステップ 6 **commit**

デフォルトのスタティック ルートは、多くの場合、単純なルータ トポロジで使用されます。次の例では、アドミニストレーティブ ディスタンス 110 でルートが設定されます。

```
configure
 router static
  address-family ipv4 unicast
  0.0.0.0/0 2.6.0.1 110
  end
```

フローティング スタティック ルートの設定

ここでは、フローティング スタティック ルートを設定する方法について説明します。

手順の概要

1. **configure**
2. **router static**
3. **vrf** *vrf-name*
4. **address-family** { **ipv4** | **ipv6** } { **unicast** | **multicast** }
5. *prefix mask* [**vrf** *vrf-name*] { *ip-address* | *interface-type* *interface-instance* } [*distance*] [**description** *text*] [**tag** *tag*] [**permanent**]
6. **commit**

手順の詳細

ステップ 1 **configure**

ステップ 2 **router static**

例：

```
RP/0/RSP0/cpu 0: router(config)# router static
```

スタティック ルート コンフィギュレーション モードを開始します。

ステップ 3 **vrf** *vrf-name*

例：

```
RP/0/RSP0/cpu 0: router(config-static)# vrf vrf_A
```

（任意）VRF コンフィギュレーション モードを開始します。

VRF が指定されていない場合、スタティック ルートはデフォルトの VRF で設定されます。

ステップ 4 **address-family** { **ipv4** | **ipv6** } { **unicast** | **multicast** }

例：

```
RP/0/RSP0/cpu 0: router(config-static-vrf)# address family ipv6 unicast
```

アドレス ファミリ モードを開始します。

ステップ 5 *prefix mask* [**vrf** *vrf-name*] { *ip-address* | *interface-type interface-instance* } [*distance*] [**description** *text*] [**tag** *tag*] [**permanent**]

例：

```
RP/0/RSP0/cpu 0: router(config-static-vrf-afi)# 2001:0DB8::/32 2001:0DB8:3000::1 201
```

アドミニストレーティブ ディスタンス 201 を設定します。

ステップ 6 **commit**

フローティング スタティック ルートは、しばしば接続失敗時のバックアップ パスの準備として使用されます。次の例では、アドミニストレーティブ ディスタンス 201 でルートが設定されます。

```
configure
 router static
  address-family ipv6 unicast
  2001:0DB8::/32 2001:0DB8:3000::1 201
  end
```

PE-CE ルータ間でのスタティック ルートの設定

このタスクでは、PE-CE ルータ間でのスタティック ルーティングの設定方法について説明します。

（注） 6VPE（IPv6 VPN Provider Edge）では、VRF フォールバックはサポートされていません。

手順の概要

1. **configure**
2. **router static**
3. **vrf** *vrf-name*
4. **address-family** { **ipv4** | **ipv6** } { **unicast** | **multicast** }
5. *prefix mask* [**vrf** *vrf-name*] { *ip-address* | *interface-type interface- path-id* } [*distance*] [**description** *text*] [**tag** *tag*] [**permanent**]
6. **commit**

手順の詳細

ステップ 1 **configure**

ステップ 2 **router static**

例：

```
RP/0/RSP0/cpu 0: router(config)# router static
```

スタティック ルート コンフィギュレーション モードを開始します。

ステップ 3 **vrf** *vrf-name*

例：

```
RP/0/RSP0/cpu 0: router(config-static)# vrf vrf_A
```

（任意）VRF コンフィギュレーション モードを開始します。

VRF が指定されていない場合、スタティック ルートはデフォルトの VRF で設定されます。

ステップ 4 **address-family** { **ipv4** | **ipv6** } { **unicast** | **multicast** }

例：

```
RP/0/RSP0/cpu 0: router(config-static-vrf)# address family ipv6 unicast
```

アドレス ファミリ モードを開始します。

ステップ 5 *prefix mask* [**vrf** *vrf-name*] { *ip-address* | *interface-type interface- path-id* } [*distance*] [**description** *text*] [**tag** *tag*] [**permanent**]

例：

```
RP/0/RSP0/cpu 0: router(config-static-vrf-afi)# 2001:0DB8::/32 2001:0DB8:3000::1 201
```

アドミニストレーティブ ディスタンス 201 を設定します。

ステップ 6 **commit**

次の例では、PE ルータと CE ルータ間のスタティック ルートが設定され、VRF がスタティック ルートに関連付けられます。

```
configure
 router static
  vrf vrf_A
  address-family ipv4 unicast
  0.0.0.0/0 2.6.0.2 120
  end
```

許可できるスタティック ルートの最大数の変更

このタスクでは、スタティック ルートの許容される最大数の変更方法について説明します。

始める前に

（注） あるルータ上で特定のアドレス ファミリに設定できるスタティック ルートの数は、デフォルトで 4000 に制限されています。**maximum path** コマンドを使用して、この上限を増大または減少させることが可能です。**maximum path** コマンドを使用して、指定されたアドレスファミリの静的ルートの設定済み最大許容数を、現在設定されている静的ルートの数よりも少なくした場合、この変更は拒否されることに注意してください。さらに、グループ化されている場合にルートのバッチをコミットした結果、設定される静的ルートの数が許可された最大数を超えたときは、バッチ内の最初の *n* 個のルートが受け入れられる、という動作も理解しておく必要があります。以前に設定されていた数が受け入れられ、残りは拒否されます。引数 *n* は、最大許容数と以前設定された数との差です。

手順の概要

1. **configure**
2. **router static**
3. **maximum path** { **ipv4** | **ipv6** } *value*
4. **commit**

手順の詳細

ステップ 1 **configure**

ステップ 2 **router static**

例：

```
RP/0/RSP0/cpu 0: router(config)# router static
```

スタティック ルート コンフィギュレーション モードを開始します。

ステップ 3 **maximum path** { **ipv4** | **ipv6** } *value*

例：

```
RP/0/RSP0/cpu 0: router(config-static)# maximum path ipv4 10000
```

許可できるスタティック ルートの最大数を変更します。

- IPv4 または IPv6 アドレス プレフィックスを指定します。
- 指定したアドレス ファミリのスタティック ルートの最大数を指定します。範囲は 1 ～ 140000 です。
- この例では、スタティック IPv4 ルートの最大数を 10000 に設定します。

ステップ 4 **commit**

インターフェイス null0 をポイントするようにスタティック ルートを設定することで、特定のプレフィックスへのトラフィックを廃棄できます。たとえば、プレフィックス 2001:0DB8:42:1/64 へのすべてのトラフィックを廃棄する必要がある場合は、次のスタティック ルートが定義されます。

```
configure
 router static
  address-family ipv6 unicast
  2001:0DB8:42:1::/64 null 0
  end
```

スタティック ルートを使用した VRF の関連付け

ここでは、VRF をスタティック ルートと関連付ける方法について説明します。

手順の概要

1. **configure**
2. **router static**
3. **vrf** *vrf-name*
4. **address-family** { **ipv4** | **ipv6** } { **unicast** | **multicast** }
5. *prefix mask* [**vrf** *vrf-name*] {**next-hop** *ip-address* | *interface-name* } {*path-id* } [*distance*] [**description** *text*] [**tag** *tag*] [**permanent**]
6. **commit**

手順の詳細

ステップ 1 **configure**

ステップ 2 **router static**

例：

```
RP/0/RSP0
/CPU0:router(config)# router static
```

スタティック ルート コンフィギュレーション モードを開始します。

ステップ 3 **vrf** *vrf-name*

例：

```
RP/0/RSP0/cpu 0: router(config-static)# vrf vrf_A
```

VRF コンフィギュレーション モードを開始します。

ステップ 4 **address-family** { **ipv4** | **ipv6** } { **unicast** | **multicast** }

例：

```
RP/0/RSP0/cpu 0: router(config-static-vrf)# address family ipv6 unicast
```

アドレス ファミリ モードを開始します。

ステップ 5 *prefix mask* [**vrf** *vrf-name*] {**next-hop** *ip-address* | *interface-name* } {*path-id* } [*distance*] [**description** *text*] [**tag** *tag*] [**permanent**]

例：

```
RP/0/RSP0/cpu 0: router(config-static-vrf-afi)# 2001:0DB8::/32 2001:0DB8:3000::1 201
```

アドミニストレーティブ ディスタンス 201 を設定します。

ステップ 6 **commit**

設定例

ここでは、次の設定例について説明します。

トラフィック廃棄の設定：例

インターフェイス null 0 をポイントするようにスタティック ルートを設定することで、特定のプレフィックスへのトラフィックを廃棄できます。たとえば、プレフィックス 2001:0DB8:42:1/64 へのすべてのトラフィックを廃棄する必要がある場合は、次のスタティック ルートが定義されます。

```
configure
 router static
  address-family ipv6 unicast
  2001:0DB8:42:1::/64 null 0
  end
```

デフォルトの固定ルートの設定：例

デフォルトのスタティック ルートは、多くの場合、単純なルータ トポロジで使用されます。次の例では、アドミニストレーティブ ディスタンス 110 でルートが設定されます。

```
configure
 router static
  address-family ipv4 unicast
  0.0.0.0/0 2.6.0.1 110
  end
```

フローティング スタティック ルートの設定：例

フローティング スタティック ルートは、しばしば接続失敗時のバックアップ パスの準備として使用されます。次の例では、アドミニストレーティブ ディスタンス 201 でルートが設定されます。

```
configure
 router static
  address-family ipv6 unicast
  2001:0DB8::/32 2001:0DB8:3000::1 201
  end
```

スタティック ルーティング向けネイティブ UCMP の設定

トラフィックが 2 つ以上のリンクで負荷分散されているネットワークでは、リンク上で等価メトリックを設定すると、Equal Cost Multipath（ECMP; 等コスト マルチパス）ネクスト ホップが作成されます。ロードバランシング中にリンクの帯域幅が考慮されないため、より高い帯域幅のリンクが十分に活用されません。この問題を回避するには、より高い帯域幅のリンクがリンクの容量に比例してトラフィックを伝送するように、不等コスト マルチパス（UCMP）をローカル（ローカル UCMP）またはネイティブ（ネイティブ UCMP）で設定できます。UCMP は、IPv4 および IPv6 のスタティック VRF ルートをサポートしています。

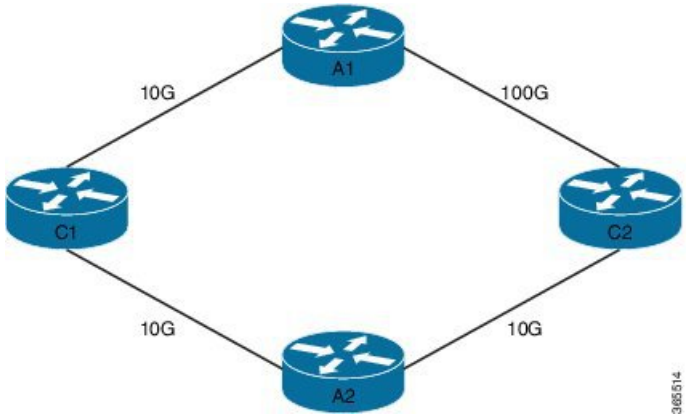
ローカル UCMP：すべてのスタティック ルートは同じリンク メトリックで設定されます。スタティック IGP は、リンクの帯域幅に基づいて負荷メトリックを計算し、リンク上のトラフィックを負荷分散します。ただし、ローカル UCMP では、（複数ホップ離れた）宛先に近いリンク間のロード バランシング時に帯域幅を考慮しません。

ネイティブ UCMP：より高い帯域幅のリンク上のスタティック ルートは、より低い帯域幅のリンク上のルートに優先されるように、より低いリンク メトリックで構成されます。スタティック IGP は、リンクの帯域幅に基づいて負荷メトリックを計算し、より高い帯域幅のリンクおよびより低い帯域幅のリンクから出るトラフィックの割合を決定します。設定されたリンク メトリックとエンドツーエンドの使用可能な帯域幅を照合することで、ネイティブ UCMP は、（複数ホップ離れた）宛先に近いリンク間でトラフィックを効果的に負荷分散できます。

設定例

次の図のトポロジについて考えます。ルータ A1 からのトラフィックのロードバランシングでは、ローカル UCMP が使用されている場合、10G と 100G の両方のリンクには等しいリンク メトリックが設定されます。負荷メトリックが高いため、スタティック IGP は 100G リンクからより多くのトラフィックを送信することを決定します。ただし、ルータ A2 からのトラフィックのロード バランシングでは、ローカル UCMP はルータ C1 および C2 へのリンク上でのみ機能します。ルータ C1 からルータ A1 およびルータ C2 からルータ A1 へのトラフィックのロードバランシングでは、ネイティブ UCMP が推奨されます。その結果、ローカル UCMP はシングルホップの宛先でのみ使用され、ネイティブ UCMP はマルチホップの宛先で使用されます。

図 *26 :* スタティック ルーティングのための不等コスト マルチパス



スタティック ルーティング用に UCMP を設定するには、次の手順を実行します。

1. グローバル コンフィギュレーション モードを開始します。

```
RP/0/0/CPU0:Router# configure
```

2. スタティック ルーティング モードを開始します。

```
RP/0/0/CPU0:Router(config)# router static
```

3. IPv4 または IPv6 アドレス ファミリ用に負荷メトリックを使用して UCMP を設定します。

```
RP/0/0/CPU0:Router(config-static)# address-family ipv4 unicast
RP/0/0/CPU0:Router(config-static-afi)# 10.10.10.1/32 GigabitEthernet 0/0/0/1 metric
 10
```

この例では、IPv4 アドレス ファミリ用に UCMP を設定しました。IPv6 アドレス ファミリ用に UCMP を設定するには、次の設定例を使用します。

```
RP/0/0/CPU0:Router(config-static)# address-family ipv6 unicast
RP/0/0/CPU0:Router(config-static-afi)# 10:10::1/64 GigabitEthernet 0/0/0/1 metric 10
```

4. スタティック設定モードを終了し、設定をコミットします。

```
RP/0/0/CPU0:Router(config-static-afi)# exit
RP/0/0/CPU0:Router(config-static)# exit
RP/0/0/CPU0:Router(config)# commit
Fri Feb 19 06:16:33.164 IST
RP/0/0/CPU0:Feb 19 06:16:34.273 : ipv4_static[1044]:
%ROUTING-IP_STATIC-4-CONFIG_NEXTHOP_ETHER_INTERFACE :
Route for 10.10.10.1 is configured via ethernet interface
```

UCMP を使用して設定する必要のあるすべてのルータで、この手順を繰り返します。

PE-CE ルータ間のスタティック ルートの設定：例

次の例では、PE ルータと CE ルータ間のスタティック ルートが設定され、VRF がスタティック ルートに関連付けられます。

```
configure
 router static
```

```
vrf vrf_A
address-family ipv4 unicast
0.0.0.0/0 2.6.0.2 120
end
```

その他の参考資料

ここでは、スタティック ルートの実装に関する関連資料について説明します。

関連資料

関連項目	マニュアル タイトル
スタティック ルート管理コマンド：コマンド構文の詳細、コマンド モード、コマンド履歴、デフォルト設定、使用上の注意事項、および例	*Routing Command Reference for Cisco ASR 9000 Series Routers*の「*Static Routing Commands*」
MPLS レイヤ 3 VPN コンフィギュレーション：コンフィギュレーションの概念、設定作業、および例	*MPLS Configuration Guide for Cisco ASR 9000 Series RoutersMPLS Configuration Guide for Cisco NCS 560 Series Routers*

標準

標準	タイトル
この機能でサポートされる新規の標準または変更された標準はありません。また、既存の標準のサポートは変更されていません。	—

MIB

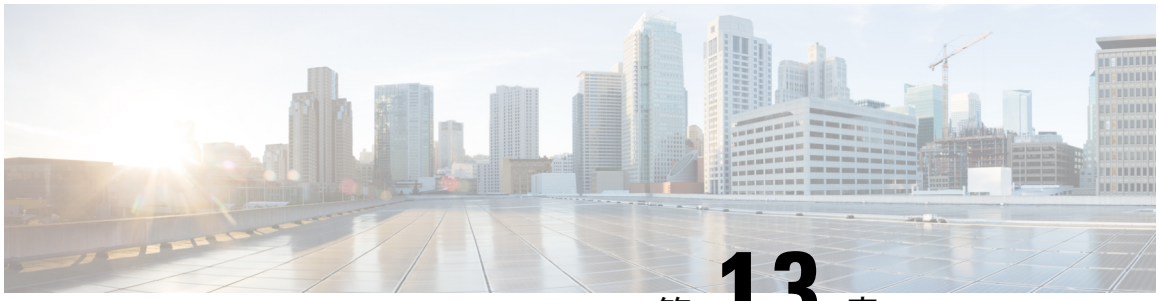
MB	MIB のリンク
—	Cisco IOS XR ソフトウェアを使用して MIB の場所を特定してダウンロードするには、次の URL にある Cisco MIB Locator を使用して、[Cisco Access Products] メニューからプラットフォームを選択します。 https://mibs.cloudapps.cisco.com/ITDIT/MIBS/servlet/index

RFC

RFC	タイトル
この機能によりサポートされた新規 RFC または改訂 RFC はありません。またこの機能による既存 RFC のサポートに変更はありません。	—

シスコのテクニカル サポート

説明	リンク
シスコのテクニカル サポート Web サイトでは、製品、テクノロジー、ソリューション、技術的なヒント、およびツールへのリンクなどの、数千ページに及ぶ技術情報が検索可能です。Cisco.com に登録済みのユーザは、このページから詳細情報にアクセスできます。	http://www.cisco.com/techsupport



第 13 章

RCMD の実装

このモジュールでは、RCMD の実装方法について説明します。

RCMD の実装の機能履歴

リリース	変更内容
リリース 4.2.0	この機能が導入されました。

ルート収束モニタリングおよび診断

Route Convergence Monitoring and Diagnostics（RCMD）は、OSPF と ISIS のコンバージェンス イベントをモニタし、SPF の実行とルートおよびルータ上のすべての LC の LDP ラベルのプロビジョニングにかかる時間についての詳細情報を収集します。

RCMD はルートの収束に関するデータを収集およびレポートするツールです。RCMD メカニズムの主な機能は次のとおりです。

- ルーティング コンポーネント間でルート フロー マーカーを使用する Lightweight および常時接続（すべてのノードおよび MC）。
- ほとんどの収束イベントと影響を受けるすべてのルートを追跡。

- 各コンバージェンス イベント ベースの統計情報とタイムラインを含むルータ内ビューを装備。
- タイムライン/SLA を測定し、指定した EEM アクションを超過時にトリガー。
- CLI/XML インターフェイスによる「ルータ上」レポート。
- 各 RCMD 対応ルータは収束に関するデータのダイジェストを提供。

RCMD で監視およびレポートされるイベントは、次のとおりです。

- OSPF および IS-IS SPF イベント（デフォルト VRF のみ）。
- 特定の外部またはエリア間のレベルのプレフィックスの追加/削除。
- LSA/LSP の変更に対する IGP フラッディングの伝搬遅延。

RCMD は次の 2 種類のモードで動作します。

- モニタリング：イベントを検出し、コンバージェンスを測定します。
- 診断：異常イベントに関する追加の（デバッグ）情報の収集。

ルート収束モニタリングおよび診断の設定

ルート収束モニタリングおよび診断を設定するには、次のタスクを実行します。

手順の概要

1. **configure**
2. **router-convergence**
3. **collect-diagnostics** *location*
4. **event-buffer-size** *number*
5. **max-events-stored** *number*
6. **monitoring-interval** *minutes*
7. **node** *node-name*
8. **protocol**
9. **priority**
10. **disable**
11. **leaf-network** *number*
12. **threshold** *value*
13. **storage-location**
14. **diagnostics** *directory-path-name*
15. **diagnostics-size**
16. **reports** *directory-path-name*
17. **reports-size**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	

ステップ 2	**router-convergence**
	例：
	`RP/0/RSP0/cpu 0:`
	`router(config)#router-convergence`
		ルータの収束モニタリング診断（rcmd）のコンフィギュレーション モードを開始します。

ステップ 3	**collect-diagnostics** *location*
	例：
	`RP/0/RSP0/cpu 0:`
	`router(config-rcmd)#collect-diagnostics 0/3/CPU0`
		指定したノードの診断を収集するように設定します。

ステップ 4	**event-buffer-size** *number*
	例：
	`RP/0/RSP0/cpu 0:`
	`router(config-rcmd)#event-buffer-size 100`
		イベント トレースを保存するためにイベント バッファ サイズを（イベント数として）設定します。

ステップ 5	**max-events-stored** *number*
	例：
	`RP/0/RSP0/cpu 0:`
	`router(config-rcmd)#max-events-stored 10`
		サーバに保存されるイベントの最大数を設定します。

ステップ 6	**monitoring-interval** *minutes*
	例：
	`RP/0/RSP0/cpu 0:`
	`router(config-rcmd)#monitoring-interval 120`
		ログを収集する間隔（分）を設定します。

ステップ 7	**node** *node-name*	指定したノードにパラメータを設定します。 `RP/0/RSP0/cpu 0: router(config-rcmd)#node`

ステップ 8	**protocol**
	例：
	`RP/0/RSP0/cpu 0: router(config-rcmd)#protocol ISIS`
	`RP/0/RSP0/cpu 0: router(config-rcmd-proto)#`
		RCMD パラメータを設定するプロトコルを指定します。
		• ISIS：ISIS プロトコルに関連するパラメータを設定するには、ISIS を設定します。
		• OSPF：OSPF プロトコルに関連するパラメータを設定するには、OSPF を設定します。

ステップ 9	**priority**
	例：
	`RP/0/RSP0/cpu 0:`
	`router(config-rcmd-proto)#priority critical`
	`RP/0/RSP0/cpu 0: router(config-rcmd-proto-prio)#`
		指定したプロトコルのルート収束モニタリングのプライオリティを設定します。
		• Critical：プライオリティが「critical」のルートのルート コンバージェンスを監視するよう設定

	コマンドまたはアクション	目的
		• High：プライオリティが「high」のルートのルート コンバージェンスを監視するよう設定 • Medium：プライオリティが「medium」のルートのルート コンバージェンスを監視するよう設定 • Low：プライオリティが「low」のルートのルート コンバージェンスを監視するよう設定

ステップ 10	**disable**
	例：
	`RP/0/RSP0/cpu 0:`
	`router(config-rcmd-proto-prio)#disable`
		指定したプライオリティのルート コンバージェンスのモニタリングを無効にします。

ステップ 11	**leaf-network** *number*
	例：
	`RP/0/RSP0/cpu 0:`
	`router(config-rcmd-proto-prio)#leaf-network 100`
		リーフ ネットワークのモニタリングをイネーブルにします。監視するリーフ ネットワークの最大数を指定します。最大数の範囲は 10 ～ 100 です。

ステップ 12	**threshold** *value*
	例：
	`RP/0/RSP0/cpu 0:`
	`router(config-rcmd-proto-prio)#threshold 1000`
		コンバージェンスのしきい値をミリ秒で指定します。しきい値は範囲内から選択します。範囲は0～4294967295 ミリ秒です。

ステップ 13	**storage-location**
	例：
	`RP/0/RSP0/cpu 0:`
	`router(config-rcmd)#storage-location`
	`RP/0/RSP0/cpu 0: router(config-rcmd-store)#`
		診断レポートを格納するディレクトリの絶対パスを指定します。

ステップ 14	**diagnostics** *directory-path-name*
	例：
	`RP/0/RSP0/cpu 0:`
	`router(config-rcmd-store)#diagnostics`
	`/disk0:/rcmd`
		診断レポートを格納するディレクトリの絶対パスを指定します。directory-path-name を設定します。
		例：/disk0:/rcmd/ または <tftp-location>/rcmd/

ステップ 15	**diagnostics-size**
	例：
	`RP/0/RSP0/cpu 0: router(config-rcmd-store)#`
	`diagnostics-size 8`
		診断ディレクトリの最大サイズを指定します。サイズを % で設定します。範囲は 5 % ～ 80 %です。

ステップ 16	**reports** *directory-path-name*
	例：
	`RP/0/RSP0/cpu 0:`
	`router(config-rcmd-store)#reports /disck0:/rcmd`
		レポートを格納するディレクトリの絶対パスを指定します。directory-path-name を設定します。
		例：/disk0:/rcmd/ または <tftp-location>/rcmd/

	コマンドまたはアクション	目的

ステップ 17	**reports-size**
	例：
	`RP/0/RSP0/cpu 0:`
	`router(config-rcmd-store)#reports-size 8`
		レポートディレクトリの最大サイズを指定します。サイズを % で設定します。範囲は 5 % ～ 80 %です。

ルート収束モニタリングおよび診断のプレフィックスモニタリング

ルート収束モニタリングおよび診断（RCMD）のプレフィックスモニタリング機能を使用すると、Open Shortest Path First（OSPF）および Intermediate System-to-Intermediate System（IS-IS）の内部ゲートウェイプロトコル（IGP）の特定のプレフィックスの収束をモニタできます。IGP では、ルート情報が作成されると、設定されているプレフィックスリストと照合してプレフィックスが確認されます。モニタするプレフィックスが見付かった場合は、モニタ対象としてマークされ、各プレフィックスの変更イベントに関する情報がキャプチャされます。RCMD のプレフィックス モニタリングでは、ネットワーク内の RCMD 対応の各ルータで特定のプレフィックスを個別にモニタします。最大で 10 個のプレフィックスをモニタできます。個々のプレフィックスのモニタリングは、特定のサービスエンドポイントの接続および可用性をモニタするためにカスタマーネットワークのエッジで有効にされているプローブを補完するものです。

IS-IS のプレフィックスに対する RCMD プレフィックスモニタリングを有効にするには、ルータの IS-IS モニタ コンバージェンス コンフィギュレーション モードで、**prefix-list** コマンドを設定します。OSPF のプレフィックスに対する RCMD プレフィックスモニタリングを有効にするには、ルータの OSPF モニタ コンバージェンス コンフィギュレーション モードで、**prefix-list** コマンドを設定します。

個別のプレフィックス モニタリングの場合、モニタリングによって OSPF または ISIS ルートの収束に影響が生じないように、プレフィックスはルート計算のために表示される前にマークされます。

ルート収束モニタリングおよび診断の OSPF タイプ 3/5/7 リンクステート アドバタイズメントのモニタリング

ルート収束モニタリングおよび診断（RCMD）の OSPF タイプ 3/5/7 のリンクステート アドバタイズメント（LSA）のモニタリング機能では、LSA のモニタリング中に LSA にフラグを付けて区別します。タイプ 3/5/7 LSA に対するルートの変更は、モニタする必要があります。ルート計算の際に、ルートの送信元がタイプ 3/5/7 LSA のようであり、ルート変更が追加または削除アクションである場合は、それらのプレフィックスをモニタする必要があります。RCMD では、タイプ 3/5/7 LSA のすべてを対象として、利用可能なパスの削除（パージ操作）と最初のパスの追加（復元操作）をすべてモニタします。OSPF タイプ 3/5/7 LSA は個々のプレフィックスごとにモニタされ、報告されます。ただし、パスの変更に関連する変更操作は全体として

の到達可能性に影響を与えないため、モニタされません。レポート用にすべてのプレフィックスがログに記録されますが、収束のトラッキングは SPF の実行で影響を受ける最初の 10 個のプレフィックスにレート制限されます。

RCMD OSPF タイプ 3/5/7 LSA のモニタリングを有効にするには、ルータの OSPF モニタコンバージェンス コンフィギュレーション モードで、**track-external-routes** と **track-summary-routes** を設定します。

IS-IS のプレフィックスの RCMD モニタリングの有効化

IS-IS のプレフィックスに対する個々のプレフィックスのモニタリングを有効にするには、次のタスクを実行します。

始める前に

個々のプレフィックスのモニタリングを有効にするには、最初に **{ipv4 | ipv6} prefix-list** コマンドを使用して、プレフィックスリストを設定します。次に、**prefix-list** コマンドでこのプレフィックスリストを使用します。

手順の概要

1. **configure**
2. **router isis** *instance-id*
3. **address-family** {**ipv4 | ipv6**} [**unicast** | **multicast**]
4. **monitor-convergence**
5. **prefix-list** *prefix-list-name*
6. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router isis** *instance-id* 例： `RP/0/RSP0/cpu 0: router(config)#router isis isp`	指定したルーティング インスタンスの IS-IS ルーティングを有効にし、ルータをルータ コンフィギュレーション モードにします。
ステップ 3	**address-family** {**ipv4	ipv6**} [**unicast**	**multicast**] 例： `RP/0/RSP0/cpu 0: router(config-isis)#address-family ipv6 unicast`	IS-IS アドレスファミリのコンフィギュレーション モードを開始します。
ステップ 4	**monitor-convergence** 例： `RP/0/RSP0/cpu 0: router(config-isis-af)#monitor-convergence`	IS-IS プロトコルのルート収束モニタリングを有効にします。

	コマンドまたはアクション	目的
ステップ 5	**prefix-list** *prefix-list-name* 例： `RP/0/RSP0/cpu 0: router(config-isis-af-rcmd)#prefix-list isis_monitor`	IS-IS のプレフィックスの個々のプレフィックス モニタリングを有効にします。
ステップ 6	**commit**	

OSPF プレフィックスの RCMD モニタリングのイネーブル化

OSPF のプレフィックスに対する個々のプレフィックスのモニタリングを有効にするには、次のタスクを実行します。

始める前に

個々のプレフィックスのモニタリングを有効にするには、最初に **{ipv4 | ipv6} prefix-list** コマンドを使用して、プレフィックスリストを設定します。次に、**prefix-list** コマンドでこのプレフィックスリストを使用します。

手順の概要

1. **configure**
2. **router ospf** *ospf-process-name*
3. **monitor-convergence**
4. **prefix-list** *prefix-list-name*
5. **commit**

手順の詳細

ステップ 1 **configure**

ステップ 2 **router ospf** *ospf-process-name*

例：

```
RP/0/RSP0/cpu 0: router(config)#router ospf 1
```

指定したルーティングプロセスに OSPF ルーティングを有効にし、ルータ コンフィギュレーション モードでルータを配置します。

ステップ 3 **monitor-convergence**

例：

```
RP/0/RSP0/cpu 0: router(config-ospf)#monitor-convergence
```

OSPF ルート収束モニタリングを有効にします。

ステップ 4 **prefix-list** *prefix-list-name*

例：

```
RP/0/RSP0/cpu 0: router(config-ospf-af-rcmd)#prefix-list ospf_monitor
```

OSPF のプレフィックスの個々のプレフィックス モニタリングを有効にします。

ステップ 5 **commit**

OSPF のプレフィックスの RCMD モニタリングのイネーブル化：例

この例では、個々の OSPF プレフィックスに対する RCMD モニタリングをイネーブルにする例を示します。

```
ipv6 prefix-list ospf_monitor
 10 permit 2001:db8::/32
!
router ospf 100
 monitor-convergence
  prefix-list ospf_monitor
```

タイプ 3/5/7 の OSPF LSA の RCMD モニタリングの有効化

タイプ 3/5/7 OSPF LSA に対する RCMD のモニタリングを有効にするには、次のタスクを実行します。

手順の概要

1. **configure**
2. **router ospf** *100*
3. **track-external-routes**
4. **track-summary-routes**
5. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router ospf** *100* 例： `RP/0/RSP0/cpu 0: router(config)#router ospf 100`	指定したルーティングプロセスに OSPF ルーティングを有効にし、ルータ コンフィギュレーションモードでルータを配置します。

	コマンドまたはアクション	目的

ステップ 3	**track-external-routes**
	例：
	`RP/0/RSP0/cpu 0:`
	`router(config-ospf-af-rcmd)#track-external-routes`
		外部の（タイプ 3/5/7）LSA プレフィックス モニタリングの追跡を有効にします。

ステップ 4	**track-summary-routes**
	例：
	`RP/0/RSP0/cpu 0:`
	`router(config-ospf-af-rcmd)#track-summary-routes`
		サマリー（エリア間）ルートのモニタリングの追跡を有効にします。

ステップ 5	**commit**	

IS-IS のプレフィックスの RCMD モニタリングのイネーブル化：例

この例では、個々の IS-IS プレフィックスに対する RCMD プレフィックス モニタリングを実行する例を示します。

```
ipv6 prefix-list isis_monitor
 10 permit 2001:db8::/32
!
router isis isp
 address-family ipv6 unicast
  monitor-convergence
   prefix-list isis_monitor
```

OSPF のプレフィックスの RCMD モニタリングのイネーブル化：例

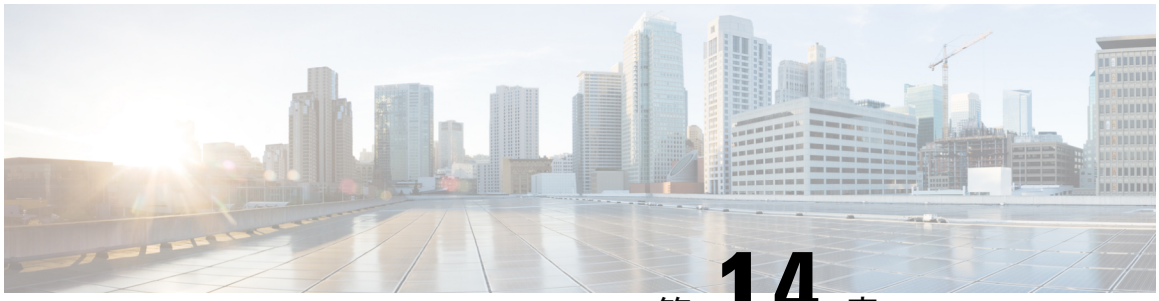
この例では、個々の OSPF プレフィックスに対する RCMD モニタリングをイネーブルにする例を示します。

```
ipv6 prefix-list ospf_monitor
 10 permit 2001:db8::/32
!
router ospf 100
 monitor-convergence
  prefix-list ospf_monitor
```

タイプ 3/5/7 の OSPF LSA の RCMD モニタリングのイネーブル化：例

この例では、OSPF 外部 LSA とサマリー ルートに対するプレフィックス モニタリングの追跡をイネーブルにする例を示します。

```
router ospf 100
 monitor-convergence
  track-external-routes
  track-summary-routes
```



第 14 章

データプレーンセキュリティの実装

データプレーンセキュリティ（DPSec）機能は、外部送信元から LISP VPN へのトラフィック注入を防止します。DPSec は、Unicast Reverse Path Forwarding（uRPF）のサポートを使用して構築されたルーティングロケータ（RLOC）ネットワークの整合性に依存します。

認証や暗号化のオーバーヘッドを発生させずに LISP 共有モードセグメンテーションを有効にするため、DPSec 機能では、ネットワーク上で URPF を適用する送信元 RLOC カプセル化解除フィルタリングというメカニズムを使用します。ネットワークに設定されている URPF は、URPF によってすでに証明されているトラフィックの許容可能 RLOC のリストを配布します。これにより、LISP 制御およびデータパケットの送信元 RLOC アドレスをスプーフィングできなくなります。DPSec 機能では、各 EID インスタンスの有効なカプセル化送信元のリストを使用して、xTR と PxTR でのカプセル化解除時に LISP データパケットのフィルタリングを行います。

（注）

- LISP 転送は Cisco ASR 9000 高密度100GE イーサネットラインカードでサポートされていますが、LISP IPv6 RLOC および LISP データプレーンセキュリティ機能は、これらのカードではサポートされていません。

データプレーンセキュリティの機能の履歴

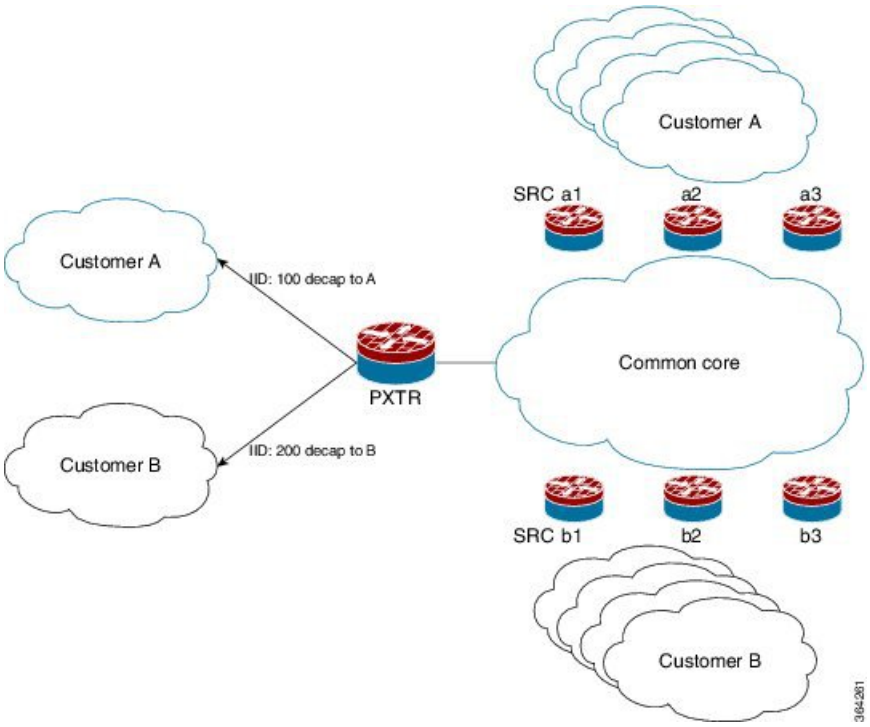
リリース 5.3.0	この機能が導入されました。

データプレーンセキュリティに関する情報

LISP データ プレーン セキュリティ機能により、LISP VPN からのトラフィックのみが VPN でカプセル化を解除できます。データプレーンセキュリティを理解するには、それがサポートしている次の機能と概念を十分に理解しておく必要があります。

送信元 RLOC カプセル化解除のフィルタリング

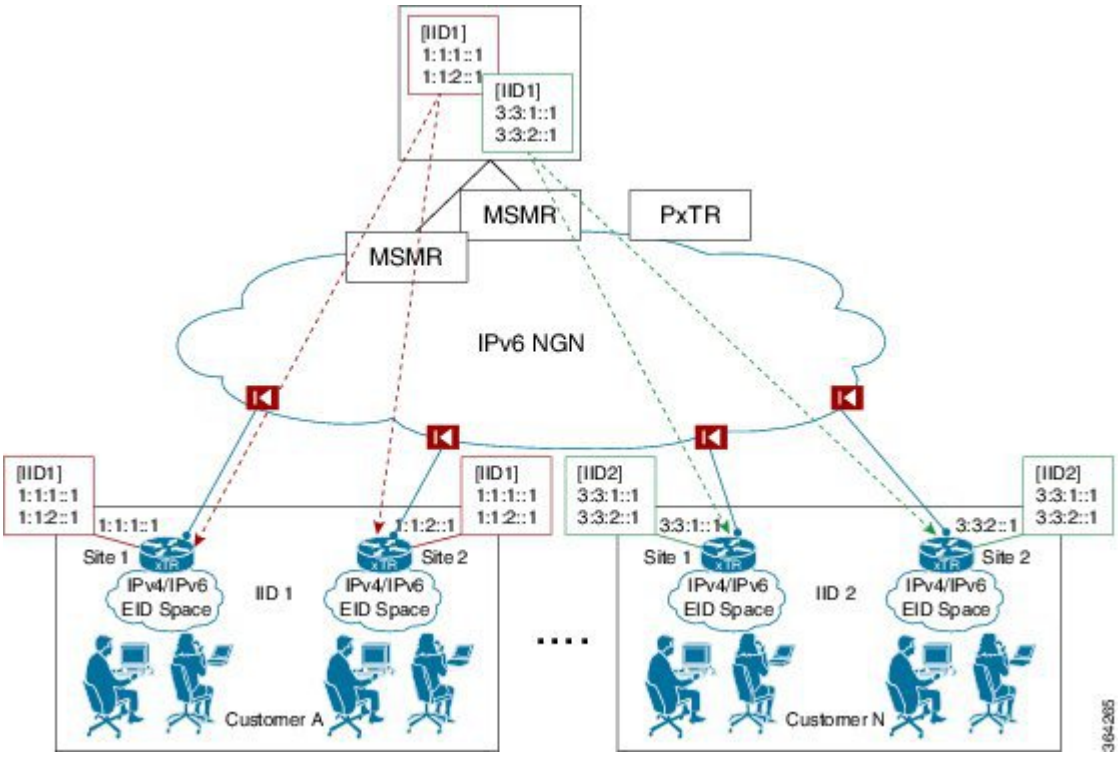
次に、共有共通 RLOC コアを介して、LISP EID インスタンス ID（IID）100 および 200 を使用しているカスタマーネットワークをそれぞれ青と黒で示した図を示します。LISP データパケットをカプセル化を解除すると、PxTR は、インスタンス ID 100 を伝送しているパケットの a1、a2、または a3 のカプセル化ヘッダー内に送信元（SRC）RLOC があることを検証します。同様に、インスタンス ID 200 の場合、PxTR は RLOC 送信元が b1、b2、または b3 であることを検証します。



有効な送信元 RLOC を伝送しない LISP カプセル化データパケットはドロップされます。RLOC 空間 URPF の適用と送信元 RLOC ベースのカプセル化解除フィルタリングを組み合わせることで、テナント VPN のメンバーではない送信元が VPN にトラフィックを挿入できないようにします。

EID インスタンスメンバーシップの配布

送信元 RLOC フィルタリング ソリューションを展開するには、マッピングシステムを介して有効な RLOC のリストをカプセル化解除を実行するボックスにプッシュするための自動化されたメカニズムが必要です。この機能は、マップサーバによって実行されます。マップサーバは、Map-Register メッセージで受信したマッピングレコード内の RLOC 情報を使用して、EID インスタンス ID と RLOC のメンバーシップリストを作成します。EID インスタンス ID で識別された VPN のパケットのカプセル化解除が必要なすべての xTR と PxTR に完全なリストがプッシュされます。



この例では、マップサーバがカスタマーごとに個別のVPN（EIDインスタンス）メンバーシップリストを作成し、リストの内容をプッシュします。カスタマー A の 2 つの xTR はそれぞれのサイトの RLOC を登録します。各ユーザは、マップサーバから、カスタマー A のすべての xTF の RLOC の完全なリストを受信します。受信したリストは、カプセル化解除トラフィックをフィルタリングし、データプレーンのセキュリティを適用するために使用されます。

PxTR が使用されている場合（VPN へのインターネット接続をVPN に提供するなど）、VPN に参加している xTR は、PxTR によって送信された LISP データパケットを受け入れ、カプセル化を解除する必要があります。PxTR によって使用される RLOC アドレスは、マップサーバによって xTR に伝達される EID インスタンス メンバーシップ リストに含まれている必要があります。PxTR は、マップサーバが PxTR RLOC を検出するために使用できるマップサーバに EID プレフィックスを登録しません。これらの RLOC はマップサーバ上に手動で設定する必要があります。

マップサーバによって構築された EID インスタンスのメンバーシップリストが有効なのは、VPN に参加しているボックスのみです。追加されたセキュリティ対策として、マップサーバは、EID インスタンスのメンバーシップリストの内容を、その VPN のメンバである xTR と PxTR にのみ伝達します。

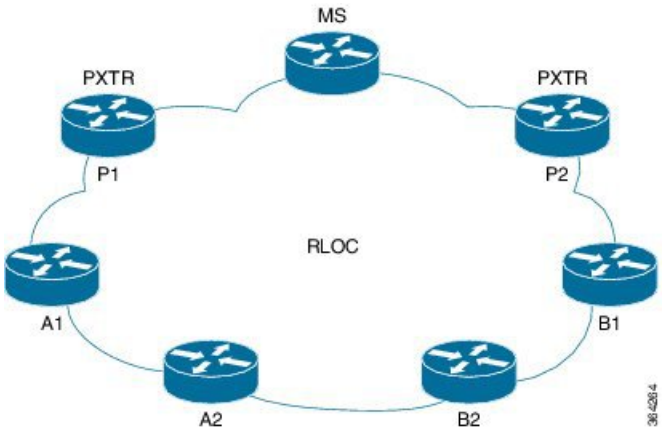
マップサーバメンバーシップの収集と配布

LISP マップサーバは、EID インスタンス単位のメンバーシップを追跡し、それを (P)xTR に配布する役割を担います。この機能を有効にするには、**map-server rloc members distribute** コマンドを使用します。このコマンドは、マップサーバを次のように設定します。

- 信頼性の高いトランスポートセッションを受け入れるためのマップ登録と設定を使用して、RLOC アドレスのリストを構築する。
- 上記のリストの (P)xTR からの TCP 接続を受け入れる。
- 受信した Map-Register メッセージから EID インスタンス単位で RLOC メンバーシップを収集し、保持する。
- (P)xTR からの信頼性の高い転送セッションを介して受信した EID インスタンスのメンバーシップ要求に対応し、メンバーシップ情報を配信する。

受信した登録から MS が収集した EID インスタンス単位のメンバーシップリストは、**map-server rloc members {add | override}** コンフィギュレーション コマンドを使用して拡張したり、完全に上書きすることができます。このコマンドを使用すると、検出した xTR RLOC メンバーシップを PxTR RLOC アドレスを使用して拡張できます。拡張されたメンバーシップリストは、信頼性の高い転送セッションを介して受信したメンバーシップ要求を許可するかどうかを決定するために使用されます。EID インスタンスに登録されている xTR からの要求のみが許可されます。拡張されたメンバーシップリストは、カプセル化解除デバイスにプッシュされてデータプレーンセキュリティ機能が実装された後、有効な xTR と PxTR の両方から送信されたカプセル化されたパケットを受け入れられるようになります。

マップサーバとの TCP 接続を確立しようとする無許可の試行を防ぐため、接続を許可する許可済みのロケータのリストが構築されます。このリストには、登録する xTR の RLOC アドレスとともに、メンバーシップリストの拡張で設定された RLOC のアドレスが含まれています。RLOC アドレスファミリごとに接続を受け入れる単一のリストがあることに注意してください（EID インスタンス固有ではありません）。



たとえば、上の図の 2 つの VPN があるネットワークを考えてみましょう。VPN A と B にはそれぞれ 2 つの xTR A1/A2 と B1/B2 があります。VPN A のメンバーシップは、「map-server rloc members add …」設定を使用して PxTR RLOC アドレス P1 を含めるように MS 上で拡張されます。VPN B のメンバーシップは、PxTR RLOC アドレス P2 を含むように拡張されます。MS が管理する結果のリストは次のようになります。

- EID インスタンス 1（VPN A）メンバーシップ：A1、A2、P1
- EID インスタンス 2（VPN B）メンバーシップ：B1、B2、P2

- TCP セッションを受け入れるロケーター：A1、A2、P1、B1、B2、P2

マップサーバは、確立された信頼性の高いトランスポートセッションごとに 1 つ以上の EID インスタンスの EID インスタンスメンバーシップ要求を受信する場合があります。PxTR は通常、MS で確立された 1 つのセッションを通じて複数のインスタンスのメンバーシップを要求します。マップサーバは、許可された要求ごとに完全なメンバーシップのリフレッシュと増分更新を提供する必要があります。

メンバーシップ要求が MS によって受信され、要求を発信しているピア (P)xTR が要求に関連する EID インスタンスのメンバではない場合、MS は要求を拒否し、(P)xTR に Membership-NACK メッセージを返します。このようなイベントは、通常の動作中に発生することがあります。これは、TCP セッションと xTR からのメンバーシップ要求を、対応する Map Register メッセージの前に受信して EID インスタンスメンバーシップに配置することがあるためです。EID インスタンスメンバーシップ要求が MS によって受け入れられた後、登録の有効期限または設定の変更が原因で、要求側 (P)xTR が EID インスタンスメンバーシップから削除された場合、MS は (P) xTR にそのインスタンスのメンバーシップ更新を受信しなくなったことを示す Membership-NACK メッセージを送信します。

マップサーバが再起動すると、メンバーシップ要求に対応する前に、まず、EID インスタンスのメンバーシップリストを検出して再構築する必要があります。特に、完全なメンバーシップリストを持たない EID インスタンスのメンバーシップリフレッシュ終了メッセージの送信は、MS でオフにする必要があります。MS では、LISP コントロールプレーンはメンバーシップリストの完了を検討する前に、登録の受信を待機します。次の条件を満たす必要があります。

- 最初の登録が受信されてから、次のいずれかの条件が満たされた後、少なくとも 1 つの登録期間が経過した（1 分）。
  - accept-more-specific site という EID プレフィックス設定が EID インスタンスに存在せず、設定済みのすべての EID プレフィックスの登録が受信されている。
  - 最初の登録が受信された時点から登録の 3 期間が経過しています。
  - 登録が受信されておらず、LISP コントロールプレーンが再起動してから 3 回の登録期間が経過しました。

EXEC コンフィギュレーション モードで **show lisp site rloc members** コマンドを使用して、メンバーシップの配布をマップサーバで管理できます。

データプレーンセキュリティの実装方法 （681 ページ） に、手順を詳細に示します。

(P)xTR でのカプセル化解除のフィルタリング

送信元 RLOC のカプセル化解除 RLOC フィルタリング機能は、**decapsulation filter rloc source** コマンドを使用して、(P)xTR 上で有効になります。この機能を有効にすると、(P)xTR はフィルタによって許可された送信元 RLOC を伝送する LISP データパケットのカプセル化解除のみを許可します。この機能を初めて有効にしたときに、フィルタがマップサーバからの EID インスタンスメンバーシップの自動検出に基づいている場合、マップサーバとの信頼性の高い転送接続が確立され、メンバーシップを受信するまで、トラフィックはドロップされます。

(P)xTR メンバーシップの検出

decapsulation filter rloc source members 設定を使用して 1 つ以上の EID インスタンスのメンバーシップ自動検出を使用したデータプレーンの送信元RLOCフィルタリング用に設定されている (P)xTR は、それらのインスタンスに設定されているマップサーバそれぞれとの信頼性の高い転送セッションを確立しようとします。1 つ以上の EID インスタンスのメンバーシップを伝達する各マップサーバを使用して、信頼性の高い1つの転送セッションが開始されます。自動検出されたメンバーシップリストは、**decapsulation filter rloc source** コマンドの **locator-set** オプションを使用して送信元 RLOC フィルタを形成するように拡張されています。各マップサーバから検出されたEIDインスタンスのメンバーシップリストは、設定されているロケータセットの内容とともにまとめてマージされ、データプレーンの送信元RLOCを定義するために使用されます。マップサーバは、最初に正常に登録した EID プレフィックスを持つ RLOC アドレスからの信頼性の高い着信転送接続のみを受け入れます。xTRは、正常に登録されたことを確認する Map-Notify を受信した後にのみ、接続を確立しようとします。特定のインスタンス ID の EID インスタンスメンバーシップを要求するには、そのインスタンスの 1 つ以上の EID プレフィックスが正常に登録されている必要があります。

マップサーバとの接続が確立されると、(P)xTR は設定内にマップサーバがある各 EID インスタンスの Membership-Request メッセージを送信します。受信した Membership-Add メッセージと Membership-Delete メッセージは、(P)xTR 上の EIDインスタンス メンバーシップ データベースを更新します。

EID インスタンス メンバーシップ データベースを再構築するために、(P)xTR は、Membership-ACK メッセージを使用してメンバーシップサービスの提供を希望することをマップサーバが示すとすぐに、Membership-Refresh-Request メッセージを発行します。(P)xTR は、検出された各メンバーシップエントリのエポックを保持します。マップサーバから Membership-Refresh-Start メッセージを受信すると、(P)xTR は、マップサーバと EID インスタンスの組み合わせについて保持するエポックを増分させ、既存のメンバーシップ状態を失効としてフラグ付けします。リフレッシュ時に受信した後続の Membership-Add メッセージは、対応するエントリのエポックを更新します。Membership-Refresh-End メッセージを受信すると、(P)xTRは、リフレッシュ時に更新されていない古いエポックを伝送している、マップサーバから受信した EID インスタンスのメンバーシップエントリを削除します。

転送する通信のフィルタリング

LISP コントロールプレーンは、RIB を通じて情報を伝達するための RIB の Opaque ファシリティをテーブル配布の一部として、すべてのFIBインスタンスまで使用します。メッセージは次のように定義されます。

- RLOC AF および EID インスタンスの粒度ごとにフィルタの有効化状態を伝達する
- RLOC フィルタのエントリを伝達する

TCP ベースの信頼性の高いトランスポートセッション

LISP は、EID インスタンスのメンバーシップの配布に xTR とマップサーバ間の TCP ベースのセッションを使用します。信頼性の高いトランスポートセッションは、アクティブセッションまたはパッシブセッションの確立を（TCP ポート 4342 を使用）をサポートしています。この

場合、xTRがアクティブロール、マップサーバがパッシブロールを担います。セッションは、送信元 RLOC フィルタリングに基づいて、マップサーバ側からの有効な RLOC からのみ受け入れられます。サポートできる同時 TCP 接続の数は、OS ごとおよびプラットフォームごとに異なります。次に、考慮する必要があるセキュリティ上の事項を示します。

- マップサーバが対応できる xTR の数は、プラットフォームで確立および保持できる TCP セッションの数で制限されます。これにより、マップサーバがホストできる VPN カスタマーの数が決定します。水平スケーリングは、VPN カスタマーを複数の Map Server 間で分割することによって実現されます。
- 同じ VPN に属しているすべての xTR は同じマップサーバに登録する必要があります。マップサーバ TCP セッションのスケール制限よりも多くの xTR を持つ VPN は使用できません。
- 最初の成果物のセッション認証は、RLOC ネットワークの整合性に依存しており、パケットの送信元アドレスを使用して TCP セッションのみをフィルタリングします。

セッションの確立、信頼性の高いトランスポートメッセージ形式、キープアライブメッセージ、エラー通知メッセージなどの TCP ベースの信頼性の高いトランスポートセッションの詳細については、http:// tools.ietf.org /id/draft-kouvelas-lisp-reliable-transport-00.txt を参照してください。

データプレーンセキュリティの実装方法

ここでは、次の手順について説明します。

送信元 RLOC ベースのカプセル化解除のフィルタリングの有効化

LISP パケットのカプセル化を解除するときに送信元検証用のカプセル化解除フィルタリストをダウンロードするように xTR またはプロキシ xTR を設定するには、lisp コンフィギュレーション モードで **decapsulation filter source** コマンドを使用します。

(P)ETR が LISP パケットのカプセル化を解除する場合は、LISP パケットの外部ヘッダー送信元アドレスを考慮せずに実行されます。送信元アドレスが信頼できるネットワーク環境では、カプセル化解除前に LISP パケットの送信元アドレスを考慮する必要がある場合があります。(P)xTR で decapsulation filter source コマンドを設定すると、デバイスはマップサーバとの信頼性の高い TCP ベースのトランスポートセッションを確立し、LISP パケットのカプセル化解除時にフィルタリストをダウンロードして使用します。**members** または **locator-set** キーワードのいずれか、あるいは両方を指定する必要があります。

members キーワードを指定すると、xTR は、登録済みの RLOC メンバーシップリストを自動的に取得するために、設定されたマップサーバとの信頼性の高いトランスポート（TCP）セッションを確立しようとします。**locator-set** が指定されている場合、そのロケータセット内に設定されているロケータに対してフィルタリングが実行されます。locator-set と「members」キーワードの両方が指定されている場合は、設定されているロケータと自動的に検出されたロケータがマージされ、その結果のリストがカプセル化が解除されたパケットに使用されます。

（注）
- (P)xTR は通常、複数のマップサーバと通信します。ただし、すべての信頼性の高いトランスポートセッションがダウンした場合、既存の（失効している可能性がある）フィルタリストが短期間（数分）使用されたままになります。その間、(P)xTR はMS を使用してセッションを再確立してメンバーシップを更新しようと試みます。
- フィルタリストをダウンロードできない場合、または既存のリストがタイムアウトになった場合、パケットはドロップされます（フェールクローズ）。
- xTR が（DHCP を介してなどで）RLOC を変更した場合は、RLOC が変更されるとすぐに、マップサーバへの登録が更新され、新しい登録済み RLOC がこの IID/VPN のすべての「メンバ」にプッシュされます（イベント駆動）。

始める前に

次の前提条件を満たしていることを確認してください。

- xTR では、TCP ベースの信頼性の高いトランスポートセッションは、UDP ベース（通常）の マップ登録プロセスが正常に完了した後にのみ確立されます。
- PxTR では、このデバイスは（通常は）マップサーバに登録されないため、信頼性の高いトランスポートセッションの確立とフィルタリストのダウンロードを可能にするために、「スタブ」（偽）のマップ登録設定を追加する必要があります。マップサーバでは、このセッションの確立を許可するために、map-server rloc members modify-discovered add コマンドに PETR RLOC を含める必要があります。

手順の概要

1. **configure**
2. **router lisp**
3. **exit**
4. **locator-set** *name IP_address*
5. **eid-table** { **default** | [**vrf** *vrf_name*]} **instance-id** *instance_id*
6. **address-family** { **ipv4** | **ipv6** } **unicast**
7. **etr map-server** *IP_address* { **key** [**clear** | **encrypted**] **LINE** | **proxy-reply** }
8. **itr map-resolver** *map-resolver-address*
9. **map-cache** *destination-EID-prefix / prefix-length* { **action** { **drop** | **map-request** | **native-forward**} | **locator** *locator-address* **priority** *priority_value* | **weight** *weight_value*
10. **database-mapping** *EID-prefix/prefixlength locator* **locator-set site priority** *priority* **weight** *weight*
11. **exit**
12. **decapsulation filter rloc source** [**locator-set** *locator_set_name*][**members**]
13. **locator-table** *name* [**default** | **vrf** *vrf_name*]
14. **commit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure**	
ステップ 2	**router lisp** 例： `RP/0/RSP0/cpu 0: router(config)# router lisp`	指定したルーティングインスタンスの LISP を有効にし、ルータを Locator and ID Separation Protocol（LISP）コンフィギュレーションモードにします。
ステップ 3	**exit** 例： `RP/0/RSP0/cpu 0: routerRP/0/0/CPU0:ios(config-lisp-afi)#exit`	ルータを LISP コンフィギュレーションモードに戻します。
ステップ 4	**locator-set** *name IP_address* 例： `RP/0/RSP0/cpu 0: router(config-lisp)#locator-set loc_sh1_vrf1 202.1.0.1`	名前付きのロケータセットサイトを設定し、ループバックまたはその他の出力トンネルルータ（ETR）のインターフェイスの RLOC IP アドレスを指定します。
ステップ 5	**eid-table** { **default** | [**vrf** *vrf_name*]} **instance-id** *instance_id* 例： `RP/0/RSP0/cpu 0: router(config-lisp)#eid-table default instance-id <IID-A>`	デフォルト（グローバル）のルーティングテーブルか、または指定した VRF を選択して、設定したインスタンス ID と関連付けます。
ステップ 6	**address-family** { **ipv4** | **ipv6** } **unicast** 例： `RP/0/RSP0/cpu 0: router(config-lisp-afi)# address-family ipv4 unicast`	IPv4 または IPv6 アドレスファミリを指定して、アドレス ファミリ コンフィギュレーション モードを開始します。 • この例では、ユニキャスト IPv4 アドレスファミリを指定します。
ステップ 7	**etr map-server** *IP_address* { **key** [**clear** | **encrypted**] **LINE** | **proxy-reply** } 例： `RP/0/RSP0/cpu 0: router(config-lisp-afi)#etr map-server 204.1.0.1 key encrypted lisp`	ロケータや認証キーなどの etr map-server（MS）に関連するオプションを指定します。
ステップ 8	**itr map-resolver** *map-resolver-address* 例： `RP/0/RSP0/cpu 0: router(config-lisp-afi)#itr map-resolver 204.1.0.1`	IPv4 EID から RLOC へのマッピングを解決するための ITR マップ要求で使用されるように LISP マップリゾルバの IPv4 または IPv6 のロケータアドレスを設定します。

	コマンドまたはアクション	目的

ステップ 9	**map-cache** *destination-EID-prefix / prefix-length* { **action** { **drop** | **map-request** | **native-forward**} | **locator** *locator-address* **priority** *priority_value* | **weight** *weight_value*
	例：
	`RP/0/RSP0/cpu 0: router(config-lisp-afi)#map-cache 12.2.0.0/24 map-request`
	`RP/0/RSP0/cpu 0: router(config-lisp-afi)#map-cache 102.2.0.0/24 map-request`
	`RP/0/RSP0/cpu 0: router(config-lisp-afi)#map-cache 103.2.0.0/24 map-request`
		静的 IPv4 EID から RLOC、または静的 IPv6 EID から RLOC のマッピング関係とそれに関連付けられたトラフィックポリシーを設定するか、あるいは宛先 IPv4 EID プレフィックスまたは宛先 IPv6 EID プレフィックスと関連付けられたパケット処理動作を静的に設定します。

ステップ 10	**database-mapping** *EID-prefix/prefixlength locator* **locator-set site priority** *priority* **weight** *weight* 例： `RP/0/RSP0/cpu 0: router(config-lisp-afi)#database-mapping 11.2.0.0/24 201.1.0.1 priority 1 weight 100`	この LISP サイトの EID-to-RLOC のマッピング関係と、それに関連するトラフィック ポリシーを設定します。 （注） この eid-table vrf での EID-to-RLOC のすべてのマッピングおよび LISP サイトのインスタンス ID が設定されるまで、この手順を繰り返します。
ステップ 11	**exit** 例： `RP/0/RSP0/cpu 0: router(config-lisp-afi)#exit`	ルータを LISP コンフィギュレーション モードに戻します。
ステップ 12	**decapsulation filter rloc source** [**locator-set** *locator_set_name*][**members**] 例： `RP/0/RSP0/cpu 0: router(config-lisp)#decapsulation filter rloc source member locator-set loc_sh1_vrf1`	送信元 RLOC ベースのカプセル化解除フィルタリング機能を有効にします。 • **members** キーワードを使用すると、設定されたマップサーバとの信頼性の高いトランスポート（TCP）セッションの確立、およびマップサーバが保持しているカプセル化解除フィルタリストのダウンロードが可能になります。 • **locator-set** キーワードが使用され、単独で含まれていた場合は locator-set で指定されたプレフィックスが使用されます。member キーワードと組み合わせて使用されている場合は（ダウンロードされた）ダイナミックリストに追加されます。

	コマンドまたはアクション	目的

ステップ 13	**locator-table** *name* [**default** | **vrf** *vrf_name*]
	例：
	`RP/0/RSP0/cpu 0:`
	`router(config-lisp)#locator-table vrf 1`
		ルーティングロケータのアドレス空間がルータの Locator/ID Separation Protocol（LISP）のインスタンス化に到達可能な Virtual Route Forwarding（VRF）テーブルを関連付けます。

ステップ 14	**commit**	

例

この例では、204.1.0.1 のマップサーバとの信頼性の高いトランスポートセッションを確立し、カプセル化解除フィルタリスト（この場合は IID 1002）をダウンロードし、カプセル化解除の前にこのフィルタリストを使用してすべての LISP カプセル化パケットの送信元チェックを行うように xTR が設定されます。

```
router lisp
 address-family ipv4 unicast
 !
 locator-set loc_sh1_vrf1
   202.1.0.1
   203.1.0.1
 !
 eid-table vrf sh1_vrf2 instance-id 1002
  address-family ipv4 unicast
   etr map-server 204.1.0.1 key encrypted lisp
   etr
   itr map-resolver 204.1.0.1
   itr
   map-cache 12.2.0.0/24 map-request
   map-cache 102.2.0.0/24 map-request
   map-cache 103.2.0.0/24 map-request
   database-mapping 11.2.0.0/24 201.1.0.1 priority 1 weight 100
   database-mapping 101.2.0.0/24 201.1.0.1 priority 1 weight 100
  !
  decapsulation filter rloc source member locator-set
     loc_sh1_vrf1
 !
 locator-table default
```

カプセル解除化フィルタリストの作成、保持、および配布

マップサーバは、サイト コンフィギュレーション モードで map-server rloc members distribute コマンドを使用して、適切な LISP デバイスに対し、インスタンス ID 単位でカプセル化解除フィルタリストを動的に作成、保持、および配布するように設定できます。設定されている場合は次のようになります。

- マップサーバは、適切な xTR との TCP ベース LISP の信頼性の高いトランスポートセッションを確立できるようにします。

- マップサーバは、登録された LISP サイトの RLOC アドレスに基づいて、LISP サイトの RLOC のリストを作成または保持（IID 単位）します。
- マップサーバは、信頼性の高いトランスポートメカニズムを介してフィルタを確立済みのデバイスにプッシュするか、または更新します。

（注）

- データプレーンセキュリティは、「map-server roc members distribute」コマンドを使用して有効になります。動的に保持されている RLOC フィルタリストに追加するか、または上書きするには、オプションコマンドの「map-server rloc members modified-discovered [add | override]」を使用します。
- この機能はカプセル化解除を実行している (P)xTR デバイスに設定されている decapsulation filter rloc source コマンドと組み合わせて使用されます。

次に、特定の LISP サイトとの信頼性の高いトランスポートセッションを作成し、カプセル化解除フィルタリストを動的に作成、保持、配布するようにマップサーバを設定する例を示します。

```
router lisp
 locator-set PxTR_set
  2001:DB8:E:F::2
  exit
 !
 eid-table vrf 1001 instance-id 1001
  map-server rloc members modify-discovered add locator-set PxTR_set
  exit
 !
---<skip>---
 !
 map-server rloc members distribute
!
```

カプセル化解除フィルタリストの追加または上書き

カプセル化解除フィルタリストを動的に作成、維持、配布するようにマップサーバが設定されている場合は、EID テーブル設定モードで map-server rloc members modify-discovered コマンドを使用することでカプセル化解除フィルタリストを追加または上書きできます。以下を使用できます。

- PxTR がアーキテクチャに含まれている場合、PITR LISP はパケットを ETR にカプセル化します。そのため、ETR には、そのカプセル化解除フィルタリスト内に PITR RLOC を含める必要があります。PITR はマップサーバに登録されないため、それらの RLOC はカプセル化解除フィルタリストに自動的には含められません。そのため、このコマンドを使用し、設定を介して追加する必要があります。
- また、PETR は、カプセル化解除時にフィルタ処理するように設定することもできますが、PETR はマップサーバに登録されないため、カプセル化解除フィルタリストを取得する手段が必要です。このコマンドの **add** 形式には、PETR でカプセル化解除フィルタリストを

取得するための信頼性の高いトランスポートセッションをマップサーバと確立するメカニズムが含まれています。

- 診断/トラブルシューティング上の理由から、カプセル化解除フィルタリスト全体を（一時的に）上書きすると便利な場合があります。

（注） カプセル化解除フィルタリストを取得するために、PETR がマップサーバに「偽の登録」を行えるよう、add 関数を含める必要があります。このコマンドに PETR RLOC を挿入すると、PETR は信頼性の高いトランスポートセッションを確立できます。

この例では、特定の LISP サイトと信頼性の高いトランスポートセッションを作成するようにマップサーバを設定し、カプセル化解除フィルタリストを作成、維持、配布します。また、静的に設定された PxTR IPv6 RLOC のアドレス（2001:db8:e:f::2）を使用して、動的に作成されたフィルタリスト（登録したサイトの RLOC アドレスで構成）を変更するようにも設定されています。

```
router lisp
 locator-set PxTR_set
  2001:DB8:E:F::2
  exit
 !
 eid-table vrf 1001 instance-id 1001
  map-server rloc members modify-discovered add locator-set PxTR_set
  ipv4 route-export site-registration
  exit
 !
---<skip>---
 !
 map-server rloc members distribute
!
```

LISP TCP の信頼性の高いトランスポートセッションのリセット

xTR と MS の間で LISP TCP の信頼性のトランスポートセッションをリセットするには、**clear lisp vrf** コマンドを EXEC モードで使用します。

手順の概要

1. **clear lisp vrf** *VRF_name* **session** {*peer_address* | *****

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**clear lisp vrf** *VRF_name* **session** {*peer_address* | ***** 例： `RP/0/0/CPU0:ios#clear lisp vrf test session *`	ピアアドレスを指定すると、そのピアへの TCP 接続がクリアされます。「*」オプションを指定すると、すべての LISP の信頼性の高いトランスポートセッションがクリアされます。

データプレーンのセキュリティ設定の確認

データプレーンのセキュリティ設定を確認するには、次のタスクを実行します。

手順の概要

1. **show lisp session**
2. **show lisp site** [**instance-id** *EID instance-ID*] **rloc members** [**registrations** [*rloc-addr*]]
3. **show lisp vrf** *vrf_name* **session** [*peer_address*]
4. **show lisp decapsulation filter**
5. **show cef vrf** [*locator-vrf*] *address_family* **lisp decapsulation** [**instance-id** *EID-instance-ID*] **detail location** *RLOC-facing LC*
6. **show controllers np struct LISP-INSTANCE-HASH detail all-entries** ［ **all** ｜*np* ］**location** *RLOC-facing LC*

手順の詳細

	コマンドまたはアクション	目的

ステップ 1	**show lisp session**
	例：
	`R11-MSMR#show lisp session`
	
	`Sessions for VRF default, total: 8, established: 7`
	`Peer State Up/Down In/Out Users`
	`2001:DB8:A:1::2 Up 00:04:13 2/7 2`
	`2001:DB8:A:2::2 Up 00:04:13 2/7 2`
	`2001:DB8:A:3::2 Up 00:03:53 2/7 2`
	`2001:DB8:B:1::2 Up 00:04:04 2/6 2`
	`2001:DB8:B:2::2 Init never 0/0 1`
	`2001:DB8:C:1::2 Up 00:03:55 2/6 2`
	`2001:DB8:C:2::2 Up 00:03:54 2/6 2`
	`2001:DB8:E:F::2 Up 00:04:04 6/19 4`
	`R11-MSMR#`
		LISP デバイスで、信頼性の高いトランスポート（TCP） セッションの現在のリストを表示するには、EXEC コンフィギュレーション モードで **show LISP session** コマンドを使用します。この例では、信頼性の高いトランスポート LISP セッションがマップサーバに表示されます。出力では、7つのセッションが確立され、1 つのセッションが Init 状態になっています（decapsulation filter rloc source member コマンドがそのサイトに適用されておらず、セッションは確立されませんでした）。

ステップ 2	**show lisp site** [**instance-id** *EID instance-ID*] **rloc members** [**registrations** [*rloc-addr*]]
	例：
	`R114-MSMR#show lisp site rloc members`
	`LISP RLOC membership for EID table default (IID 0), 5 entries`
	
	`RLOC Origin Valid`
		収集および設定された EID インスタンスのメンバーシップを表示するには、EXEC コンフィギュレーション モードで **show lisp site** コマンドを使用します。出力の 「origin」列には、RLOC メンバが手動で設定されたのか、または受信した登録から自動的に収集されたのか、あるいはその両方かが示されます。「valid」列には、RLOC が (P)xTR に配布される有効なメンバであるかどうかが示されます。リストされ

	コマンドまたはアクション	目的
	1.2.3.4 config Yes 10.0.1.2 registration Yes 10.0.2.2 config & registration Yes 13:12::1 config Yes 2001:DB8:2:3::2 registration Yes	ているRLOCは、それが登録から収集されたものであっても、「modify-discovered」設定に「override」オプションが使用されており、指定されたロケータセットにそのRLOCが含まれていない場合は有効でない可能性があります。オプションの「registrations」キーワードが指定されている場合、このコマンドはメンバーシップエントリに関与する登録のリストを表示します。
ステップ3	**show lisp vrf** *vrf_name* **session** [*peer_address*] 例： On xTR: RP/0/RSP1/CPU0:VKG-1#sh lisp vrf default session Sessions for VRF default, total: 1, established: 1 Peer State Up/Down In/Out Users 204.1.0.1 Up 06:49:05 0/1 1 RP/0/RSP1/CPU0:VKG-1# On MSMR: sh lisp vrf default session Sessions for VRF default, total: 2, established: 2 Peer State Up/Down In/Out Users 201.1.0.1 Up 06:48:49 1/0 0 202.1.0.1 Up 06:48:36 2/0 0 RP/0/RSP0/CPU0:VKG-4#	
ステップ4	**show lisp decapsulation filter** 例： RP/0/RSP0/CPU0:lisp9-a9k-1#show lisp eid-table se2 decapsulation filter LISP decapsulation filter for EID table vrf se2 (IID 16777212), 5 entries Source RLOC Added by 22:22::10 MS 190::190 33:33::20 MS 190::190 88:88::30 MS 190::190 99:99::30 MS 190::190 110:110::40 MS 190::190 RP/0/RSP0/CPU0:lisp9-a9k-1#	LISP デバイスで、選択した送信元 RLOC のカプセル化解除フィルタに関連するデータを表示するには、EXEC コンフィギュレーション モードで **show lisp decapsulation filter** コマンドを使用します。この例では、カプセル化解除フィルタ情報は (P)xTR for Instance-ID (IID 16777212 に表示されます。この出力では、5 つの送信元 RLOC アドレスが定義されており、このリストのすべてのメンバがマップサーバとの信頼性の高いトランスポートセッションによって提供されるリスト内で定義されています。

	コマンドまたはアクション	目的

ステップ 5	**show cef vrf** [*locator-vrf*] *address_family* **lisp decapsulation** [**instance-id** *EID-instance-ID*] **detail location** *RLOC-facing LC*
	例：
	`RP/0/RSP0/CPU0:lisp9-a9k-1#show cef ipv6 lisp decapsulation instance-id 16777212 loc 0/0/cpu0`
	
	`Number of EID tables handling LISP payload received in this table: 3`
	
	`Transport LISP ipv6 packets received in VRF: default, Instance ID: 16777212`
	`Payload IPv4 is : decapsulated`
	`Payload IPv6 is : decapsulated`
	`Payload switched in VRF : se2 (0xe00000 1d /0xe080001d)`
	`H/W driver signalled : active`
	`Binding in retry : no`
	
	`Source RLOC Prefix Filter : enabled`
	`Lookup statistics (s/w) :`
	`Misses : 8`
	`Matches (historic) : 0`
	`H/W driver signalled : active`
	`Binding in retry : no`
	`Platform space : allocated`
	
	`Len Prefix Action Matches Attributes`
	
	`128 22:22::10 accept 0 h/w [active, plt space]`
	`128 33:33::20 accept 0 h/w [active, plt space]`
	`128 88:88::30 accept 0 h/w [active, plt space]`
	`128 99:99::30 accept 0 h/w [active, plt space]`
	`128 110:110::40 accept 0 h/w [active, plt space]`
		この例では、カプセル化解除フィルタのサマリー情報が (P)xTR に表示されます。

ステップ 6	**show controllers np struct LISP-INSTANCE-HASH detail all-entries** [**all** | *np*] **location** *RLOC-facing LC*
	例：
	`RP/0/RSP0/CPU0:lisp9-a9k-1#show controllers np struct LISP-INSTANCE-HASH detail all-entries np0 location 0/0/cpu0`
	
	`Node: 0/0/CPU0:`
	`----------`
	
	`NP: 0`
	`Struct 114: LISP_INSTANCE_HASH_STR (maps to uCode Str=79)`
	`Struct is a LOGICAL entity inside a shared`	

	コマンドまたはアクション	目的
	PHYSICAL resource Reserved Entries: Logical 0, Physical 0 Used Entries: Logical 79, Physical 79 Max Entries: Logical 86016, Physical 86016 Entries Shown: Logical 79 --- Entry 1: >> Key: 4affffe 00000099 00990000 00000000 00000000 0030 Size: 22 Mask: ffffffff ffffffff ffffffff ffffffff ffffffff ffff Size: 22 Result: 51000000 1e000000 Size: 8 Entry 2: >> Key: 4976adf1 1c005858 0e1e0000 00000000 00000000 0000 Size: 22 Mask: ffffffff ffffffff ffffffff ffffffff ffffffff ffff Size: 22 Result: 51000000 1c000000 Size: 8 Entry 3: >> Key: 4976adf1 1c006e6e 0e280000 00000000 00000000 0000 Size: 22 Mask: ffffffff ffffffff ffffffff ffffffff ffffffff ffff Size: 22 Result: 51000000 1c000000 Size: 8 Entry 4: >> Key: 41000000 20002121 0b140000 00000000 00000000 0000 Size: 22 Mask: ffffffff ffffffff ffffffff ffffffff ffffffff ffff Size: 22 Result: 51000000 1f000000 Size: 8 Entry 5: >> Key: 4**affffc** 0000**0099 00990000 00000000 00000000 0030** Size: 22 Mask: ffffffff ffffffff ffffffff ffffffff ffffffff ffff Size: 22 Result: 51000000 **1d**000000 Size: 8 Entry 6: >> Key: 4a0003e8 19000033 00330003 00000000 00000000 0020 Size: 22 Mask: ffffffff ffffffff ffffffff ffffffff ffffffff ffff Size: 22 Result: 51000000 19000000 Size: 8 Entry 7: >> <Snipped> End NP Show Structure Display	

その他の参考資料

以降の項では、LISP の実装に関する関連資料について説明します。

関連資料

関連項目	マニュアル タイトル
LISP コマンド：コマンドシンタックスの詳細、コマンドモード、コマンド履歴、デフォルト設定、使用上の注意事項、および例	*Routing Command Reference for Cisco ASR 9000 Series Routers*
LISP コンフィギュレーション ガイド、Cisco IOS リリース	『IP Routing: LISP Configuration Guide, Cisco IOS Release 15M&T』

標準

標準	タイトル
draft-kouvelas-lisp-reliable-transport-00.txt	『*LISP Reliable Transport*』（C. Cassar、I. Kouvelas、および D. Lewis）
RFC 6830	*Locator/ID Separation Protocol*（*LISP*）
RFC 6832	*Interworking between Locator/ID Separation Protocol (LISP) and Non-LISP Sites*
RFC 6833	*Locator/ID Separation Protocol (LISP) Map-Server Interface*

シスコのテクニカル サポート

説明	リンク
シスコのテクニカル サポート Web サイトでは、製品、テクノロジー、ソリューション、技術的なヒント、およびツールへのリンクなどの、数千ページに及ぶ技術情報が検索可能です。Cisco.com に登録済みのユーザは、このページから詳細情報にアクセスできます。	http://www.cisco.com/techsupport