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。

このモジュールでは、BGP の概念および設定情報を示しています。

ヒント `openconfig-network-instance.yang` OpenConfig データモデルを使用して、BGP をプログラムで設定し、運用データを取得することができます。データモデルの使用を開始するには、「*Programmability Configuration Guide for Cisco 8000 Series Routers*」を参照してください。

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ー 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。ユーザー 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 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） BGP プロセスの ASN の変更は、**commit replace** では現在サポートされていません。

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ー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ーが設定できます。

（注） IOS-XR リリース 7.3.1 以降では、ユーザー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、ルータはデフォルト制限値を適用しません。

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。

過剰パスの破棄の設定

最大プレフィックス設定での過剰パスの破棄オプションを使用すると、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をすべて廃棄できます。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。

（注）
- 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。
- システムの物理メモリが不足すると、bgp プロセスが終了し、bpm を手動でリスタートする必要があります。手動でリスタートするには、**process restart bpm** コマンドを使用します。

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** または **end** コマンドを使用します。

手順の詳細

ステップ 1 **configure**

例：

```
RP/0/RP0/cpu 0: router# configure
```

XR 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/RP0/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/RP0/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/RP0/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/RP0/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** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

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//CPU0:router# configure
RP/0//CPU0:router(config)# router bgp 10
RP/0//CPU0:router(config-bgp)# neighbor 10.0.0.1
RP/0//CPU0:router(config-bgp-nbr)# address-family ipv4 unicast
RP/0//CPU0:router(config-bgp-nbr-af)# maximum-prefix 1000 discard-extra-paths
RP/0//CPU0:router(config-bgp-vrf-af)# commit
```

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//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
```

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 ラベル付きユニキャスト

ユニファイド MPLS とも呼ばれる BGP ラベル付きユニキャスト（LU）機能は、多数の IGP の境界（AS 内）または多数の自律システム（AS 間）で分離されたプロバイダーエッジ（PE）ルータ間の MPLS 転送を提供します。自律システム境界ルータ（ASBR）を使用すると、PE のループバックプレフィックスとその MPLS ラベルバインディング（エリア境界ルータ（ABR）間の iBGP、および自律システム境界ルータ間の eBGP）をアドバタイズできます。PE が異な

る自律システム（AS）にある場合は、PE 間でマルチホップ eBGP を使用して VPN ルートを交換できます。BGP LU 接続を持つ PE 間で 6PE およびその他のサービスを実行できます。

BGP LU 機能により、IGP ラベル付きプレフィックススケールおよび隣接関係スケールの値が小さくなります。BGP LU を使用してルータが設定されていない場合は、スケール値の低下を防ぐ必要があります。そのため、BGP LU 機能を有効にする前に、hw-module コマンドを設定する必要があります。hw-module コマンドの設定を有効にするには、ルータを再起動します。

機能制限

- Cisco 8000 は、vrf 単位のラベルモードのみをサポートしています。
- トランスポートアンダーレイとして LDP またはセグメントルーティング（SR）を使用できます。トランスポートアンダーレイとして TE を使用することはできません。
- BGP PIC エッジ機能はサポートされていません。
- BGP LU を介した L3VPN および 6VPE 機能はサポートされていません。
- BGP PIC コア機能がサポートされています。
- **label-allocation-mode** はリリース 7.4.1 で廃止されました。このコマンドの機能は、設定された address-family で label mode コマンドを使用して実行できます。

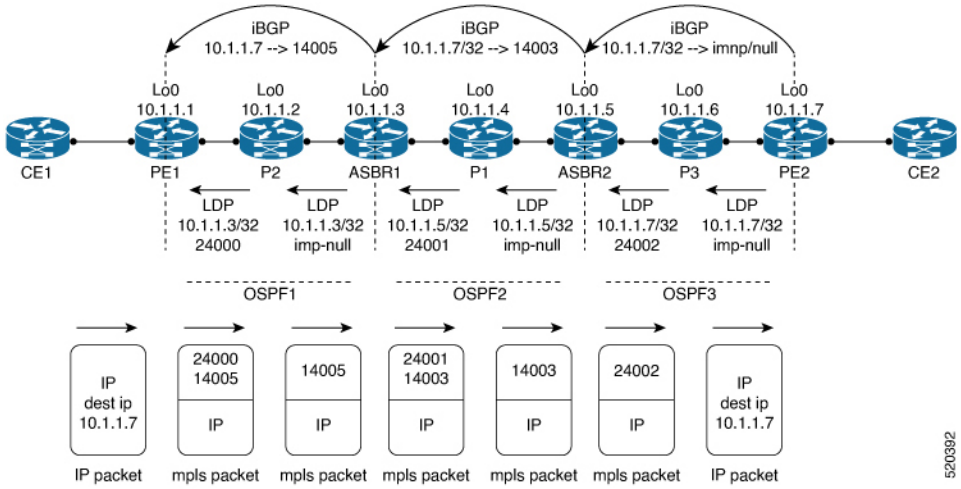
サポートされる機能

サポートされる機能は次のとおりです。

- inter-AS オプション C を使用した BGP LU
- LDP またはセグメントルーティングを使用した MPLS トランスポートを介した 6PE。
- BGP PIC コア

トポロジ

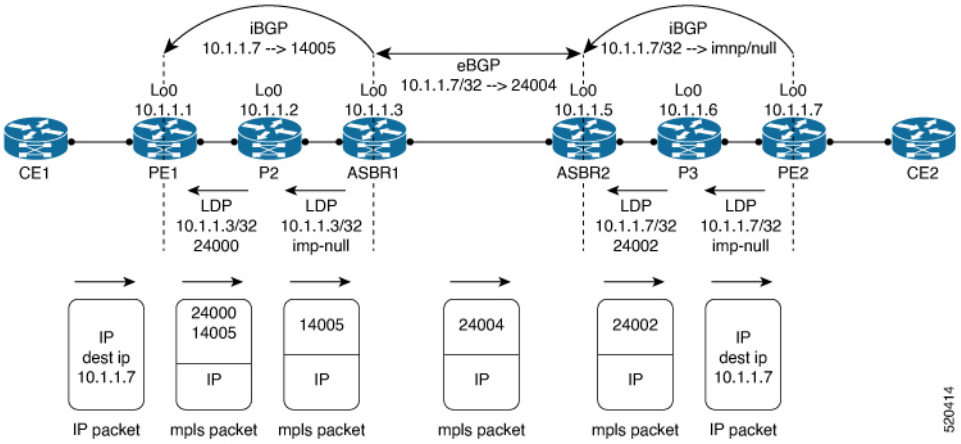
図 *1 : BGP* ラベル付きユニキャスト（自律システム内）のコントロールプレーンおよびデータプレーン



上の図は、MPLS 接続を介して PE1 が PE2 に接続される方法を示しています。PE1 と PE2 は、同じ AS 内の多数のエリアによって分離されています。OSPF1、OSPF2、および OSPF3 の 3 つのネットワークエリアがあるとします。これらの各エリアは、個別の OSPF を実行しています。LDP は、これらの各エリア間のトランスポートとして機能します。プロバイダーエッジルータ PE1 と PE2 間の接続を確立するには、P3、ASBR2、P1 および ASBR1、P2 を介して PE2 から PE1 に iBGP を送信します。PE1 のループバックアドレスと PE2 のループバックアドレス間の接続を確立するために、PE1 は PE2 のループバックアドレスを学習する必要があります。

PE2 のループバックアドレス 10.1.1.7 は、iBGP を介して ASBR2 に BGP ラベルをアドバタイズします。このアドレスは、暗黙的なヌルラベルとしてアドバタイズされます。ASBR2 は、ループバックアドレス 10.1.1.7 にローカルラベル 14003 を割り当て、ASBR1 に送信します。ASBR1 は、ループバックアドレス 10.1.1.7 に独自のラベル 14005 を割り当て、それを PE1 に送信します。PE1 は、ループバックアドレス 10.1.1.7 のプレフィックスと BGP ラベル 14005 を学習しました。PE1 の BGP ネクストホップは ASBR1 です。PE1 がトラフィックを PE2 に送信すると、PE1 は BGP-LU ラベルとトランスポート LDP ラベルの 2 つのラベルを追加します。トランスポート LDP ラベル 24000 は、BGP-LU ラベル 14005 の上にあります。PE1 がループバックアドレス 10.1.1.7 宛ての IP パケットを送信するときに、PE1 はトランスポート LDP ラベルと BGP-LU ラベルをインポーズします。トランスポート LDP ラベルは、パケットを ASBR1 に伝送します。ASBR1 が IP パケットを受信します。これには、BGP-LU ラベル 14005 のみが含まれています。ASBR1 が BGP-LU ラベルを 14005 から 14003 にスワップし、トランスポート LDP ラベル 24001 をインポーズして、IP パケットを ASBR2 に送信します。ASBR2 がパケットを受信します。ASBR2 のループバックアドレス 10.1.1.7 の BGP-LU ラベルは暗黙的なヌルです。トランスポートラベルのみが 24002 にプッシュされます。ASBR2 は、トランスポートを伝送するトランスポートラベルを PE2 に送信します。

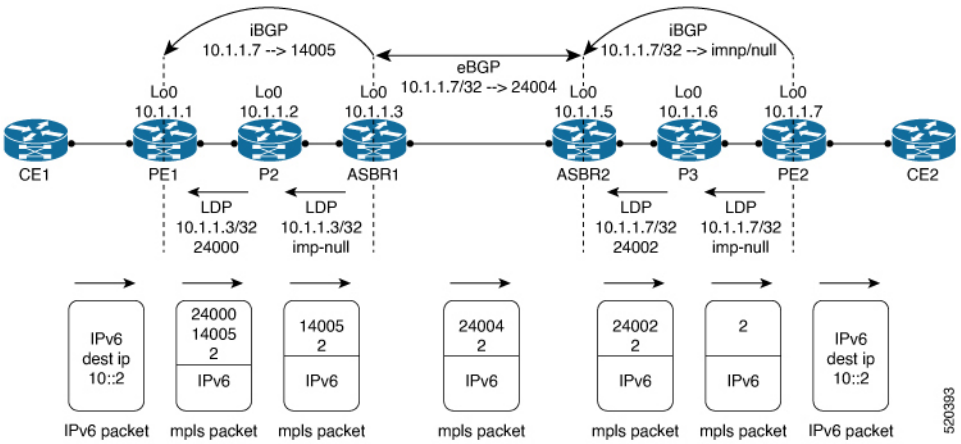
図 *2 : BGP* ラベル付きユニキャスト（自律システム内オプション *C*）のコントロールプレーンおよびデータプレーン



ASBR2 は、BGP パス 10.1.1.7 よりも IGP MPLS パスを優先します。これは、LDP ローカルラベルを BGP ラベルとして ASBR1 にアドバタイズします。LDP スワップ操作が ASBR2 で実行されます。

上の図は、eBGP を使用して MPLS 接続を介して PE1 が PE2 に接続される方法を示しています。上記のシナリオでは、ASBR1 と ASBR2 の間に eBGP が存在します。PE2 は、iBGP を介して暗黙的なヌル値を持つ BGP-LU ラベルを ASBR2 にアドバタイズします。ループバックアドレスは、IGP を介して ASBR2 に認識されます。ASBR2 は、ldp ラベル 24002 の IGP パスを優先します。ASBR2 は、ローカルラベル 24004 をループバック 10.1.1.7 に割り当てます。これは、ローカルラベル 24004 を ASBR1 にアドバタイズします。ASBR1 がローカルラベル 14005 を作成し、PE1 にアドバタイズします。これにより、PE1 がループバックアドレス 10.1.1.7 を認識します。IP パケットには、BGP ラベル 14005 とトランスポートラベル 24000 の 2 つのラベルがあります。PE1 は、IP パケットを ASBR1 に送信します。ASBR1 が受信した IP パケットには、BGP LU ラベル 14005 のみがあります。ASBR1 は、BGP-LU ラベルを 14005 から 24004 にスワップします。IP パケットが ASBR2 に到達し、そこで LDP ラベル 24002 がプッシュされ、パケットが PE2 に送信されます。

図 *3 : BGP LU* を介した *6PE*（*Inter-AS* オプション *C*）のコントロールプレーンおよびデータプレーン



上の図は、複数の AS 間でマルチホップ eBGP を使用して MPLS 接続を介して PE1 が PE2 に接続される方法を示しています。マルチホップ BGP は、PE1 と PE2 の間に存在します。PE1 と PE2 は、ラベルを使用してマルチホップ eBGP で 6PE ルートを交換できます。6PE のラベル値は、v6 の明示的なヌルです。PE2 が v6 プレフィックス 10::2/128 をアドバタイズする場合、ラベルは常に明示的なヌルラベルです。BGP ラベルと LDP ラベルが、上の 2 つのラベルを構成します。6PE ラベルは、v6 の明示的なヌルである下のラベルを構成します。v6 パケットは、宛先 IP 10:2 の PE1 に到達します。ここでラベルインポジションが行われます。値 2 の 6PE ラベルが最初にインポーズされ、次に BGP ラベル 14005 がインポーズされ、次に BGP LU ネクストホップのネクストホップ LDP ラベル 14005 がインポーズされます。ASBR1 は BGP-LU ラベルを 14005 から 24004 にスワップし、パケットを ASBR2 に転送します。ASBR2 は 6PE ラベル 2 の上に LDP ラベルを追加し、LDP ラベルがポップされる P3 に転送するため、PE2 は 6PE の明示的なヌルラベルのみを持つパケットを受信します。PE2 は v6 ルックアップを実行し、パケットを転送します。

BGP ラベル付きユニキャストの設定

```
Router(config)# hw-module profile cef bgplu enable
Router(config)# router bgp 1
Router(config-bgp)# bgp router-id 2001:DB8::1
Router(config-bgp)# address-family ipv6 unicast
Router(config-bgp-af)# redistribute connected route-policy set-lbl-idx
Router(config-bgp-af)# allocate-label all
Router(config-bgp-af)# exit
Router(config-bgp)# neighbor 2001:DB8::2
Router(config-bgp)# remote-as 1
Router(config-bgp)# update-source Loopback 0
Router(config-bgp)# address-family ipv6 labeled-unicast
Router(config-bgp)# route-policy pass-all in
Router(config-bgp)# route-policy pass-all out

/* Note: Restart the router for the hw-module command configuration to take effect. */
```

実行コンフィギュレーション

```
!
hw-module profile cef bgplu enable
!
router bgp 1
 bgp router-id 2001:DB8::1
 address-family ipv6 unicast
 redistribute connected route-policy set-lbl-idx
 allocate-label all
!
 neighbor 2001:DB8::2
 remote-as 1
 update-source Loopback0

!
 address-family ipv6 labeled-unicast
 route-policy pass-all in
 route-policy pass-all out
!
```

確認

SME は、以下の必要な show 出力を提供します。

```
Router # show bgp ipv6 unicast labels
  Network              Next Hop         Rcvd Label      Local Label

Router# show bgp ipv6 unicast labels
Network            Next Hop        Rcvd Label      Local Label
```

BGP ラベル付きユニキャスト PIC エッジのコンバージェンス

表 *1:* 機能の履歴（表）

機能名	リリース情報	機能説明
BGP ラベル付きユニキャスト PIC エッジのコンバージェンス	リリース 7.7.1	この機能により、入力プロバイダーエッジルータに障害が発生した場合、または PE ルータ接続が失われて別の PE ルータを接続する必要がある場合に、BGP ラベル付きユニキャスト（LU）ルートのコンバージェンス時間が 1 秒未満に短縮されます。この機能により、BGP LU ルートのプライマリパスで障害が発生した場合に、トラフィックのドロップが最小限に抑えられます。

BGP ラベル付きユニキャスト（LU）PIC エッジ機能を使用すると、プライマリパスとバックアップパスの両方を作成して、ルーティング情報ベース（RIB）、転送情報ベース（FIB）、および Cisco Express Forwarding に保存することができます。ルータが障害を検出するとバックアップパスまたは代替パスによりただちに引き継がれるため、この機能により 1 秒未満での高速フェールオーバーとコンバージェンスが可能になります。

BGP LU PIC エッジが機能するには、入力 PE や自律システム境界ルータ（ASBR）などのエッジ iBGP デバイスが BGP PIC をサポートしていて、バックアップ BGP ネクストホップを受信する必要があります。

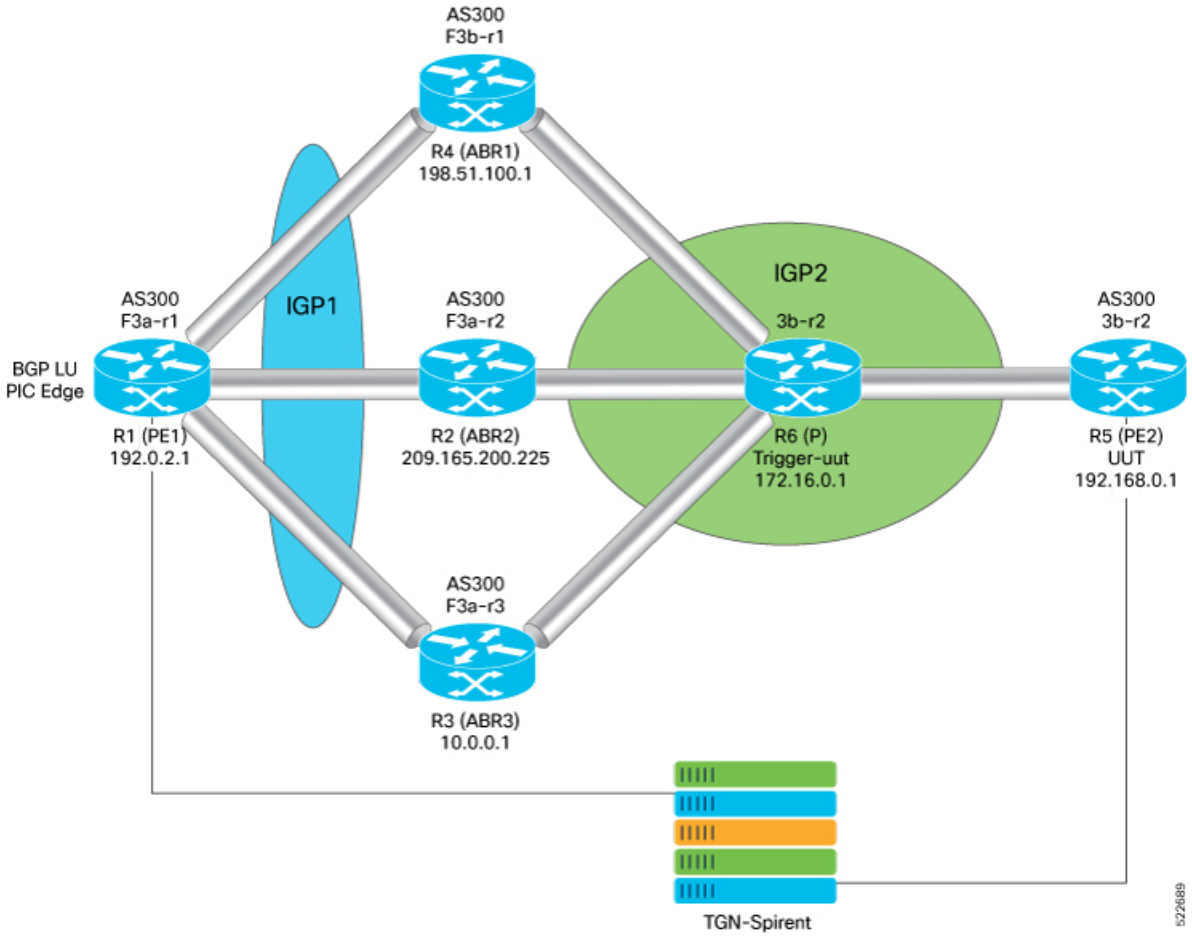
次のトポロジ図は、BGP ラベル付きユニキャスト PIC エッジのコンバージェンス機能を示しています。トポロジの説明を次に示します。

- BGP LU PIC エッジ機能が、プロバイダーエッジルータの PE1 で有効になっています。
- PE1 は、リモート PE ルータの PE2 から BGP LU プレフィックスを学習します。
- PE1 は、エリア境界ルータの ABR1、ABR2、および ABR3 を介してトラフィックをルーティングします。そのうちの 1 つで障害が発生すると、障害が発生した ABR の事前にプログラムされたバックアップがトラフィックをルーティングします。
- PE1 は、エリア境界ルータの ABR1、ABR2、および ABR3 を介してトラフィックをルーティングします。
- PE2 はバックアップまたは代替ネクストホップとしてマークされ、PE1 の FIB にプログラムされます。

- PE1 は、PE2 が ABR1 を介して到達不能であることを学習すると、PE1 のプレフィックスの BGP ネクストホップを ABR2 にただちに変更します。
- プレフィックスの数に関係なく、スイッチオーバーは 1 秒未満で行われます。
- 複数の BGP プレフィックスに対する更新が保留中であっても、1 秒未満でコンバージェンスが行われます。

トポロジ

図 *4 : BGP LU PIC* エッジ



注意事項と制約事項

この機能は、ルータが複数の内部 BGP パスと複数の外部 BGP パスを転送テーブルにインストールできるようにする BGP マルチパスをサポートします。マルチパスにより、BGP は複数のリンク間でトラフィックをロードバランシングできます。

コンバージェンス時間は、BGP LU ルートスケールとは無関係です。

BGP ラベル付きユニキャスト PIC エッジのコンバージェンスの設定

BGP ラベル付きユニキャスト PIC エッジのコンバージェンスを設定するには、次の手順を実行します。

- BGP ラベル付きユニキャストを設定し、ルートポリシーを BGP アドレスファミリにアタッチします。
- BGP ラベル付きユニキャストマルチパスを設定し、ルートポリシーを BGP アドレスファミリにアタッチします

```
Router(config)# route-policy BGP-PIC-EDGE
Router(config-rpl)# set path-selection backup 1 install
Router(config-rpl)# end-policy
Router(config)# end
Router(config)# router bgp 200
Router(config-bgp)# bgp router-id 10.0.0.1
Router(config-bgp)#  address-family ipv4 unicast
Router(config-bgp-af)# additional-paths receive
Router(config-bgp-af)# additional-paths send
Router(config-bgp-af)# additional-paths selection route-policy BGP-PIC-EDGE


/*Perform the following steps to configure BGP labeled unicast multipath and attach
route-policy to BGP address families: */
Router(config)# route-policy BGP-PIC-EDGE-MULTIPATH
Router(config-rpl)# set path-selection backup 1 install multipath-protect
Router(config)# end-policy
Router(config)# router bgp 200
Router(config)# bgp router-id 192.168.1.0
Router(config)# address-family ipv4 unicast
Router(config)# maximum-paths ibgp 2
Router(config)# additional-paths receive
Router(config)# additional-paths send
Router(config)# additional-paths selection route-policy BGP-PIC-EDGE-MULTIPATH
```

実行コンフィギュレーション

```
route-policy BGP-PIC-EDGE
 set path-selection backup 1 install
 end-policy
router bgp 200
 bgp router-id 192.168.1.0
 address-family ipv4 unicast
  additional-paths receive
  additional-paths send
  additional-paths selection route-policy BGP-PIC-EDGE

route-policy BGP-PIC-EDGE-MULTIPATH
 set path-selection backup 1 install multipath-protect
 end-policy
router bgp 200
 bgp router-id 192.168.1.0
 address-family ipv4 unicast
  maximum-paths ibgp 2
  additional-paths receive
  additional-paths send
  additional-paths selection route-policy BGP-PIC-EDGE-MULTIPATH
```

確認

バックアップパスが確立されていることを確認します。

```
Router# show cef 192.0.2.1/32
192.168.0.0/32, version 31, internal 0x5000001 0x40 (ptr 0x901d2370) [1], 0x0 (0x90d2beb8),
 0xa08 (0x91c74378)
 Prefix Len 32, traffic index 0, precedence n/a, priority 4
   via 203.0.113.1/32, 3 dependencies, recursive [flags 0x6000]  << Primary Path
    path-idx 0 NHID 0x0 [0x90319650 0x0]
    recursion-via-/32
    next hop 192.51.100.1/32 via 24006/0/21
    next hop 209.165.200.225/32 Hu0/0/0/25   labels imposed {24002 24000}
    next hop 10.0.0.1/32 Hu0/0/0/26   labels imposed {24002 24000}
   via 203.0.113.2/32, 2 dependencies, recursive, backup [flags 0x6100]  << Backup Path
    path-idx 1 NHID 0x0 [0x903197b8 0x0]
    recursion-via-/32
    next hop 209.165.200.225/32 via 24005/0/21
    next hop 192.51.100.1/32 Hu0/0/0/25   labels imposed {24001 24000}
    next hop 10.0.0.1/32 Hu0/0/0/26   labels imposed {24001 24000}
```

ブラック ボックス モニタリング

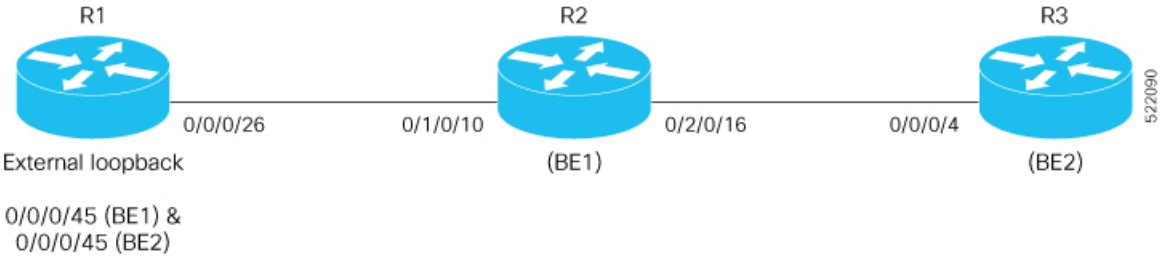
表 *2:* 機能の履歴（表）

機能名	リリース情報	機能説明
ブラック ボックス モニタリング	リリース 7.3.2	この機能を使用すると、ルータ上に転送パスを設定し、ネットワークデバイスに固有のシステムメトリックについてお客様の回線をプローブするために使用できます。このようなモニタリングは、お客様とのサービスレベル契約を維持するために役立ちます。

この機能では、GRE カプセル化およびカプセル化解除インフラストラクチャ全体でダミーの BGP セッションを確立する技術が使用されます。ダミーの BGP セッションを終了するために、ルータは、自身にピアリングしているピアリングファブリックで設定されているアドレスにピアリングします。

ルータは、本質的には自身にピアリングしている PF で設定されているアドレスにピアリングする必要があります。これを機能させる唯一の方法は、2 つのインターフェイスを物理ケーブルで相互に接続することです。2 つのインターフェイスが相互に接続された後、BGP セッションが開始されるように、インターフェイスの 1 つを VRF に配置します。ルータは通常、自身への BGP セッションの確立を試みないため、VRF を使用してルーティングテーブルを分離する必要があります。もう一方のインターフェイスでは、これは、PF のピアリングインターフェイスでの通常の設定と同じ設定を持つ、グローバル vrf の「通常の」インターフェイスです。

設定例



BGP および GRE トンネルを設定するには、次の手順を実行します。

```
/* Configure the Local Proxy ARP on the Bundle-Ether interfaces.*/
Router(config)# interface Bundle-Ether1.1
Router(config-if)# ipv4 address 10.1.1.1 255.255.255.240
Router(config-if)# local-proxy-arp
Router(config-if)# encapsulation dot1q 12
Router(config-if)# ipv4 access-group acl-aa ingress

Router(config-if)# exit
Router(config)# interface Bundle-Ether2.1
Router(config-if)# vrf aa
Router(config-if-vrf)# ipv4 address 10.1.1.2 255.255.255.240
Router(config-if-vrf)# local-proxy-arp
Router(config-if-vrf)# encapsulation dot1q 12

/* Configure a bundle on FortyGigE interfaces.*/
Router(config)# interface FortyGigE 0/0/0/46
Router(config-if)# bundle id 1 mode on
Router(config-if)# exit
Router(config)# interface FortyGigE0/0/0/47
Router(config-if)# bundle id 2 mode on

/* Configure the access list.*/
Router(config-if)# ipv4 access-list acl-aa
Router(config-if)# 1 permit icmp any host 10.1.1.1 echo-reply
Router(config-if)# 2 permit ipv4 any any nexthop1 ipv4 100.100.2.2
Router(config-if)# 10 permit tcp any eq bgp any
Router(config-if)# 20 permit tcp any any eq bgp

/* Configure BGP.*/
Router(config)# router bgp 100
Router(config-bgp)# bgp router-id 10.10.10.10
Router(config-bgp)# bgp log neighbor changes detail
Router(config-bgp)# address-family ipv4 unicast
Router(config-bgp)# maximum-paths ebgp 64
Router(config-bgp)# maximum-paths ibgp 64

/* Apply route policy. */
Router(config)# address-family vpnv4 unicast
Router(config-af)# vrf aa
Router(config-af)# rd auto
Router(config-af)# exitexit
Router(config)# address-family ipv4 unicast
Router(config)# exit
Router(config)# neighbor 10.1.1.1
Router(config-nbr)# remote-as 200
Router(config-nbr)# ebgp-multihop 4
Router(config-nbr)# exit
Router(config)# address-family ipv4 unicast
```

```
Router(config-af)#send-community-ebgp
Router(config-af)# route-policy pass-all in
Router(config-af)# route-policy pass-all out

/* Configure loopback interfaces. */
Router(config)# interface Loopback1001
Router(config-if)# ipv4 address 10.10.10.10 255.255.255.255
Router(config)# exit
Router(config)# interface Loopback1002
Router(config-if)# vrf aa
Router(config-if-vrf)# ipv4 address 10.10.10.10 255.255.255.255

/* Configure a class map. */
Router(config)# class-map type traffic match-all aa
Router(config-cmap)# match protocol gre
Router(config-cmap)# match destination-address ipv4 10.10.10.10 255.255.255.255
Router(config-cmap)# end-class-map

/* Configure a policy map. */
Router(config)# policy-map type pbr pmap1
Router(config-pmap)# class type traffic aa
Router(config-pmap-c)# decapsulate gre
Router(config-pmap-c)# class type traffic class-default
Router(config-pmap-c)# end-policy-map

/* Configure VRF policy. */
Router(config)# vrf-policy
Router(config-vrf)# vrf default address-family ipv4 policy type pbr input pmap1
Router(config)# interface tunnel-ip 1100
Router(config-if)#ipv4 unnumbered Loopback1001
Router(config-if)#tunnel mode gre ipv4 encap
Router(config-if)#tunnel source Loopback1001
Router(config-if)#tunnel destination 200.1.2.1
Router(config-if)#logging events link-status
```

実行コンフィギュレーション

```
interface Bundle-Ether1.1
 ipv4 address 10.1.1.1 255.255.255.240
 local-proxy-arp
 encapsulation dot1q 12
 ipv4 access-group aa-acl ingress

interface Bundle-Ether2.1
 vrf aa
 ipv4 address 10.1.1.2 255.255.255.240
 local-proxy-arp
 encapsulation dot1q 12

interface FortyGigE0/0/0/46
 bundle id 1 mode on

interface FortyGigE0/0/0/47
 bundle id 2 mode on
ipv4 access-list aa-acl
 1 permit icmp any host 10.1.1.1 echo-reply
 2 permit ipv4 any any nexthop1 ipv4 100.100.2.2
 10 permit tcp any eq bgp any
 20 permit tcp any any eq bgp

router bgp 100
```

```
 bgp router-id 10.10.10.10
 bgp log neighbor changes detail
 address-family ipv4 unicast
  maximum-paths ebgp 64
  maximum-paths ibgp 64
 !
 address-family vpnv4 unicast
 !
 vrf aa
  rd auto
  address-family ipv4 unicast
  !
  neighbor 10.1.1.1
   remote-as 200
   ebgp-multihop 4
   address-family ipv4 unicast
    send-community-ebgp
    route-policy pass-all in
    route-policy pass-all out

interface Loopback1001
 ipv4 address 10.10.10.10 255.255.255.255
RP/0/RP0/CPU0:SF-DD#sh run int loopback 1002
interface Loopback1002
 vrf aa
 ipv4 address 10.10.10.10 255.255.255.255

class-map type traffic match-all aa
 match protocol gre
 match destination-address ipv4 10.10.10.10 255.255.255.255
 end-class-map

policy-map type pbr pmap1
 class type traffic aa
  decapsulate gre
 class type traffic class-default
 end-policy-map
!
vrf-policy
 vrf default address-family ipv4 policy type pbr input pmap1

interface tunnel-ip1100
 ipv4 unnumbered Loopback1001
 tunnel mode gre ipv4 encap
 tunnel source Loopback1001
 tunnel destination 200.1.2.1
 logging events link-status
```

確認

ブラック ボックス モニタリングの設定を確認します。

```
Router# show bgp vrf aa neighbors
BGP neighbor is 10.1.1.1, vrf aa
 Remote AS 200, local AS 100, external link
 Remote router ID 200.1.2.1
  BGP state = Established, up for 00:12:35
  NSR State: None
  Last read 00:00:30, 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 00:00:30, attempted 19, written 19
  Second last write 00:01:30, 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  Sep 29 05:50:49.983 last full not set pulse count 30
  Last write pulse rcvd before reset 00:00:00
Connections established 1; dropped 0
  Local host: 10.1.1.2, Local port: 52660, IF Handle: 0x00000000
  Foreign host: 10.1.1.1, Foreign port: 179
  Last reset 00:00:00
  External BGP neighbor may be up to 4 hops away.
```

BGP ラベル付きユニキャストバージョン 6

表 *3:* 機能の履歴（表）

機能名	リリース情報	機能説明
BGP ラベル付きユニキャストバージョン 6	リリース 7.3.16	この機能は、IPv6 を介した BGP ラベル付きユニキャスト（LU）機能を拡張します。この機能は、L3VPN や 6PVE などのサービスを実行するための PE 間の接続を提供します。この機能により、PE は自律システム（AS）の境界を越えてトラフィックを転送できます。 BGP LU を使用すると、IGP の境界を越えて MPLS トラフィックを転送できます。ルータは、IGP の境界を越えてループバックとラベルバインディングをアドバタイズすることで、同じローカル IGP を共有していないリモートエリアの他のルータと通信します。

BGP ラベル付きユニキャストの概要

ユニファイド MPLS とも呼ばれる BGP ラベル付きユニキャスト（LU）機能は、多数の IGP の境界（AS 内）または多数の自律システム（AS 間）で分離されたプロバイダーエッジ（PE）ルータ間の MPLS 転送を提供します。自律システム境界ルータ（ASBR）を使用すると、PE のループバックプレフィックスとその MPLS ラベルバインディング（エリア境界ルータ（ABR）間の iBGP、および自律システム境界ルータ間の eBGP）をアドバタイズできます。PE が異なる自律システム（AS）にある場合は、PE 間でマルチホップ eBGP を使用して VPN ルートを交換できます。BGP LU 接続を持つ PE 間で 6PE およびその他のサービスを実行できます。

BGP LU 機能により、IGP ラベル付きプレフィックススケールおよび隣接関係スケールの値が小さくなります。BGP LU を使用してルータが設定されていない場合は、スケール値の低下を防ぐ必要があります。そのため、BGP LU 機能を有効にする前に、hw-module コマンドを設定する必要があります。hw-module コマンドの設定を有効にするには、ルータを再起動します。

BGP ラベル付きユニキャストバージョン 6（BGP LU v6）機能は、IPv6 を介した BGP ラベル付きユニキャスト（LU）機能を拡張します。

機能制限

- BGP LU を介した 6VPE 機能はサポートされていません。
- AFI 間はサポートされていません。
- BGP PIC コア機能はサポートされていません。
- 同じネイバーでの 6PE との共存はサポートされていません。
- BGP LU バージョン 6 IPv6 ユニキャストアドレス ファミリの共存はサポートされていません。
- BGP LU v6 を介した VPNV6 はサポートされていません。
- リンクローカル アドレスはサポートされていません。
- BGP LU 自体がトランスポートである書き換えのケースはサポートされていません。
- Carrier Supporting Carrier バージョン 6 はサポートされていません。
- BGP LU バージョン 6 を使用した Inter-AS オプション C はサポートされていません。

BGP ラベル付きユニキャストバージョン 6 の設定

```
Router(config)# hw-module profile cef bgplu enable
Router(config)# router bgp 1
Router(config-bgp)# bgp router-id 2001:DB8::1
Router(config-bgp)# address-family ipv6 unicast
Router(config-bgp-af)# redistribute connected route-policy set-lbl-idx
Router(config-bgp-af)# allocate-label all
Router(config-bgp-af)# exit
Router(config-bgp)# neighbor 2001:DB8::2
Router(config-bgp)# remote-as 1
Router(config-bgp)# update-source Loopback 0
Router(config-bgp)# address-family ipv6 labeled-unicast
Router(config-bgp)# route-policy pass-all in
Router(config-bgp)# route-policy pass-all out
Router(config-bgp)# commit
```

（注） **hw-module profile cef bgplu enable** コマンドを有効にするには、ルータをリロードします。

実行コンフィギュレーション

```
hw-module profile cef bgplu enable
router bgp 1
 bgp router-id 2001:DB8::1
 address-family ipv6 unicast
  redistribute connected route-policy set-lbl-idx
  allocate-label all
  exit
 neighbor 2001:DB8::2
 remote-as 1
```

```
 update-source Loopback 0
  address-family ipv6 labeled-unicast
   route-policy pass-all in
   route-policy pass-all out
```

確認

BGP LU が設定されていることを確認します。

```
Router# show hw-module profile cef
Thu Jun 17 00:06:32.974 UTC
-----------------------------------------------------------------------------------------
Knob                          Status        Applied   Action
-----------------------------------------------------------------------------------------
BGPLU                        Configured      Yes       None
LPTS ACL                     Unconfigured    Yes       None
Dark Bandwidth               Unconfigured    Yes       None
MPLS Per Path Stats          Unconfigured    Yes       None
Tunnel TTL Decrement         Unconfigured    Yes       None
High-Scale No-LDP-Over-TE    Unconfigured    Yes       None
IPv6 Hop-limit Punt          Unconfigured    Yes       None
IP Redirect Punt             Unconfigured    Yes       None
```

ルートパスの詳細を、BGP およびトランスポートラベル情報とともに確認します。

```
Router# show cef ipv6 192:168:9::80/128
Wed Jun 16 07:42:04.789 UTC
192:168:9::80/128, version 27, internal 0x5000001 0x40 (ptr 0x93f2d478) [1], 0x0
(0x93ef6cc0), 0xa08 (0x9460a8a8)
 Updated Jun 16 07:36:00.189
 Prefix Len 128, traffic index 0, precedence n/a, priority 4, encap-id 0x1001000000001
   via 10:0:1::51/128, 3 dependencies, recursive [flags 0x6000]
    path-idx 0 NHID 0x0 [0x94720660 0x0]
    recursion-via-/128
    next hop 10:0:1::51/128 via 16061/0/21
     next hop fe80::7af8:c2ff:fee4:20c0/128 Hu0/0/0/27   labels imposed {16061 25001}
/*
16061 - Transport Label
25001 – BGP Label
*/
```

BGP プロセスの BGP LU バージョン 6 ルートと BGP ラベル情報を確認します。

```
Router# show bgp ipv6 unicast labels
Wed Jun 16 07:34:58.968 UTC
BGP router identifier 10.0.1.50, local AS number 1
BGP generic scan interval 60 secs
Non-stop routing is enabled
BGP table state: Active
Table ID: 0xe0800000   RD version: 6
BGP main routing table version 6
BGP NSR Initial initsync version 3 (Reached)
BGP NSR/ISSU Sync-Group versions 0/0
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        Rcvd Label      Local Label
*> 192:168::/64        192:168:1::70   nolabel         24006
*>i192:168:9::80/128   10:0:1::51      25001           nolabel

Processed 2 prefixes, 2 paths
```

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。

ピアリング インターフェイスの IPv6 アドレスとしてのネクスト ホップ

BGP を使用すると、IPv4 セッションで IPv6 プレフィックスを伝送できます。IPv6 プレフィックスのネクスト ホップは、ネクスト ホップ ポリシーを使用して設定できます。ポリシーが設定されていない場合、ネクスト ホップはピアリング インターフェイスの IPv6 アドレスとして設定されます（いずれかのインターフェイスが設定されている場合は、IPv6 ネイバーインターフェイスまたは IPv6 の更新送信元インターフェイス）。

ネクスト ホップ ポリシーが設定されておらず、IPv6 ネイバー インターフェイスも IPv6 更新送信元インターフェイスも設定されていない場合は、ネクスト ホップは IPv4 射影 IPv6 アドレスになります。

グローバルアドレスを使用した IPv6 マルチプロトコル BGP ピアリング

いずれかのインターフェイスを除くすべての ECMP リンクがシャットダウンされると、ネクストホップがグローバルアドレスからリンクローカルアドレスに変更され、数秒間の過渡時間の間、すべてのフローのトラフィック損失が発生します。

その後、BGP テーブルポリシーで **set next-hop ipv6-global** コマンドを設定して、妨げられていないパスでトラフィック損失を回避できます。

BGP は、マルチパスルートのグローバル IPv6 アドレスネクストホップをインストールし、単一パスルートの linklocal と ifhandle をインストールして、ebgp neighbor を直接接続します。BGP テーブルポリシーで **set next-hop ipv6-global** コマンドを次のように設定して、グローバル ipv6 アドレスネクストホップを設定できます。

```
route-policy RESILIENT-HASH-V6
  if destination in (1000:1000::/32 le 128) or destination in (2000:1000::/32 le 128)
then
    set load-balance ecmp-consistent
    set next-hop ipv6-global
    pass
  endif
```

```
  pass
end-policy
```

範囲を指定した IPv4 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 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 ユニキャストに属していることを示します。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

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。

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。

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。

```
Router# configuration
Router(config)# router bgp 140
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。

```
Router(config)# router bgp 112
Router(config-bgp)# address-family ipv4 unicast
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。

```
Router(config)# router bgp 140
Router(config-bgp)# neighbor 10.0.0.1
Router(config-bgp-nbr)#
```

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。

```
Router(config)# router bgp 140
```

```
Router(config-bgp)# vrf vrf_A
Router(config-bgp-vrf)#
```

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/RP0/cpu 0: router(config)# router bgp 112
RP/0/RP0/cpu 0: router(config-bgp)# vrf vrf_A
RP/0/RP0/cpu 0: router(config-bgp-vrf)# neighbor 11.0.1.2
RP/0/RP0/cpu 0: router(config-bgp-vrf-nbr)# address-family ipv4 unicast
RP/0/RP0/cpu 0: router(config-bgp-vrf-nbr-af)#
```

VPNv6 アドレス ファミリ コンフィギュレーション モード

次に、VPNv6 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。

```
Router(config)# router bgp 150
Router(config-bgp)# address-family vpnv6 unicast
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。

```
Router(config)# router bgp 100
Router(config-bgp)# address-family l2vpn vpls-vpws
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 ソフトウェアでは、この設定は次のとおりです。

```
Router(config-bgp)# neighbor 192.23.1.2
Router(config-bgp-nbr)# remote-as 2002
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 ソフトウェアでは、この設定は次のとおりです。

```
Router(config-bgp)# neighbor 2002::2
Router(config-bgp-nbr)# remote-as 2023
Router(config-bgp-nbr)# address-family ipv6 unicast
Router(config-bgp-nbr-af)# next-hop-self
Router(config-bgp-nbr-af)# route-policy one 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 または IPv6 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

```
Router(config)# router bgp 140
Router(config-bgp)# af-group afmcast1 address-family ipv4 unicast
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。

```
Router# router bgp 140
Router(config-bgp)# session-group session1
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。

```
Router(config)# router bgp 123
Router(config-bgp)# neighbor-group nbrgroup1
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。

```
Router(config)# router bgp 140
Router(config-bgp)# neighbor-group nbrgroup1
Router(config-bgp-nbrgrp)# address-family ipv4 unicast
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。

```
Router(config)# router bgp 140
Router(config-bgp)# neighbor-group AS_1
Router(config-bgp-nbrgrp)# advertisement-interval 15
Router(config-bgp-nbrgrp)# exit
Router(config-bgp)# neighbor 10.1.1.1
Router(config-bgp-nbr)# remote-as 1
Router(config-bgp-nbr)# use neighbor-group AS_1
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。

```
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。

```
Router(config)# router bgp 140
Router(config-bgp)# session-group AS_2
Router(config-bgp-sngrp)# advertisement-interval 15
Router(config-bgp-sngrp)# exit
Router(config-bgp)# neighbor-group AS_1
Router(config-bgp-nbrgrp)# advertisement-interval 20
Router(config-bgp-nbrgrp)# exit
Router(config-bgp)# neighbor 192.168.0.1
Router(config-bgp-nbr)# remote-as 1
Router(config-bgp-nbr)# use session-group AS_2
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。

```
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。

```
Router(config)# router bgp 150
Router(config-bgp)# session-group AS_2
Router(config-bgp-sngrp)# advertisement-interval 20
Router(config-bgp-sngrp)# exit
Router(config-bgp)# neighbor-group AS_1
Router(config-bgp-nbrgrp)# advertisement-interval 15
Router(config-bgp-nbrgrp)# exit
Router(config-bgp)# neighbor 192.168.1.1
Router(config-bgp-nbr)# remote-as 1
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。

```
 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。

```
Routerconfig)# router bgp 140
Router(config-bgp)# session-group ADV
Router(config-bgp-sngrp)# advertisement-interval 15
Router(config-bgp-sngrp)# exit
Router(config-bgp)# neighbor-group ADV_2
Router(config-bgp-nbrgrp)# advertisement-interval 25
Router(config-bgp-nbrgrp)# address-family ipv4 unicast
Router(config-bgp-nbrgrp-af)# route-policy POLICY_1 in
Router(config-bgp-nbrgrp-af)# exit
Router(config-bgp-nbrgrp)# exit
Router(config-bgp)# exit
Router(config-bgp)# neighbor 192.168.2.2
Router(config-bgp-nbr)# remote-as 1
```

```
Router(config-bgp-nbr)# use session-group ADV
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。

```
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。

```
Router(config)# router bgp 140
Router(config-bgp)# neighbor-group AS_1
Router(config-bgp-nbrgrp)# remote-as 1
Router(config-bgp-nbrgrp)# exit
Router(config-bgp)# neighbor-group adv_15
Router(config-bgp-nbrgrp)# remote-as 10
Router(config-bgp-nbrgrp)# advertisement-interval 15
Router(config-bgp-nbrgrp)# exit
Router(config-bgp)# neighbor 10.0.101.5
Router(config-bgp-nbr)# use neighbor-group AS_1
Router(config-bgp-nbr)# exit
Router(config-bgp)# neighbor 10.0.101.10
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。

```
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。

```
Router(config)# router bgp 142
Router(config-bgp)# af-group GROUP_3 address-family ipv4 unicast
Router(config-bgp-afgrp)# next-hop-self
Router(config-bgp-afgrp)# route-policy POLICY_1 in
Router(config-bgp-afgrp)# exit
Router(config-bgp)# session-group GROUP_2
Router(config-bgp-sngrp)# advertisement-interval 15
Router(config-bgp-sngrp)# exit
Router(config-bgp)# neighbor-group GROUP_1
Router(config-bgp-nbrgrp)# use session-group GROUP_2
Router(config-bgp-nbrgrp)# ebgp-multihop 3
Router(config-bgp-nbrgrp)# address-family ipv4 unicast
Router(config-bgp-nbrgrp-af)# weight 100
Router(config-bgp-nbrgrp-af)# send-community-ebgp
Router(config-bgp-nbrgrp-af)# exit
Router(config-bgp-nbrgrp)# exit
Router(config-bgp)# neighbor 192.168.0.1
Router(config-bgp-nbr)# remote-as 2
Router(config-bgp-nbr)# use neighbor-group GROUP_1
```

```
Router(config-bgp-nbr)# address-family ipv4 unicast
Router(config-bgp-nbr-af)# use af-group GROUP_3
Router(config-bgp-nbr-af)# weight 200
```

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。

```
Router(config)# router bgp 142
Router(config-bgp)# af-group GROUP_3 address-family ipv4 unicast
Router(config-bgp-afgrp)# next-hop-self
Router(config-bgp-afgrp)# route-policy POLICY_1 in
Router(config-bgp-afgrp)# exit
Router(config-bgp)# session-group GROUP_2
Router(config-bgp-sngrp)# advertisement-interval 15
Router(config-bgp-sngrp)# exit
Router(config-bgp)# neighbor-group GROUP_1
Router(config-bgp-nbrgrp)# use session-group GROUP_2
Router(config-bgp-nbrgrp)# ebgp-multihop 3
Router(config-bgp-nbrgrp)# address-family ipv4 unicast
Router(config-bgp-nbrgrp-af)# weight 100
Router(config-bgp-nbrgrp-af)# send-community-ebgp
Router(config-bgp-nbrgrp-af)# exit
Router(config-bgp-nbrgrp)# exit
Router(config-bgp)# neighbor 192.168.0.1
Router(config-bgp-nbr)# remote-as 2
Router(config-bgp-nbr)# use neighbor-group GROUP_1
Router(config-bgp-nbr)# address-family ipv4 unicast
Router(config-bgp-nbr-af)# use af-group GROUP_3
Router(config-bgp-nbr-af)# weight 200
```

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。

```
Router(config)# router bgp 140
Router(config-bgp)# af-group GROUP_3 address-family ipv4 unicast
Router(config-bgp-afgrp)# remove-private-as
Router(config-bgp-afgrp)# route-policy POLICY_1 in
Router(config-bgp-afgrp)# exit
Router(config-bgp)# af-group GROUP_1 address-family ipv4 unicast
Router(config-bgp-afgrp)# use af-group GROUP_2
Router(config-bgp-afgrp)# maximum-prefix 2500 75 warning-only
Router(config-bgp-afgrp)# default-originate
Router(config-bgp-afgrp)# exit
Router(config-bgp)# af-group GROUP_2 address-family ipv4 unicast
Router(config-bgp-afgrp)# use af-group GROUP_3
Router(config-bgp-afgrp)# send-community-ebgp
Router(config-bgp-afgrp)# send-extended-community-ebgp
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。

```
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。

```
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。

```
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。

```
Router(config)# router bgp 113
Router(config-bgp)# session-group GROUP_1
Router(config-bgp-sngrp)# use session-group GROUP_2
Router(config-bgp-sngrp)# update-source Loopback 0
Router(config-bgp-sngrp)# exit
Router(config-bgp)# session-group GROUP_2
Router(config-bgp-sngrp)# use session-group GROUP_3
Router(config-bgp-sngrp)# ebgp-multihop 2
Router(config-bgp-sngrp)# exit
Router(config-bgp)# session-group GROUP_3
Router(config-bgp-sngrp)# dmz-link-bandwidth
```

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。

```
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。

```
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。

```
Router# show bgp session-group GROUP_3 users
```

```
Session: s:GROUP_1 s:GROUP_2
```

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。

```
Router(config)# router bgp 113
Router(config-bgp)# session-group GROUP_1
Router(config-bgp-sngrp)# use session-group GROUP_2
Router(config-bgp-sngrp)# update-source Loopback 0
Router(config-bgp-sngrp)# exit
Router(config-bgp)# session-group GROUP_2
Router(config-bgp-sngrp)# use session-group GROUP_3
Router(config-bgp-sngrp)# ebgp-multihop 2
Router(config-bgp-sngrp)# exit
Router(config-bgp)# session-group GROUP_3
Router(config-bgp-sngrp)# dmz-link-bandwidth
```

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。

```
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。

```
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。

```
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。

```
Router(config)# router bgp 140
Router(config-bgp)# af-group GROUP_3 address-family ipv4 unicast
Router(config-bgp-afgrp)# remove-private-as
Router(config-bgp-afgrp)# soft-reconfiguration inbound
Router(config-bgp-afgrp)# exit
Router(config-bgp)# af-group GROUP_2 address-family ipv4 unicast
Router(config-bgp-afgrp)# use af-group GROUP_3
Router(config-bgp-afgrp)# send-community-ebgp
Router(config-bgp-afgrp)# send-extended-community-ebgp
Router(config-bgp-afgrp)# capability orf prefix both
Router(config-bgp-afgrp)# exit
Router(config-bgp)# session-group GROUP_3
Router(config-bgp-sngrp)# timers 30 90
Router(config-bgp-sngrp)# exit
Router(config-bgp)# neighbor-group GROUP_1
Router(config-bgp-nbrgrp)# remote-as 1982
Router(config-bgp-nbrgrp)# use neighbor-group GROUP_2
Router(config-bgp-nbrgrp)# address-family ipv4 unicast
Router(config-bgp-nbrgrp-af)# exit
Router(config-nbrgrp)# exit
Router(config-bgp)# neighbor-group GROUP_2
Router(config-bgp-nbrgrp)# use session-group GROUP_3
Router(config-bgp-nbrgrp)# address-family ipv4 unicast
Routerconfig-bgp-nbrgrp-af)# use af-group GROUP_2
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。

```
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。

```
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。

```
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 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、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。

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。

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 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 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。

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。。

（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。

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

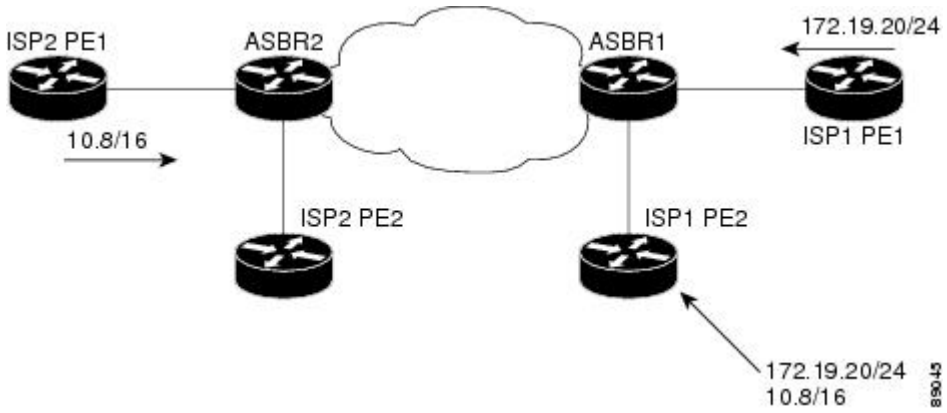
アドバタイズされる単一のパスには、次のような集約コスト コミュニティなどがあります。

{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 *5:* 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。

```
Router(config)# route-policy ISP2_PE1
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。

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 DMZ 総帯域幅

表 *4:* 機能の履歴（表）

機能名	リリース情報	機能説明
iBGP ピアへのリンク帯域幅拡張コミュニティの削除	リリース 7.3.2	緩衝地帯（DMZ）リンク帯域幅拡張コミュニティにより、BGP は複数の内部 BGP（iBGP）学習パスを介してトラフィックを送信できます。送信されるトラフィックは、自律システムから外部に出るために使用されるリンクの帯域幅に比例します。デフォルトでは、iBGP は DMZ リンク帯域幅コミュニティを伝達します。この機能により、サービス プロバイダー ネットワークでルーティングポリシーを制御するために使用されるコミュニティパラメータが、認識されていないネットワークゾーンまたは不要なネットワークゾーンに公開されるリスクを最小限に抑えられます。

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。
- 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
```

iBGP ピアへのリンク帯域幅拡張コミュニティの削除

緩衝地帯（DMZ）リンク帯域幅拡張コミュニティにより、BGP は複数の内部 BGP（iBGP）学習パスを介してトラフィックを送信できます。送信されるトラフィックは、自律システムから外部に出るために使用されるリンクの帯域幅に比例します。デフォルトでは、iBGP は DMZ リンク帯域幅コミュニティを伝達します。iBGP ピアへのリンク帯域幅拡張コミュニティの削除機能により、DMZ リンク帯域幅コミュニティを柔軟に削除して、認識されないネットワークゾーンまたは不要なネットワークゾーンにコミュニティパラメータが公開されるリスクを最小限に抑えることができます。

設定例

ユーザーがルートポリシーを設定して拡張コミュニティを削除できるようにするには、次の手順を実行します。

```
/* Delete all the extended communities. */
Router(config)# route-policy dmz_del_all
Router(config-rpl)# delete extcommunity bandwidth all
Router(config-rpl)# pass
Router(config-rpl)# end-policy

/* Delete only the extended communities that match an extended community mentioned in
the list. */
Router(config)# route-policy dmz_CE1_del_non_match
Router(config-rpl)# if destination in (10.9.9.9/32) then
Router(config-rpl-if)# delete extcommunity bandwidth in (10:7000)
Router(config-rpl-if)# endif
Router(config-rpl)# pass
Router(config-rpl)# end-policy

/* Delete all the extended communities. */
Router(config)# route-policy dmz_del_param2($a,$b)
Router(config-rpl)# if destination in (10.9.9.9/32) then
Router(config-rpl-if)# delete extcommunity bandwidth in ($a:$b)
Router(config-rpl-if)# endif
Router(config-rpl)# pass
Router(config-rpl)# end-policy
```

確認

ユーザーが特定の拡張コミュニティを削除できる設定を確認します。

```
Router# show bgp 10.9.9.9/32
Fri Aug 27 13:15:05.833 EDT
BGP routing table entry for 10.9.9.9/32
Versions:
Process bRIB/RIB SendTblVer
Speaker 15 15
Last Modified: Aug 27 13:06:45.000 for 00:08:21
Paths: (3 available, best #1)
Advertised IPv4 Unicast paths to peers (in unique update groups):
13.13.13.5
Path #1: Received by speaker 0
Advertised IPv4 Unicast paths to peers (in unique update groups):
13.13.13.5
10
10.10.10.1 from 10.10.10.1 (192.168.0.1)
Origin incomplete, metric 0, localpref 100, valid, external, best, group-best, multipath
Received Path ID 0, Local Path ID 1, version 15
Extended community: LB:10:48
Origin-AS validity: (disabled)
Path #2: Received by speaker 0
Not advertised to any peer
10
11.11.11.3 from 11.11.11.3 (192.168.0.3)
Origin incomplete, metric 0, localpref 100, valid, external, multipath
Received Path ID 0, Local Path ID 0, version 0
Extended community: LB:10:48
Origin-AS validity: (disabled)
Path #3: Received by speaker 0
Not advertised to any peer
10
12.12.12.4 from 12.12.12.4 (192.168.0.4)
Origin incomplete, metric 0, localpref 100, valid, external, multipath
Received Path ID 0, Local Path ID 0, version 0
Extended community: LB:10:48
Origin-AS validity: (disabled)

22:35 30-09-2021
```

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
  !
```

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、「パスの比較」の項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、「パスのペアの比較」の項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、「パスのペアの比較」の項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、「パスのペアの比較」の項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、「パスのペアの比較」の項で説明されている 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。

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、「デフォルトの BGP アドミニストレーティブ ディスタンス」の項のアドミニストレーティブ ディスタンスを使用します。

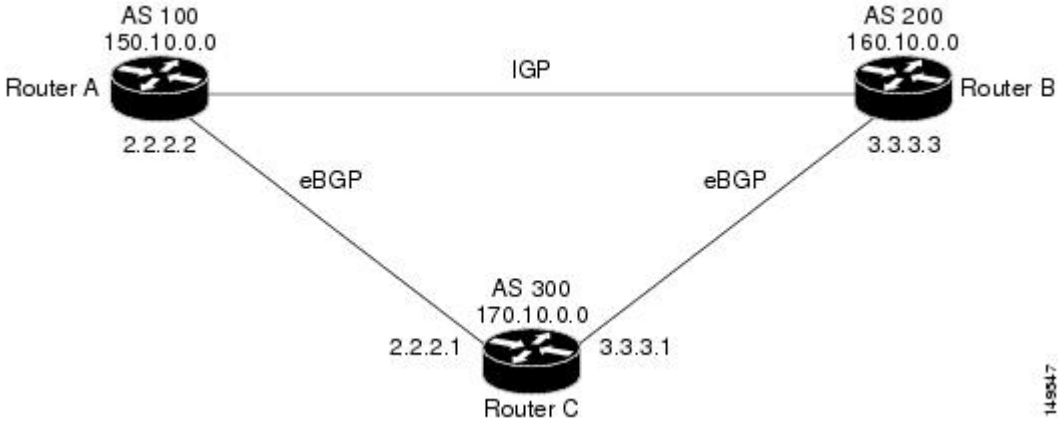
表 *5:* デフォルトの *BGP* アドミニストレーティブ ディスタンス

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 *6 :* バックドアの例



「バックドアの例」の項では、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。

```
Router(config)# router bgp 100
Router(config-bgp)# address-family ipv4 unicast
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。

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 の最適なルート リフレクタ

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

BGP ORR は次の方法でベスト パスを計算します。

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

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

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

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

BGP ORR には次のような利点があります。

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

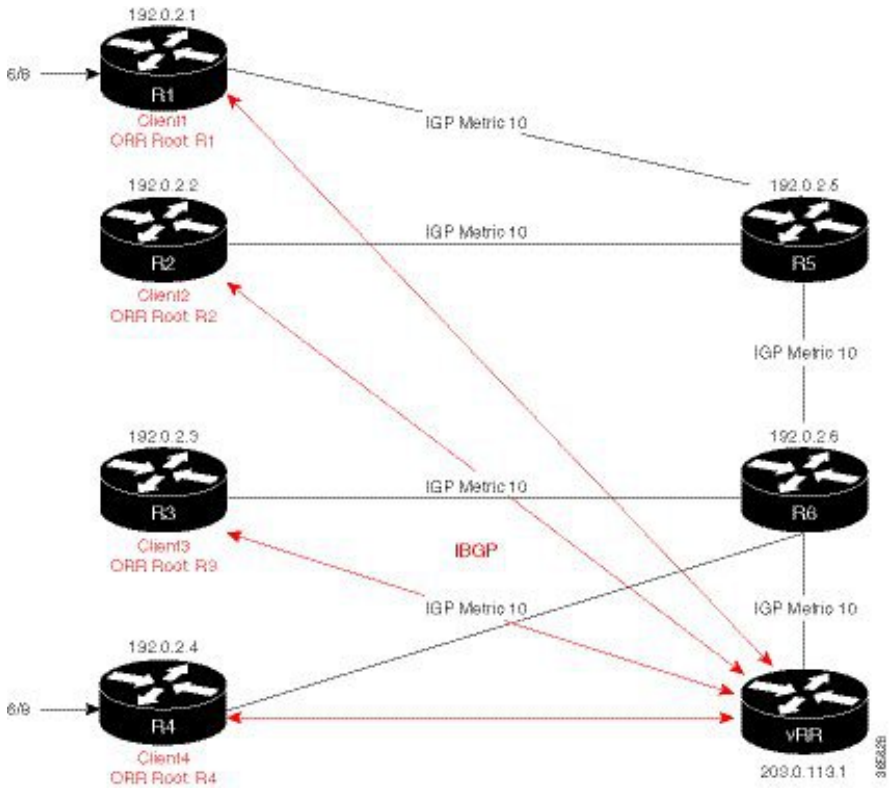
使用例

次のような、BGP ルート リフレクタのトポロジを検討します。

- ルータ R1、R2、R3、R4、R5、R6 がルート リフレクタ クライアントである。

- ルータ R1 および R4 が vRR に 6/8 プレフィックスをアドバタイズする。

図 *7: BGP-ORR* トポロジ



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

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

BGP ORR の設定には、次のことが含まれます。

- 最短出力点を決定する必要のあるクライアントの RR 上の ORR を有効にする
- ORR 設定をネイバーに適用する

R2（RR クライアント）の vRR 上の ORR を有効にする

たとえば、R2 の最短出力点を判別するには、vRR 上の ORR に R2 の IP アドレス（192.0.2.2）を設定します。次のように、AS 番号に 6500、orr（ルート）ポリシー名に g1 を使用します。

```
router bgp 6500
```

```
 address-family ipv4 unicast
   optimal-route-reflection g1 192.0.2.2
commit
```

ORR 設定をネイバーに適用する

次に、ORR ポリシーを BGP ネイバー R2 に適用します。これにより、orr（ルート）ポリシー g1 に設定されたルート IP アドレス（192.0.2.2）を使用して計算されたベストパスを、RR は R2 にアドバタイズできるようになります。

```
router bgp 6500
 neighbor 192.0.2.2
  address-family ipv4 unicast
   optimal-route-reflection g1
commit
```

ルートルータでの MPLS トラフィック エンジニアリングの設定

ルートルータは、RR で設定されたルートアドレスと一致するマルチプロトコル ラベル スイッチング（MPLS）TE ルータ ID をアドバタイズします。そのため、この MPLS TE ルータ ID をアドバタイズするには、最小限の MPLS TE 設定を使用してルートルータを設定する必要があります。設定する必要がある最小限のコマンドセットは、ルートルータのオペレーティングシステムによって異なります。

次に、ルートルータでの設定例を示します。

```
router isis 100

is-type level-2-only

net 49.0001.0000.0000.0001.00

distribute link-state

  metric-style wide

  mpls traffic-eng level-2-only

  mpls traffic-eng router-id Loopback0

!

mpls traffic-eng
```

確認

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

```
R2# show bgp 6.0.0.0/8
Tue Apr  5 20:21:58.509 UTC
BGP routing table entry for 6.0.0.0/8
Versions:
  Process           bRIB/RIB  SendTblVer
  Speaker                  8           8
```

```
Last Modified: Apr  5 20:00:44.022 for 00:21:14
Paths: (1 available, best #1)
  Not advertised to any peer
  Path #1: Received by speaker 0
  Not advertised to any peer
  Local
    192.0.2.1 (metric 20) from 203.0.113.1 (192.0.2.1)
      Origin incomplete, metric 0, localpref 100, valid, internal, best, group-best
      Received Path ID 0, Local Path ID 1, version 8
      Originator: 192.0.2.1, Cluster list: 203.0.113.1
```

上記の例では、出力の状態として、R2 のベスト パスが、IP アドレスが 192.0.2.1 でパスのメトリックが 20 である R1 経由であることが表示されています。

ORR によって計算された R2 のベスト パスを判別するには、**show bgp** コマンドを vRR から実行します。R2 には他のピアとは異なるベスト パスがある（または異なるポリシーが設定されている）ため、R2 は独自のアップデートグループを持ちます。

```
VRR#show bgp 6.0.0.0/8
Thu Apr 28 13:36:42.744 UTC
BGP routing table entry for 6.0.0.0/8
Versions:
Process bRIB/RIB SendTblVer
Speaker 13 13
Last Modified: Apr 28 13:36:26.909 for 00:00:15
Paths: (2 available, best #2)
Advertised to update-groups (with more than one peer):
0.2
Path #1: Received by speaker 0
ORR bestpath for update-groups (with more than one peer):
0.1
Local, (Received from a RR-client)
192.0.2.1 (metric 30) from 192.0.2.1 (192.0.2.1)
Origin incomplete, metric 0, localpref 100, valid, internal, add-path
Received Path ID 0, Local Path ID 2, version 13
Path #2: Received by speaker 0
Advertised to update-groups (with more than one peer):
0.2
ORR addpath for update-groups (with more than one peer):
0.1
Local, (Received from a RR-client)
192.0.2.4 (metric 20) from 192.0.2.4 (192.0.2.4)
Origin incomplete, metric 0, localpref 100, valid, internal, best, group-best
Received Path ID 0, Local Path ID 1, version 13
```

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

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

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

Update group for IPv4 Unicast, index 0.1:
Attributes:
Neighbor sessions are IPv4
```

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

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

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

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

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

Number of mapping entries: 8
```

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。

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。

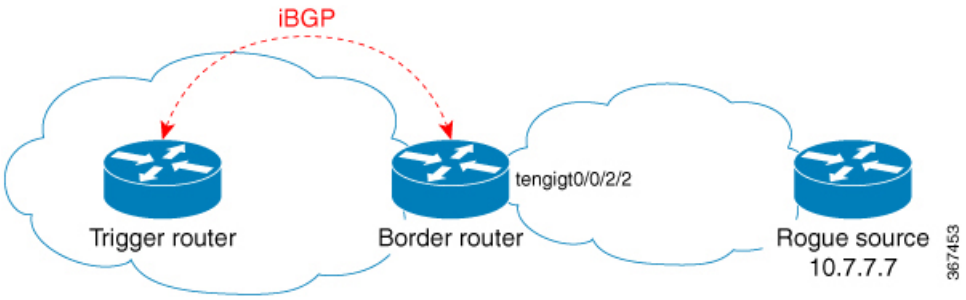
RTBH フィルタリングは、対象のプレフィックスのネクストホップをヌルインターフェイスに設定します。対象を宛先とするトラフィックは、入力時にドロップされます。

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 *8 : 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
```

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 最大セグメントサイズ

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」の項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」の項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。

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。

l2vpn_mgr プロセスが再起動されると、NSR クライアント（te-control）は、**Ready** 状態と **Not Ready** 状態を繰り返します。これは予想される動作であり、トラフィックが損失することはありません。

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 の最適外部パス

最適外部パス機能では、ローカルで選択された最適パスが内部ピアからのパスの場合における、iBGP およびルート リフレクタ ピアへの最適外部パスのアドバタイズメントをサポートしています。BGP では各宛先に対して最適パスを 1 つとバックアップ パスを 1 つ選択します。デフォルトでは、最適パスを 1 つ選択します。さらに、BGP では、1 つのプレフィックスに対する残りの外部パスのうちから別の最適パスを選択します。1 つのパスのみが最適外部パスとして選択され、バックアップ パスとして他の PE に送信されます。BGP では、最適パスが iBGP パスの場合のみ最適外部パスを計算します。最適パスが eBGP パスの場合、最適外部パス計算は不要です。

最適外部パスを決定する手順を次に示します。

1. プレフィックスに利用可能なパスの全セットから最適パスを決定します。
2. 現在の最適パスを除外します。
3. このプレフィックスのすべての内部パスを除外します。
4. 残りのパスから、現在の最適パスと同じネクスト ホップを持つすべてのパスを除外します。
5. 残りのパスのセットに対して最適パス アルゴリズムを再度実行し、最適外部パスを決定します。

BGP では、1 つのプレフィックスに対する外部およびコンフェデレーションの BGP パスを考慮して最適外部パスを計算します。BGP では、最適パスおよび最適外部パスを次のようにアドバタイズします。

- プライマリ PE 上：プレフィックスの最適パスを内部と外部の両方のピアにアドバタイズ
- バックアップ PE 上：あるプレフィックスに対して選択された最適パスを外部ピアにアドバタイズし、このプレフィックスに対して選択された最適外部パスを内部ピアにアドバタイズ

BGP プレフィックス独立コンバージェンス

BGP プレフィックス独立コンバージェンス（PIC）機能を使用すると、プライマリパスで障害が発生した場合にバックアップパスをアクティブにすることができます。

ネットワークは高速再ルーティング（FRR）を使用して次のベストパス（バックアップパス）を計算し、それを BGP および IP ルーティング情報ベース（RIB）に保存します。RIB は、バックアップパス情報を転送情報ベース（FIB）と共有します。ラインカードが PIC に対応している場合、BGP PIC 機能は、FIB のバックアップパス情報を使用して、ネットワーク障害時にこのパスに素早く切り替えます。

プレフィックス依存コンバージェンスの使用上の欠点

標準 BGP ネットワークでは、BGP ルータが宛先プレフィックスへのベストパスのみをアドバタイズします。したがって、自律システムでは、BGP を実行しているルータは宛先プレフィックスへのすべての可能なパスを認識していません。リンクまたはネットワークで障害が発生してベストパスが失敗した場合は、次のプロセスが実行されます。

1. 障害が発生したベストパスをアドバタイズしている影響を受けた BGP ルータがパスの取り消しをアナウンスします。
2. 影響を受けた BGP ルータからベストパスの取り消しを受信した BGP ルータはそれら自体のベストパスを取り消し、宛先プレフィックスへのベストパスを再計算します。
3. BGP ルータは、再計算されたベストパスをすべての隣接ルータにアドバタイズします。
4. 隣接する BGP ルータからの新しいベストパスを受信する各 BGP ルータは、それ自体のベストパスを再度評価し、場合によっては取り消してベストパスを再計算します。
5. ベストパスを再計算した BGP ルータは、ネットワーク内の新しいパスを再度アドバタイズします。

このプロセスは、すべての BGP ルータが宛先プレフィックスへのベストパスを得るまで繰り返されるため、ネットワークのコンバージェンスにはかなりの時間がかかります。この形式のコンバージェンスをプレフィックス依存コンバージェンスといいます。ルートリフレクタがネットワーク内に設定されている場合、コンバージェンスにはさらに時間がかかります。

プレフィックス非依存コンバージェンスを使用する場合の利点

BGP ネットワークでプレフィックス非依存コンバージェンスが設定されていると、すべての BGP ルータが外部のベストパスを宛先プレフィックスにアドバタイズします。これは、すべての BGP ルータが宛先プレフィックスへの複数の外部ベストパスを認識していることを示します。

各 BGP ルータは、使用可能な外部ベストパスからバックアップパスを選択し、そのパスを FIB にダウンロードします。そのため、各 BGP ルータの FIB には、ベストパスと宛先プレフィックスへの外部ベストパスが含まれています。リンクまたはネットワークに障害が発生してベストパスが失敗した場合、影響を受けた BGP ルータ上の FIB は、障害が発生したパスを使用するすべてのルートを、単一の操作で外部ベストパスに切り替えることができます。この形式の

コンバージェンスにかかる時間は最小限であるため、大規模なネットワーク展開で推奨されます。

ルートリフレクタを使用したプレフィックス非依存コンバージェンスの使用

カスタマーエッジルータからリモートプロバイダーエッジルータへのトラフィックでは、（プライマリ PE からの）プライマリパスと（バックアップ PE からの）バックアップパスを選択するために、BGP `local-pref` 属性が使用されます。リモートプロバイダーエッジルータがバックアップ PE からバックアップ（外部ベスト）パスを受信した場合でも、バックアップ PE はプライマリ PE から iBGP ベストパスを受信するとコアネットワークからのバックアップパスを取り消しします。そのため、PIC を機能させるには、プライマリおよびバックアップ（外部ベスト）のパスをネットワーク内で事前にプログラムしておく必要があります。

プライマリパスで障害が発生した場合は、行われる次のプロセスによりコンバージェンスの遅延が発生します。

1. プライマリ PE は、プライマリパスを取り出すためにプロバイダー コア ネットワークに要求を送信します。
2. バックアップ PE は、新しいプライマリ（ベスト）パスとしてバックアップ（外部ベスト）パスをアドバタイズします。
3. リモート PE は、プライマリ PE からの取り消し要求を受信した時点のプライマリパスとバックアップ PE からの新しいプライマリパスを再計算します。
4. FIB 内のすべてのプレフィックスが新しいプライマリパスで更新された後で、ネットワーク内のトラフィックが再開されます。

そのため、コンバージェンスは、PE ルータによってアドバタイズされたプレフィックスによって異なるため、低速になります。

プレフィックス非依存コンバージェンスを導入することで、次の変更が行われます。

- プライマリパスとバックアップパスは、RIB と FIB で事前にプログラミングされる。
- すべてのプロバイダーエッジルータが FIB からバックアップパスを受信する。
- プライマリパスで障害が発生した場合、FIB は LDI 変更してバックアップパスを含め、このルートに沿ってすぐにトラフィックを転送する。

（注） BGP PIC 機能をルートリフレクタとともに使用するには、プロバイダーエッジルータを VRF のコンテキスト内で一意のルート識別子（RD）を使用して設定する必要があります。それ以外の場合、異なる PE からのパスは同じネットワークに属していると見なされ、ルートリフレクタはベストバックアップパスを正確に計算できません。

バックアップパスの選択プロセス

次の手順を使用して、RIB および FIB でプログラミングされるベストバックアップパスを特定します。

1. ベストパスアルゴリズムを使用して、プレフィックスに使用可能な一連のパスからベストパスを特定します。
2. ベストパスを除外します。
3. ベストパスと同じネクストホップを持つすべてのパスを除外します。
4. 残りの一連のパスでベストパスアルゴリズムを再度実行し、ベストバックアップパスを特定します。

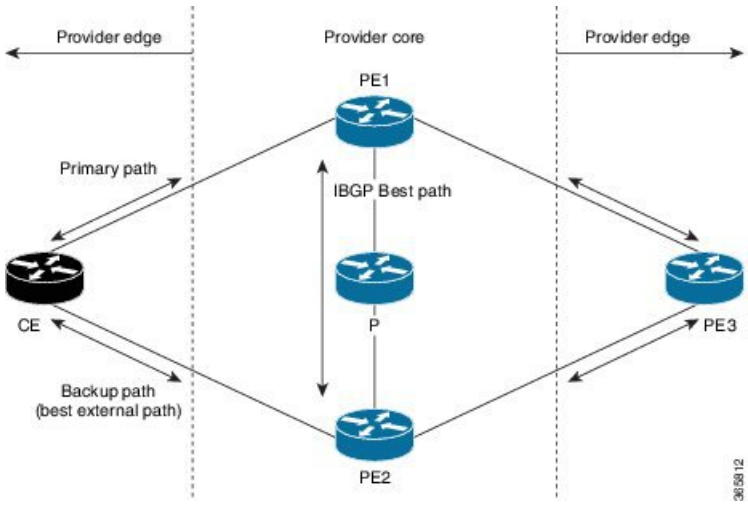
プロバイダー エッジ ネットワークでの BGP PIC の設定

この項では、プロバイダーエッジ ネットワークに BGP PIC を設定する手順について説明します。

トポロジ

次の図に示すトポロジについて考えてみましょう。

図 *9:* プロバイダーエッジ ネットワークでのプレフィックス独立コンバージェンス



カスタマーエッジルータ CE からプロバイダーエッジルータ PE3 へのトラフィックの場合、BGP `local-pref` 属性を使用して CE-PE1-PE3 をプライマリパス、CE-PE2-PE3 をバックアップパスとして選択します。PE1-P-PE2 は、プロバイダー コア ネットワーク内部のベストパスです。

はじめる前に

BGP PIC 機能を設定する前に、次のように設定されていることを確認します。

1. トポロジにしたがってループバック インターフェイスとネットワーク インターフェイスが設定されている。
2. プロバイダー コア ネットワーク用に VRF が設定されている。

コンフィギュレーション

図に示すトポロジの BGP PIC 機能を設定するには、この項での設定を使用します。

ルータ PE1

ルータ CE からルータ PE3 へのトラフィックの場合、ルータ CE からの eBGP パスはルータ PE1 上のプライマリパスとして保存されます。

次に示すように、ルータ PE2 がアドバタイズしたバックアップパス（外部ベスト）パスをインストールするようにルータ PE1 を設定し、ローカルラベルがコンバージェンス時に保持する必要がある期間を設定します。

```
Router(config)# router bgp 10
Router(config-bgp)# vrf foo
Router(config-bgp-vrf)# address-family ipv4 unicast
Router(config-bgp-vrf-af)# additional-path install
Router(config-bgp-vrf-af)# label-retention 10
```

ルータ PE2

バックアップ CE-PE2 パスを外部ベストパスとしてインストールし、アドバタイズするようにルータ PE2 を設定します。

```
Router(config)# router bgp 10
Router(config-bgp)# vrf foo
Router(config-bgp-vrf)# address-family ipv4 unicast
Router(config-bgp-vrf-af)# advertise-best-external label-alloc-mode
Router(config-bgp-vrf-af)# additional-path install
```

ルータ PE3

ルータ PE1 からの iBGP パス（CE-PE1）はルータ PE3 のプライマリパスとして保存されます。次に示すように、iBGP バックアップパスの CE-PE2 を設定します。

```
Router(config)# router bgp 10
Router(config-bgp)# vrf foo
Router(config-bgp-vrf)# address-family ipv4 unicast
Router(config-bgp-vrf-af)# additional-path install
```

BGP PIC の確認

ルータ PE3 で次のコマンドを実行して、BGP PIC 機能が動作していることを確認します。

1. FIB にバックアップパスが存在することを確認します。

```
Router# show cef 1.1.1.1/32 detail
Fri Oct 10 10:24:33.079 UTC
1.1.1.1/32, version 1, internal 0x40000001 (0xa94c0574) [1], 0x0 (0x0), 0x0
(0x0)
Updated Oct 9 16:49:06.795
Prefix Len 32, traffic index 0, precedence routine (0)
gateway array (0xa8d9b130) reference count 4, flags 0x80200, source rib
(3),
```

```
[1 type 3 flags 0x901101 (0xa8ec6b90) ext 0x0 (0x0)]
LW-LDI[type=0, refc=0, ptr=0x0, sh-ldi=0x0]
Level 1 - Load distribution: 0
[0] via 12.24.0.1, recursive
via 12.24.0.1, 3 dependencies, recursive
next hop 12.24.0.1 via 12.24.0.1/32
via 12.24.0.2, 3 dependencies, recursive, backup
next hop 12.24.0.2 via 12.24.0.2/32
Load distribution: 0 (refcount 1)
Hash OK Interface Address
0 Y MgmtEth0/RP0/CPU0/0 12.24.0.1
```

2. BGP のバックアップ（外部ベスト）パスが存在することを確認します。

```
Router# show bgp vrf foo 206.1.1.1/32
BGP routing table entry for 206.1.1.1/32
Versions:
Process bRIB/RIB SendTblVer
Speaker 6 6
Local Label: 3
Paths: (1 available, best #1)
Advertised to peers (in unique update groups):
100.100.100.1
Path #1: Received by speaker 0
1.1.1.1 from 1.1.1.1 (200.200.200.1)
Origin incomplete, metric 0, localpref 100, weight 32768, valid,
internal, best
2.2.2.2 from 2.2.2.2 (100.100.100.1)
Origin incomplete, metric 0, localpref 100, weight 32768, valid,
external, backup, best-external
```

自律システム間での BGP PIC の設定

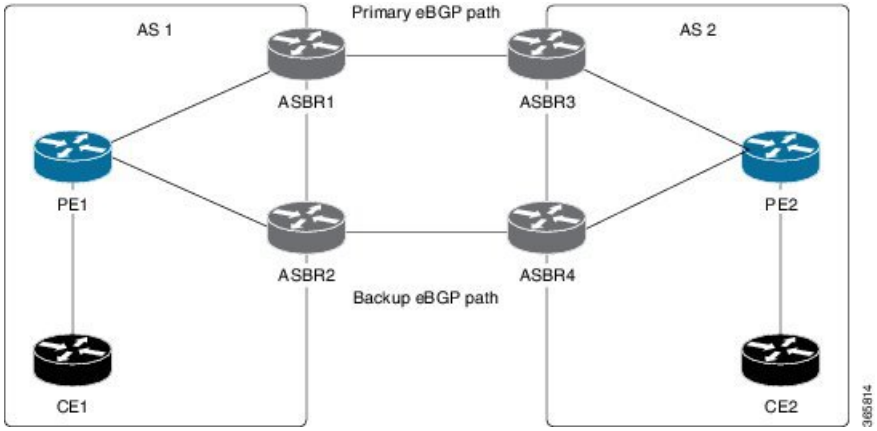
この項では、自律システム間に BGP PIC を設定する手順について説明します。。

（注） BGP PIC は、オプション A とオプション B のシナリオでのみサポートされています。次の項では、オプション B の設定例について説明します。

トポロジ

たとえば、次の図に示すトポロジを考えてみましょう。

図 *10 :* 自律システム間のプレフィックス非依存コンバージェンス



ルータ PE1 からルータ PE2 へのトラフィックの場合、ASBR1 がプライマリルータで、ASBR2 がバックアップルータです。ASBR1-ASBR3 eBGP パスはプライマリパスです。ASBR2-ASBR4 eBGP パスはバックアップパスです。ルータ PE2 からルータ PE1 へのトラフィックの場合、ASBR3 がプライマリルータで、ASBR4 がバックアップルータです。ASBR3-ASBR1 eBGP パスはプライマリパスで、ASBR4-ASBR2 eBGP パスはバックアップパスです。

はじめる前に

BGP PIC 機能を設定する前に、図のトポロジに従ってループバックとネットワークのインターフェイスが設定されていることを確認します。

コンフィギュレーション

図に示すトポロジの BGP PIC 機能を設定するには、この項での設定を使用します。

ルータ ASBR1

次に示すように、ルータ ASBR2 がアドバタイズしたバックアップパス（外部ベスト）パスをインストールするようにルータ ASBR1 を設定し、ローカルラベルがコンバージェンス時に保持する必要がある期間を設定します。

```
Router(config)# router bgp 10
Router(config-bgp)# address-family vpnv4 unicast
Router(config-bgp-af)# additional-path install
Router(config-bgp-af)# label-retention 10
```

指定された設定は、ルータ PE1 からルータ PE2 へのトラフィック用です。同様に、ルータ PE2 からルータ PE1 へのトラフィック用に ルータ ASBR3 を設定します。

ルータ ASBR2

次に示すように、ASBR2-ASBR4 バックアップ（外部ベスト）パスをインストールしてアドバタイズするようにルータ ASBR2 を設定します。

```
Router(config)# router bgp 10
Router(config-bgp)# address-family vpnv4 unicast
Router(config-bgp-af)# advertise-best-external label-alloc-mode
```

```
Router(config-bgp-af)# additional-path install
```

指定された設定は、ルータ PE1 からルータ PE2 へのトラフィック用です。同様に、ルータ PE2 からルータ PE1 へのトラフィック用に ルータ ASBR4 を設定します。

BGP PIC の確認

ルータ PE2（ルータ PE1 からルータ PE2 へのトラフィック用）またはルータ PE1（ルータ PE2 からルータ PE1 へのトラフィック用）で次のコマンドを実行して、BGP PIC 機能が動作していることを確認します。

1. FIB にバックアップパスが存在することを確認します。

```
Router# show cef 1.1.1.1/32 detail

Fri Oct 10 10:24:33.079 UTC
1.1.1.1/32, version 1, internal 0x40000001 (0xa94c0574) [1], 0x0 (0x0), 0x0
(0x0)
Updated Oct 9 16:49:06.795
Prefix Len 32, traffic index 0, precedence routine (0)
gateway array (0xa8d9b130) reference count 4, flags 0x80200, source rib
(3),
[1 type 3 flags 0x901101 (0xa8ec6b90) ext 0x0 (0x0)]
LW-LDI[type=0, refc=0, ptr=0x0, sh-ldi=0x0]
Level 1 - Load distribution: 0
[0] via 12.24.0.1, recursive
via 12.24.0.1, 3 dependencies, recursive
next hop 12.24.0.1 via 12.24.0.1/32
via 12.24.0.2, 3 dependencies, recursive, backup
next hop 12.24.0.2 via 12.24.0.2/32
Load distribution: 0 (refcount 1)
Hash OK Interface Address
0 Y MgmtEth0/RP0/CPU0/0 12.24.0.1
```

2. BGP のバックアップ（外部ベスト）パスが存在することを確認します。

```
Router# show bgp vrf foo 206.1.1.1/32

BGP routing table entry for 206.1.1.1/32
Versions:
Process bRIB/RIB SendTblVer
Speaker 6 6
Local Label: 3
Paths: (1 available, best #1)
Advertised to peers (in unique update groups):
100.100.100.1
Path #1: Received by speaker 0
1.1.1.1 from 1.1.1.1 (200.200.200.1)
Origin incomplete, metric 0, localpref 100, weight 32768, valid,
internal, best
2.2.2.2 from 2.2.2.2 (100.100.100.1)
Origin incomplete, metric 0, localpref 100, weight 32768, valid,
external, backup, best-external
```

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

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

表 *6:* 機能の履歴（表）

機能名	リリース情報	機能説明
ネイバーごとの追加パス制御	リリース 7.3.15	この機能により、ネイバー アウトバウンド ポリシー設定に基づいて、追加パスのアドバタイズメントを柔軟かつきめ細かく制御できます。 これは、単一のパス選択とは異なり、さまざまなパス選択手順の組み合わせの設定を可能にし、アドバタイズされるパスタイプをより細かく制御できるようにネイバー アウトバウンド ポリシーを拡張することで行われます。 この機能により、追加のパスを管理するための運用効率が向上し、一般的なクラスタ化されたネットワークアーキテクチャでのパスのスケールを縮小できます。 この機能がないと、メモリのパススケール制限が影響を受け、過剰な数のパスが原因でコントロールプレーンのコンバージェンスの問題が発生します。

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。

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。

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。

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** または **end** コマンドを使用します。

手順の詳細

ステップ 1 **configure**

例：

```
RP/0/RP0/cpu 0: router# configure
```

モードを開始します。

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。

ステップ 3 **address-family** {**ipv4**|**ipv6**} {**unicast**|**multicast**}

例：

```
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：

```
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** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

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
```

BGP の累積 IGP 属性

表 *7:* 機能の履歴（表）

機能名	リリース情報	機能説明
BGP の累積 IGP 属性	リリース 7.3.2	この機能により、1 回の管理で複数の連続した BGP 自律システムを導入できます。 IGP と同様に、BGP が IGP メトリックに基づいてルーティングを決定できるようにすることが可能です。

BGP AIGP の概要

BGP の累積 IGP（AIGP）属性は、オプションの非推移的な BGP パス属性です。IANA が AIGP 属性に属性タイプコードを割り当てました。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 または LDP は、ローカルエリアでのみプレフィックスまたはラベル情報を伝送します。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ラベルを伝送します。その後、ルートまたはラベルは、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。

AIGP によるプレフィックスの生成

累積内部ゲートウェイ プロトコル（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有効化されている。
- ルートが、ボーダー ゲートウェイ プロトコル（BGP）に再配布される内部ゲートウェイ プロトコル（IGP）ルートである。AIGP 属性に割り当てられた値が、ルートの iGP ネクストホップの値であるか、ルートポリシーによって設定された値である。
- 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。

設定例

AIGP によりプレフィックスを生成します。

```
Router(config)# route-policy aip_policy
Router(config-rpl)# set aigp-metric igp-cost
Router(config-rpl)# exit
Router(config)# router bgp 100
Router(config-bgp)# address-family ipv4 unicast
Router(config-bgp-af)# redistribute ospf route-policy aip_policy
```

実行コンフィギュレーション

```
route-policy aip_policy
 set aigp-metric igp-cost
!
router bgp 100
 address-family ipv4 unicast
  redistribute ospf route-policy aip_policy
```

確認

AIGP 属性のステータスを確認します。

```
Router# show bgp 10.0.0.1
Thu Sep 30 21:21:15.279 EDT
BGP routing table entry for 10.0.0.1/32
Versions:
Process bRIB/RIB SendTblVer
Speaker 4694 4694
Last Modified: Sep 30 21:20:09.000 for 00:01:06
Paths: (2 available, best #1)
Not advertised to any peer
Path #1: Received by speaker 0
Not advertised to any peer
Local
192.168.0.1 (metric 2) from 192.168.0.1 (192.168.0.6)
Received Label 24000
Origin IGP, localpref 80, aigp metric 900, valid, internal, best, group-best,
labeled-unicast
Received Path ID 1, Local Path ID 1, version 4694
Originator: 192.168.0.6, Cluster list: 192.168.0.1
Total AIGP metric 902 <-- AIGP attribute received.
```

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。

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。

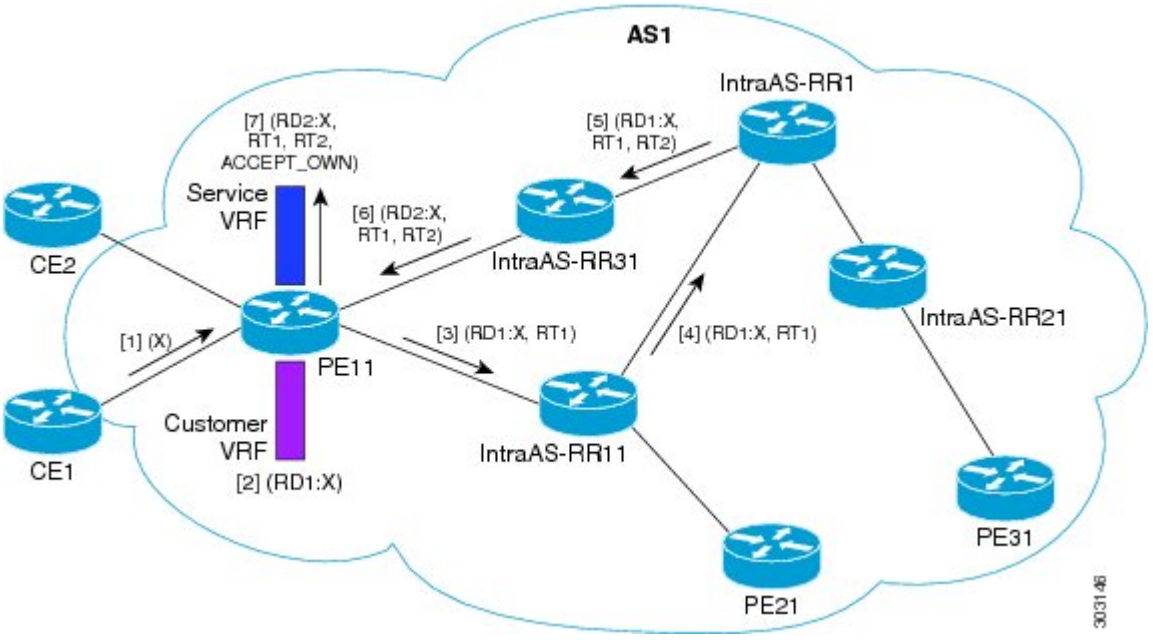
BGP Accept Own の応用例の 1 つは、MPLS VPN ネットワーク内のエクストラネットの自動設定です。エクストラネットの設定では、ある VRF にあるルートは同じ PE の別の VRF にインポートされます。通常、エクストラネットのメカニズムでは、別の VRF からのプレフィックスのインポートを制御するために、エクストラネット VRF のインポート RT またはインポートポリシーを編集する必要があります。ただし、Accept Own 機能を使用すると、ルート リフレクタは、PE で設定変更することなく、その制御をアサートできます。このように Accept Own 機能によって、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

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** 例： `RP/0/RP0/cpu 0: router# configure`	モードを開始します。
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。
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

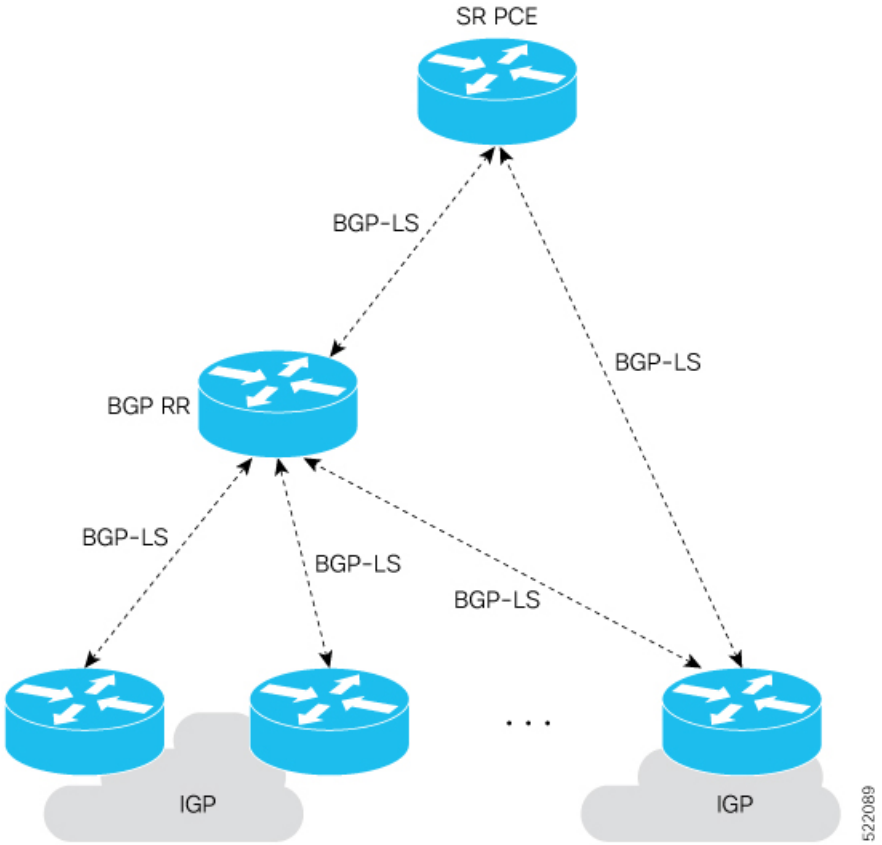
BGP リンクステート（LS）は、BGP を介して内部ゲートウェイプロトコル（IGP）リンクステート情報を伝送するために元々定義されたアドレスファミリ識別子（AFI）およびサブアドレスファミリ識別子（SAFI）です。BGP-LS の BGP ネットワーク層到達可能性情報（NLRI）のエンコーディング形式と、BGP-LS 属性と呼ばれる新しい BGP パス属性は、RFC7752 で定義されています。各リンクステートオブジェクト（具体的には、ノード、リンク、またはプレ

フィックス）の識別キーは NLRI でエンコードされ、オブジェクトのプロパティは BGP-LS 属性でエンコードされます。

（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-LS のアプリケーションの例は、セグメントルーティングパス計算要素（SR-PCE）です。SR-PCE は、トポロジ内のノードの SR 機能と、それらのノードへの SR セグメントのマッピングを学習できます。これにより、SR-PCE は SR-TE に基づいてパス計算を実行し、基盤となる IGP ベースの分散ベストパス計算とは異なるパスでトラフィックを誘導できます。

次の図に、一般的な展開のシナリオを示します。各 IGP エリアで、1 つ以上のノード（BGP スピーカー）が BGP-LS を使用して設定されています。これらの BGP スピーカーは、1 つ以上のルートリフレクタに接続することによって iBGP メッシュを形成します。このようにして、すべての BGP スピーカー（特にルートリフレクタ）は、すべての IGP エリア（および eBGP ピアのその他の AS）からリンクステート情報を取得します。



BGP ネイバーとのリンクステート情報の交換

次の例は、リンクステート情報を BGP ネイバーと交換する方法を示しています。

```
Router# configure
Router(config)# router bgp 1
Router(config-bgp)# neighbor 10.0.0.2
Router(config-bgp-nbr)# remote-as 1
Router(config-bgp-nbr)# address-family link-state link-state
Router(config-bgp-nbr-af)# exit
```

IGP リンクステートデータベース配信

特定の BGP ノードは、複数の独立したルーティングドメインに接続できます。BGP-LS への IGP リンクステートデータベース配信は、OSPF と IS-IS の両方のプロトコルでサポートされていて、これらの複数のドメインにまたがるパスまたはこれらの複数のドメインを含むパスを構築するコントローラまたはアプリケーションに、この情報を配信することができます。

BGP-LS を使用して OSPFv2 リンクステートデータを配布するには、ルータ コンフィギュレーション モードで **distribute link-state** コマンドを使用します。

```
Router# configure
Router(config)# router ospf 100
Router(config-ospf)# distribute link-state instance-id 32
```

使用上のガイドラインと制限事項

- BGP-LS は IS-IS および OSPFv2 サポートしています。
- BGP-LS の（Instance-ID と呼ばれる）識別子フィールドは、NLRI が属する IGP ルーティングドメインを識別します。同じ IGP ルーティングインスタンスからのリンクステートオブジェクト（ノード、リンク、またはプレフィックス）を表す NLRI は、同じ Instance-ID 値を使用する必要があります。
- BGP-LS が動作しているネットワークにプロトコルインスタンスが 1 つしかない場合は、Instance-ID 値を **0** に設定することを推奨します。
- 特定の IGP ドメイン内のすべての BGP-LS プロデューサに一貫した BGP-LS の Instance-ID 値を割り当てます。
- 異なる Instance-ID 値を持つ NLRI は、異なる IGP ルーティングインスタンスからのものと見なされます。
- 異なる IGP ドメインで動作するルーティング プロトコル インスタンスには、一意の Instance-ID 値を割り当てる必要があります。これにより、ネットワーク内の複数の BGP-LS プロデューサによって BGP-LS 経由でトポロジがアドバタイズされる場合でも、BGP-LS コンシューマ（SR-PCE など）は Instance-ID 値に基づいて正確な分離されたマルチドメイントポロジを構築できます。
- BGP-LS の Instance-ID の設定ガイドラインに従わない場合、複数の BGP-LS プロデューサが展開されていると、BGP-LS コンシューマで同じノード、リンク、またはプレフィックスの重複するリンクステートオブジェクトが確認される可能性があります。これにより、

BGP-LS コンシューマが不正確なネットワーク全体のトポロジを取得する可能性もあります。

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。

ステップ 1 **configure**

例：

```
RP/0/RP0/cpu 0: router# configure
```

モードを開始します。

ステップ 2 **router bgp** *as-number*

例：

```
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：

```
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：

```
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：

```
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** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

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。

ステップ 1 **configure**

例：

```
RP/0/RP0/cpu 0: router# configure
```

モードを開始します。

ステップ 2 **router bgp** *as-number*

例：

```
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：

```
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：

```
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** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

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。

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。

ステップ 1 **configure**

例：

```
RP/0/RP0/cpu 0: router# configure
```

モードを開始します。

ステップ 2 **prefix-set** *prefix-set-name*

例：

```
Router(config)# prefix-set PERMANENT-NETWORK-IPv4
Router(config-pfx)# 1.1.1.1/32,
Router(config-pfx)# 2.2.2.2/32,
Router(config-pfx)# 3.3.3.3/32
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：

```
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：

```
Router(config)# route-policy POLICY-PERMANENT-NETWORK-IPv4
Router(config-rpl)# if destination in PERMANENT-NETWORK-IPv4 then
Router(config-rpl)# pass
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：

```
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：

```
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：

```
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。

ステップ 8 **permanent-network route-policy** *route-policy-name*

例：

```
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** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

ステップ 10 **show bgp {ipv4 | ipv6} unicast** *prefix-set*

例：

```
show 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

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。

ステップ 1 **configure**

例：

```
RP/0/RP0/cpu 0: router# configure
```

モードを開始します。

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：

```
Router(config-bgp)# neighbor 10.255.255.254
```

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 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：

```
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：

```
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** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

ステップ 8 **show bgp {ipv4 | ipv6} unicast neighbor** *ip-address*

例：

```
Router# show 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。

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 ルートアドバタイズメントの遅延

表 *8:* 機能の履歴（表）

機能名	リリース情報	機能説明

BGP ルートアドバタイズメントの遅延	リリース 7.5.3	早すぎるBGPルートのアドバタイズメントによるトラフィック損失と、ネットワークでの後続のパケット損失を防ぐことができるようになりました。これを実現するには、ルーティング情報ベース（RIB）がルーティングテーブル内の転送情報ベース（FIB）と同期されるまで、ルータでのBGPの起動の遅延時間を設定します。これにより、BGPアップデートの生成が遅延し、ネットワークでのトラフィック損失が防止されます。 遅延は最小で 1 秒、最大で 600 秒に設定できます。 この機能では、**update wait-install delay startup** コマンドが導入されています。

BGP がトラフィックを転送する場合、RIB がトラフィックを転送する準備ができるまで RIB からのフィードバックを待機します。RIB の準備が整うと、BGP はルート更新を BGP ネイバーおよびピアグループに送信します。RIB が FIB で同期される前にルートをアドバタイズすると、トラフィック損失が発生します。この問題を回避するには、トラフィック損失が発生しないように、ルータが BGP の起動プロセスを遅延させて BGP アップデートの生成を遅延させる必要があります。

これを行うには、**update wait-install delay startup** コマンドを設定して、BGP アップデートの生成を遅延させる必要があります。**show bgp process** コマンドは、最後のルータのリロード以降の BGP プロセス更新の遅延を表示します。

この機能を使用すると、最小および最大の遅延期間を設定できます。遅延の範囲は 1 ～ 600 秒です。その結果、ネットワークトラフィックの損失が回避されます。

制約事項

この機能は、次のアドレス ファミリ インジケータ（AFI）に適用されます。

- IPv4 ユニキャスト
- IPv6 ユニキャスト
- VPNv4 ユニキャスト
- VPNv6 ユニキャスト

設定

1. IOS XR コンフィギュレーション モードを開始します。

```
Router# configure
```

2. BGP 自律システム番号（AS 番号）を指定します。

```
Router(config)# router bgp 1
```

3. アドレスファミリ（Pv4、IPv6、VPNv4、または VPNv6）オプションから IP アドレスを指定します。

```
Router(config-bgp)# address-family {ipv4| ipv6| vpnv4| vpn6} unicast
```

次の例を参考にしてください。

```
Router(config-bgp)# address-family ipv4 unicast
```

4. BGP プロセスの遅延をスケジュールして、RIB が同期されるまでルートがピアにアドバタイズされないようにします。

```
Router(config-bgp-af)# update wait-install delay startup (time in seconds)
```

次の例を参考にしてください。

```
Router(config-bgp-af)# update wait-install delay startup 10
```

5. 変更を確定します。

```
Router(config-bgp-af)#commit
```

（注） 遅延時間の範囲は 1 ～ 600 秒です。

実行コンフィギュレーション

```
configure
router bgp 1
 address-family ipv4 unicast
  update wait-install delay startup 10
!
```

検証の例

次のコマンドは、BGP プロセス更新の遅延を表示します。

```
Router# show running-config router bgp 1
router bgp 1
address-family ipv4 unicast
update wait-install delay startup 10
```

次の作業

次に、**show bgp process** コマンドを実行できます。**show bgp process** コマンドの **Update wait-install enabled** セクションには、最後のルータのリロード以降の BGP プロセス更新の遅延が表示されます。

```
Router# show bgp process
Wed Aug 24 00:40:48.649 PDT

BGP Process Information:
BGP is operating in STANDALONE mode
Autonomous System number format: ASPLAIN
Autonomous System: 100
Router ID: 192.168.0.2 (manually configured)
Default Cluster ID: 192.168.0.2
Active Cluster IDs:  192.168.0.2
------------------------------
```

```
------------------------------
Update wait-install enabled:
  ack request 2, ack rcvd 2, slow ack 0
  startup delay 10 secs

--More--
```

VRF での default-originate

BGP は、VRF ごとの設定に基づいて、プロバイダーエッジのネイバーにデフォルトルートをアドバタイズします。

ユーザー定義の Martian アドレスチェック

Cisco 8000 シリーズ ルータで BGP を設定すると、ルータが特定の IP アドレスプレフィックスを持つ特定のサイトにアクセスするのを防ぐことができます。これらのルータは、このような IP アドレスからのパケットをドロップします。このような IP アドレスは、Martian アドレスと呼ばれます。ただし、コマンド **default-martian-check disable** を設定することで、BGP IPv4 アドレスファミリまたは BGP IPv6 アドレスファミリの設定が使用されているルータがこれらのサイトにアクセスできるようにすることが可能です。これらのサイトは、次のような特定の IPv4 および IPv6 プレフィックスを持つサイトです。

- 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

機能制限

OSPF または IS-IS プロトコルを使用するルータは、**default-martian-check disable** コマンドが設定されていても、これらのサイトにアクセスできません。

設定例

Martian アドレスからのルートを許可するには、次の手順を実行します。

1. BGP IPv4 または BGP IPv6 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。
2. アドレスファミリ修飾子をユニキャストアドレスとして設定します。
3. Martian アドレスチェックを無効にします。

設定

```
/* Enter BGP IPv4 or BGP IPv6 address-family configuration mode. */
Router# configure
Router(config)# router bgp 100

/* Configure the address-family modifier as unicast. */
Router(config-bgp)# address-family ipv4 unicast

/* Disable the martian address check. */
Router(config-bgp-af)# default-martian-check disable
Router(config-bgp-af)# commit
```

確認

Martian アドレスチェックを有効にしたか無効にしたかを確認するには、**show bgp ipv4 unicast** コマンドまたは **show bgp ipv6 unicast** コマンドを使用します。

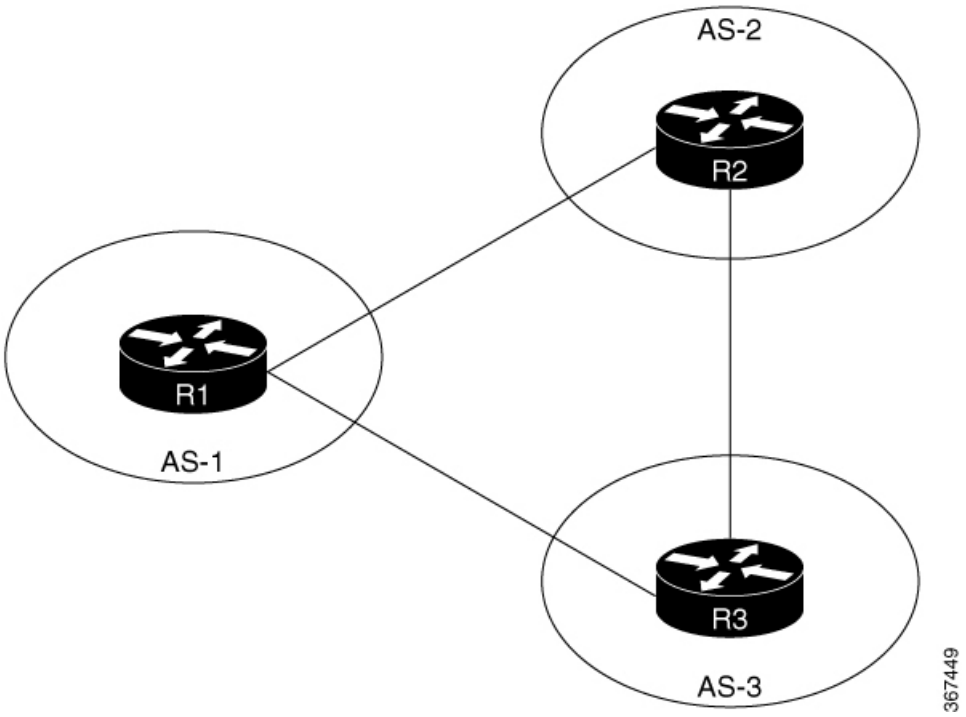
```
Router# show bgp ipv6 unicast
BGP router identifier 2.2.2.1, local AS number 1
BGP generic scan interval 60 secs
Non-stop routing is enabled
BGP table state: Active
Table ID: 0xe0800000 RD version: 29
BGP main routing table version 29
BGP NSR Initial initsync version 4 (Reached)
BGP NSR/ISSU Sync-Group versions 0/0
Dampening enabled
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
*>i::/0              1:1:1:1:1:1:1:1         100        0            i
* i192:1::/112       1.1.1.1                   0      100            0 ?
*>i                  1:1:1:1:1:1:1:1           0      100            0 ?
* iff11:1123::/64    1.1.1.1                   2      100            0 ?
*>i                  1:1:1:1:1:1:1:1           2      100            0 ?
```

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 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）をモニターできるようになります。BMP サーバー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ーできます。また、1 つ以上の BMP サーバーに接続するように 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 サーバー（プライマリ サーバーとして設定）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ー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 サーバーが存在する場合があります。設定された BMP サーバー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 サーバー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 サーバー間の「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 サーバー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 サーバーの設定は、その特定の BMP サーバーによってモニター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 サーバー間のセッションは、プレーン TCP（暗号化/カプセル化なし）で動作します。BMP サーバーとの 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 サーバーは、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 サーバーへ）のみです。

ルータでは、最大 8 台の BMP サーバーを設定できます。各 BMP サーバーはサーバー 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 サーバーを正常に設定すると、BGP スピーカーは BMP サーバー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、ユーザーは複数（独立かつ非同期）の BMP サーバー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 サーバーとの直接 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 サーバーに送信されます。

BMP サーバー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 サーバー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 サーバーへの TCP 接続が試行されます。サーバー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。すでにモニターされている 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 サーバーに送信します。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 サーバー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。モニターするネイバが複

数ある場合、それらが有効になっている BMP サーバーに基づいて、各ネイバにはリフレッシュ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 サーバー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 サーバーに送信されます。

（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 サーバーが起動すると、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 サーバー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 サーバー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 サーバー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サーバー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ーするように、**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ーの設定では、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、サーバー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 サーバー上でさまざまな遅延タイマーを設定できます。

BGP：複数のクラスタ ID

BGP—複数のクラスタ ID機能により、iBGP ネイバー（通常はルート リフレクタ）は複数のクラスタ ID（グローバル クラスタ ID と、クライアント（ネイバー）に割り当てられる追加クラスタ ID）を持つことができます。この機能が導入される前は、デバイスは単一のグローバル クラスタ ID を持つことができました。

ネットワーク管理者がネイバーごとのクラスタ ID を設定すると、次のようになります。

- CLUSTER_LIST に基づくループ防止メカニズムが、複数のクラスタ ID を考慮するように自動的に変更されます。
- クラスタ ID に基づいてクライアント間のルート リフレクションを無効にすることができます。

制約事項

BGP 複数のクラスタ ID 機能は、デフォルトの VRF でのみ動作します。

BGP フロースペックの概

表 *9:* 機能の履歴（表）

機能名	リリース情報	機能説明
6,000 個のルールへの BGP フロースペックの拡張	リリース 7.5.2	Cisco 8800 シリーズ ルータには 6,000 個の BGP フロースペックルールを、および Cisco 8100 および 8200 シリーズ ルータには 3,000 個の BGP フロースペックルールを割り当てることができるようになりました。この機能により、分散型サービス妨害（DDoS）攻撃に対する強化された緩和策が提供されます。 以前のリリースでは、2,000 個の BGP フロースペックルールを割り当てることができました。これらは表面的な拡張可能数です。この数は、AccessList（ACL）、Quality of Service（QoS）、Local Path Transport Switching（LPTS）などの、他の交差する機能に応じて異なります。

BGP 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。

BGP フロースペック機能を使用すると、BGP アップデートを介して特定のフローを、IPv4 と IPv6 の送信元、IPv4 と IPv6 の宛先、L4 パラメータ、パケットの詳細（長さ、フラグメント、宛先ポート、送信元ポートなど）、実行する必要があるアクション（トラフィックのドロップ、一定レートでのポリシング、トラフィックのリダイレクトなど）と照合する命令を作成できます。BGP アップデートでは、フロースペックの一致基準はネットワーク層到達可能性情報（BGP NLRI）によって表され、アクションは BGP 拡張コミュニティによって表されます。

BGP フロースペック機能を使用して、DDoS 攻撃を軽減できます。DDoS 攻撃がネットワーク内の特定のホストで発生した場合、境界ルータにフロースペックアップデートを送信して、攻撃トラフィックをポリシングまたはドロップしたり、他の場所にリダイレクトしたりできます。たとえば、有害なトラフィックをフィルタ処理してトラフィックをクリーンにし、影響を受けるホストに正常なトラフィックのみを転送するアプライアンスに送信します。

フロースペックがルータで受信され、該当するラインカードでプログラムされると、それらのラインカードのアクティブな L3 ポートは、フロースペックルールに従って入力トラフィックの処理を開始します。

BGP フロースペック機能は、特定のインターフェイス上で MAP-E および PBR と共存することはできません。PBR を使用して BGP フロースペックを設定した場合、ルータはエラーまたはシステムメッセージを表示しません。ルータは BGP フロースペック設定を無視し、この機能は機能しません。

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 ハードウェアの機能によって異なる場合があります。「サポートされている一致基準とアクション」および「トラフィック フィルタリング アクション」の項で、サポートされている一致（タプル定義）とアクションパラメータに関する情報を提供しています。

（注）
- Cisco 8800 シリーズ ルータでは、最大 6,000 個のフロースペックルールがサポートされています。
- Cisco 8200 および 8100 シリーズ ルータでは、最大 3,000 個のフロースペックルールがサポートされています。

サポートされている一致基準とアクション

表 *10 :* 機能の履歴（表）

機能名	リリース名	説明
強化されたセキュリティのための追加の BGP フロースペックアクション	リリース 7.3.3	このリリースでは、分散型サービス妨害（DDoS）攻撃に対するセキュリティを強化するために、追加の BGP フロースペックアクションが導入されています。 • ネクストホップ VRF のみをリダイレクト：トラフィックを別の自律システム番号（ASN）にリダイレクトします。 • レート制限と IPv4 または IPv6 ネクストホップのリダイレクト：指定されたネクストホップ IPv4 または IPv6 アドレスにトラフィックをリダイレクトします。ポリサーレートによりトラフィックが規制されます。 • レート制限とネクストホップ VRF のリダイレクト：VRF を介してネクストホップ IPv4 アドレスにトラフィックをリダイレクトします。ポリサーレートによりトラフィックが規制されます。このアクションは、Q200 Silicon One ASIC でのみサポートされます。

表 *11* : 機能の履歴（表）

機能名	リリース名	説明
BGP フロースペック NLRI タイプ	リリース 7.3.15	BGP フロースペックは、IP トラフィックに適用される NLRI にエンコードされた複数の一致基準で構成されます。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必要があります。ネットワーク層到達可能性情報（NLRI）は、ルーティング情報と一致基準を BGP ピア間で交換し、宛先に到達する方法を示します。 次の NLRI タイプがサポートされています。 • タイプ 7：IPv4 または IPv6 の ICMP タイプ • タイプ 8：IPv4 または IPv6 の ICMP コード • タイプ 9：IPv4 TCP フラグ（2バイトに予約ビットを含む） • タイプ 10：IPv4 パケット長 • タイプ 11：IPv4 または IPv6 の DSCP • タイプ 12：IPv4 のフラグメンテーションビット

機能名	リリース名	説明
BGP フロースペックアクション	リリース 7.3.15	この機能は、BGP 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提供します。トラフィック フィルタリング フロースペックが、指定されたルールに基づいて適用されます。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。 • DSCP の設定 • IPv4 または IPv6 ネクストホップのリダイレクト

制約事項

ネクストホップ VRF のリダイレクトが使用されている場合、BGP フロースペックの統計はサポートされません。

ポリサーレート制限が使用されている場合は、BGP フロースペックの統計がサポートされます。

ポリサーが関連付けられている場合にのみ、リダイレクトアクションで BGP フロースペックの統計がサポートされます。リダイレクトアクション単独では、BGP フロースペックの統計はサポートされません。

L3VPN VRF はサポートされていません。ただし、プレーン VRF はサポートされています。

概要

フロースペック 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、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 スペック NLRI タイプは、オプションである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。

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**： **match protocol** {*protocol-value* | *[min-value – max-value]*}	単一の値 （注） 複数値の範囲はサポートされていません

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 または 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、ポート、送信元ポート、および宛先ポートコンポーネントは FALSE と評価されます。パケットがフラグメント化されていて、これが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。 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。 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。 **構文**： エンコーディング：<type (1 octet), [op, value]+> **match**{**ipv4** | **ipv6**}**icmp-type** {value | min-value -max-value}	単一の値 （注） 複数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 のパケット長 （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 • IPv4 または IPv6 のサポートは、最初のフラグメントパケットではないパケットに使用できます。	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。そのため、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。 （注） DSCP にはフロースペック統計は含まれません。 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 のフラグメンテーションビット （注） IPv4 のサポートは、最初のフラグメントパケットではないパケットに使用できます。 IPv6 BGP フロースペックは、タイプ 12 の NRLI をサポートしていません。	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、2 タプルアクションの組み合わせを制限なしで指定できます。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。

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。

（注） BGP フロースペックアクションのレート制限とリダイレクトは、同時にはサポートされません。

BGP フロースペックアクションのリダイレクトは、ネクストホップ IPv4 および IPv6 でのみサポートされ、ネクストホップ VRF IPv4 および IPv6 ではサポートされません。

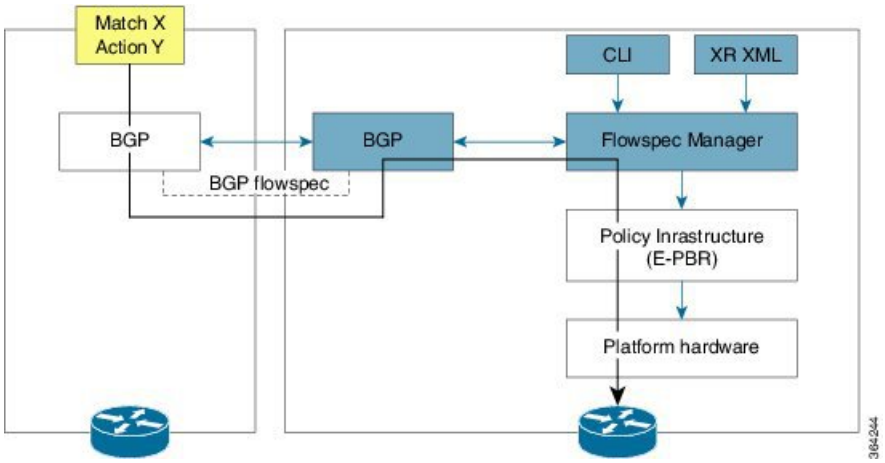
タイプ	Extended Community	PBR アクション	説明
0x8006	traffic-rate 0 traffic-rate <rate>	ドロップ ポリシング	トラフィックレート拡張コミュニティは、自律システム境界を越えた非過渡的な拡張コミュニティであり、次の拡張コミュニティのエンコーディングを使用します。 最初の 2 つのオクテットで 2 オクテットの ID を伝送します。この ID は 2 バイトの AS 番号から割り当てることができます。4 バイトの AS 番号がローカルに存在する場合は、このような AS 番号の最下位 2 バイトを使用できます。この値は情報です。残りの 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）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
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>

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 フィールドで記述された IP パケットを照合し、関連付けられた MP_REACH_NLRI の Network Address of Next-Hop フィールドで指定された IPv4 または IPv6 アドレスにリダイレクトまたはコピーします。 （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。 （注） リダイレクト IP NH は、デフォルトの VRF でのみサポートされています。 **コマンド構文** redirect {ipv4 | ipv6} next-hop {ipv4-address | ipv6-address}

BGP フロースペッククライアント/サーバーコントローラモデル

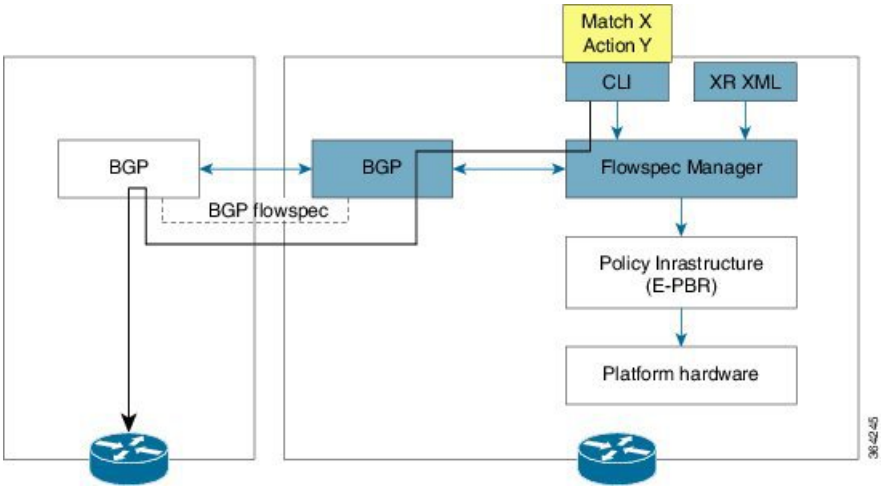
BGP フロースペックモデルは、クライアントとサーバーコントローラで構成されます。コントローラは、フロースペック NRLI エントリの送信または挿入を実行します。クライアント（BGP スピーカーとして機能）はその NRLI を受信し、コントローラからの指示に従って動作するようにハードウェア転送をプログラムします。このモデルの図を下に示します。

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

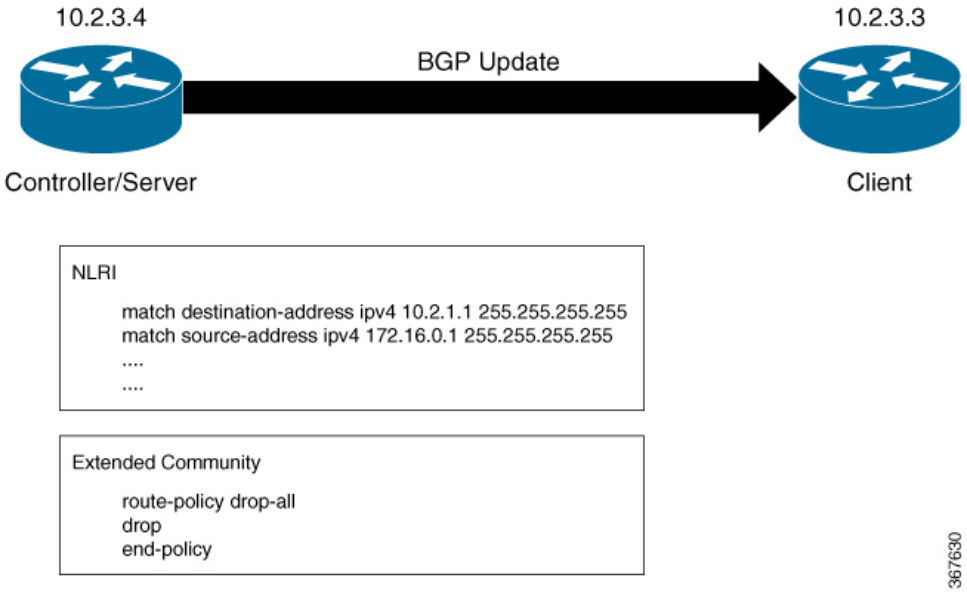


コントローラは、CLI を使用して NRLI インジェクション用のエントリを提供するように設定されます。

BGP フロースペックの設定

以下の項では、BGP フロースペック機能の設定方法について説明します。

図 *12 : BGP* フロースペック



IP アドレス 10.2.3.4 のコントローラまたはサーバーが、IP アドレス 10.2.3.3 のクライアントにフロースペック NLRI を送信します。NLRI は一致基準で構成され、クライアントはこの基準に基づいて処理します。トラフィックは、設定された基準に基づいてドロップされるか受け入れられます。

次の項では、クライアントで BGP フロースペックを設定する方法について説明します。

```
/*Configure BGP Flowspec */
Router(config)# flowspec
Router(config-flowspec)# address-family ipv4
Router(config-flowspec-af)# local-install interface-all
Router(config-flowspec-af)# exit
Router(config-flowspec)# address-family ipv6
Router(config-flowspec-af)# local-install interface-all
Router(config-flowspec-af)# exit

/* Configure the policy to accept all presented routes without modifying the routes */
Router(config)# route-policy pass-all
Router(config)# pass
Router(config)# end-policy

/* Configure the policy to reject all presented routes without modifying the routes */
Router(config)# route-policy drop-all
Router(config)# drop
Router(config)# end-policy

/* Configure BGP towards flowspec server */
Router(config)# router bgp 1
Router(config-bgp)# nsr
Router(config-bgp)# bgp router-id 10.2.3.3
Router(config-bgp)# address-family ipv4 flowspec
Router(config-bgp-af)# exit
Router(config-bgp)# address-family ipv6 flowspec
Router(config-bgp-af)# exit
Router(config-bgp)# neighbor 10.2.3.4
Router(config-bgp-nbr)# remote-as 1
Router(config-bgp-nbr)# address-family ipv4 flowspec
Router(config-bgp-nbr-af)# route-policy pass-all in
Router(config-bgp-nbr-af)# route-policy drop-all out
Router(config-bgp-af)# exit
Router(config-bgp-nbr)# address-family ipv6 flowspec
Router(config-bgp-nbr-af)# route-policy pass-all in
Router(config-bgp-nbr-af)# route-policy drop-all out
Router(config-bgp-nbr-af)# exit
Router(config-bgp-nbr)# update-source Loopback0

/* Disable BGP Flowspec */
Router(config)# interface bundle-ether 3.1
Router(config-subif)# ipv4 flowspec disable
Router(config-subif)# ipv6 flowspec disable

The following section describes how you can configure BGP Flowspec on the server:
/* Configure the policy to accept all presented routes without modifying the routes */
Router(config)# route-policy pass-all
Router(config)# pass
Router(config)# end-policy

/* Configure the policy to reject all presented routes without modifying the routes */
Router(config)# route-policy drop-all
Router(config)# drop
```

```
Router(config)# end-policy

/* Configure BGP towards flowspec client */
Router(config)# router bgp 1
Router(config-bgp)# nsr
Router(config-bgp)# bgp router-id 10.2.3.4
Router(config-bgp)# address-family ipv4 flowspec
Router(config-bgp-af)# exit
Router(config-bgp)# address-family ipv6 flowspec
Router(config-bgp-af)# exit
Router(config-bgp)# neighbor 10.2.3.3
Router(config-bgp-nbr)# remote-as 1
Router(config-bgp-nbr)# address-family ipv4 flowspec
Router(config-bgp-nbr-af)# route-policy pass-all in
Router(config-bgp-nbr-af)# route-policy pass-all out
Router(config-bgp-nbr-af)# exit
Router(config-bgp-nbr)# update-source Loopback0

/* Configure IPv4 flowspec to be advertised to client. Define traffic classes. */
Router(config)# class-map type traffic match-all ipv4_fragment
Router(config-cmap)# match destination-address ipv4 10.2.1.1 255.255.255.255
Router(config-cmap)# match source-address ipv4 172.16.0.1 255.255.255.255

Router(config-cmap)# end-class-map
Router(config)# class-map type traffic match-all ipv4_icmp
Router(config-cmap)# match destination-address ipv4 10.2.1.1 255.255.255.255
Router(config-cmap)# match source-address ipv4 172.16.0.1 255.255.255.255
Router(config-cmap)# end-class-map

/* Define a policy map and associate it with traffic classes.
Router(config)# policy-map type pbr scale_ipv4
Router(config-pmap)# class type traffic ipv4_fragment
Router(config-pmap-c)# drop
Router(config-pmap-c)# exit
Router(config-pmap)# class type traffic ipv4_icmp
Router(config-pmap-c)# exit
Router(config-pmap)# class type traffic class-default
Router(config-pmap-c)# end-policy-map
Router(config)# flowspec
Router(config)# address-family ipv4
Router(config-af)# service-policy type pbr scale_ipv4

/* Configure IPv6 flowspec to be advertised to client. Define traffic classes. */
Router(config)# class-map type traffic match-all ipv6_tcp
Router(config-cmap)# match destination-address ipv6 70:1:1::5a/128
Router(config-cmap)# match source-address ipv4 ipv6 80:1:1::5a/128
Router(config-cmap)# match destination-port 22
Router(config-cmap)# match source-port 4000
Router(config-cmap)# end-class-map
Router(config)# class-map type traffic match-all ipv6_icmp
Router(config-cmap)# match destination-address ipv6 70:2:1::1/128
Router(config-cmap)# match source-address ipv4 ipv6 80:2:1::1/128
Router(config-cmap)# end-class-map

/* Define a policy map and associate it with traffic classes.
Router(config)# policy-map type pbr scale_ipv6
Router(config-pmap)# class type traffic ipv6_tcp
Router(config-pmap-c)# exit
Router(config-pmap)# class type traffic ipv6_icmp
Router(config-pmap-c)# exit
Router(config-pmap)# class type traffic class-default
Router(config-pmap-c)# end-policy-map
```

```
Router(config)# flowspec
Router(config)# address-family ipv6
Router(config-af)# service-policy type pbr scale_ipv6

/* Class map configuration with DSCP */
Router(config-map)# class-map type traffic match-all class_dscp_5
Router(config-cmap)# match destination-address ipv4 192.0.2.254 255.255.255.0
Router(config-cmap)# match dscp 10-12

/* Policy map configuration with IPv4 Redirect and Rate Limiter */
Router(config-pmap)#class type traffic class_dscp_5
Router(config-pmap-c)#redirect ipv4 nexthop 10.26.245.2
Router(config-pmap-c)#police rate 5 mbps
Router(config-pmap-c)# root
```

実行コンフィギュレーション

```
/* Client-side configuration */

flowspec
address-family ipv4
local-install interface-all
!
address-family ipv6
local-install interface-all
!
!
route-policy pass-all
pass
end-policy
!
route-policy drop-all
drop
end-policy
!
router bgp 1
nsr
bgp router-id 10.2.3.3
address-family ipv4 flowspec
!
address-family ipv6 flowspec
!
neighbor 10.2.3.4
remote-as 1
address-family ipv4 flowspec
route-policy pass-all in
route-policy drop-all out
!
address-family ipv6 flowspec
route-policy pass-all in
route-policy drop-all out
!
update-source Loopback0
!
!
vrf vrf1
address-family ipv4 unicast
import route-target
4787:13
!
export route-target
4787:13
!
!
```

```
address-family ipv6 unicast
import route-target
4787:13
!
export route-target
4787:13
!
!
!
router static
vrf vrf1
address-family ipv4 unicast
10.0.0.0/8 200.255.55.2
!
!
!
/* Disable the flowspec. This is optional configuration */
interface Bundle-Ether3.1
ipv4 flowspec disable
ipv6 flowspec disable
!
/* Server-side Configuration */
route-policy pass-all
pass
end-policy
!
route-policy drop-all
drop
end-policy
!
router bgp 1
nsr
bgp router-id 10.2.3.4
address-family ipv4 flowspec
!
address-family ipv6 flowspec
!
neighbor 10.2.3.3
remote-as 1
address-family ipv4 flowspec
route-policy drop-all in
route-policy pass-all out
exit
update-source Loopback0
!
!
class-map type traffic match-all ipv4_fragment
match destination-address ipv4 10.2.1.1 255.255.255.255
end-class-map
!
class-map type traffic match-all ipv4_icmp
match destination-address ipv4 10.2.1.1 255.255.255.255
match source-address ipv4 172.16.0.1 255.255.255.255
end-class-map
!
policy-map type pbr scale_ipv4
class type traffic ipv4_fragment
drop
!
class type traffic ipv4_icmp
!
!
class type traffic class-default
!
```

```
end-policy-map
!
flowspec
address-family ipv4
service-policy type pbr scale_ipv4
!
!
class-map type traffic match-all ipv6_tcp
match destination-address ipv6 70:1:1::5a/128
match source-address ipv6 80:1:1::5a/128
match protocol tcp
match destination-port 22
match source-port 4000
end-class-map
!
class-map type traffic match-all ipv6_icmp
match destination-address ipv6 70:2:1::1/128
match source-address ipv6 80:2:1::1/128
end-class-map
!
policy-map type pbr scale_ipv6
class type traffic ipv6_tcp
!
!
class type traffic ipv6_icmp
!
!
class type traffic class-default
!
!
flowspec
address-family ipv6
service-policy type pbr scale_ipv6
!
!
```

確認

次の show の出力は、クライアント側からのフロースペックのステータスを示しています。

```
Router# show bgp ipv4 flowspec
GP router identifier 202.158.0.1, local AS number 4787
BGP generic scan interval 60 secs
Non-stop routing is enabled
BGP table state: Active
Table ID: 0x0 RD version: 7506
BGP main routing table version 7506
BGP NSR Initial initsync version 130 (Reached)
BGP NSR/ISSU Sync-Group versions 7506/0
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
*>iDest:10.1.1.1/32,Proto:=6,DPort:=80,SPort:=3000,Length:=200,DSCP:=10/176
0.0.0.0 10 0 ?
*>iDest:10.1.1.2/32,Proto:=6,DPort:=80,SPort:=3000,Length:=200,DSCP:=10/176
0.0.0.0 10 0 ?
*>iDest:10.1.1.3/32,Proto:=6,DPort:=80,SPort:=3000,Length:=200,DSCP:=10/176
0.0.0.0 10 0 ?
*>iDest:10.1.1.4/32,Proto:=6,DPort:=80,SPort:=3000,Length:=200,DSCP:=10/176
0.0.0.0 10 0 ?
*>iDest:10.1.1.5/32,Proto:=6,DPort:=80,SPort:=3000,Length:=200,DSCP:=10/176
0.0.0.0 10 0 ?
```

```
Router# show bgp ipv6 flowspec

BGP router identifier 202.158.0.1, local AS number 4787
BGP generic scan interval 60 secs
Non-stop routing is enabled
BGP table state: Active
Table ID: 0x0 RD version: 1503
BGP main routing table version 1504
BGP NSR Initial initsync version 2 (Reached)
BGP NSR/ISSU Sync-Group versions 1504/0
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
*>iDest:70:1:1::1/0-128,Source:80:1:1::1/0-128,NH:=6,DPort:=22,SPort:=4000,TCPFlags:=0x10,Length:=300,DSCP:=12/464
202:158:2::1 100 0 i
*>iDest:70:1:1::2/0-128,Source:80:1:1::2/0-128,NH:=6,DPort:=22,SPort:=4000,TCPFlags:=0x10,Length:=300,DSCP:=12/464
202:158:2::1 100 0 i
*>iDest:70:1:1::3/0-128,Source:80:1:1::3/0-128,NH:=6,DPort:=22,SPort:=4000,TCPFlags:=0x10,Length:=300,DSCP:=12/464
202:158:2::1 100 0 i
*>iDest:70:1:1::4/0-128,Source:80:1:1::4/0-128,NH:=6,DPort:=22,SPort:=4000,TCPFlags:=0x10,Length:=300,DSCP:=12/464
202:158:2::1 100 0 i
*>iDest:70:1:1::5/0-128,Source:80:1:1::5/0-128,NH:=6,DPort:=22,SPort:=4000,TCPFlags:=0x10,Length:=300,DSCP:=12/464
202:158:2::1 100 0 i

Router# show bgp vpnv4 flowspec
BGP router identifier 202.158.0.1, local AS number 4787
BGP generic scan interval 60 secs
Non-stop routing is enabled
BGP table state: Active
Table ID: 0x0 RD version: 0
BGP main routing table version 5
BGP NSR Initial initsync version 3 (Reached)
BGP NSR/ISSU Sync-Group versions 5/0
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
Route Distinguisher: 202.158.0.1:0 (default for vrf customer_1)
*>iDest:202.158.3.2/32,Source:202.158.1.2/32/96
0.0.0.0 100 0 i
Route Distinguisher: 202.158.0.2:1
*>iDest:202.158.3.2/32,Source:202.158.1.2/32/96
0.0.0.0 100 0 i
Processed 2 prefixes, 2 paths

Router# show bgp vpnv6 flowspec
BGP router identifier 202.158.0.1, local AS number 4787
BGP generic scan interval 60 secs
Non-stop routing is enabled
BGP table state: Active
Table ID: 0x0 RD version: 0
BGP main routing table version 5
BGP NSR Initial initsync version 4 (Reached)
BGP NSR/ISSU Sync-Group versions 5/0
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
Route Distinguisher: 202.158.0.1:0 (default for vrf customer_1)
```

```
*>iDest:200:158:3::2/0-128,Source:200:158:1::2/0-128,NH:=6,DPort:=22,SPort:=4000,Length:=300,DSCP:=12/440
0.0.0.0 100 0 i
Route Distinguisher: 202.158.0.2:1
*>iDest:200:158:3::2/0-128,Source:200:158:1::2/0-128,NH:=6,DPort:=22,SPort:=4000,Length:=300,DSCP:=12/440
0.0.0.0 100 0 i
Processed 2 prefixes, 2 paths

Router# show bgp ipv6 flowspec summary
BGP router identifier 202.158.0.1, local AS number 4787
BGP generic scan interval 60 secs
Non-stop routing is enabled
BGP table state: Active
Table ID: 0x0 RD version: 1503
BGP main routing table version 1504
BGP NSR Initial initsync version 2 (Reached)
BGP NSR/ISSU Sync-Group versions 1504/0
BGP scan interval 60 secs
BGP is operating in STANDALONE mode.
Process RcvTblVer bRIB/RIB LabelVer ImportVer SendTblVer StandbyVer
Speaker 1504 1504 1504 1504 1504 1504
Neighbor Spk AS MsgRcvd MsgSent TblVer InQ OutQ Up/Down St/PfxRcd
200.255.1.5 0 4787 6957 2957 1504 0 0 04:48:02 0
200.255.1.6 0 50011 3015 3010 0 0 0 05:27:50 (NoNeg)
202.158.2.1 0 4787 1548 1648 1504 0 0 1d01h 750 <-- this
many flowspecs were received from server
202.158.3.1 0 4787 1683 1644 1504 0 0 1d01h 751
202.158.4.1 0 4787 1543 1649 1504 0 0 1d01h 0

Router# show bgp vpnv4 flowspec summary
BGP router identifier 202.158.0.1, local AS number 4787
BGP generic scan interval 60 secs
Non-stop routing is enabled
BGP table state: Active
Table ID: 0x0 RD version: 0
BGP main routing table version 5
BGP NSR Initial initsync version 3 (Reached)
BGP NSR/ISSU Sync-Group versions 5/0
BGP scan interval 60 secs
BGP is operating in STANDALONE mode.
Process RcvTblVer bRIB/RIB LabelVer ImportVer SendTblVer StandbyVer
Speaker 5 5 5 5 5 5
Neighbor Spk AS MsgRcvd MsgSent TblVer InQ OutQ Up/Down St/PfxRcd
202.158.2.1 0 4787 1549 1648 5 0 0 1d01h 1 <-- this
many flowspecs were received from server
202.158.3.1 0 4787 1684 1644 5 0 0 1d01h 0
202.158.4.1 0 4787 1543 1649 5 0 0 1d01h 0

Router# show bgp vpnv6 flowspec summary
BGP router identifier 202.158.0.1, local AS number 4787
BGP generic scan interval 60 secs
Non-stop routing is enabled
BGP table state: Active
Table ID: 0x0 RD version: 0
BGP main routing table version 5
BGP NSR Initial initsync version 4 (Reached)
BGP NSR/ISSU Sync-Group versions 5/0
BGP scan interval 60 secs
BGP is operating in STANDALONE mode.
Process RcvTblVer bRIB/RIB LabelVer ImportVer SendTblVer StandbyVer
Speaker 5 5 5 5 5 5
Neighbor Spk AS MsgRcvd MsgSent TblVer InQ OutQ Up/Down St/PfxRcd
202.158.2.1 0 4787 1549 1649 5 0 0 1d01h 1 <-- this
many flowspecs were received from server
202.158.3.1 0 4787 1684 1645 5 0 0 1d01h 0
```

```
202.158.4.1 0 4787 1543 1650 5 0 0 1d01h 0

Router# show flowspec ipv4 detail
AFI: IPv4
Flow :Dest:10.1.1.1/32,Proto:=6,DPort:=80,SPort:=3000,Length:=200,DSCP:=10
Actions :Traffic-rate: 0 bps (bgp.1)
Statistics (packets/bytes)
Matched : 18174999/3707699796
Transmitted : 0/0
Dropped : 18174999/3707699796

Router# show flowspec ipv6 detail
AFI: IPv6
Flow
:Dest:70:1:1::1/0-128,Source:80:1:1::1/0-128,NH:=6,DPort:=22,SPort:=4000,TCPFlags:=0x10,Length:=300,DSCP:=12
Actions :Traffic-rate: 1000000 bps DSCP: cs1 Nexthop: 202:158:2::1 (bgp.1)
Statistics (packets/bytes)
Matched : 64091597/19483845488
Transmitted : 33973978/10328089312
Dropped : 30117619/9155756176

Router# show flowspec vrf customer_1 ipv4 detail
VRF: customer_1 AFI: IPv4
Flow :Dest:202.158.3.2/32,Source:202.158.1.2/32
Actions :Traffic-rate: 250000000 bps DSCP: cs6 Redirect: VRF dirty_dancing
Route-target: ASN2-4787:666 (bgp.1)
Statistics (packets/bytes)
Matched : 37260786850/4098686553500
Transmitted : 21304093027/2343450232970
Dropped : 15956693823/1755236320530

Router# show flowspec vrf customer_1 ipv6 detail
VRF: customer_1 AFI: IPv6
Flow
:Dest:200:158:3::2/0-128,Source:200:158:1::2/0-128,NH:=6,DPort:=22,SPort:=4000,Length:=300,DSCP:=12
Actions :Traffic-rate: 250000000 bps DSCP: cs6 Redirect: VRF dirty_dancing
Route-target: ASN2-4787:666 (bgp.1)
Statistics (packets/bytes)
Matched : 16130480136/4903665961344
Transmitted : 8490755776/2581189755904
Dropped : 7639724360/2322476205440

Router# show flowspec ipv4 nlri
AFI: IPv4
NLRI (hex) :0x0120460101010381060581500691 0bb80a81c80b810a
Actions :Traffic-rate: 0 bps (bgp.1)

Router# show flowspec ipv6 nlri
AFI: IPv6
NLRI (hex)
:0x0180000070000100010000000000000000000102800000800001000100000000000000000010381060581160691 0fa00981100a91012c0b810c
Actions :Traffic-rate: 1000000 bps DSCP: cs1 Nexthop: 202:158:2::1 (bgp.1)

Router# show flowspec vrf customer_1 ipv4 nlri
VRF: customer_1 AFI: IPv4
NLRI (hex) :0x0120ca9e03020220ca9e0102
Actions :Traffic-rate: 250000000 bps DSCP: cs6 Redirect: VRF dirty_dancing
Route-target: ASN2-4787:666 (bgp.1)

Router# show flowspec vrf customer_1 ipv6 nlri
VRF: customer_1 AFI: IPv6
NLRI (hex)
:0x0180000200015800030000000000000000000202800002000158000100000000000000000020381060581160691 0fa00a91012c0b810c
Actions :Traffic-rate: 250000000 bps DSCP: cs6 Redirect: VRF dirty_dancing
```

```
Route-target: ASN2-4787:666 (bgp.1)

Router# show policy-map transient type pbr
policy-map type pbr __bgpfs_default_IPv4
handle:0x36000004
table description: L3 IPv4 and IPv6
class handle:0x760013eb sequence 1024
match destination-address ipv4 10.1.1.1 255.255.255.255
match protocol tcp
match destination-port 80
match source-port 3000



Router# show flowspec ipv4 detail
  Flow           :Dest:192.0.2.254/24,DSCP:>=10&<=12
    Actions      :Traffic-rate: 5000000 bps Nexthop: 10.26.245.2  (bgp.1)
    Statistics                          (packets/bytes)
      Matched             :             1169087/233817400
      Transmitted         :              369952/73990400
      Dropped             :              799135/159827000
```

BGP 拡張ルート保持

表 *12*：機能の履歴（表）

機能名	リリース名	説明
BGP 拡張ルート保持	リリース 7.3.3	この機能を使用すると、障害が発生した BGP ピアからの古いルーティング情報を、グレースフルリスタート属性で設定されている期間よりも長く維持できます。ただし、この機能により、BGP ネイバーは古いルートを新しいルートと見なします。

BGP ピアに障害が発生すると、拡張ルート保持機能によってルート保持ポリシーがルートに適用され、ルート属性が変更されます。この機能は、ネイバーのインバウンドポリシーが原因で発生する変更に加えて、ルート属性を変更します。この機能により、BGP ホールドタイマーが期限切れになった場合、または設定されたグレースフル リスタート タイマー内で BGP セッションが受信スピーカーとして再確立できなかった場合に、LLGR の代わりにルート保持ポリシーを使用できます。

LLGR を適用する場合、古いルートがアドバタイズされたときに LLGR_STALE コミュニティを削除することはできず、ルートはそれを最も優先度の低いものとして扱います。また、次の条件下では、LLGR 機能をアドバタイズしていないネイバーに古いルートをアドバタイズできます。

- ネイバーが内部（IBGP またはコンフェデレーション）ネイバーである必要があります。

- NO_EXPORT コミュニティが古いルートにアタッチされている必要があります。
- 古いルートで、そのLOCAL_PREFコミュニティがゼロに設定されている必要があります。

この機能により、古いルートを eBGP ネイバーに柔軟にアドバタイズし、iBGP システム内に保持されている古いルートのローカルプリファレンス値を指定することができます。

機能制限

- ネイバーはグレースフルリスタートに対応している必要があります。
- BGP ネイバーに障害が発生すると、グレースフル リスタート タイマーが有効な間はグレースフルリスタート機能が適用されます。
- グレースフル リスタート タイマーが期限切れになると、拡張ルート保持機能が開始されます。
- ソフト再設定のインバウンド設定は必須の設定です。必要に応じて、インバウンドポリシーを設定します。
- 拡張ルート保持機能は、BGP ピアがダウンした場合、つまりホールドダウンタイマーの期限が切れた場合にのみ開始されます。
- 他のトリガー（タイマーの期限切れなど）の場合、ルートは古いと示されず、ルートが消去されます。
- 拡張ルート保持機能は、次のアドレスファミリモードにのみ適用されます。
  - IPv4 および IPv6 ユニキャスト アドレス ファミリ モード
  - IPv4 および IPv4 ラベル付きユニキャスト アドレス ファミリ モード
- 同じネイバーに LLGR 機能と拡張ルート保持機能の両方を設定することはできません。
- 拡張ルート保持機能を設定すると、機能属性は送信されません。

設定例

CLUSTER_LIST 属性の使用方法

CLUSTER_LIST 伝播ルールは、リリースによって異なり、デバイスが実行しているシスコ ソフトウェア リリースが BGP—複数のクラスタ ID 機能の実装前に生成されたか実装後に生成されたかによって決まります。CLUSTER_LIST に基づくループ防止についても同様です。

CLUSTER_LIST の動作については、以下で説明します。Classic は、複数クラスタ ID 機能の実装前にリリースされたソフトウェアの動作を指します。MCID は、複数クラスタ ID 機能の実装後にリリースされたソフトウェアの動作を指します。

CLUSTER_LIST 伝播ルール

- Classic：ルートを反映する前に、RR はグローバル クラスタ ID を CLUSTER_LIST に追加します。受信したルートに CLUSTER_LIST 属性がない場合、RR はそのグローバル クラスタ ID を持つ新しい CLUSTER_LIST 属性を作成します。
- MCID：ルートを反映する前に、RR はルートの送信元であるネイバーのクラスタ ID を CLUSTER_LIST に追加します。受信したルートに CLUSTER_LIST 属性がない場合、RR はそのクラスタ ID を持つ新しい CLUSTER_LIST 属性を作成します。この動作には、スピーカーのクライアントではないネイバーが含まれます。ルートの送信元である非クライアント ネイバーにクラスタ ID が関連付けられていない場合、RR はグローバル クラスタ ID を使用します。

CLUSTER_LIST に基づくループ防止

- Classic：ルートを受信すると、RR は、RR のグローバル クラスタ ID がルートの CLUSTER_LIST に含まれている場合にはそのルートを破棄します。
- MCID：ルートを受信すると、RR は、RR のグローバル クラスタ ID またはいずれかの iBGP ネイバーに割り当てられているクラスタ ID がルートの CLUSTER_LIST に含まれている場合にはそのルートを破棄します。

ネイバー単位のクラスタ ID の設定

ネイバーごとにクラスタ ID を設定するには、iBGP ピア（通常はルートリフレクタ）でこのタスクを実行します。ネイバーごとにクラスタ ID を設定すると、CLUSTER_LIST に基づくループ防止メカニズムが複数のクラスタ ID を考慮するように自動的に変更されます。また、クラスタ ID に基づいてクライアント間のルート リフレクションを無効化できるようになります。別のコマンドを使用してルート リフレクションを無効にすることができるように、ネイバーにタグが付けられます

（注） ネイバーのクラスタ ID を変更すると、BGP により、すべての iBGP ピアについてインバウンド ソフト リフレッシュとアウトバウンド ソフト リフレッシュが自動的に実行されます。

```
Router> enable
Router # configure terminal
Router(config)# router bgp 65000
Router(config-router)# neighbor 192.168.1.2
Router(config-router)# remote-as 65000
Router(config-router)# cluster-id 0.0.0.1
Router(config-router)# end
```

実行コンフィギュレーション

```
!
 !
 router bgp 65000
  neighbor 192.168.1.2
   remote-as 65000
    cluster-id 0.0.0.1
```

確認

次に、グローバルでも、ネイバー単位でも、クラスタ ID が何らかのレベルで設定されている場合に、ネイバーの状態に関係なく、その ID がアクティブなクラスタ ID に追加される例を示します。BGP は、この機能のネイバー状態を追跡しません。

```
Router# show bgp process detail

BGP Process Information:
BGP is operating in STANDALONE mode
Autonomous System number format: ASPLAIN
Autonomous System: 65000
Router ID: 10.10.1.92 (manually configured)
Default Cluster ID: 10.10.1.92
Active Cluster IDs:  10.10.1.92, 10.10.3.93, 10.10.4.20
                     10.10.5.20, 198.51.100.254
...

Router# show configuration commit change last 1

Building configuration...
!! IOS XR Configuration 6.1.3
router bgp 65000
neighbor 198.51.100.254                        <<< not operational, no AFs etc
  remote-as 65000
  cluster-id 198.51.100.254
!
!
end
```

指定したクラスタ ID のクライアント間リフレクションの無効化

（注） 特定のクラスタ ID についてリフレクション状態がソフトウェアによって変更されると、BGP はアウトバウンド ソフト リフレッシュをすべてのクライアントに送信します。

```
Router# configure terminal
Router(config)# router bgp 65000
Router(config-bgp)# address-family ipv4 unicast
Router(config-bgp-af)# bgp client-to-client reflection cluster-id 0.0.0.1 disable
Router(config-bgp)# commit
```

実行コンフィギュレーション

```
!
router bgp 65000
   address-family ipv4 unicast
    bgp client-to-client reflection cluster-id 0.0.0.1 disable
```

確認

次に、クラスタ ID のクライアント間リフレクションが無効になっていることを示す show コマンドの出力を示します。

```
Router# show bgp process
BGP Process Information:
BGP is operating in STANDALONE mode
Autonomous System number format: ASPLAIN
```

```
Autonomous System: 65000
Router ID: 0.0.0.0
Active Cluster IDs:  0.0.0.1
Fast external fallover enabled
Platform RLIMIT max: 2147483648 bytes
Maximum limit for BMP buffer size: 409 MB
Default value for BMP buffer size: 307 MB
Current limit for BMP buffer size: 307 MB
Current utilization of BMP buffer limit: 0 B
Neighbor logging is enabled
Enforce first AS enabled
Default local preference: 100
Default keepalive: 60
Non-stop routing is enabled
Update delay: 120
Generic scan interval: 60

Address family: IPv4 Unicast
Dampening is not enabled
Client reflection is not enabled in global config
Dynamic MED is Disabled
Dynamic MED interval : 10 minutes
Dynamic MED Timer : Not Running
Dynamic MED Periodic Timer : Not Running
Scan interval: 60
Total prefixes scanned: 0
Prefixes scanned per segment: 100000
Number of scan segments: 1
Nexthop resolution minimum prefix-length: 0 (not configured)
Main Table Version: 2
Table version synced to RIB: 2
Table version acked by RIB: 2
IGP notification: IGPs notified
RIB has converged: version 0
RIB table prefix-limit reached ?  [No], version 0
Permanent Network Unconfigured

Node                Process     Nbrs Estb Rst Upd-Rcvd Upd-Sent Nfn-Rcv Nfn-Snt
node0_0_CPU0        Speaker        1    0   2        0        0       0       3
```

BGP の実装方法

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 タイマーの調整

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。

BGP ネイバーにタイマーを設定するには、次の作業を実行します。

ステップ 1 **configure**

例：

```
RP/0/RP0/cpu 0: router# configure
```

モードを開始します。

ステップ 2 **router bgp** *as-number*

例：

```
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：

```
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：

```
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 **timers** *keepalive hold-time*

例：

```
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** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

BGP ルーティングの有効化

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。

ステップ 1 **configure**

例：

```
RP/0/RP0/cpu 0: router# configure
```

モードを開始します。

ステップ 2 **route-policy** *route-policy-name*

例：

```
Routing(config)# route-policy drop-as-1234
Routing(config-rpl)# if as-path passes-through '1234' then
Routing(config-rpl)# apply check-communities
Routing(config-rpl)# else
Routing(config-rpl)# pass
Routing(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：

```
Routing(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** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

ステップ 5 **configure**

例：

```
RP/0/RP0/cpu 0: router# configure
```

モードを開始します。

ステップ 6 **router bgp** *as-number*

例：

```
Routing(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：

```
Routing(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。

ステップ 8 **address-family** { **ipv4** | **ipv6** } **unicast**

例：

```
Routing(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：

```
Routing(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：

```
Routing(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：

```
Routing(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：

```
Routing(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：

```
Routing(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** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

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** または **end** コマンドを使用します。

手順の詳細

ステップ 1 **configure**

例：

```
RP/0/RP0/cpu 0: router# configure
```

モードを開始します。

ステップ 2 **router bgp** *as-number* [**instance** *instance name*]

例：

```
RP/0/RSP0/CPU0:router(config)# router bgp 100 instance inst1
```

ユーザーが指定した 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** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

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。

ステップ 1 **configure**

例：

```
RP/0/RP0/cpu 0: router# configure
```

モードを開始します。

ステップ 2 **router bgp** *as-number*

例：

```
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：

```
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：

```
Router(config-bgp)# bgp confederation peers 1091
Router(config-bgp)# bgp confederation peers 1092
Router(config-bgp)# bgp confederation peers 1093
Router(config-bgp)# bgp confederation peers 1094
Router(config-bgp)# bgp confederation peers 1095
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** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

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、このアップデートでローカル 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 .19 .69.1 のルータは通常の eBGP スピーカーであり、このピアからのアップデートは、自律システム 666 のピアから受け取る通常の eBGP アップデート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 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 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です。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
```

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。

```
Router# configure
Router(config)# lpts pifib hardware police location 0/2/CPU0
Router(config-pifib-policer-per-node)#flow bgp configured rate 4000
Router(config-pifib-policer-per-node)#flow bgp known rate 4000
Router(config-pifib-policer-per-node)#flow bgp default rate 4000
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 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。

ステップ 1 **configure**

例：

```
RP/0/RP0/cpu 0: router# configure
```

モードを開始します。

ステップ 2 **router bgp** *as-number*

例：

```
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：

```
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** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

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。

ステップ 1 **configure**

例：

```
RP/0/RP0/cpu 0: router# configure
```

モードを開始します。

ステップ 2 **router bgp** *as-number*

例：

```
Routing(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：

```
Routing(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** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

BGP ウェイトの設定

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。ネイバーから受信しルートに重みを割り当てるには、次の作業を実行します。

ステップ 1 **configure**

例：

```
RP/0/RP0/cpu 0: router# configure
```

モードを開始します。

ステップ 2 **router bgp** *as-number*

例：

```
Routing(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：

```
Routing(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：

```
Routing(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：

```
Routing(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：

```
Routing(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** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

次のタスク

新たに設定したウェイトを反映するには、clear bgp コマンドを使用します。

BGP 最適パス計算の調整

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。BGP 最適パスは、次の 3 つのステップで構成されます。

- ステップ 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。
- ステップ 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。
- ステップ 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、ステップ 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）は、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。

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。

ステップ 1 **configure**

例：

```
RP/0/RP0/cpu 0: router# configure
```

モードを開始します。

ステップ 2 **router bgp** *as-number*

例：

```
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：

```
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：

```
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：

```
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：

```
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：

```
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** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

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。

（注） インターネット bgp テーブルなどの、より大きな規模でのスーパーネットアドレスの BGP 集約を展開するときに最適な CPU 使用率を実現するには、次のことを推奨します。

- /24 を超えないサイズの集約サブネットを使用します。
- ネットワークの規模とチャーンに基づいてサブネットマスクサイズを調整します。
- デフォルトルート 0.0.0.0 をアドバタイズする場合は、集約として 0.0.0.0 の変わりに **default-originate** または **network** 0.0.0.0 CLI を使用します。

ステップ 1 **configure**

例：

```
RP/0/RP0/cpu 0: router# configure
```

モードを開始します。

ステップ 2 **router bgp** *as-number*

例：

```
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：

```
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：

```
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** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

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。

ステップ 1 **configure**

例：

```
RP/0/RP0/cpu 0: router# configure
```

モードを開始します。

ステップ 2 **router bgp** *as-number*

例：

```
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：

```
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：

```
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。

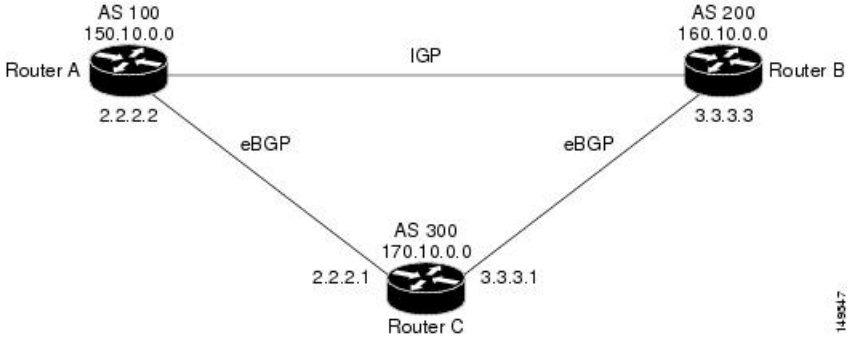
ステップ 5 **commit** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

バックドア：例



ここでは、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。

```
Router(config)# router bgp 100
Router(config-bgp)# address-family ipv4 unicast
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。

BGP 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。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、次に示すアドミニストレーティブ ディスタンスを使用します。

表 *13 :* 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。

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。

ステップ 1 **configure**

ステップ 2 **router bgp** *as-number*

例：

```
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：

```
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 **distance bgp** *external-distance internal-distance local-distance*

例：

```
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** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

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。

ステップ 1 **configure**

例：

```
RP/0/RP0/cpu 0: router# configure
```

モードを開始します。

ステップ 2 **router bgp** *as-number*

例：

```
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：

```
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：

```
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：

```
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：

```
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：

```
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。

ステップ 8 **route-policy** *route-policy-name* { **in** | **out** }

例：

```
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：

```
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：

```
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：

```
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：

```
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：

```
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** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

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 および 192.168 .7.0 に関する情報を隣接ルータと共有します。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 192.16831.7.0 255.255.0.0
  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 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。

この作業では、**bgp cluster-id** オプションを使用して、クラスタに対応するルートリフレクタの1つとしてルータを設定します。**cluster-id** オプションは、BGP ネイバーアドレスファミリ（config-bgp-nbr-af）モードでも使用できます。ルータが、クラスタ ID のリスト内の、ルータ自身のクラスタ ID と同じである最初のクラスタ ID を持つ BGP ルートを受け入れることができるようにするには、**cluster-id allow-equal** コマンドを使用します。ルーティングループを回避するために、このコマンドは慎重に使用する必要があります。

ステップ 1 **configure**

例：

```
RP/0/RP0/cpu 0: router# configure
```

モードを開始します。

ステップ 2 **router bgp** *as-number*

例：

```
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：

```
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：

```
 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：

```
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。

ステップ 6 **address-family** { **ipv4** | **ipv6** } **unicast**

例：

```
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：

```
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** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

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 プレフィックスのリフレクタ 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
```

BGP MD5 認証の概要

BGP は、Message Digest 5（MD5）認証というメカニズムを、クリア テキストまたは暗号化されたパスワードを使用して 2 つの BGP ピア間での TCP セグメントの認証に提供します。

MD5 認証は BGP ネイバー レベルで設定します。MD5 認証を使用する BGP ピアは同じパスワードで設定します。パスワード認証に失敗した場合、パケットはセグメントに従って転送されません。

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

	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 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： `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**	

BGP 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。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、次に示すアドミニストレーティブ ディスタンスを使用します。

表 *14 :* 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。

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。

ステップ 1 **configure**

ステップ 2 **router bgp** *as-number*

例：

```
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：

```
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 **distance bgp** *external-distance internal-distance local-distance*

例：

```
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** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

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

	コマンドまたはアクション	目的
ステップ 1	**configure** **例**： `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**： `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**： `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**： `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**： `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** または **end** コマンドを使用します。	**commit** ：設定の変更を保存し、コンフィギュレーション セッションに留まります。 **end** ：次のいずれかのアクションの実行をユーザーに要求します。

	コマンドまたはアクション	目的
		• **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。 • **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。 • **Cancel** ：設定の変更をコミットせずに、コンフィギュレーションセッションに留まります。

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。

ステップ 1 **configure**

例：

```
Router# configure
```

XR 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：

```
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：

```
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：

```
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：

```
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：

```
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：

```
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：

```
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：

```
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：

```
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：

```
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：

```
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：

```
Router(config-bgp-nbr)# address-family ipv4 unicast
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** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

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。

ステップ 1 **configure**

例：

```
Router# configure
```

ステップ 2 **router bgp** *as-number*

例：

```
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：

```
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：

```
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：

```
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：

```
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：

```
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：

```
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：

```
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：

```
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：

```
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：

```
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：

```
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。

ステップ 14 **address-family ipv4 unicast**

例：

```
Router(config-bgp-nbr)# address-family ipv4 unicast
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** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

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

	コマンドまたはアクション	目的
ステップ 1	**configure** 例： `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： `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： `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： `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： `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** または **end** コマンドを使用します。	**commit** ：設定の変更を保存し、コンフィギュレーション セッションに留まります。 **end** ：次のいずれかのアクションの実行をユーザーに要求します。 • **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。 • **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。 • **Cancel** ：設定の変更をコミットせずに、コンフィギュレーションセッションに留まります。

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。

ステップ 1 **configure**

例：

```
Router# configure
```

XR 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：

```
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：

```
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：

```
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：

```
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：

```
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：

```
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：

```
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：

```
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：

```
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：

```
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：

```
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：

```
Router(config-bgp-nbr)# address-family ipv4 unicast
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** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

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。

ステップ 1 **configure**

例：

```
Router# configure
```

ステップ 2 **router bgp** *as-number*

例：

```
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：

```
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：

```
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：

```
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：

```
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：

```
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：

```
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：

```
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：

```
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：

```
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：

```
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：

```
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。

ステップ 14 **address-family ipv4 unicast**

例：

```
Router(config-bgp-nbr)# address-family ipv4 unicast
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** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

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。

ステップ 1 **configure**

ステップ 2 **route-policy** *name*

例：

```
Router(config)# route-policy drop-as-1234
  Router(config-rpl)# if as-path passes-through '1234' then
  Router(config-rpl)# apply check-communities
  Router(config-rpl)# else
  Router(config-rpl)# pass
  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：

```
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：

```
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：

```
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：

```
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：

```
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

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

ステップ 1 **configure**

例：

```
RP/0/RP0/cpu 0: router# configure
```

モードを開始します。

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。

ステップ 3 **attribute-filter group** *attribute-filter group name*

例：

```
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：

```
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）では変更のシビラティ（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** または **end** コマンドを使用します。

手順の詳細

ステップ 1 configure

例：

```
RP/0/RP0/cpu 0: router# configure
```

モードを開始します。

ステップ 2 router bgp *as-number*

例：

```
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：

```
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：

```
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** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

BGP 更新でのネクスト ホップ処理のディセーブル化

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ー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。

ステップ 1 **configure**

例：

```
RP/0/RP0/cpu 0: router# configure
```

モードを開始します。

ステップ 2 **router bgp** *as-number*

例：

```
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：

```
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：

```
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：

```
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：

```
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** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

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。

ステップ 1 **configure**

例：

```
RP/0/RP0/cpu 0: router# configure
```

モードを開始します。

ステップ 2 **router bgp** *as-number*

例：

```
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：

```
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：

```
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** | | **mvpn** | **rt-filter** | **tunnel**} | **ipv6** {**labeled-unicast** | **mvpn** | **unicast**}}

例：

```
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**
- **mvpn**
- **rt-filter**
- **tunnel**
- **unicast**

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：

```
Router(config-bgp-nbr-af)# send-community-ebgp
```

または

```
Router(config-bgp-nbr-af)# send-extended-community-ebgp
```

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。

ステップ 7 **commit** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

BGP コスト コミュニティの設定

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。ユーザー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。BGP コスト コミュニティを設定するには、次の作業を実行します。

ステップ 1 **configure**

例：

```
RP/0/RP0/cpu 0: router# configure
```

モードを開始します。

ステップ 2 **route-policy** *name*

例：

```
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：

```
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：

```
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：

```
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*]

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。

ステップ 7 次のいずれかを実行します。

- **neighbor** *ip-address* **remote-as** *as-number*
- **route-policy** *route-policy-name* { **in** | **out** }

ステップ 8 **commit** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

ステップ 9 **show bgp** *ip-address*

例：

```
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。

（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** または **end** コマンドを使用します。

手順の詳細

ステップ 1 **configure**

例：

```
RP/0/RP0/cpu 0: router# configure
```

モードを開始します。

ステップ 2 **router bgp** *as-number*

例：

```
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：

```
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：

```
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：

```
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** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

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）
- IPv4 および IPv6 アドレスファミリ

BGP 永続化設定：例

次に、BGP ネイバー 10.3.3.3 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 10.3.3.3
  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。ユーザーが 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 コミュニ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、そのコミュニティに影響を与えます。

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** または **end** コマンドを使用します。

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** 例： `RP/0/RP0/cpu 0: router# configure`	モードを開始します。
ステップ 2	**router bgp** *as-number* 例： `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： `Router(config-bgp)# graceful-maintenance activate all-neighhbors`	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

	コマンドまたはアクション	目的
		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** または **end** コマンドを使用します。	**commit** ：設定の変更を保存し、コンフィギュレーション セッションに留まります。 **end** ：次のいずれかのアクションの実行をユーザーに要求します。 • **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。 • **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。 • **Cancel** ：設定の変更をコミットせずに、コンフィギュレーションセッションに留まります。

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** または **end** コマンドを使用します。

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** 例：	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/RP0/cpu 0: router# configure`	
ステップ 2	**router bgp** *as-number* 例： `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： `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： `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** または **end** コマンドを使用します。	**commit** ：設定の変更を保存し、コンフィギュレーション セッションに留まります。 **end** ：次のいずれかのアクションの実行をユーザーに要求します。 • **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。 • **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。 • **Cancel** ：設定の変更をコミットせずに、コンフィギュレーションセッションに留まります。

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** または **end** コマンドを使用します。

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** 例： `RP/0/RP0/cpu 0: router# configure`	モードを開始します。
ステップ 2	**router bgp** *as-number* 例： `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： `Router(config-bgp)# neighbor-group AS_1`	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 4	**graceful-maintenance activate** 例： `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** または **end** コマンドを使用します。	**commit** ：設定の変更を保存し、コンフィギュレーション セッションに留まります。 **end** ：次のいずれかのアクションの実行をユーザーに要求します。 • **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。 • **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。 • **Cancel** ：設定の変更をコミットせずに、コンフィギュレーションセッションに留まります。

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。

ステップ 1 **configure**

例：

```
RP/0/RP0/cpu 0: router# configure
```

モードを開始します。

ステップ 2 **router bgp** *as-number*

例：

```
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：

```
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：

```
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：

```
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
```

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
```

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ー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ー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 サーバーまたはキャッシュ サーバー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。

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** または **end** コマンドを使用します。

手順の詳細

ステップ 1 **configure**

例：

```
RP/0/RP0/cpu 0: router# configure
```

モードを開始します。

ステップ 2 **interface** *type interface-path-id*

例：

```
Router(config-if)# interface FourHundredGige 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、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：

```
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** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション モードに留まります。

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
!

Router#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
!

Router#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 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。

ステップ 1 **configure**

例：

```
RP/0/RP0/cpu 0: router# configure
```

モードを開始します。

ステップ 2 **router bgp** *as-number*

例：

```
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：

```
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：

```
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：

```
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** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

BGP での MDT アドレス ファミリ セッションの設定

BGP で IPv4 マルチキャスト配信ツリー（MDT）サブアドレス ファミリ識別子（SAFI）セッションを設定するには、次のタスクを実行します。これは MVPNv6 ネットワーク配信にも使用できます。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **address-family** { **ipv4** | **ipv6** } **unicast**
4. **exit**
5. **address-family** { **vpnv4 | vpnv6** } **unicast**
6. **exit**
7. **address-family ipv4 mdt**
8. **exit**
9. **neighbor** *ip-address*
10. **remote-as** *as-number*
11. **update-source** *interface-type interface-id*
12. **address-family** { **ipv4** | **ipv6** } **unicast**

13. **exit**
14. **address-family {vpnv4 | vpnv6} unicast**
15. **exit**
16. **address-family ipv4 mdt**
17. **exit**
18. **vrf** *vrf-name*
19. **rd** { *as-number:nn* | *ip-address:nn* | **auto** }
20. **address-family** { **ipv4** | **ipv6** } **unicast**
21. 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*]
22. **commit** または **end** コマンドを使用します。

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** 例： `RP/0/RP0/cpu 0: router# configure`	モードを開始します。
ステップ 2	**router bgp** *as-number* 例： `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： `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 4	**exit** 例： `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。

	コマンドまたはアクション	目的
ステップ 5	**address-family** { **vpnv4** | **vpnv6** } **unicast** 例： `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。 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。 （注） マルチキャスト MVPN を設定している場合は必須です。MVPNv6 を設定する場合は **vpnv6** キーワードを使用します。
ステップ 6	**exit** 例： `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 7	**address-family ipv4 mdt** 例： `Router(config-bgp)# address-family ipv4 mdt`	マルチキャスト配信ツリー（MDT）アドレス ファミリを指定します。
ステップ 8	**exit** 例： `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 9	**neighbor** *ip-address* 例： `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 10	**remote-as** *as-number* 例： `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 11	**update-source** *interface-type interface-id* 例： `Router(config-bgp-nbr)# update-source loopback 0`	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。 *interface-type interface-id* 引数では、ATM、POS、ループバックなどのインターフェイスのタイプと ID 番号を指定します。使用できるインターフェイス タイプとその ID 番号のリストをすべて表示するには、CLI ヘルプ（?）を使用します。
ステップ 12	**address-family** { **ipv4** | **ipv6** } **unicast** 例： `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。

	コマンドまたはアクション	目的
		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	**exit** 例： `Router(config-bgp-nbr-af)# exit`	（任意）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 14	**address-family {vpnv4 | vpnv6} unicast** 例： `Router(config-bgp-nbr)# address-family vpnv4 unicast`	（任意）指定されたアドレス ファミリのアドレス ファミリ コンフィギュレーション サブモードを開始します。 （注） マルチキャスト MVPN を設定している場合は必須です。MVPNv6 を設定する場合は **vpnv6** キーワードを使用します。
ステップ 15	**exit** 例： `Router(config-bgp-nbr-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 16	**address-family ipv4 mdt** 例： `Router(config-bgp)# address-family ipv4 mdt`	マルチキャスト配信ツリー（MDT）アドレス ファミリを指定します。
ステップ 17	**exit** 例： `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 18	**vrf** *vrf-name* 例： `Router(config-bgp)# vrf vpn1`	（任意）PE ルータの特定の VRF の BGP ルーティングを有効にします。 （注） マルチキャスト MVPN を設定している場合は必須です。
ステップ 19	**rd** { *as-number:nn* | *ip-address:nn* | **auto** } 例： `Router(config-bgp-vrf)# rd 1:1`	（任意）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。

	コマンドまたはアクション	目的
		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。 （注） マルチキャスト MVPN を設定している場合は必須です。
ステップ 20	**address-family** { **ipv4** | **ipv6** } **unicast** 例： `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 21	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*] 例： `Router(config-bgp-vrf-af)# redistribute eigrp 23`	（任意）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。 （注） マルチキャスト MVPN を設定している場合は必須です。
ステップ 22	**commit** または **end** コマンドを使用します。	**commit** ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

	コマンドまたはアクション	目的
		end ：次のいずれかのアクションの実行をユーザーに要求します。 • **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。 • **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。 • **Cancel** ：設定の変更をコミットせずに、コンフィギュレーションセッションに留まります。

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** または **end** コマンドを使用します。

手順の詳細

ステップ 1 **configure**

例：

```
RP/0/RP0/cpu 0: router# configure
```

モードを開始します。

ステップ 2 **router bgp** *as-number*

例：

```
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：

```
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：

```
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** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

ネイバー機能の抑制

BGP スピーカは機能ネゴシエーション機能を使用することによって、ピアでサポートされている BGP 拡張機能を学習できます。機能ネゴシエーションによって、リンクの両側の BGP ピアがサポートする機能セットだけを BGP に使用させることができます。ネイバー機能の抑制機能は、オープンメッセージの交換時にネイバー機能のネゴシエーションをオフにします。これは、機能オプションを認識しない、非常に古い顧客構内機器デバイスとの相互運用性のために必要です。

設定

ネイバーモード、セッショングループモード、およびネイバーグループモードに導入されたコマンド。

手順の概要

1. **configure**
2. **router bgp** *as-number*
3. **neighbor** *ip-address*
4. **capability suppress all**
5. **commit** または **end** コマンドを使用します。

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** 例： `RP/0/RP0/cpu 0: router# configure`	モードを開始します。
ステップ 2	**router bgp** *as-number* 例： `Router(config)# router bgp 4`	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： `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	**capability suppress all** 例： `Router(config-bgp-nbr)# capability suppress all`	ネイバー機能をオフにします。
ステップ 5	**commit** または **end** コマンドを使用します。	**commit** ：設定の変更を保存し、コンフィギュレーション セッションに留まります。 **end** ：次のいずれかのアクションの実行をユーザーに要求します。 • **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。 • **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。 • **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

BGP ダイナミック ネイバー

以前の IOS-XR は、明示的に設定されるか、または静的ネイバーの設定をサポートしていました。BGP ダイナミック ネイバーのサポートは、IP アドレスの範囲で定義されたリモート ネイバーのグループへの BGP ピアリングを可能にします。各範囲は、サブネット IP アドレスとして設定できます。

大きい BGP ネットワークで BGP ダイナミック ネイバーを実装すると CLI 設定の量と複雑さが軽減され、CPU とメモリの使用量が節約されます。IPv4 と IPv6 の両方のピアリングがサポートされています。IPv4 と IPv6 の両方のピアリングがサポートされています。

アドレス範囲を使用した BGP ダイナミック ネイバーの設定

既存のネイバーコマンドを拡張し、アドレスの代わりにプレフィックスを使用できるようにします。

次のタスクでは、リモート BGP ピアとしてルータ B を設定します。サブネット範囲を設定した後、そのサブネット範囲内に IP アドレスがあるルータ B によって TCP セッションが開始され、新しい BGP ネイバーが動的に確立されます。

サブネット範囲の初期設定とピア ネイバーのアクティベーションの後は、ダイナミック BGP ネイバーの作成時にルータ A でさらに CLI を設定する必要はありません。

ステップ 1 **configure**

例：

```
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：

```
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** *address prefix*

例：

```
Router(config-bgp)# neighbor 10.0.0.0/16
```

BGP ルーティング用のネイバー コンフィギュレーション モードを開始し、サブネット範囲内で BGP ダイナミックネイバーを設定します。

（注） 静的ネイバーで現在サポートされているすべてのコマンド（ネイバーグループ、セッショングループ、および af グループを使用したアドレスファミリと継承を含む）は、次のコマンドを除き、ダイナミックネイバー範囲でサポートされています。

- session-open-mode
- local address

ステップ 4 **remote-as** *as-number*

例：

```
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。

ステップ 5 **update-source** *type interface-id*

例：

```
Router(config-bgp-nbr)# update-source FourHundredGige 0/0/0/0
```

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。

type 引数と interface-id 引数では、インターフェイスのタイプと ID 番号を指定します。使用できるインターフェイス タイプとその ID 番号のリストをすべて表示するには、CLI ヘルプ（?）を使用します。

ステップ 6 **address-family ipv4 unicast**

例：

```
Router(config-bgp-nbr)# address-family ipv4 unicast
```

IPv4 ユニキャスト 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 モードを開始します。

ステップ 7 **commit** コマンドまたは **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

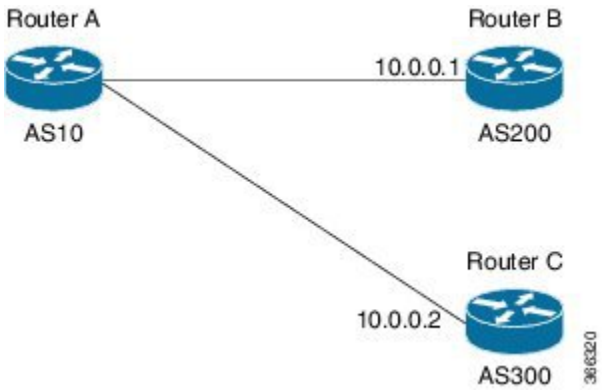
end ：次のいずれかのアクションを実行することをユーザーに要求します。

- **Yes**：設定変更を保存し、コンフィギュレーション セッションを終了します。
- **No**：設定変更をコミットせずにコンフィギュレーション セッションを終了します。
- **Cancel**：設定変更をコミットせずに、コンフィギュレーション モードに留まります。

リモート AS リスト

次のタスクでは、リモート BGP ピアとしてルータ B とルータ C を設定します。ルータ B とルータ C の両方は異なる自律システムにあります。

リモートルータの自律システムを使用してリストが作成された後、そのリストは **remote-as-list** コマンドを使用してネイバーモードで設定されます。



設定

```
Router# configure
Router(config)# router bgp as-number
Router(config-bgp)# as-list name
Router(config-bgp)# neighbor address prefix
Router(config-bgp-nbr)# remote-as-list name
Router(config-bgp-nbr)# address-family ipv4 unicast
Router# commit
```

maximum-peers と idle-watch のタイムアウト

次に、maximum-peers コマンドと idle-watch timeout コマンドをリモート BGP ピアに設定するタスクを示します。

ステップ 1 **configure**

例：

```
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：

```
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** *address prefix*

例：

```
Router(config-bgp)# neighbor 10.0.0.0/16
```

BGP ルーティング用のネイバー コンフィギュレーション モードを開始し、サブネット範囲内で BGP ダイナミック ネイバーを設定します。

ステップ 4 **maximum-peers** *number*

例：

```
Router(config-bgp-nbr)# maximum-peers 16
```

これは、範囲内で許可されるダイナミック ネイバー インスタンス数の上限を設定するために使用されます。

ステップ 5 **idle-watch-time** *number*

例：

```
Router(config-bgp)# idle-watch-time 120
```

アイドル状態の TCP インスタンスを削除するまでの待機時間を設定します。

ステップ 6 **commit** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

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

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。

ステップ 1 **show bgp neighbors**

例：

```
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。

ステップ 2 **soft** [**in** [**prefix-filter**] | **out**]

例：

```
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。

ステップ 1 **show bgp neighbors**

例：

```
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。

ステップ 2 clear bgp ipv4 unicast *ip-address*soft out

例：

```
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。

clear bgp { **ipv4** { **unicast** | **labeled-unicast** | **all** | **tunnel tunnel** | **mdt** } | **ipv6 unicast** | **all** | **labeled-unicast** } | **all** { **unicast** | **multicast** | **all** | **labeled-unicast** | **mdt** | **tunnel** } | **vpnv4 unicast** | **vrf** { *vrf-name* | **all** } { **ipv4 unicast** | **labeled-unicast** } | **ipv6 unicast** } | **vpnv6 unicast** } { ***** | *ip-address* | **as** *as-number* | **external** } [**graceful**] **soft** [**in** [**prefix-filter**] | **out**] **clear bgp** { **ipv4 | ipv6**} { **unicast** | **labeled-unicast** }

例：

```
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。
- **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。

ステップ 1 clear bgp ipv4 *ip-address*

例：

```
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：

```
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：

```
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

ステップ 1 **show bgp cidr-only**

例：

```
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：

```
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：

```
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：

```
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。

ステップ 5 **show bgp neighbors** *ip-address* [**advertised-routes** | **dampened-routes** | **flap-statistics** | **performance-statistics** | **received** *prefix-filter* | **routes**]

例：

```
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：

```
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：

```
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：

```
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。

ステップ 1 **show bgp process**

例：

```
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：

```
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：

```
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：

```
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：

```
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：

```
Router# show bgp summary
```

BGP 接続すべての状況を表示します。

ステップ 7 **show placement program bgp**

例：

```
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：

```
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。

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** または **end** コマンドを使用します。

手順の詳細

ステップ 1 **configure**

例：

```
RP/0/RP0/cpu 0: router# configure
```

モードを開始します。

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。

ステップ 3 **address-family** {**ipv4**|**ipv6**} {**unicast**|**multicast**}

例：

```
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：

```
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** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

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
```

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。

ステップ 1 **configure**

例：

```
RP/0/RP0/cpu 0: router# configure
```

モードを開始します。

ステップ 2 **route-policy** *aigp_policy*

例：

```
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：

```
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。

ステップ 4 **exit**

例：

```
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：

```
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：

```
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：

```
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** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

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

BGP Accept Own を設定するには、次の作業を実行します。

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。

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**]

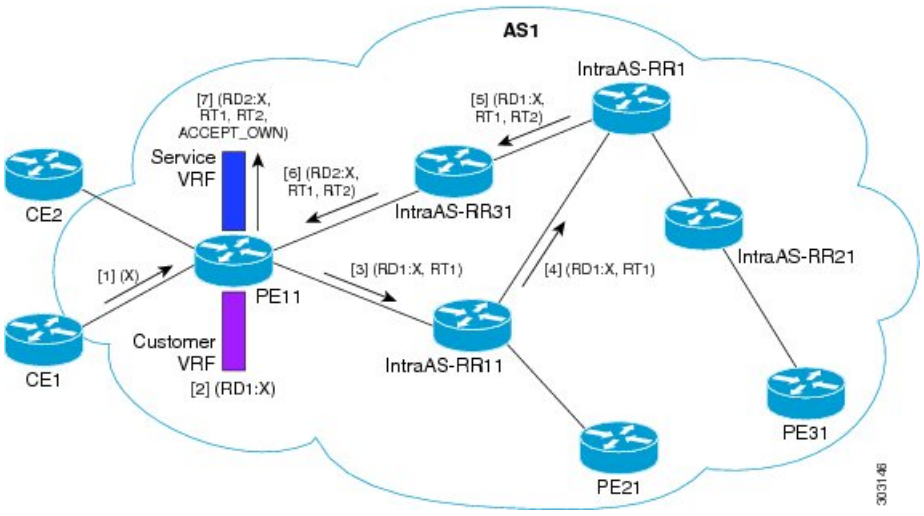
例：

```
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 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。

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 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。

ステップ 1 **configure**

例：

```
RP/0/RP0/cpu 0: router# configure
```

モードを開始します。

ステップ 2 **router bgp** *as-number*

例：

```
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：

```
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：

```
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：

```
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** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

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。

ステップ 1 **configure**

例：

```
RP/0/RP0/cpu 0: router# configure
```

モードを開始します。

ステップ 2 **prefix-set** *prefix-set-name*

例：

```
Router(config)# prefix-set PERMANENT-NETWORK-IPv4
Router(config-pfx)# 1.1.1.1/32,
Router(config-pfx)# 2.2.2.2/32,
Router(config-pfx)# 3.3.3.3/32
```

```
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：

```
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：

```
Router(config)# route-policy POLICY-PERMANENT-NETWORK-IPv4
Router(config-rpl)# if destination in PERMANENT-NETWORK-IPv4 then
Router(config-rpl)# pass
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：

```
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：

```
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：

```
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。

ステップ 8 **permanent-network** **route-policy** *route-policy-name*

例：

```
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** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

ステップ 10 **show bgp {ipv4 | ipv6} unicast** *prefix-set*

例：

```
show 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。

ステップ 1 **configure**

例：

```
RP/0/RP0/cpu 0: router# configure
```

モードを開始します。

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：

```
Router(config-bgp)# neighbor 10.255.255.254
```

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 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：

```
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：

```
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** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

ステップ 8 **show bgp {ipv4 | ipv6} unicast neighbor** *ip-address*

例：

```
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

手順

	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 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： `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： `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。 • **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： `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： `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。

	コマンドまたはアクション	目的
ステップ 7	**dmz-link-bandwidth** 例： `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**	

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
```

BGP の大型コミュニティの設定

BGP コミュニティはコミュニティ属性を使用して、宛先をグループ化し、宛先グループでの承認、拒否、優先、または再配布などのルーティングの決定を適用する方法を提供します。BGP コミュニティ属性は、2 つの 16 ビット部分に分割される 1 つ以上の 4 バイト値で構成される可変長属性です。上位の 16 ビットが AS 番号を表し、下位ビットが AS の演算子によって割り当てられたローカルに定義された値を表します。

4 バイトの ASN（RFC6793）の採用以降、4 バイトの ASN、およびルートにタグ付けする AS 固有の値をエンコードするのに 4 バイトでは不十分なため、BGP コミュニティ属性は 4 バイトの ASN に対応できなくなりました。BGP 拡張コミュニティは、グローバル管理者フィールドとして 4 バイトの AS のエンコードを許可しますが、ローカル管理者フィールドには利用可能なスペースが 2 バイトしかありません。そのため、6 バイトの拡張コミュニティ属性も適切ではありません。この制限を打開するには、12 バイトの BGP 大型コミュニティを設定します。これはオプションの属性であり、自律システム番号をグローバル管理者としてエンコードする最上位 4 バイト値と、ローカル値をエンコードする残りの 4 バイトの割り当て済みの数字を提供します。

BGP コミュニティと同様に、ルータはルートポリシー言語（RPL）を使用して BGP 大型コミュニティを BGP ルータに適用でき、他のルータはルートに付加されたコミュニティに基づいてアクションを実行できます。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。

他のコマンドで大型コミュニティを指定する場合は、コロンで区切った 3 つの負ではない 10 進整数として指定します（たとえば、1:2:3）。各整数は 32 ビットで格納されます。各整数の有効な範囲は 0 ～ 4294967295 です。

ルートポリシー ステートメントでは、BGP 大型コミュニティの各整数を次のいずれかの表現で置き換えることができます。

- [x..y]：この表現は、x と y の範囲（両端の値を含む）を指定します。
- *：この表現は任意の数値を表します。
- peeras：この表現は、必要に応じてコミュニティの送信元または送信先のネイバーの AS 番号で置き換えられます。
- not-peeras：この表現は、peeras 以外の任意の数値と一致します。
- private-as：この表現は、プライベート ASN 範囲（[64512..65534] および [4200000000..4294967294]）の任意の数値を指定します。

これらの表現は、ポリシー一致ステートメントでも使用できます。

IOS 正規表現（ios-regex）と DFA 形式の正規表現（dfa-regex）は、大型コミュニティ ポリシーのすべての match 文と delete 文に使用できます。たとえば、IOS 正規表現 ios-regex '^5:.*:7$' は、表現 5:*:7 と同等です。

send-community-ebgp コマンドは、BGP 大型コミュニティを含むように拡張されています。BGP スピーカーで大型コミュニティを ebgp ネイバーに送信するには、このコマンドが必要です。

制限とガイドライン

次に、BGP 大型コミュニティに適用される制限とガイドラインを示します。

- BGP コミュニティ属性のすべての機能を BGP 大型コミュニティ属性に使用できます。
- BGP スピーカーで大型コミュニティを ebgp ネイバーに送信するには、**send-community-ebgp** コマンドが必要です。
- よく知られた大型コミュニティはありません。
- peeras 表現は、大型コミュニティ セットでは使用できません。
- peeras 表現は、neighbor-in または neighbor-out 付加ポイントで適用されるルート ポリシーに含まれる、大型コミュニティの match 文または delete 文でのみ使用できます。
- not-peeras 表現は、大型コミュニティセットまたはポリシーの set 文では使用できません。

設定例：大型コミュニティ セット

大型のコミュニティ セットは 1 セットの大型コミュニティを定義します。ルートポリシーの match 文および set 文では、名前付きの大型コミュニティ セットが使用されます。

次の例は、名前付きの大型コミュニティ セットを作成する方法を示しています。

```
RP/0/RP0/CPU0:router(config)# large-community-set catbert
RP/0/RP0/CPU0:router(config-largecomm)#  1: 2: 3,
RP/0/RP0/CPU0:router(config-largecomm)#  peeras:2:3
RP/0/RP0/CPU0:router(config-largecomm)# end-set
```

設定例：大型コミュニティの設定

次の例に、**set large-community** {*large-community-set-name* | *inline-large-community-set* | *parameter* } [**additive**] コマンドを使用して、ルートで BGP 大型コミュニティ属性を設定する方法を示します。名前付きの大型コミュニティセットまたはインライン セットを指定できます。**additive** キーワードは、ルート内にすでに存在する大型コミュニティを保持し、新しい大型コミュニティのセットを追加します。ただし、**additive** キーワードを指定してもエントリが重複することはありません。

特定の大型コミュニティがルートに付加されている場合に、set 文の additive キーワードで同じ大型コミュニティを再度指定しても、指定した大型コミュニティは再追加されません。マージ操作を行うと、重複エントリが削除されます。これは、peeras キーワードにも適用されます。

この例の peeras 表現は、必要に応じて BGP 大型コミュニティの送信元または送信先のネイバーの AS 番号で置き換えられます。

```
RP/0/RP0/CPU0:router(config)# route-policy mordac
RP/0/RP0/CPU0:router(config-rpl)# set large-community (1:2:3, peeras:2:3)
RP/0/RP0/CPU0:router(config-rpl)# end-set
RP/0/RP0/CPU0:router(config)# large-community-set catbert
RP/0/RP0/CPU0:router(config-largecomm)#  1: 2: 3,
RP/0/RP0/CPU0:router(config-largecomm)#  peeras:2:3
RP/0/RP0/CPU0:router(config-largecomm)# end-set
RP/0/RP0/CPU0:router(config)# route-policy wally
RP/0/RP0/CPU0:router(config-rpl)# set large-community catbert additive
RP/0/RP0/CPU0:router(config-rpl)# end-set
```

この例では、ASN が 1 のネイバーにルートポリシー mordac が適用されると、大型コミュニティ (1:2:3) が一度だけ設定されます。

（注） 大型コミュニティを ebgp ネイバーに送信するには、**send-community-ebgp** コマンドを設定する必要があります。

設定例：大型コミュニティの matches-any

次の例に、大型コミュニティ セットの要素で一致を確認するルート ポリシーの設定方法を示します。これはブール型の条件であり、ルート内の大型コミュニティのいずれかが、一致条件内の大型コミュニティのいずれかに一致した場合に true を返します。

```
RP/0/RP0/CPU0:router(config)# route-policy elbonia
RP/0/RP0/CPU0:router(config-rpl)# if large-community matches-any (1:2:3, 4:5:*) then
RP/0/RP0/CPU0:router(config-rpl)#   set local-preference 94
RP/0/RP0/CPU0:router(config-rpl)#  endif
RP/0/RP0/CPU0:router(config-rpl)# end-policy
```

設定例：大型コミュニティの matches-every

次の例は、ステートメント内のすべての match 指定がルート内の 1 つ以上の大型コミュニティに一致する必要があるルート ポリシーの設定方法を示しています。

```
RP/0/RP0/CPU0:router(config)# route-policy bob
RP/0/RP0/CPU0:router(config-rpl)# if large-community matches-every (*:*:3, 4:5:*) then
RP/0/RP0/CPU0:router(config-rpl)#   set local-preference 94
RP/0/RP0/CPU0:router(config-rpl)#  endif
RP/0/RP0/CPU0:router(config-rpl)# end-policy
```

この例では、次の大型コミュニティ セットを含むルートが TRUE を返します。

- (1:1:3, 4:5:10)
- (4:5:3)：この単一の大型コミュニティは両方の仕様に一致します。
- (1:1:3, 4:5:10, 7:6:5)

次の大型コミュニティ セットを含むルートは FALSE を返します。

(1:1:3, 5:5:10)：指定 (4:5:*) は一致しません。

設定例：大型コミュニティの matches-within

次の例に、大型コミュニティ セット内で照合するルート ポリシーの設定方法を示します。これは **large-community matches-any** コマンドに似ていますが、ルート内のすべての大型コミュニティが 1 つ以上の match 指定に一致する必要があります。大型コミュニティがないルートは一致することに注意してください。

```
RP/0/RP0/CPU0:router(config)# route-policy bob
RP/0/RP0/CPU0:router(config-rpl)# if large-community matches-within (*:*:3, 4:5:*) then
RP/0/RP0/CPU0:router(config-rpl)#   set local-preference 103
RP/0/RP0/CPU0:router(config-rpl)#  endif
RP/0/RP0/CPU0:router(config-rpl)# end-policy
```

たとえば、次の大型コミュニティ セットを含むルートは TRUE を返します。

- (1:1:3, 4:5:10)
- (4:5:3)
- (1:2:3, 6:6:3, 9:4:3)

次の大型コミュニティ セットを含むルートは FALSE を返します。

(1:1:3, 4:5:10, 7:6:5)：大型コミュニティ (7:6:5) は一致しません

設定例：コミュニティの matches-within

次の例に、コミュニティ セットの要素内で照合するルート ポリシーの設定方法を示します。このコマンドは **community matches-any** コマンドに似ていますが、ルート内のすべてのコミュニティが 1 つ以上の match 指定に一致する必要があります。コミュニティがないルートは一致します。

```
RP/0/RP0/CPU0:router(config)# route-policy bob
RP/0/RP0/CPU0:router(config-rpl)# if community matches-within (*:3, 5:*)  then
```

```
RP/0/RP0/CPU0:router(config-rpl)#   set local-preference 94
RP/0/RP0/CPU0:router(config-rpl)#  endif
RP/0/RP0/CPU0:router(config-rpl)# end-policy
```

たとえば、次のコミュニティ セットを含むルートは TRUE を返します。

- (1:3, 5:10)
- (5:3)
- (2:3, 6:3, 4:3)

次のコミュニティ セットを含むルートは FALSE を返します。

(1:3, 5:10, 6:5)：コミュニティ (6:5) は一致しません。

設定例：大型コミュニティの is-empty

次の例では、**large-community is-empty** 句を使用した、大型コミュニティ属性が設定されていないルートのフィルタリングを示します。

```
RP/0/RP0/CPU0:router(config)# route-policy lrg_comm_rp4
RP/0/RP0/CPU0:router(config-rpl)# if large-community is-empty then
RP/0/RP0/CPU0:router(config-rpl)#   set local-preference 104
RP/0/RP0/CPU0:router(config-rpl)#  endif
RP/0/RP0/CPU0:router(config-rpl)# end-policy
```

設定例：属性フィルタ グループ

次の例に、大型コミュニティ属性を使用して属性フィルタ グループを設定し、BGP ネイバーに適用する方法を示します。フィルタは、BGP のパス属性と、BGP アップデート メッセージの受信時に実行するアクションを指定します。BGP ネイバーから指定の属性のいずれかが含まれているアップデート メッセージを受信すると、指定したアクションが実行されます。この例では、dogbert という属性フィルタが作成されて BGP ネイバー 10.0.1.101 に適用されます。このフィルタは、大型コミュニティ属性と破棄アクションを指定します。つまり、ネイバー 10.0.1.101 からの BGP アップデート メッセージで大型コミュニティの BGP パス属性を受信した場合、メッセージを処理する前にその属性が破棄されます。

```
RP/0/RP0/CPU0:router(config)# router bgp 100
RP/0/RP0/CPU0:router(config-bgp)# attribute-filter group dogbert
RP/0/RP0/CPU0:router(config-bgp-attrfg)# attribute LARGE-COMMUNITY discard
RP/0/RP0/CPU0:router(config-bgp-attrfg)# neighbor 10.0.1.101
RP/0/RP0/CPU0:router(config-bgp-nbr)# remote-as 6461
RP/0/RP0/CPU0:router(config-bgp-nbr)# update in filtering
RP/0/RP0/CPU0:router(config-nbr-upd-filter)# attribute-filter group dogbert
```

設定例：大型コミュニティの削除

次の例は、**delete large-community** コマンドを使用してルート ポリシーから指定の BGP 大型コミュニティを削除する方法を示しています。

```
RP/0/RP0/CPU0:router(config)# route-policy lrg_comm_rp2
RP/0/RP0/CPU0:router(config-rpl)# delete large-community in (ios-regex '^100000:’)
```

```
RP/0/RP0/CPU0:router(config-rpl)# delete large-community all
RP/0/RP0/CPU0:router(config-rpl)# delete large-community not in (peeras:*:*, 41289:*:*)
```

確認

次の例では、**show bgp large-community***list-of-large-communities* [**exact-match**] コマンドで指定した大型コミュニティを含むルートが表示されます。オプション キーワード exact-match を使用すると、リストされるルートには指定した大型コミュニティのみが含まれます。このキーワードを指定しない場合は、表示されるルートに追加の大型コミュニティが含まれることがあります。

```
RP/0/0/CPU0:R1# show bgp large-community 1:2:3 5:6:7
Thu Mar 23 14:40:33.597 PDT
BGP router identifier 4.4.4.4, local AS number 3
BGP generic scan interval 60 secs
Non-stop routing is enabled
BGP table state: Active
Table ID: 0xe0000000   RD version: 66
BGP main routing table version 66
BGP NSR Initial initsync version 3 (Reached)
BGP NSR/ISSU Sync-Group versions 66/0
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
* 10.0.0.3/32         10.10.10.3               0     94      0 ?
* 10.0.0.5/32         10.11.11.5               0             0 5 ?
```

次の例では、**show bgp** *ip-address/ prefix-length* コマンドを使用して、ネットワークに接続されている大型コミュニティを表示します。

```
RP/0/0/CPU0:R4# show bgp 10.3.3.3/32
Thu Mar 23 14:36:15.301 PDT
BGP routing table entry for 10.3.3.3/32
Versions:
  Process           bRIB/RIB  SendTblVer
  Speaker                 42          42
Last Modified: Mar 22 20:04:46.000 for 18:31:30
Paths: (1 available, best #1)
  Advertised to peers (in unique update groups):
    10.11.11.5
  Path #1: Received by speaker 0
  Advertised to peers (in unique update groups):
    10.11.11.5
  Local
    10.10.10.3 from 10.10.10.3 (10.3.3.3)
      Origin incomplete, metric 0, localpref 94, valid, internal, best, group-best
      Received Path ID 0, Local Path ID 0, version 42
      Community: 258:259 260:261 262:263 264:265
      Large Community: 1:2:3 5:6:7 4123456789:4123456780:4123456788
```

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 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

  neighbor 10.0.101.60
    remote-as 65000
    address-family ipv4 unicast

  neighbor 10.0.101.61
    remote-as 65000
    address-family ipv4 unicast

  neighbor 10.0.101.62
    remote-as 3
    address-family ipv4 unicast
      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
```

BGP アップデート グループの表示：例

次に、**show bgp update-group** 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。設定するルータは隣接ルータとの間でネットワーク 172 .16 .0.0 および 192.168 .7.0 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（自律システム番号は同一）が指定され、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、このアップデートでローカル 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 .19 .69.1 のルータは通常の eBGP スピーカーであり、このピアからのアップデートは、自律システム 666 のピアから受け取る通常の eBGP アップデート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 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 は自律システム 701 からの通常の 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
```

次に、同じ例の自律システム 701 からの BGP スピーカー 192 .168 .200.205 からの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 からの通常の 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
```

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
```

BGP MDT アドレス ファミリ設定：例

BGP での MDT アドレス ファミリの設定例を次の例に示します。

```
router bgp 10
```

```
bgp router-id 10.0.0.2
address-family ipv4 unicast
address-family vpnv4 unicast
address-family ipv4 mdt


 !
 neighbor 1.1.1.1


remote-as 11
update-source Loopback0
address-family ipv4 unicast
address-family vpnv4 unicast
address-family ipv4 md

!
```

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
```

最適外部パス アドバタイズメント設定：例

次に、最適外部パス アドバタイズメントを設定する例を示します。

```
router bgp 100
 address-family l2vpn vpls-vpws
  advertise best-external
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 100
 address-family l2vpn vpls-vpws
```

```
    additional-paths install backup
  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
```

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。

show bgp neighbor の出力には、同じプレフィックスが取り消されて再アドバタイズされた場合のプレフィックス アドバタイズ カウントの累積数が表示されます。

```
Router#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 FourHundredGige0/1/0/0
 bandwidth 8000000
 ipv4 address 11.11.11.11 255.255.255.0
 ipv6 address 11:11:0:1::11/64
!
interface FourHundredGige0/0/0/0
 bandwidth 7000000
 ipv4 address 11.11.12.11 255.255.255.0
 ipv6 address 11:11:0:2::11/64
!
interface FourHundredGige0/3/0/0
 bandwidth 6000000
 ipv4 address 11.11.13.11 255.255.255.0
 ipv6 address 11:11:0:3::11/64
!
interface FourHundredGige0/4/0/0
 bandwidth 5000000
 ipv4 address 11.11.14.11 255.255.255.0
 ipv6 address 11:11:0:4::11/64
!
interface FourHundredGige0/0/0/0
 bandwidth 4000000
 ipv4 address 11.11.15.11 255.255.255.0
 ipv6 address 11:11:0:5::11/64
!
interface FourHundredGige0/2/0/0
 bandwidth 3000000
 ipv4 address 11.11.16.11 255.255.255.0
 ipv6 address 11:11:0:6::11/64
!
interface FourHundredGige0/3/0/0
 bandwidth 2000000
 ipv4 address 11.11.17.11 255.255.255.0
 ipv6 address 11:11:0:7::11/64
!
interface FourHundredGige0/3/0/0
 bandwidth 1000000
 ipv4 address 11.11.18.11 255.255.255.0
 ipv6 address 11:11:0:8::11/64
!
interface FourHundredGige0/4/0/0
 description CONNECTED TO IXIA 1/3
```

```
 transceiver permit pid all
!
interface FourHundredGige0/4/0/0
 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 10.20.20.20
 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 10.11.13.12
  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
```

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ー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ー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 サーバーまたはキャッシュ サーバー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。

宛先ベースのフロータグ伝達の設定

宛先ベースのフローのタグ機能では、着信パケットの宛先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。

（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有効にすることはできません。

コンフィギュレーション

宛先ベースのフロータグの伝達を設定するには、次の設定例を使用します。

```
/* Configure a route policy for flow-tag propagation */
Router(config)# prefix-set FLOWTAG36
Router(config-pfx)# 10.1.30.0/24
Router(config-pfx)# end-set
Router(config)# prefix-set FLOWTAG38
Router(config-pfx)# 10.1.40.0/24
Router(config-pfx)# end-set

Router(config)# route-policy SETFLOWTAG
Router(config-rpl)# if destination in FLOWTAG36 then set flow-tag 36 endif
Router(config-rpl)# if destination in FLOWTAG38 then set flow-tag 38 endif
Router(config-rpl)# end-policy
Router(config)# commit
Tue Apr  3 15:10:07.223 IST

/* Configure the class map and policy map for flow-tag propagation */
Router(config)# class-map type traffic match-any FLOWMATCH36
Router(config-cmap)# match flow-tag 36
Router(config-cmap)# end-class-map

Router(config)# class-map type traffic match-any FLOWMATCH38
Router(config-cmap)# match flow-tag 38
Router(config-cmap)# end-class-map

Router(config)# policy-map type pbr FLOWMATCH
Router(config-pmap)# class type traffic FLOWMATCH36
Router(config-pmap-c)# redirect ipv4 nexthop 20.20.20.1
Router(config-pmap-c)# exit
Router(config-pmap)# class type traffic FLOWMATCH38
```

```
Router(config-pmap-c)# drop
Router(config-pmap-c)# exit
Router(config-pmap)# class type traffic DEFAULT
Router(config-pmap-c)# exit
Router(config-pmap)# end-policy-map

/* Configure BGP with flow-tag propagation */
 Router(config)# router bgp 10
 Router(config-bgp)# bgp router-id 1.1.1.1
 Router(config-bgp)# address-family ipv4 unicast
 Router(config-bgp-af)# table-policy SETFLOWTAG
 Router(config-bgp-af)# redistribute static
 Router(config-bgp-af)# bgp attribute-download
 Router(config-bgp-af)# redistribute connected
 Router(config-bgp-af)# exit

 Router(config-bgp)# neighbor 20.20.20.1/24
 Router(config-bgp-nbr)# remote-as 20
 Router(config-bgp-nbr)# address-family ipv4 unicast
 Router(config-bgp-nbr-af)# route-policy BGPIN in
 Router(config-bgp-nbr-af)# route-policy BGPOUT out
 Router(config-bgp-nbr-af)# exit
 Router(config-bgp-nbr)# exit
 Router(config-bgp)# exit

 Router(config)# route-policy BGPIN
 Router(config-rpl)# pass
 Router(config-rpl)# end-policy
 Router(config)# route-policy BGPOUT
 Router(config-rpl)# pass
 Router(config-rpl)# end-policy

/* Enter the interface configuration mode and enable flow tag on an interface. */
Router(config)# interface FourHundredGige 0/0/0/0
Router(config-if)# ipv4 address 10.10.10.1 255.255.255.0
Router(config-if)# service-policy type pbr input FLOWMATCH
Router(config-if)# no shut

/* Commit the configuration */
Router(config-if)# commit
Mon Mar 19 07:59:01.081 IST
RP/0/0/CPU0:Mar 19 07:59:01.537 : ifmgr[403]: %PKT_INFRA-LINK-3-UPDOWN : Interface
FourHundredGige0/1/0/0, changed state to Down
RP/0/0/CPU0:Mar 19 07:59:01.619 : ifmgr[403]: %PKT_INFRA-LINK-3-UPDOWN : Interface
FourHundredGige0/2/0/0, changed state to Up

 /* Validate the configuraton */
Router(config)# do show run
Mon Mar 19 08:03:31.106 IST
Building configuration...
!! IOS XR Configuration 0.0.0
!! Last configuration change at Mon Mar 19 08:02:55 2018 by UNKNOWN
…
class-map type traffic match-any FLOWMATCH36
 match flow-tag 36
 end-class-map
!
!
class-map type traffic match-any FLOWMATCH40
 match flow-tag 40
 end-class-map
!
policy-map type pbr FLOWMATCH
```

```
 class type traffic FLOWMATCH36
  transmit
 !
 class type traffic FLOWMATCH40
  transmit
 !
 class type traffic class-default
 !
 end-policy-map
!
interface FourHundredGige0/1/0/0
 ipv4 forwarding-enable
 ipv6 address 2000::2/64
!
interface FourHundredGige0/2/0/0
 service-policy type pbr input FLOWMATCH
 ipv4 address 10.10.10.1 255.255.255.0
!
interface FourHundredGige0/3/0/0
 ipv4 forwarding-enable
 ipv6 address 3000::2/64
!
…
!
prefix-set FLOWTAG36
  10.1.30.0/24
end-set
!
prefix-set FLOWTAG40
  10.1.40.0/24
end-set
!
route-policy SETFLOWTAG
  if destination in FLOWTAG36 then
    set flow-tag 36
  endif
  if destination in FLOWTAG40 then
    set flow-tag 40
  endif
end-policy
!
!
router bgp 10
bgp router-id 1.1.1.1
address-family ipv4 unicast
table-policy SETFLOWTAG
redistribute static
bgp attribute-download
redistribute connected
!
neighbor 20.20.20.1/24
remote-as 20
address-family ipv4 unicast
route-policy BGPIN in
route-policy BGPOUT out
!
route-policy BGPIN
 pass
end-policy
route-policy BGPOUT
 pass
end-policy
!
```

宛先ベースのフロータグの伝達が正常に設定されました。

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。

（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** または **end** コマンドを使用します。

手順の詳細

ステップ 1 **configure**

例：

```
RP/0/RP0/cpu 0: router# configure
```

モードを開始します。

ステップ 2 **router bgp** *as-number*

例：

```
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：

```
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：

```
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：

```
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** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

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ーするには、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** または **end** コマンドを使用します。

手順の詳細

ステップ 1 **configure**

例：

```
RP/0/RP0/cpu 0: router# configure
```

モードを開始します。

ステップ 2 **router bgp** *as-number*

例：

```
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：

```
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：

```
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** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

ルーティング テーブル更新時のポリシーの適用

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。この機能をイネーブルにするには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。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。

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。

ステップ 1 **configure**

例：

```
RP/0/RP0/cpu 0: router# configure
```

モードを開始します。

ステップ 2 **router bgp** *as-number*

例：

```
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。

ステップ 3 **address-family** { **ipv4** | **ipv6** } **unicast**

例：

```
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：

```
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** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

ルーティング ポリシーの適用：例

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。

```
Router(config)# route-policy pass-all
Router(config-rpl)# pass
Router(config-rpl)# end-policy
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。

```
Router(config)# router bgp 1
Router(config-bgp)# neighbor 192.168.40.24
Router(config-bgp-nbr)# remote-as 21
Router(config-bgp-nbr)# address-family ipv4 unicast
Router(config-bgp-nbr-af)# route-policy pass-all in
Router(config-bgp-nbr-af)# route-policy pass-all out
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。

```
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
```

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。

ステップ 1 **configure**

ステップ 2 **route-policy** *name*

例：

```
Router(config)# route-policy drop-as-1234
  Router(config-rpl)# if as-path passes-through '1234' then
  Router(config-rpl)# apply check-communities
  Router(config-rpl)# else
  Router(config-rpl)# pass
  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：

```
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：

```
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：

```
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：

```
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：

```
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**

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。

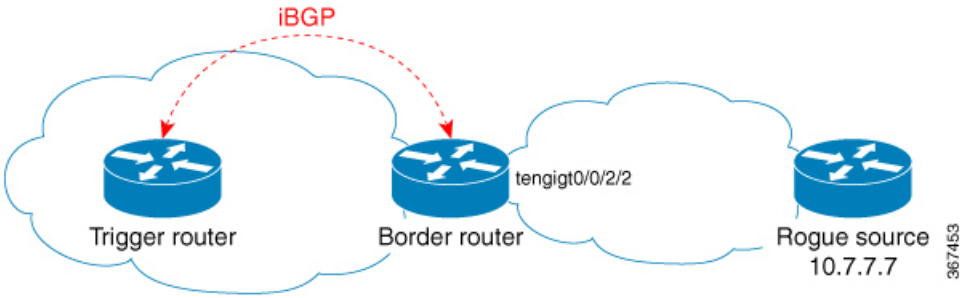
RTBH フィルタリングは、対象のプレフィックスのネクストホップをヌルインターフェイスに設定します。対象を宛先とするトラフィックは、入力時にドロップされます。

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 *13 : 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
```

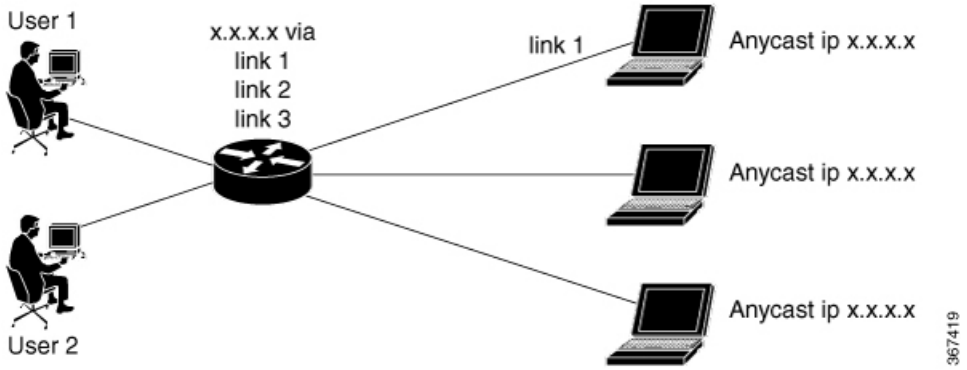
復元力のあるハッシュとフローの自動回復

復元力のあるハッシュとフローの自動回復機能は、ECMP パスの障害発生時にデフォルトの等コストマルチパス（ECMP）動作を選択的にオーバーライドするオプションを提供します。この機能により、非アクティブなリンクのみを介したフローのリダイレクトが可能になり、既存のすべてのフローが新しいリンクに再ハッシュされないようにすることができます。また、この機能には、リンクまたはサーバーが復旧したときに、それらをセッションで再利用できるようにするオプションも用意されています。

ECMP パス障害

復元力のあるハッシュとフローの自動回復機能が導入される前は、ECMP は宛先に向かう多数の使用可能なパスを介してトラフィックをロードバランシングしていました。1 つのパスに障害が発生すると、トラフィックは新しいパスのセットを介して再ハッシュされ、パスごとに新しいネクストホップが選択されます。

図 *14 : ECMP* パス障害

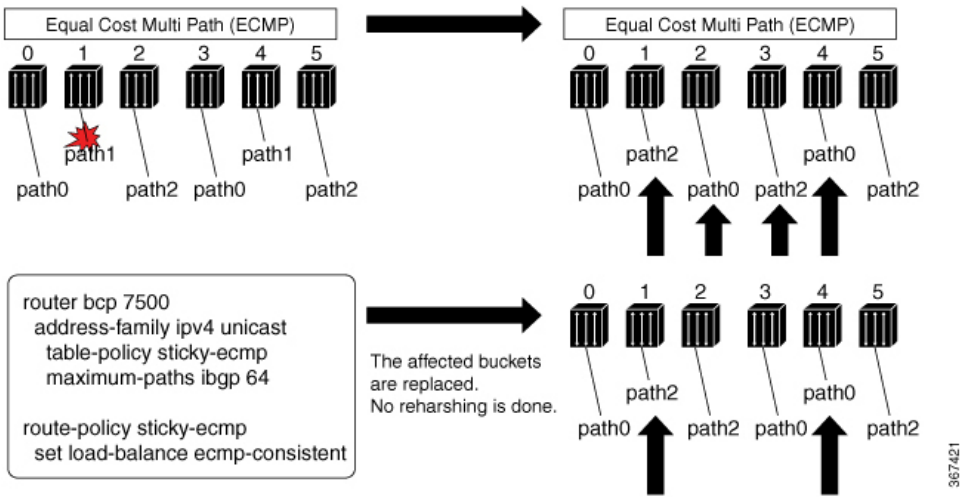


たとえば、図に示されているリンク 1、リンク 2、およびリンク 3 の 3 つのリンクで、障害が発生する前にリンク 1 を使用していたトラフィックフローは、障害が発生したのはリンク 2 のみであるにも関わらず障害後にリンク 3 を使用します。

従来のコアネットワークでは、このトラフィックフローの再配布によって問題が発生することはありません。エンドツーエンドの接続が維持され、ユーザーにはこれに伴う問題が発生しないためです。ただし、データセンター環境では、トラフィックフローの再配布によるロードバランシングが問題を引き起こす可能性があります。

複数のサーバーが ECMP を介して接続されているデータセンター環境では、この再ハッシュによってアクティブリンクのトラフィックが失われると、TCP セッションがリセットされます。

図 *15*：復元力のあるハッシュとフローの自動回復



上の図は、パス 1 で障害が発生した場合に、パスの完全な再ハッシュがどのように行われるかを示しています。ただし、復元力のあるハッシュとフローの自動回復機能が設定されている場合は、影響を受けるバケットのみが置き換えられます。再ハッシュは行われません。RPL を使用して、復元力のあるハッシュとフローの自動回復を必要とするプレフィックスを定義します。各プレフィックスにはパスリストがあります。たとえば、プレフィックス 'X' には、パスリスト、つまりパス 0、パス 1、パス 2 があります。たとえば、パス 1 に障害が発生し、復元力のあるハッシュとフローの自動回復機能を設定していた場合、新しいパスリストは、（パス 0、パス 2、およびパス 0）になるデフォルトの再ハッシュロジックではなく、（パス 0、パス 0、およびパス 2）になります。

パス 1 がアクティブになると、復元力のあるハッシュとフローの自動回復機能が設定されていない場合、再ハッシュは行われず、次のいずれかが発生するまでパスは使用されません。

- ECMP への新しいパスの追加
- **clear route** コマンドの使用。
- テーブルポリシーの削除、コミット、テーブルポリシーの追加、コミット
- **cef consistent-hashing auto-recovery** コマンドの設定

パス1がアクティブになると、復元力のあるハッシュとフローの自動回復機能が設定されている場合、セッションは自動的に再シャッフルされます。これにより、障害が発生したパスから新しいサーバーに移動されたセッションが、アクティブになった元のサーバーに再ハッシュされます。したがって、これらのセッションだけが中断されます。

永続的なロードバランシング

従来のECMPまたは等コストマルチパスは、宛先に向かう多数の使用可能なパスを介してトラフィックのロードバランシングを行います。1 つのパスに障害が発生すると、トラフィックは使用可能なパス数で再シャッフルされます。このフロー分散が、データセンターのロードバランシングで問題になる可能性があります。

永続的なロードバランシングまたはスティッキECMPは、既存のパスのフローを再ハッシュせず、障害が発生したサーバーのバケット割り当てのみを置き換えるようにプレフィックスを定義します。これには、サーバーへの確立されたセッションが再ハッシュされないという利点があります。

次の項では、永続的なロードバランシングの設定方法について説明します。

```
/*Configure persistent load balancing. */

Router(config)# router bgp 7500
Router(config-bgp)# address-family ipv4 unicast
Router(config-bgp-af)# table-policy sticky-ecmp
Router(config-bgp-af)# bgp attribute-download
Router(config-bgp-af)# maximum-paths ebgp 64
Router(config-bgp-af)# maximum-paths ibgp 32
Router(config-bgp-af)# exit
Router(config-bgp)# exit
Router(config)# route-policy sticky-ecmp
Router(config-rpl)# if destination in (192.1.1.1/24) then
Router(config-rpl-if)# set load-balance ecmp-consistent
Router(config-rpl-if)# else
Router(config-rpl-else)# pass
Router(config-rpl-else)# endif
RP/0/0/CPU0:ios(config-rpl)# end-policy
RP/0/0/CPU0:ios(config)#

/* Enable autocovery and hence recover the original hashing state
after failed paths become active. */
Router(config)# cef consistent-hashing auto-recovery

/* Recover to the original hashing state after failed paths come up
and avoid affecting newly formed flows after path failure.  */
Router(config)# clear route 192.0.2.0/24
```

実行コンフィギュレーション

```
/* Configure persistent loadbalancing. */
router bgp 7500
 address-family ipv4 unicast
  table-policy sticky-ecmp
  bgp attribute-download
  maximum-paths ebgp 64
  maximum-paths ibgp 32
```

```
cef consistent-hashing auto-recovery

clear route 192.0.2.0/24
```

確認

永続的なロードバランシングを使用したパス分散が設定されていることを確認します。

次の show 出力は、リンク障害が発生する前のパス分散のステータスを示しています。この出力では、3 つの異なる GigabitEthernet インターフェイスを通過する 3 つのネクストホップ（10.1/2/3.0.1）で 3 つのパスが識別されています。

```
show cef 192.0.2.0/24
 LDI Update time Sep  5 11:22:38.201
   via 10.1.0.1/32, 3 dependencies, recursive, bgp-multipath [flags 0x6080]
    path-idx 0 NHID 0x0 [0x57ac4e74 0x0]
    next hop 10.1.0.1/32 via 10.1.0.1/32
   via 10.2.0.1/32, 3 dependencies, recursive, bgp-multipath [flags 0x6080]
    path-idx 1 NHID 0x0 [0x57ac4a74 0x0]
    next hop 10.2.0.1/32 via 10.2.0.1/32
   via 10.3.0.1/32, 3 dependencies, recursive, bgp-multipath [flags 0x6080]
    path-idx 2 NHID 0x0 [0x57ac4f74 0x0]
    next hop 10.3.0.1/32 via 10.3.0.1/32


    Load distribution (consistent): 0 1 2 (refcount 1)

    Hash  OK  Interface                 Address
    0     Y   GigabitEthernet0/0/0/0    10.1.0.1
    1     Y   GigabitEthernet0/0/0/1    10.2.0.1
    2     Y   GigabitEthernet0/0/0/2    10.3.0.1
```

次の show 出力は、リンク障害が発生した後のパス分散のステータスを示しています。バケット 1 が GigabitEthernet 0/0/0/0 に置き換えられていることと "*" 記号により、このパスが障害が発生したパスの代替であることが示されています。

```
show cef 192.0.2.0/24
 LDI Update time Sep  5 11:23:13.434
   via 10.1.0.1/32, 3 dependencies, recursive, bgp-multipath [flags 0x6080]
    path-idx 0 NHID 0x0 [0x57ac4e74 0x0]
    next hop 10.1.0.1/32 via 10.1.0.1/32
   via 10.3.0.1/32, 3 dependencies, recursive, bgp-multipath [flags 0x6080]
    path-idx 1 NHID 0x0 [0x57ac4f74 0x0]
    next hop 10.3.0.1/32 via 10.3.0.1/32

    Load distribution (consistent) : 0 1 2 (refcount 1)
    Hash  OK  Interface                 Address
    0     Y   GigabitEthernet0/0/0/0    10.1.0.1
 1*    Y   GigabitEthernet0/0/0/0    10.1.0.1
    2     Y   GigabitEthernet0/0/0/2    10.3.0.1
```

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。

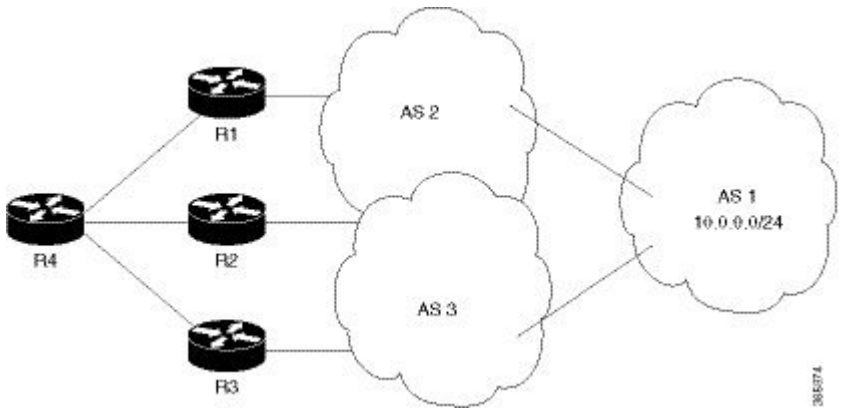
BGP 選択的マルチパスの使用時には、次の動作に注意してください。

- 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。

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 *16 : 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
Router(config)# router bgp 1
Router(config-bgp)# address-family ipv4 unicast
Router(config-bgp-af)# maximum-paths ibgp 4 selective
Router(config-bgp-af)# maximum-paths ebgp 5 selective
Router(config-bgp-af)# commit

/* To configure selective multipath for eiBGP
Router(config)# router bgp 1
Router(config-bgp)# address-family ipv4 unicast
Router(config-bgp-af)# maximum-paths eibgp 6 selective
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。

```
Router(config-bgp)# neighbor 1.1.1.1
Router(config-bgp-nbr)# address-family ipv4 unicast
Router(config-bgp-nbr-af)# multipath
Router(config-bgp-nbr-af)# commit

Router(config-bgp-nbr)# neighbor 2.2.2.2
Router(config-bgp-nbr)# address-family ipv4 unicast
Router(config-bgp-nbr-af)# multipath
Router(config-bgp-nbr-af)# commit

Router(config-bgp-nbr)# neighbor 3.3.3.3
Router(config-bgp-nbr)# address-family ipv4 unicast
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。

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** コマンドは、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。
- **remove-private-as** コマンドは、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。
- **remove-private-as** コマンドは、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 ファミリごとにネイバーに適用できます（アドレス ファミリ コンフィギュレーション モード）。そのため、この機能をあるアドレス ファミリのネイバーには適用して、別のアドレスファミリでは適用しないようにすることで、機能が設定されているアドレス ファミリのみのアウトバウンド側のアップデート メッセージに影響を与えることができます。

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。

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 Multi-Instance および Multi-AS

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。

RPKI キャッシュ サーバーの設定

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

RPKI サーバーのコンフィギュレーション モードで RPKI キャッシュ サーバー パラメータを設定します。RPKI サーバー 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。

ステップ 1 **configure**

例：

```
RP/0/RP0/cpu 0: router# configure
```

モードを開始します。

ステップ 2 **router bgp** *as-number*

例：

```
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 cache** {*host-name* | *ip-address*}

例：

```
Router(config-bgp)#rpki server 10.2.3.4
```

RPKI サーバー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 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：

```
Router(config-bgp-rpki-server)#transport ssh port 22
```

または

```
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*：TCP および SSH プロトコルを介した RPKI キャッシュ転送用のポート番号を指定します。ポート番号の範囲は 1 ～ 65535 です。

（注）
- SSH は、デフォルトのポート番号 22 に加えてカスタムポートをサポートします。
- 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 5 （任意） **username** *user_name*

例：

```
Router(config-bgp-rpki-server)#username ssh_rpki_cache
```

RPKI キャッシュ サーバーの（SSH）ユーザー名を指定します。

ステップ 6 （任意） **password**

例：

```
Router(config-bgp-rpki-server)#password ssh_rpki_pass
```

RPKI キャッシュ サーバーの（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 7 **preference** *preference_value*

例：

```
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 8 **purge-time** *time*

例：

```
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 9 次のいずれかのコマンドを使用します。

- **refresh-time** *time*
- **refresh-time off**

例：

```
Router(config-bgp-rpki-server)#refresh-time 20
```

または

```
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。

ステップ 10 次のいずれかのコマンドを使用します。

- **response-time** *time*
- **response-time off**

例：

```
Router(config-bgp-rpki-server)#response-time 30
```

または

```
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 11 **shutdown**

例：

```
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。

ステップ 12 **commit** または **end** コマンドを使用します。

commit ：設定の変更を保存し、コンフィギュレーション セッションに留まります。

end ：次のいずれかのアクションの実行をユーザーに要求します。

- **Yes** ：設定の変更を保存し、コンフィギュレーション セッションを終了します。
- **No** ：設定の変更をコミットせずに、コンフィギュレーション セッションを終了します。
- **Cancel** ：設定の変更をコミットせずに、コンフィギュレーション セッションに留まります。

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、シビラティ（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 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
```

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、LCOIR、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 の大型コミュニティの設定

BGP コミュニティはコミュニティ属性を使用して、宛先をグループ化し、宛先グループでの承認、拒否、優先、または再配布などのルーティングの決定を適用する方法を提供します。BGP コミュニティ属性は、2 つの 16 ビット部分に分割される 1 つ以上の 4 バイト値で構成される可変長属性です。上位の 16 ビットが AS 番号を表し、下位ビットが AS の演算子によって割り当てられたローカルに定義された値を表します。

4 バイトの ASN（RFC6793）の採用以降、4 バイトの ASN、およびルートにタグ付けする AS 固有の値をエンコードするのに 4 バイトでは不十分なため、BGP コミュニティ属性は 4 バイトの ASN に対応できなくなりました。BGP 拡張コミュニティは、グローバル管理者フィールドとして 4 バイトの AS のエンコードを許可しますが、ローカル管理者フィールドには利用可能なスペースが 2 バイトしかありません。そのため、6 バイトの拡張コミュニティ属性も適切ではありません。この制限を打開するには、12 バイトの BGP 大型コミュニティを設定します。これはオプションの属性であり、自律システム番号をグローバル管理者としてエンコードする最上位 4 バイト値と、ローカル値をエンコードする残りの 4 バイトの割り当て済みの数字を提供します。

BGP コミュニティと同様に、ルータはルート ポリシー言語（RPL）を使用して BGP 大型コミュニティを BGP ルータに適用でき、他のルータはルートに付加されたコミュニティに基づいてアクションを実行できます。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。

他のコマンドで大型コミュニティを指定する場合は、コロンで区切った 3 つの負ではない 10 進整数として指定します（たとえば、1:2:3）。各整数は 32 ビットで格納されます。各整数の有効な範囲は 0 ～ 4294967295 です。

ルートポリシー ステートメントでは、BGP 大型コミュニティの各整数を次のいずれかの表現で置き換えることができます。

- [x..y]：この表現は、x と y の範囲（両端の値を含む）を指定します。
- *：この表現は任意の数値を表します。
- peeras：この表現は、必要に応じてコミュニティの送信元または送信先のネイバーの AS 番号で置き換えられます。
- not-peeras：この表現は、peeras 以外の任意の数値と一致します。

- private-as：この表現は、プライベート ASN 範囲（[64512..65534] および [4200000000..4294967294]）の任意の数値を指定します。

これらの表現は、ポリシー一致ステートメントでも使用できます。

IOS 正規表現（ios-regex）と DFA 形式の正規表現（dfa-regex）は、大型コミュニティ ポリシーのすべての match 文と delete 文に使用できます。たとえば、IOS 正規表現 ios-regex '^5:.*:7$' は、表現 5:*:7 と同等です。

send-community-ebgp コマンドは、BGP 大型コミュニティを含むように拡張されています。BGP スピーカーで大型コミュニティを ebgp ネイバーに送信するには、このコマンドが必要です。

制限とガイドライン

次に、BGP 大型コミュニティに適用される制限とガイドラインを示します。

- BGP コミュニティ属性のすべての機能を BGP 大型コミュニティ属性に使用できます。
- BGP スピーカーで大型コミュニティを ebgp ネイバーに送信するには、**send-community-ebgp** コマンドが必要です。
- よく知られた大型コミュニティはありません。
- peeras 表現は、大型コミュニティ セットでは使用できません。
- peeras 表現は、neighbor-in または neighbor-out 付加ポイントで適用されるルート ポリシーに含まれる、大型コミュニティの match 文または delete 文でのみ使用できます。
- not-peeras 表現は、大型コミュニティセットまたはポリシーの set 文では使用できません。

設定例：大型コミュニティ セット

大型のコミュニティ セットは 1 セットの大型コミュニティを定義します。ルートポリシーの match 文および set 文では、名前付きの大型コミュニティ セットが使用されます。

次の例は、名前付きの大型コミュニティ セットを作成する方法を示しています。

```
Router(config)# large-community-set catbert
Router(config-largecomm)#  1: 2: 3,
Router(config-largecomm)#  peeras:2:3
Router(config-largecomm)# end-set
```

設定例：大型コミュニティの設定

次の例に、**set large-community** {*large-community-set-name* | *inline-large-community-set* | *parameter* } [**additive**] コマンドを使用して、ルートで BGP 大型コミュニティ属性を設定する方法を示します。名前付きの大型コミュニティセットまたはインライン セットを指定できます。**additive** キーワードは、ルート内にすでに存在する大型コミュニティを保持し、新しい大型コミュニティのセットを追加します。ただし、**additive** キーワードを指定してもエントリが重複することはありません。

特定の大型コミュニティがルートに付加されている場合に、set 文の additive キーワードで同じ大型コミュニティを再度指定しても、指定した大型コミュニティは再追加されません。マージ操作を行うと、重複エントリが削除されます。これは、peeras キーワードにも適用されます。

この例の peeras 表現は、必要に応じて BGP 大型コミュニティの送信元または送信先のネイバーの AS 番号で置き換えられます。

```
Router(config)# route-policy mordac
Router(config-rpl)# set large-community (1:2:3, peeras:2:3)
Router(config-rpl)# end-set
Router(config)# large-community-set catbert
Router(config-largecomm)#  1: 2: 3,
Router(config-largecomm)#  peeras:2:3
Router(config-largecomm)# end-set
Router(config)# route-policy wally
Router(config-rpl)# set large-community catbert additive
Router(config-rpl)# end-set
```

この例では、ASN が 1 のネイバーにルートポリシー mordac が適用されると、大型コミュニティ (1:2:3) が一度だけ設定されます。

（注） 大型コミュニティを ebgp ネイバーに送信するには、**send-community-ebgp** コマンドを設定する必要があります。

設定例：大型コミュニティの matches-any

次の例に、大型コミュニティ セットの要素で一致を確認するルート ポリシーの設定方法を示します。これはブール型の条件であり、ルート内の大型コミュニティのいずれかが、一致条件内の大型コミュニティのいずれかに一致した場合に true を返します。

```
Router(config)# route-policy elbonia
Router(config-rpl)# if large-community matches-any (1:2:3, 4:5:*) then
Router(config-rpl)#   set local-preference 94
Router(config-rpl)#  endif
Router(config-rpl)# end-policy
```

設定例：大型コミュニティの matches-every

次の例は、ステートメント内のすべての match 指定がルート内の 1 つ以上の大型コミュニティに一致する必要があるルート ポリシーの設定方法を示しています。

```
Router(config)# route-policy bob
Router(config-rpl)# if large-community matches-every (*:*:3, 4:5:*) then
Router(config-rpl)#   set local-preference 94
Router(config-rpl)#  endif
Router(config-rpl)# end-policy
```

この例では、次の大型コミュニティ セットを含むルートが TRUE を返します。

- (1:1:3, 4:5:10)
- (4:5:3)：この単一の大型コミュニティは両方の仕様に一致します。
- (1:1:3, 4:5:10, 7:6:5)

次の大型コミュニティ セットを含むルートは FALSE を返します。

(1:1:3, 5:5:10)：指定 (4:5:*) は一致しません。

設定例：大型コミュニティの matches-within

次の例に、大型コミュニティ セット内で照合するルート ポリシーの設定方法を示します。これは **large-community matches-any** コマンドに似ていますが、ルート内のすべての大型コミュニティが 1 つ以上の match 指定に一致する必要があります。大型コミュニティがないルートは一致することに注意してください。

```
Router(config)# route-policy bob
Router(config-rpl)# if large-community matches-within (*:*:3, 4:5:*) then
Router(config-rpl)#   set local-preference 103
Router(config-rpl)#  endif
Router(config-rpl)# end-policy
```

たとえば、次の大型コミュニティ セットを含むルートは TRUE を返します。

- (1:1:3, 4:5:10)
- (4:5:3)
- (1:2:3, 6:6:3, 9:4:3)

次の大型コミュニティ セットを含むルートは FALSE を返します。

(1:1:3, 4:5:10, 7:6:5)：大型コミュニティ (7:6:5) は一致しません

設定例：コミュニティの matches-within

次の例に、コミュニティ セットの要素内で照合するルート ポリシーの設定方法を示します。このコマンドは **community matches-any** コマンドに似ていますが、ルート内のすべてのコミュニティが 1 つ以上の match 指定に一致する必要があります。コミュニティがないルートは一致します。

```
Router(config)# route-policy bob
Router(config-rpl)# if community matches-within (*:3, 5:*)  then
Router(config-rpl)#   set local-preference 94
Router(config-rpl)#  endif
Router(config-rpl)# end-policy
```

たとえば、次のコミュニティ セットを含むルートは TRUE を返します。

- (1:3, 5:10)
- (5:3)
- (2:3, 6:3, 4:3)

次のコミュニティ セットを含むルートは FALSE を返します。

(1:3, 5:10, 6:5)：コミュニティ (6:5) は一致しません。

設定例：大型コミュニティの is-empty

次の例では、**large-community is-empty** 句を使用した、大型コミュニティ属性が設定されていないルートのフィルタリングを示します。

```
Router(config)# route-policy lrg_comm_rp4
Router(config-rpl)# if large-community is-empty then
Router(config-rpl)#   set local-preference 104
Router(config-rpl)#  endif
Router(config-rpl)# end-policy
```

設定例：属性フィルタ グループ

次の例に、大型コミュニティ属性を使用して属性フィルタ グループを設定し、BGP ネイバーに適用する方法を示します。フィルタは、BGP のパス属性と、BGP アップデート メッセージの受信時に実行するアクションを指定します。BGP ネイバーから指定の属性のいずれかが含まれているアップデートメッセージを受信すると、指定したアクションが実行されます。この例では、dogbert という属性フィルタが作成されて BGP ネイバー 10.0.1.101 に適用されます。このフィルタは、大型コミュニティ属性と破棄アクションを指定します。つまり、ネイバー 10.0.1.101 からの BGP アップデート メッセージで大型コミュニティの BGP パス属性を受信した場合、メッセージを処理する前にその属性が破棄されます。

```
Router(config)# router bgp 100
Router(config-bgp)# attribute-filter group dogbert
Router(config-bgp-attrfg)# attribute LARGE-COMMUNITY discard
Router(config-bgp-attrfg)# neighbor 10.0.1.101
Router(config-bgp-nbr)# remote-as 6461
Router(config-bgp-nbr)# update in filtering
Router(config-nbr-upd-filter)# attribute-filter group dogbert
```

設定例：大型コミュニティの削除

次の例は、**delete large-community** コマンドを使用してルート ポリシーから指定の BGP 大型コミュニティを削除する方法を示しています。

```
Router(config)# route-policy lrg_comm_rp2
Router(config-rpl)# delete large-community in (ios-regex '^100000:')
Router(config-rpl)# delete large-community all
Router(config-rpl)# delete large-community not in (peeras:*:*, 41289:*:*)
```

確認

次の例では、**show bgp large-community***list-of-large-communities* [**exact-match**] コマンドで指定した大型コミュニティを含むルートが表示されます。オプション キーワード exact-match を使用すると、リストされるルートには指定した大型コミュニティのみが含まれます。このキーワードを指定しない場合は、表示されるルートに追加の大型コミュニティが含まれることがあります。

```
Router:R1# show bgp large-community 1:2:3 5:6:7
Thu Mar 23 14:40:33.597 PDT
BGP router identifier 4.4.4.4, local AS number 3
BGP generic scan interval 60 secs
Non-stop routing is enabled
```

```
BGP table state: Active
Table ID: 0xe0000000   RD version: 66
BGP main routing table version 66
BGP NSR Initial initsync version 3 (Reached)
BGP NSR/ISSU Sync-Group versions 66/0
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
* 10.0.0.3/32         10.10.10.3               0     94      0 ?
* 10.0.0.5/32         10.11.11.5               0             0 5 ?
```

次の例では、**show bgp** *ip-address/ prefix-length* コマンドを使用して、ネットワークに接続されている大型コミュニティを表示します。

```
Router:R4# show bgp 10.3.3.3/32
Thu Mar 23 14:36:15.301 PDT
BGP routing table entry for 10.3.3.3/32
Versions:
  Process           bRIB/RIB  SendTblVer
  Speaker                 42          42
Last Modified: Mar 22 20:04:46.000 for 18:31:30
Paths: (1 available, best #1)
  Advertised to peers (in unique update groups):
    10.11.11.5
  Path #1: Received by speaker 0
  Advertised to peers (in unique update groups):
    10.11.11.5
  Local
    10.10.10.3 from 10.10.10.3 (10.3.3.3)
      Origin incomplete, metric 0, localpref 94, valid, internal, best, group-best
      Received Path ID 0, Local Path ID 0, version 42
      Community: 258:259 260:261 262:263 264:265
      Large Community: 1:2:3 5:6:7 4123456789:4123456780:4123456788
```

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。

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

BGP の Management Information Base（MIB）

Cisco IOS XR では RFC 4273 に定義されている MIB および OSPFv2/v3 のトラップが完全にサポートされています。RFC 4273 には、BGP ルーティングプロトコル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。

MIBS の詳細については、MIB Locator を使用してください。

仮想ルーティング転送ネクスト ホップ ルーティング ポリシー

表 *15 :* 機能の履歴（表）

機能名	リリース名	説明
仮想ルーティング転送ネクスト ホップ ルーティング ポリシー	リリース 7.11.1	BGP ネクストホップ接続点でルートポリシーを有効にして、特定のプレフィックスの BGP に配信される通知を制限できるようになりました。これにより、ルーティングの決定をより適切に制御できるようになり、各 VRF インスタンスの正確なトラフィック エンジニアリングとセキュリティ コンプライアンスが可能になり、各 VRF に固有の冗長パスを確立することができます。 この機能により、次の変更が導入されました。 **CLI** : 変更されたコマンド : • `nexthop route-policy` コマンドが VRF アドレス ファミリ コンフィギュレーション モードに拡張されました。 **YANG データ モデル** • 次の新しい XPath `Cisco-IOS-XR-ipv4-bgp-cfg.yang` • `Cisco-IOS-XR-um-router-bgp-cfg` （GitHub、「YANG Data Models Navigator」を参照）

VRF ネクストホップポリシーの設定

VRF テーブルでネクスト ホップ ルート ポリシーを有効にするには、次の手順を実行します。

- ルートポリシーを設定して、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。
- 特定のプレフィックスに対してBGPに配信される通知を制限するために、ルートポリシーを定義します。
- ルートポリシーで設定された条件に一致するルートのプレフィックスをドロップします。
- BGP 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 モードを開始します。
- VRF を設定します。
- IPv4 または IPv6 アドレスファミリを設定します。
- ネクストホップを使用してルート ポリシー フィルタリングを設定します。

```
Router(config)# route-policy nh-route-policy
Router(config-rpl)# if destination in (10.1.1.0/24) and protocol in (connected, static)
 then
Router(config-rpl-if)# drop
Router(config-rpl-if)# endif
Router(config-rpl)# end-policy
Router(config-rpl)# exit
Router(config)# router bgp 500
Router(config-bgp)# vrf vrf10
Router(config-bgp-vrf)# address-family ipv4 unicast
Router(config-bgp-vrf-af)# nexthop route-policy nh-route-policy
```

実行コンフィギュレーション

```
route-policy nh-route-policy
 if destination in (10.1.1.0/24) and protocol in (connected, static) then
  drop
  endif
end-policy
!

router bgp 500
 vrf vrf10
  address-family ipv4 unicast
    nexthop route-policy nh-route-policy
```

確認

設定されたネクストルートホップポリシーがVRFテーブルで有効になっていることを確認します。"BGP table nexthop route policy" フィールドが、指定された VRF インスタンス VRF1 の BGP ルートのネクストホップを決定するために使用されるルートポリシーを示しています。

```
Router# show bgp vrf vrf1 ipv4 unicast
Fri Jul  7 15:51:16.309 +0530
BGP VRF vrf1, state: Active
```

```
BGP Route Distinguisher: 1:1
VRF ID: 0x6000000b
BGP router identifier 10.1.1.1, local AS number 65001
Non-stop routing is enabled
BGP table state: Active
Table ID: 0xe000000b   RD version: 1356
BGP table nexthop route policy: nh-route-policy --> This is the same route policy that
was configured.
BGP main routing table version 1362
BGP NSR Initial initsync version 1355 (Reached)
BGP NSR/ISSU Sync-Group versions 1362/0

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
Route Distinguisher: 1:1 (default for vrf vrf1)
Route Distinguisher Version: 1356
*> 10.1.1.0/24      0.0.0.0       0       32768  ?
*> 192.0.2.0/24     10.1.1.1      0       32768  ?
*> 198.50.100.0/24  10.1.1.1      0               101  i
```

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