BGP 4

BGP is an interdomain routing protocol designed to provide loop-free routing between separate routing domains that contain independent routing policies (autonomous systems).

Information About BGP 4

BGP Version 4 Functional Overview

BGP is an interdomain routing protocol designed to provide loop-free routing links between organizations. BGP is designed to run over a reliable transport protocol; it uses TCP (port 179) as the transport protocol because TCP is a connection-oriented protocol. The destination TCP port is assigned 179, and the local port is assigned a random port number. Cisco software supports BGP version 4 and it is this version that has been used by Internet service providers (ISPs) to help build the Internet. RFC 1771 introduced and discussed a number of new BGP features to allow the protocol to scale for Internet use. RFC 2858 introduced multiprotocol extensions to allow BGP to carry routing information for IP multicast routes and multiple Layer 3 protocol address families, including IPv4, IPv6, and CLNS.

BGP is mainly used to connect a local network to an external network to gain access to the Internet or to connect to other organizations. When connecting to an external organization, external BGP (eBGP) peering sessions are created. Although BGP is referred to as an exterior gateway protocol (EGP), many networks within an organization are becoming so complex that BGP can be used to simplify the internal network used within the organization. BGP peers within the same organization exchange routing information through internal BGP (iBGP) peering sessions.

BGP uses a path-vector routing algorithm to exchange network reachability information with other BGP-speaking networking devices. Network reachability information is exchanged between BGP peers in routing updates. Network reachability information contains the network number, path-specific attributes, and the list of autonomous system numbers that a route must transit to reach a destination network. This list is contained in the AS-path attribute. BGP prevents routing loops by rejecting any routing update that contains the local autonomous system number because this indicates that the route has already traveled through that autonomous system and a loop would therefore be created. The BGP path-vector routing algorithm is a combination of the distance-vector routing algorithm and the AS-path loop detection.

BGP selects a single path, by default, as the best path to a destination host or network. The best path selection algorithm analyzes path attributes to determine which route is installed as the best path in the BGP routing table. Each path carries well-known mandatory, well-known discretionary, and optional transitive attributes that are used in BGP best path analysis. Cisco software provides the ability to influence BGP path selection by altering some of these attributes using the command-line interface (CLI.) BGP path selection can also be influenced through standard BGP policy configuration. For more details about using BGP to influence path selection and configuring BGP policies to filter traffic, see the "BGP 4 Prefix Filter and Inbound Route Maps" module and the "BGP Prefix-Based Outbound Route Filtering" module.

BGP uses the best-path selection algorithm to find a set of equally good routes. These routes are the potential multipaths. In Cisco IOS Release 12.2(33)SRD and later releases, when there are more equally good multipaths available than the maximum permitted number, the oldest paths are selected as multipaths.

BGP can be used to help manage complex internal networks by interfacing with Interior Gateway Protocols (IGPs). Internal BGP can help with issues such as scaling the existing IGPs to match the traffic demands while maintaining network efficiency.

Note BGP requires more configuration than other routing protocols and the effects of any configuration changes must be fully understood. Incorrect configuration can create routing loops and negatively impact normal network operation.

BGP Router ID

BGP uses a router ID to identify BGP-speaking peers. The BGP router ID is a 32-bit value that is often represented by an IPv4 address. By default, the Cisco software sets the router ID to the IPv4 address of a loopback interface on the router. If no loopback interface is configured on the device, the software chooses the highest IPv4 address configured on a physical interface of the device to represent the BGP router ID. The BGP router ID must be unique to the BGP peers in a network.

BGP-Speaker and Peer Relationships

A BGP-speaking device does not discover another BGP-speaking device automatically. A network administrator usually manually configures the relationships between BGP-speaking devices. A peer device is a BGP-speaking device that has an active TCP connection to another BGP-speaking device. This relationship between BGP devices is often referred to as a neighbor, but because this can imply the idea that the BGP devices are directly connected with no other device in between, the term *neighbor* will be avoided whenever possible in this document. A BGP speaker is the local device, and a peer is any other BGP-speaking network device.

When a TCP connection is established between peers, each BGP peer initially exchanges all its routes—the complete BGP routing table—with the other peer. After this initial exchange, only incremental updates are sent when there has been a topology change in the network, or when a routing policy has been implemented or modified. In the periods of inactivity between these updates, peers exchange special messages called keepalives.

A BGP autonomous system is a network that is controlled by a single technical administration entity. Peer devices are called external peers when they are in different autonomous systems and internal peers when they are in the same autonomous system. Usually, external peers are adjacent and share a subnet; internal peers may be anywhere in the same autonomous system.

BGP Peer Session Establishment

When a BGP routing process establishes a peering session with a peer, it goes through the following state changes:

- Idle—The initial state that the BGP routing process enters when the routing process is enabled or when the device is reset. In this state, the device waits for a start event, such as a peering configuration with a remote peer. After the device receives a TCP connection request from a remote peer, the device initiates another start event to wait for a timer before starting a TCP connection to a remote peer. If the device is reset, the peer is reset and the BGP routing process returns to the Idle state.
- Connect—The BGP routing process detects that a peer is trying to establish a TCP session with the local BGP speaker.
- Active—In this state, the BGP routing process tries to establish a TCP session with a peer device using the ConnectRetry timer. Start events are ignored while the BGP routing process is in the Active state. If the BGP routing process is reconfigured or if an error occurs, the BGP routing process will release system resources and return to an Idle state.
- OpenSent—The TCP connection is established, and the BGP routing process sends an OPEN message to the remote peer, and transitions to the OpenSent state. The BGP routing process can receive other OPEN messages in this state. If the connection fails, the BGP routing process transitions to the Active state.
- OpenReceive—The BGP routing process receives the OPEN message from the remote peer and waits for an initial keepalive message from the remote peer. When a keepalive message is received, the BGP routing process transitions to the Established state. If a notification message is received, the BGP routing process transitions to the Idle state. If an error or configuration change occurs that affects the peering session, the BGP routing process sends a notification message with the Finite State Machine (FSM) error code and then transitions to the Idle state.
- Established—The initial keepalive is received from the remote peer. Peering is now established with the remote neighbor and the BGP routing process starts exchanging update message with the remote peer. The hold timer restarts when an update or keepalive message is received. If the BGP process receives an error notification, it will transition to the Idle state.

BGP Session Reset

Whenever the routing policy changes due to a configuration change, BGP peering sessions must be reset by using the **clear ip bgp** command. Cisco software supports the following three mechanisms to reset BGP peering sessions:

- Hard reset—A hard reset tears down the specified peering sessions including the TCP connection and deletes routes coming from the specified peer.
- Soft reset—A soft reset uses stored prefix information to reconfigure and activate BGP routing tables without tearing down existing peering sessions. Soft reconfiguration uses stored update information, at the cost of additional memory for storing the updates, to allow you to apply new BGP policy without disrupting the network. Soft reconfiguration can be configured for inbound or outbound sessions.
- Dynamic inbound soft reset—The route refresh capability, as defined in RFC 2918, allows the local device to reset inbound routing tables dynamically by exchanging route refresh requests to supporting peers. The route refresh capability does not store update information locally for nondisruptive policy changes. It instead relies on dynamic exchange with supporting peers. Route refresh must first be advertised

through BGP capability negotiation between peers. All BGP devices must support the route refresh capability. To determine if a BGP device supports this capability, use the **show ip bgp neighbors** command. The following message is displayed in the output when the device supports the route refresh capability:

```
Received route refresh capability from peer.
```

The **bgp soft-reconfig-backup** command was introduced to configure BGP to perform inbound soft reconfiguration for peers that do not support the route refresh capability. The configuration of this command allows you to configure BGP to store updates (soft reconfiguration) only as necessary. Peers that support the route refresh capability are unaffected by the configuration of this command.

BGP Route Aggregation

BGP peers store and exchange routing information and the amount of routing information increases as more BGP speakers are configured. The use of route aggregation reduces the amount of information involved. Aggregation is the process of combining the attributes of several different routes so that only a single route is advertised. Aggregate prefixes use the classless interdomain routing (CIDR) principle to combine contiguous networks into one classless set of IP addresses that can be summarized in routing tables. Fewer routes now need to be advertised.

Two methods are available in BGP to implement route aggregation. You can redistribute an aggregated route into BGP or you can use a form of conditional aggregation. Basic route redistribution involves creating an aggregate route and then redistributing the routes into BGP. Conditional aggregation involves creating an aggregate route and then advertising or suppressing the advertising of certain routes on the basis of route maps, autonomous system set path (AS-SET) information, or summary information.

The **bgp suppress-inactive** command configures BGP to not advertise inactive routes to any BGP peer. A BGP routing process can advertise routes that are not installed in the routing information database (RIB) to BGP peers by default. A route that is not installed into the RIB is an inactive route. Inactive route advertisement can occur, for example, when routes are advertised through common route aggregation. Inactive route advertisements can be suppressed to provide more consistent data forwarding.

BGP Route Aggregation Generating AS_SET Information

AS_SET information can be generated when BGP routes are aggregated using the **aggregate-address** command. The path advertised for such a route is an AS_SET consisting of all the elements, including the communities, contained in all the paths that are being summarized. If the AS_PATHs to be aggregated are identical, only the AS_PATH is advertised. The ATOMIC-AGGREGATE attribute, set by default for the **aggregate-address** command, is not added to the AS_SET.

Routing Policy Change Management

Routing policies for a peer include all the configurations for elements such as a route map, distribute list, prefix list, and filter list that may impact inbound or outbound routing table updates. Whenever there is a change in the routing policy, the BGP session must be soft-cleared, or soft-reset, for the new policy to take effect. Performing inbound reset enables the new inbound policy configured on the device to take effect. Performing outbound reset causes the new local outbound policy configured on the device to take effect without resetting the BGP session. As a new set of updates is sent during outbound policy reset, a new inbound policy of the neighbor can also take effect. This means that after changing inbound policy, you must do an

inbound reset on the local device or an outbound reset on the peer device. Outbound policy changes require an outbound reset on the local device or an inbound reset on the peer device.

There are two types of reset: hard reset and soft reset. The table below lists their advantages and disadvantages.

Table 1: Advantages and Disadvantages of Hard and Soft Resets

Type of Reset	Advantages	Disadvantages
Hard reset	No memory overhead.	The prefixes in the BGP, IP, and Forwarding Information Base (FIB) tables provided by the neighbor are lost. A hard reset is not recommended.
Outbound soft reset	No configuration, and no storing of routing table updates.	Does not reset inbound routing table updates.
Dynamic inbound soft reset	Does not clear the BGP session and cache. Does not require storing of routing table updates, and has no memory overhead.	Both BGP devices must support the route refresh capability. **Note** Does not reset outbound routing table updates.
Configured inbound soft reset (uses the **neighbor soft-reconfiguration** router configuration command)	Can be used when both BGP devices do not support the automatic route refresh capability. The **bgp soft-reconfig-backup** command was introduced to configure inbound soft reconfiguration for peers that do not support the route refresh capability.	Requires preconfiguration. Stores all received (inbound) routing policy updates without modification; is memory-intensive. Recommended only when absolutely necessary, such as when both BGP devices do not support the automatic route refresh capability. **Note** Does not reset outbound routing table updates.

Once you have defined two devices to be BGP neighbors, they will form a BGP connection and exchange routing information. If you subsequently change a BGP filter, weight, distance, version, or timer, or if you make a similar configuration change, you must reset BGP connections in order for the configuration change to take effect.

A soft reset updates the routing table for inbound and outbound routing updates. Cisco software supports soft reset without any prior configuration. This soft reset allows the dynamic exchange of route refresh requests and routing information between BGP devices, and allows the subsequent readvertisement of the respective outbound routing table. There are two types of soft reset:

- When soft reset is used to generate inbound updates from a neighbor, it is called dynamic inbound soft reset.
- When soft reset is used to send a new set of updates to a neighbor, it is called outbound soft reset.

To use soft reset without preconfiguration, both BGP peers must support the soft route refresh capability, which is advertised in the OPEN message sent when the peers establish a TCP session.

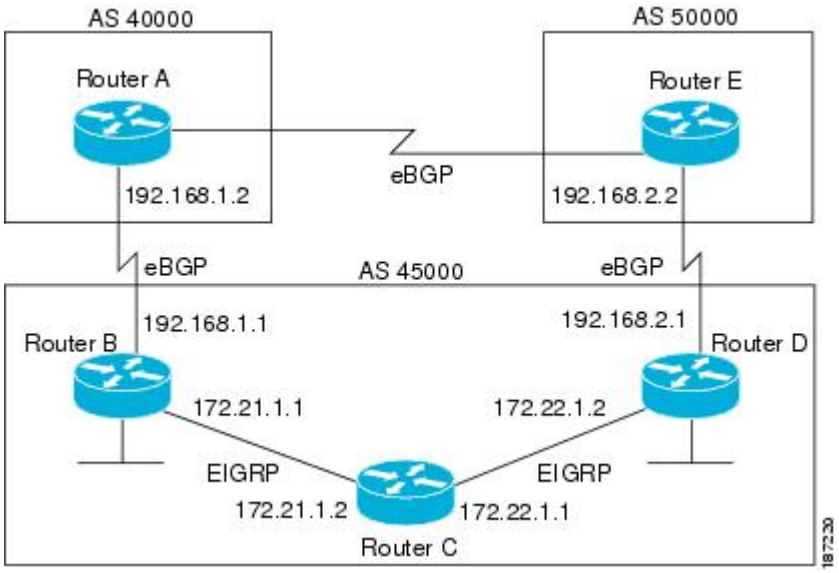
BGP Peer Groups

Often, in a BGP network, many neighbors are configured with the same update policies (that is, the same outbound route maps, distribute lists, filter lists, update source, and so on). Neighbors with the same update policies can be grouped into BGP peer groups to simplify configuration and, more importantly, to make configuration updates more efficient. When you have many peers, this approach is highly recommended.

BGP Backdoor Routes

In a BGP network topology with two border devices using eBGP to communicate to a number of different autonomous systems, using eBGP to communicate between the two border devices may not be the most efficient routing method. In the figure below, Router B as a BGP speaker will receive a route to Router D through eBGP, but this route will traverse at least two autonomous systems. Router B and Router D are also connected through an Enhanced Interior Gateway Routing Protocol (EIGRP) network (any IGP can be used here), and this route has a shorter path. EIGRP routes, however, have a default administrative distance of 90, and eBGP routes have a default administrative distance of 20, so BGP will prefer the eBGP route. Changing the default administrative distances is not recommended because changing the administrative distance may lead to routing loops. To cause BGP to prefer the EIGRP route, you can use the **network backdoor** command. BGP treats the network specified by the **network backdoor** command as a locally assigned network, except that it does not advertise the specified network in BGP updates. In the figure below, this means that Router B will communicate to Router D using the shorter EIGRP route instead of the longer eBGP route.

Figure 1: BGP Backdoor Route Topology



Restrictions for BGP

- Path MTU discovery is not supported on the Cisco RSP3 Module.
- VPLS is not supported on the Cisco RSP3 Module.
- 6PE/VPE is not supported on the Cisco RSP3 Module.

How to Configure BGP 4

Configuring a basic BGP network consists of a few required tasks and many optional tasks. A BGP routing process must be configured and BGP peers must be configured, preferably using the address family configuration model. If the BGP peers are part of a VPN network, the BGP peers must be configured using the IPv4 VRF address family task.

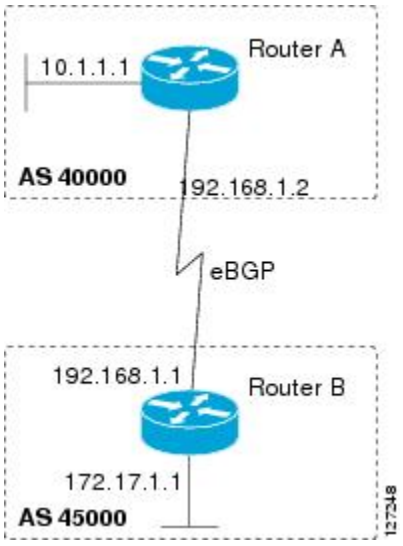
Configuring a BGP Routing Process

Perform this task to configure a BGP routing process. You must perform the required steps at least once to enable BGP. The optional steps here allow you to configure additional features in your BGP network. Several of the features, such as logging neighbor resets and immediate reset of a peer when its link goes down, are enabled by default but are presented here to enhance your understanding of how your BGP network operates.

Note A device that runs Cisco software can be configured to run only one BGP routing process and to be a member of only one BGP autonomous system. However, a BGP routing process and autonomous system can support multiple concurrent BGP address family and subaddress family configurations.

The configuration in this task is done at Router A in the figure below and would need to be repeated with appropriate changes to the IP addresses (for example, at Router B) to fully achieve a BGP process between the two devices. No address family is configured here for the BGP routing process, so routing information for the IPv4 unicast address family is advertised by default.

Figure 2: BGP Topology with Two Autonomous Systems



SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*

4. **network** *network-number* [**mask** *network-mask*] [**route-map** *route-map-name*]
5. **bgp router-id** *ip-address*
6. **timers bgp** *keepalive holdtime*
7. **bgp fast-external-fallover**
8. **bgp log-neighbor-changes**
9. **end**
10. **show ip bgp** [*network*] [*network-mask*]

DETAILED STEPS

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **router bgp** *autonomous-system-number*

Example:

```
Device(config)# router bgp 40000
```

Configures a BGP routing process, and enters router configuration mode for the specified routing process.

- Use the *autonomous-system-number* argument to specify an integer, from 0 and 65534, that identifies the device to other BGP speakers.

Step 4 **network** *network-number* [**mask** *network-mask*] [**route-map** *route-map-name*]

Example:

```
Device(config-router)# network 10.1.1.0 mask 255.255.255.0
```

(Optional) Specifies a network as local to this autonomous system and adds it to the BGP routing table.

- For exterior protocols, the **network** command controls which networks are advertised. Interior protocols use the **network** command to determine where to send updates.

Step 5 **bgp router-id** *ip-address*

Example:

```
Device(config-router)# bgp router-id 10.1.1.99
```

(Optional) Configures a fixed 32-bit router ID as the identifier of the local device running BGP.

- Use the *ip-address* argument to specify a unique router ID within the network.

Note Configuring a router ID using the **bgp router-id** command resets all active BGP peering sessions.

Step 6 **timers bgp** *keepalive holdtime*

Example:

```
Device(config-router)# timers bgp 70 120
```

(Optional) Sets BGP network timers.

- Use the *keepalive* argument to specify the frequency, in seconds, with which the software sends keepalive messages to its BGP peer. By default, the keepalive timer is set to 60 seconds.
- Use the *holdtime* argument to specify the interval, in seconds, after which the software, having not received a keepalive message, declares a BGP peer dead. By default, the holdtime timer is set to 180 seconds.

Step 7 **bgp fast-external-fallover**

Example:

```
Device(config-router)# bgp fast-external-fallover
```

(Optional) Enables the automatic resetting of BGP sessions.

- By default, the BGP sessions of any directly adjacent external peers are reset if the link used to reach them goes down.

Step 8 **bgp log-neighbor-changes**

Example:

```
Device(config-router)# bgp log-neighbor-changes
```

(Optional) Enables logging of BGP neighbor status changes (up or down) and neighbor resets.

- Use this command for troubleshooting network connectivity problems and measuring network stability. Unexpected neighbor resets might indicate high error rates or high packet loss in the network and should be investigated.

Step 9 **end**

Example:

```
Device(config-router)# end
```

Exits router configuration mode and enters privileged EXEC mode.

Step 10 **show ip bgp** [*network*] [*network-mask*]

Example:

```
Device# show ip bgp
```

(Optional) Displays the entries in the BGP routing table.

Note Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.

Examples

The following sample output from the **show ip bgp** command shows the BGP routing table for Router A in the figure above after this task has been configured on Router A. You can see an entry for the network 10.1.1.0 that is local to this autonomous system.

```
BGP table version is 12, local router ID is 10.1.1.99
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network          Next Hop            Metric LocPrf Weight Path
*> 10.1.1.0/24      0.0.0.0                  0         32768 i
```

Troubleshooting Tips

Use the **ping** command to check basic network connectivity between the BGP routers.

Configuring a BGP Peer

Perform this task to configure BGP between two IPv4 devices (peers). The address family configured here is the default IPv4 unicast address family, and the configuration is done at Router A in the figure above. Remember to perform this task for any neighboring devices that are to be BGP peers.

Before you begin

Before you perform this task, perform the "Configuring a BGP Routing Process" task.

Note By default, neighbors that are defined using the **neighbor remote-as** command in router configuration mode exchange only IPv4 unicast address prefixes. To exchange other address prefix types, such as IPv6 prefixes, neighbors must also be activated using the **neighbor activate** command in address family configuration mode for the other prefix types, such as IPv6 prefixes.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** *ip-address* **remote-as** *autonomous-system-number*
5. **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]
6. **neighbor** *ip-address* **activate**
7. **end**
8. **show ip bgp** [*network*] [*network-mask*]

9. **show ip bgp neighbors** [*neighbor-address*]

DETAILED STEPS

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **router bgp** *autonomous-system-number*

Example:

```
Device(config)# router bgp 40000
```

Enters router configuration mode for the specified routing process.

Step 4 **neighbor** *ip-address* **remote-as** *autonomous-system-number*

Example:

```
Device(config-router)# neighbor 192.168.1.1 remote-as 45000
```

Adds the IP address of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local device.

Step 5 **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]

Example:

```
Device(config-router)# address-family ipv4 unicast
```

Specifies the IPv4 address family and enters address family configuration mode.

- The **unicast** keyword specifies the IPv4 unicast address family. By default, the device is placed in configuration mode for the IPv4 unicast address family if the **unicast** keyword is not specified with the **address-family ipv4** command.
- The **multicast** keyword specifies IPv4 multicast address prefixes.
- The **vrf** keyword and *vrf-name* argument specify the name of the virtual routing and forwarding (VRF) instance to associate with subsequent IPv4 address family configuration mode commands.

Step 6 **neighbor** *ip-address* **activate**

Example:

```
Device(config-router-af)# neighbor 192.168.1.1 activate
```

Enables the neighbor to exchange prefixes for the IPv4 unicast address family with the local device.

Step 7 **end**

Example:

```
Device(config-router-af)# end
```

Exits address family configuration mode and returns to privileged EXEC mode.

Step 8 **show ip bgp** [*network*] [*network-mask*]

Example:

```
Device# show ip bgp
```

(Optional) Displays the entries in the BGP routing table.

Note Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.

Step 9 **show ip bgp neighbors** [*neighbor-address*]

Example:

```
Device(config-router-af)# show ip bgp neighbors 192.168.2.2
```

(Optional) Displays information about the TCP and BGP connections to neighbors.

Note Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.

Examples

The following sample output from the **show ip bgp** command shows the BGP routing table for Router A in the figure above after this task has been configured on Router A and Router B. You can now see an entry for the network 172.17.1.0 in autonomous system 45000.

```
BGP table version is 13, local router ID is 10.1.1.99
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network          Next Hop            Metric LocPrf Weight Path
*> 10.1.1.0/24      0.0.0.0                  0         32768 i
*> 172.17.1.0/24    192.168.1.1              0             0 45000 i
```

The following sample output from the **show ip bgp neighbors** command shows information about the TCP and BGP connections to the BGP neighbor 192.168.1.1 of Router A in the figure above after this task has been configured on Router A:

```
BGP neighbor is 192.168.1.1,  remote AS 45000, external link
```

```
  BGP version 4, remote router ID 172.17.1.99
  BGP state = Established, up for 00:06:55
  Last read 00:00:15, last write 00:00:15, hold time is 120, keepalive intervals
  Configured hold time is 120,keepalive interval is 70 seconds, Minimum holdtims
  Neighbor capabilities:
    Route refresh: advertised and received (old & new)
    Address family IPv4 Unicast: advertised and received
  Message statistics:
    InQ depth is 0
    OutQ depth is 0
                         Sent       Rcvd
    Opens:                  1          1
    Notifications:          0          0
    Updates:                1          2
    Keepalives:            13         13
    Route Refresh:          0          0
    Total:                 15         16
  Default minimum time between advertisement runs is 30 seconds
 For address family: IPv4 Unicast
  BGP table version 13, neighbor version 13/0
 Output queue size : 0
  Index 1, Offset 0, Mask 0x2
  1 update-group member
                               Sent       Rcvd
  Prefix activity:             ----       ----
    Prefixes Current:             1          1 (Consumes 52 bytes)
    Prefixes Total:               1          1
    Implicit Withdraw:            0          0
    Explicit Withdraw:            0          0
    Used as bestpath:           n/a          1
    Used as multipath:          n/a          0
                                 Outbound    Inbound
  Local Policy Denied Prefixes:  --------    -------
    AS_PATH loop:                     n/a          1
    Bestpath from this peer:            1        n/a
    Total:                              1          1
  Number of NLRIs in the update sent: max 0, min 0
  Connections established 1; dropped 0
  Last reset never
Connection state is ESTAB, I/O status: 1, unread input bytes: 0
Connection is ECN Disabled
Local host: 192.168.1.2, Local port: 179
Foreign host: 192.168.1.1, Foreign port: 37725
Enqueued packets for retransmit: 0, input: 0  mis-ordered: 0 (0 bytes)
Event Timers (current time is 0x12F4F2C):
Timer          Starts    Wakeups            Next
Retrans            14          0             0x0
TimeWait            0          0             0x0
AckHold            13          8             0x0
SendWnd             0          0             0x0
KeepAlive           0          0             0x0
GiveUp              0          0             0x0
PmtuAger            0          0             0x0
DeadWait            0          0             0x0
iss:  165379618  snduna:  165379963  sndnxt:  165379963     sndwnd:  16040
irs: 3127821601  rcvnxt: 3127821993  rcvwnd:      15993  delrcvwnd:    391
SRTT: 254 ms, RTTO: 619 ms, RTV: 365 ms, KRTT: 0 ms
minRTT: 12 ms, maxRTT: 300 ms, ACK hold: 200 ms
Flags: passive open, nagle, gen tcbs
IP Precedence value : 6
Datagrams (max data segment is 1460 bytes):
Rcvd: 20 (out of order: 0), with data: 15, total data bytes: 391
Sent: 22 (retransmit: 0, fastretransmit: 0, partialack: 0, Second Congestion: 04
```

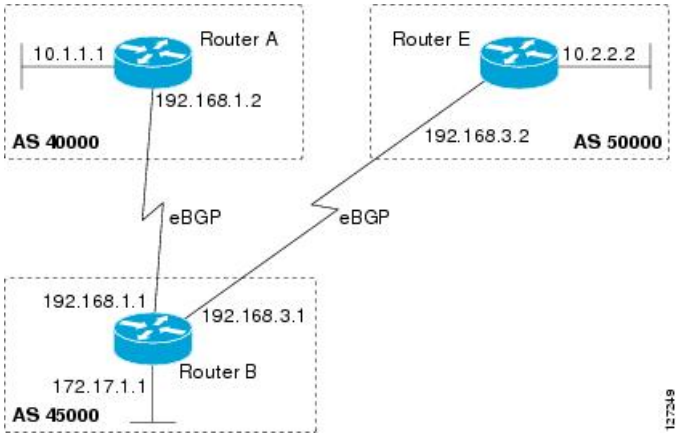
Troubleshooting Tips

Use the **ping** command to verify basic network connectivity between the BGP devices.

Configuring a BGP Peer for the IPv4 VRF Address Family

Perform this optional task to configure BGP between two IPv4 devices (peers) that must exchange IPv4 VRF information because they exist in a VPN. The address family configured here is the IPv4 VRF address family, and the configuration is done at Router B in the figure below with the neighbor 192.168.3.2 at Router E in autonomous system 50000. Remember to perform this task for any neighboring devices that are to be BGP IPv4 VRF address family peers.

Figure 3: BGP Topology for IPv4 VRF Address Family



Before you begin

Before you perform this task, perform the “Configuring a BGP Routing Process” task.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **vrf forwarding** *vrf-name*
5. **ip address** *ip-address mask* [**secondary** [**vrf** *vrf-name*]]
6. **exit**
7. **ip vrf** *vrf-name*
8. **rd** *route-distinguisher*
9. **route-target** {**import** | **export** | **both**} *route-target-ext-community*
10. **exit**
11. **router bgp** *autonomous-system-number*
12. **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]
13. **neighbor** *ip-address* **remote-as** *autonomous-system-number*
14. **neighbor** {*ip-address* | *peer-group-name*} **maximum-prefix** *maximum* [*threshold*] [**restart** *restart-interval*] [**warning-only**]

15. **neighbor** *ip-address* **activate**
16. **end**

DETAILED STEPS

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **interface** *type number*

Example:

Enters interface configuration mode.

Step 4 **vrf forwarding** *vrf-name*

Example:

```
Device(config-if)# vrf forwarding vpn1
```

Associates a VPN VRF instance with an interface or subinterface.

Step 5 **ip address** *ip-address mask* [**secondary** [**vrf** *vrf-name*]]

Example:

```
Device(config-if)# ip address 192.168.3.1 255.255.255.0
```

Sets an IP address for an interface.

Step 6 **exit**

Example:

```
Device(config-if)# exit
```

Exits interface configuration mode and enters global configuration mode.

Step 7 **ip vrf** *vrf-name*

Example:

```
Device(config)# ip vrf vpn1
```

Configures a VRF routing table and enters VRF configuration mode.

- Use the *vrf-name* argument to specify a name to be assigned to the VRF.

Step 8 **rd** *route-distinguisher*

Example:

```
Device(config-vrf)# rd 45000:5
```

Creates routing and forwarding tables and specifies the default route distinguisher for a VPN.

- Use the *route-distinguisher* argument to add an 8-byte value to an IPv4 prefix to create a unique VPN IPv4 prefix.

Step 9 **route-target** {**import** | **export** | **both**} *route-target-ext-community*

Example:

```
Device(config-vrf)# route-target both 45000:100
```

Creates a route target extended community for a VRF.

- Use the **import** keyword to import routing information from the target VPN extended community.
- Use the **export** keyword to export routing information to the target VPN extended community.
- Use the **both** keyword to import both import and export routing information to the target VPN extended community.
- Use the *route-target-ext-community* argument to add the route target extended community attributes to the VRF's list of import, export, or both (import and export) route target extended communities.

Step 10 **exit**

Example:

```
Device(config-vrf)# exit
```

Exits VRF configuration mode and enters global configuration mode.

Step 11 **router bgp** *autonomous-system-number*

Example:

```
Device(config)# router bgp 45000
```

Enters router configuration mode for the specified routing process.

Step 12 **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]

Example:

```
Device(config-router)# address-family ipv4 vrf vpn1
```

Specifies the IPv4 address family and enters address family configuration mode.

- Use the **unicast** keyword to specify the IPv4 unicast address family. By default, the device is placed in configuration mode for the IPv4 unicast address family if the **unicast** keyword is not specified with the **address-family ipv4** command.
- Use the **multicast** keyword to specify IPv4 multicast address prefixes.

- Use the **vrf** keyword and *vrf-name* argument to specify the name of the VRF instance to associate with subsequent IPv4 address family configuration mode commands.

Step 13 **neighbor** *ip-address* **remote-as** *autonomous-system-number*

Example:

```
Device(config-router-af)# neighbor 192.168.3.2 remote-as 50000
```

Adds the IP address of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local device.

Step 14 **neighbor** {*ip-address* | *peer-group-name*} **maximum-prefix** *maximum* [*threshold*] [**restart** *restart-interval*] [**warning-only**]

Example:

```
Device(config-router-af)# neighbor 192.168.3.2 maximum-prefix 10000 warning-only
```

Controls how many prefixes can be received from a neighbor.

- Use the *maximum* argument to specify the maximum number of prefixes allowed from the specified neighbor. The number of prefixes that can be configured is limited only by the available system resources on a device.
- Use the *threshold* argument to specify an integer representing a percentage of the maximum prefix limit at which the device starts to generate a warning message.
- Use the **warning-only** keyword to allow the device to generate a log message when the maximum prefix limit is exceeded, instead of terminating the peering session.

Step 15 **neighbor** *ip-address* **activate**

Example:

```
Device(config-router-af)# neighbor 192.168.3.2 activate
```

Enables the neighbor to exchange prefixes for the IPv4 VRF address family with the local device.

Step 16 **end**

Example:

```
Device(config-router-af)# end
```

Exits address family configuration mode and enters privileged EXEC mode.

Troubleshooting Tips

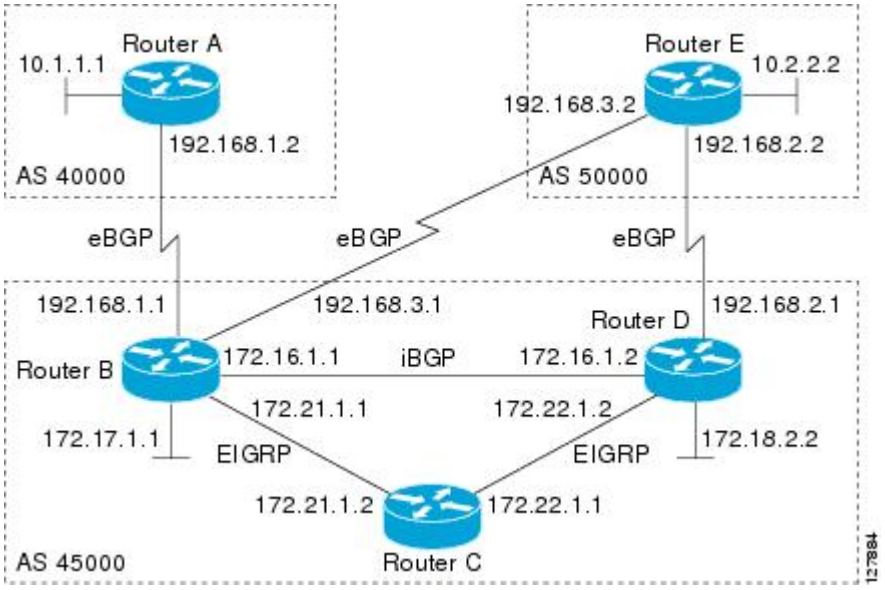
Use the **ping vrf** command to verify basic network connectivity between the BGP devices, and use the **show ip vrf** command to verify that the VRF instance has been created.

Customizing a BGP Peer

Perform this task to customize your BGP peers. Although many of the steps in this task are optional, this task demonstrates how the neighbor and address family configuration command relationships work. Using the example of the IPv4 multicast address family, neighbor address family-independent commands are configured before the IPv4 multicast address family is configured. Commands that are address family-dependent are then configured and the **exit address-family** command is shown. An optional step shows how to disable a neighbor.

The configuration in this task is done at Router B in the figure below and would need to be repeated with appropriate changes to the IP addresses, for example, at Router E to fully configure a BGP process between the two devices.

Figure 4: BGP Peer Topology



Note By default, neighbors that are defined using the **neighbor remote-as** command in router configuration mode exchange only IPv4 unicast address prefixes. To exchange other address prefix types, such as IPv6 prefixes, neighbors must also be activated using the **neighbor activate** command in address family configuration mode for the other prefix types, such as IPv6 prefixes.

Note When you configure an artificial default route with the **Neighbor** ip-address**default-originate** command, it obtains only the desired path attribute value changes if the route-map statement is added directly as an appendix.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **no bgp default ipv4-unicast**

5. **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*
6. **neighbor** {*ip-address* | *peer-group-name*} **description** *text*
7. **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]
8. **network** *network-number* [**mask** *network-mask*] [**route-map** *route-map-name*]
9. **neighbor** {*ip-address* | *peer-group-name*} **activate**
10. **neighbor** {*ip-address* | *peer-group-name*} **advertisement-interval** *seconds*
11. **neighbor** {*ip-address* | *peer-group-name*} **default-originate** [**route-map** *map-name*]
12. **exit-address-family**
13. **neighbor** {*ip-address* | *peer-group-name*} **shutdown**
14. **end**
15. **show ip bgp ipv4 multicast** [*command*]
16. **show ip bgp neighbors** [*neighbor-address*] [**received-routes** | **routes** | **advertised-routes** | **paths** *regexp* | **dampened-routes** | **received prefix-filter**]

DETAILED STEPS

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **router bgp** *autonomous-system-number*

Example:

```
Device(config)# router bgp 45000
```

Enters router configuration mode for the specified routing process.

Step 4 **no bgp default ipv4-unicast**

Example:

```
Device(config-router)# no bgp default ipv4-unicast
```

Disables the IPv4 unicast address family for the BGP routing process.

Note Routing information for the IPv4 unicast address family is advertised by default for each BGP routing session configured with the **neighbor remote-as** router configuration command unless you configure the **no bgp default ipv4-unicast** router configuration command before configuring the **neighbor remote-as** command. Existing neighbor configurations are not affected.

Step 5 **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*

Example:

```
Device(config-router)# neighbor 192.168.3.2 remote-as 50000
```

Adds the IP address of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local device.

Step 6 **neighbor** {*ip-address* | *peer-group-name*} **description** *text*

Example:

```
Device(config-router)# neighbor 192.168.3.2 description finance
```

(Optional) Associates a text description with the specified neighbor.

Step 7 **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]

Example:

```
Device(config-router)# address-family ipv4 multicast
```

Specifies the IPv4 address family and enters address family configuration mode.

- The **unicast** keyword specifies the IPv4 unicast address family. By default, the device is placed in configuration mode for the IPv4 unicast address family if the **unicast** keyword is not specified with the **address-family ipv4** command.
- The **multicast** keyword specifies IPv4 multicast address prefixes.
- The **vrf** keyword and *vrf-name* argument specify the name of the VRF instance to associate with subsequent IPv4 address family configuration mode commands.

Step 8 **network** *network-number* [**mask** *network-mask*] [**route-map** *route-map-name*]

Example:

```
Device(config-router-af)# network 172.17.1.0 mask 255.255.255.0
```

(Optional) Specifies a network as local to this autonomous system and adds it to the BGP routing table.

- For exterior protocols the **network** command controls which networks are advertised. Interior protocols use the **network** command to determine where to send updates.

Step 9 **neighbor** {*ip-address* | *peer-group-name*} **activate**

Example:

```
Device(config-router-af)# neighbor 192.168.3.2 activate
```

Enables the exchange of information with a BGP neighbor.

Step 10 **neighbor** {*ip-address* | *peer-group-name*} **advertisement-interval** *seconds*

Example:

```
Device(config-router-af)# neighbor 192.168.3.2 advertisement-interval 25
```

(Optional) Sets the minimum interval between the sending of BGP routing updates.

Step 11 **neighbor** {*ip-address* | *peer-group-name*} **default-originate** [**route-map** *map-name*]

Example:

```
Device(config-router-af)# neighbor 192.168.3.2 default-originate
```

(Optional) Permits a BGP speaker--the local device--to send the default route 0.0.0.0 to a peer for use as a default route.

Step 12 **exit-address-family**

Example:

```
Device(config-router-af)# exit-address-family
```

Exits address family configuration mode and enters router configuration mode.

Step 13 **neighbor** {*ip-address* | *peer-group-name*} **shutdown**

Example:

```
Device(config-router)# neighbor 192.168.3.2 shutdown
```

(Optional) Disables a BGP peer or peer group.

Note If you perform this step you will not be able to run either of the subsequent **show** command steps because you have disabled the neighbor.

Step 14 **end**

Example:

```
Device(config-router)# end
```

Exits router configuration mode and enters privileged EXEC mode.

Step 15 **show ip bgp ipv4 multicast** [*command*]

Example:

```
Device# show ip bgp ipv4 multicast
```

(Optional) Displays IPv4 multicast database-related information.

- Use the *command* argument to specify any multiprotocol BGP command that is supported. To see the supported commands, use the ? prompt on the CLI.

Step 16 **show ip bgp neighbors** [*neighbor-address*] [**received-routes** | **routes** | **advertised-routes** | **paths** *regexp* | **dampened-routes** | **received prefix-filter**]

Example:

```
Device# show ip bgp neighbors 192.168.3.2
```

(Optional) Displays information about the TCP and BGP connections to neighbors.

Examples

The following sample output from the **show ip bgp ipv4 multicast** command shows BGP IPv4 multicast information for Router B in the figure above after this task has been configured on Router B and Router E. Note that the networks local to each device that were configured under IPv4 multicast address family appear in the output table.

```
BGP table version is 3, local router ID is 172.17.1.99
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network          Next Hop            Metric LocPrf Weight Path
*> 10.2.2.0/24      192.168.3.2              0             0 50000 i
*> 172.17.1.0/24    0.0.0.0                  0         32768 i
```

The following partial sample output from the **show ip bgp neighbors** command for neighbor 192.168.3.2 shows general BGP information and specific BGP IPv4 multicast address family information about the neighbor. The command was entered on Router B in the figure above after this task had been configured on Router B and Router E.

```
BGP neighbor is 192.168.3.2,  remote AS 50000, external link
 Description: finance
  BGP version 4, remote router ID 10.2.2.99
  BGP state = Established, up for 01:48:27
  Last read 00:00:26, last write 00:00:26, hold time is 120, keepalive intervals
  Configured hold time is 120,keepalive interval is 70 seconds, Minimum holdtims
  Neighbor capabilities:
    Route refresh: advertised and received (old & new)
    Address family IPv4 Unicast: advertised
    Address family IPv4 Multicast: advertised and received
!
 For address family: IPv4 Multicast
  BGP table version 3, neighbor version 3/0
 Output queue size : 0
  Index 1, Offset 0, Mask 0x2
  1 update-group member
    Uses NEXT_HOP attribute for MBGP NLRIs
                                 Sent       Rcvd
  Prefix activity:               ----       ----
    Prefixes Current:               1          1 (Consumes 48 bytes)
    Prefixes Total:                 1          1
    Implicit Withdraw:              0          0
    Explicit Withdraw:              0          0
    Used as bestpath:             n/a          1
    Used as multipath:            n/a          0
                                  Outbound    Inbound
  Local Policy Denied Prefixes:   --------    -------
    Bestpath from this peer:                1        n/a
    Total:                                  1          0
  Number of NLRIs in the update sent: max 0, min 0
  Minimum time between advertisement runs is 25 seconds
  Connections established 8; dropped 7
  Last reset 01:48:54, due to User reset
Connection state is ESTAB, I/O status: 1, unread input bytes: 0
Connection is ECN Disabled
Local host: 192.168.3.1, Local port: 13172
Foreign host: 192.168.3.2, Foreign port: 179
!
```

Removing BGP Configuration Commands Using a Redistribution

BGP CLI configuration can become quite complex even in smaller BGP networks. If you need to remove any CLI configuration, you must consider all the implications of removing the CLI. Analyze the current running configuration to determine the current BGP neighbor relationships, any address family considerations, and even other routing protocols that are configured. Many BGP CLI commands affect other parts of the CLI configuration.

Perform this task to remove all the BGP configuration commands used in a redistribution of BGP routes into EIGRP. A route map can be used to match and set parameters or to filter the redistributed routes to ensure that routing loops are not created when these routes are subsequently advertised by EIGRP. When removing BGP configuration commands you must remember to remove or disable all the related commands. In this example, if the **route-map** command is omitted, then the redistribution will still occur and possibly with unexpected results as the route map filtering has been removed. Omitting just the **redistribute** command would mean that the route map is not applied, but it would leave unused commands in the running configuration.

For more details on BGP CLI removal, see the "BGP CLI Removal Considerations" concept in the "Cisco BGP Overview" module.

To view the redistribution configuration before and after the CLI removal, see the "Examples: Removing BGP Configuration Commands Using a Redistribution Example" section.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **no route-map** *map-name*
4. **router eigrp** *autonomous-system-number*
5. **no redistribute** *protocol* [*as-number*]
6. **end**
7. **show running-config**

DETAILED STEPS

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **no route-map** *map-name*

Example:

```
Device(config)# no route-map bgp-to-eigrp
```

Removes a route map from the running configuration.

- In this example, a route map named bgp-to-eigrp is removed from the configuration.

Step 4 **router eigrp** *autonomous-system-number*

Example:

```
Device(config)# router eigrp 100
```

Enters router configuration mode for the specified routing process.

Step 5 **no redistribute** *protocol* [*as-number*]

Example:

```
Device(config-router)# no redistribute bgp 45000
```

Disables the redistribution of routes from one routing domain into another routing domain.

- In this example, the configuration of the redistribution of BGP routes into the EIGRP routing process is removed from the running configuration.

Note If a route map was included in the original **redistribute** command configuration, remember to remove the **route-map** command configuration as in Step 3 in this example task.

Note Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.

Step 6 **end**

Example:

```
Device(config-router)# end
```

Exits router configuration mode and enters privileged EXEC mode.

Step 7 **show running-config**

Example:

```
Device# show running-config
```

(Optional) Displays the current running configuration on the router.

- Use this command to verify that the **redistribute** and **route-map** commands are removed from the router configuration.

Monitoring and Maintaining Basic BGP

The tasks in this section are concerned with the resetting and display of information about basic BGP processes and peer relationships. Once you have defined two devices to be BGP neighbors, they will form a BGP connection and exchange routing information. If you subsequently change a BGP filter, weight, distance, version, or timer, or make a similar configuration change, you may have to reset BGP connections for the configuration change to take effect.

Configuring Inbound Soft Reconfiguration When Route Refresh Capability Is Missing

Perform this task to configure inbound soft reconfiguration using the **bgp soft-reconfig-backup** command for BGP peers that do not support the route refresh capability. BGP peers that support the route refresh capability are unaffected by the configuration of this command. Note that the memory requirements for storing the inbound update information can become quite large.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **bgp log-neighbor-changes**
5. **bgp soft-reconfig-backup**
6. **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*
7. **neighbor** {*ip-address* | *peer-group-name*} **soft-reconfiguration** [**inbound**]
8. **neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**}
9. Repeat Steps 6 through 8 for every peer that is to be configured with inbound soft reconfiguration.
10. **exit**
11. **route-map** *map-name* [**permit** | **deny**] [*sequence-number*]
12. **set ip next-hop** *ip-address*
13. **end**
14. **show ip bgp neighbors** [*neighbor-address*]
15. **show ip bgp** [*network*] [*network-mask*]

DETAILED STEPS

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **router bgp** *autonomous-system-number*

Example:

```
Device(config)# router bgp 45000
```

Enters router configuration mode for the specified routing process.

Step 4 **bgp log-neighbor-changes**

Example:

```
Device(config-router)# bgp log-neighbor-changes
```

Enables logging of BGP neighbor resets.

Step 5 **bgp soft-reconfig-backup**

Example:

```
Device(config-router)# bgp soft-reconfig-backup
```

Configures a BGP speaker to perform inbound soft reconfiguration for peers that do not support the route refresh capability.

- This command is used to configure BGP to perform inbound soft reconfiguration for peers that do not support the route refresh capability. The configuration of this command allows you to configure BGP to store updates (soft reconfiguration) only as necessary. Peers that support the route refresh capability are unaffected by the configuration of this command.

Step 6 **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*

Example:

```
Device(config-router)# neighbor 192.168.1.2 remote-as 40000
```

Adds the IP address of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local device.

Step 7 **neighbor** {*ip-address* | *peer-group-name*} **soft-reconfiguration** [**inbound**]

Example:

```
Device(config-router)# neighbor 192.168.1.2 soft-reconfiguration inbound
```

Configures the Cisco software to start storing updates.

- All the updates received from this neighbor will be stored unmodified, regardless of the inbound policy. When inbound soft reconfiguration is done later, the stored information will be used to generate a new set of inbound updates.

Step 8 **neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**}

Example:

```
Device(config-router)# neighbor 192.168.1.2 route-map LOCAL in
```

Applies a route map to incoming or outgoing routes.

- In this example, the route map named LOCAL will be applied to incoming routes.

Step 9 Repeat Steps 6 through 8 for every peer that is to be configured with inbound soft reconfiguration.

—

Step 10 **exit**

Example:

```
Device(config-router)# exit
```

Exits router configuration mode and enters global configuration mode.

Step 11 **route-map** *map-name* [**permit** | **deny**] [*sequence-number*]

Example:

```
Device(config)# route-map LOCAL permit 10
```

Configures a route map and enters route-map configuration mode.

- In this example, a route map named LOCAL is created.

Step 12 **set ip next-hop** *ip-address*

Example:

```
Device(config-route-map)# set ip next-hop 192.168.1.144
```

Specifies where output packets that pass a match clause of a route map for policy routing.

- In this example, the ip address is set to 192.168.1.144.

Step 13 **end**

Example:

```
Device(config-route-map)# end
```

Exits route-map configuration mode and enters privileged EXEC mode.

Step 14 **show ip bgp neighbors** [*neighbor-address*]

Example:

```
Device# show ip bgp neighbors 192.168.1.2
```

(Optional) Displays information about the TCP and BGP connections to neighbors.

Note Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.

Step 15 **show ip bgp** [*network*] [*network-mask*]

Example:

```
Device# show ip bgp
```

(Optional) Displays the entries in the BGP routing table.

Note Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.

Examples

The following partial output from the **show ip bgp neighbors** command shows information about the TCP and BGP connections to the BGP neighbor 192.168.2.1. This peer supports route refresh.

```
BGP neighbor is 192.168.1.2,  remote AS 40000, external link
 Neighbor capabilities:
   Route refresh: advertised and received(new)
```

The following partial output from the **show ip bgp neighbors** command shows information about the TCP and BGP connections to the BGP neighbor 192.168.3.2. This peer does not support route refresh so the soft-reconfig inbound paths for BGP peer 192.168.3.2 will be stored because there is no other way to update any inbound policy updates.

```
BGP neighbor is 192.168.3.2,  remote AS 50000, external link
 Neighbor capabilities:
   Route refresh: advertised
```

The following sample output from the **show ip bgp** command shows the entry for the network 172.17.1.0. Both BGP peers are advertising 172.17.1.0/24, but only the received-only path is stored for 192.168.3.2.

```
BGP routing table entry for 172.17.1.0/24, version 11
Paths: (3 available, best #3, table Default-IP-Routing-Table, RIB-failure(4))
Flag: 0x820
 Advertised to update-groups:
    1
 50000
   192.168.3.2 from 192.168.3.2 (172.17.1.0)
     Origin incomplete, metric 0, localpref 200, valid, external
 50000, (received-only)
   192.168.3.2 from 192.168.3.2 (172.17.1.0)
     Origin incomplete, metric 0, localpref 100, valid, external
 40000
   192.168.1.2 from 192.168.1.2 (172.16.1.0)
     Origin incomplete, metric 0, localpref 200, valid, external, best
```

Resetting and Displaying Basic BGP Information

Perform this task to reset and display information about basic BGP processes and peer relationships.

SUMMARY STEPS

1. **enable**
2. **clear ip bgp** {***** | *autonomous-system-number* | *neighbor-address*} [**soft** [**in** | **out**]]
3. **show ip bgp** [*network-address*] [*network-mask*] [**longer-prefixes**] [**prefix-list** *prefix-list-name* | **route-map** *route-map-name*] [**shorter prefixes** *mask-length*]

4. **show ip bgp neighbors** [*neighbor-address*] [**received-routes** | **routes** | **advertised-routes** | **paths** *regexp* | **dampened-routes** | **received** *prefix-filter*]
5. **show ip bgp paths**
6. **show ip bgp summary**

DETAILED STEPS

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **clear ip bgp** {***** | *autonomous-system-number* | *neighbor-address*} [**soft** [**in** | **out**]]

Example:

```
Device# clear ip bgp *
```

Clears and resets BGP neighbor sessions:

- In the example provided, all BGP neighbor sessions are cleared and reset.

Step 3 **show ip bgp** [*network-address*] [*network-mask*] [**longer-prefixes**] [**prefix-list** *prefix-list-name* | **route-map** *route-map-name*] [**shorter prefixes** *mask-length*]

Example:

```
Device# show ip bgp 10.1.1.0 255.255.255.0
```

Displays all the entries in the BGP routing table:

- In the example provided, the BGP routing table information for the 10.1.1.0 network is displayed.

Step 4 **show ip bgp neighbors** [*neighbor-address*] [**received-routes** | **routes** | **advertised-routes** | **paths** *regexp* | **dampened-routes** | **received** *prefix-filter*]

Example:

```
Device# show ip bgp neighbors 192.168.3.2 advertised-routes
```

Displays information about the TCP and BGP connections to neighbors.

- In the example provided, the routes advertised from the device to BGP neighbor 192.168.3.2 on another device are displayed.

Step 5 **show ip bgp paths**

Example:

```
Device# show ip bgp paths
```

Displays information about all the BGP paths in the database.

Step 6 **show ip bgp summary**

Example:

```
Device# show ip bgp summary
```

Displays information about the status of all BGP connections.

Aggregating Route Prefixes Using BGP

BGP peers exchange information about local networks, but this can quickly lead to large BGP routing tables. CIDR enables the creation of aggregate routes (or *supernets*) to minimize the size of routing tables. Smaller BGP routing tables can reduce the convergence time of the network and improve network performance. Aggregated routes can be configured and advertised using BGP. Some aggregations advertise only summary routes and other methods of aggregating routes allow more specific routes to be forwarded. Aggregation applies only to routes that exist in the BGP routing table. An aggregated route is forwarded if at least one more specific route of the aggregation exists in the BGP routing table. Perform one of the following tasks to aggregate routes within BGP:

Redistributing a Static Aggregate Route into BGP

Use this task to redistribute a static aggregate route into BPG. A static aggregate route is configured and then redistributed into the BGP routing table. The static route must be configured to point to interface null 0 and the prefix should be a superset of known BGP routes. When a device receives a BGP packet, it will use the more specific BGP routes. If the route is not found in the BGP routing table, then the packet will be forwarded to null 0 and discarded.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip route** *prefix mask* {*ip-address* | *interface-type interface-number* [*ip-address*]} [*distance*] [*name*] [**permanent** | **track** *number*] [**tag** *tag*]
4. **router bgp** *autonomous-system-number*
5. **redistribute static**
6. **end**

DETAILED STEPS

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **ip route** *prefix mask* {*ip-address* | *interface-type interface-number* [*ip-address*]} [*distance*] [*name*] [**permanent** | **track** *number*] [**tag** *tag*]

Example:

```
Device(config)# ip route 172.17.0.0 255.0.0.0 null 0
```

Creates a static route.

Step 4 **router bgp** *autonomous-system-number*

Example:

```
Device(config)# router bgp 45000
```

Enters router configuration mode for the specified routing process.

Step 5 **redistribute static**

Example:

```
Device(config-router)# redistribute static
```

Redistributes routes into the BGP routing table.

Step 6 **end**

Example:

```
Device(config-router)# end
```

Exits router configuration mode and returns to privileged EXEC mode.

Configuring Conditional Aggregate Routes Using BGP

Use this task to create an aggregate route entry in the BGP routing table when at least one specific route falls into the specified range. The aggregate route is advertised as originating from your autonomous system. For more information, see the "BGP Route Aggregation Generating AS_SET Information" section.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **aggregate-address** *address mask* [**as-set**]
5. **end**

DETAILED STEPS

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **router bgp** *autonomous-system-number*

Example:

```
Device(config)# router bgp 45000
```

Enters router configuration mode for the specified routing process.

Step 4 **aggregate-address** *address mask* [**as-set**]

Example:

```
Device(config-router)# aggregate-address 172.17.0.0 255.0.0.0 as-set
```

Creates an aggregate entry in a BGP routing table.

- A specified route must exist in the BGP table.
- Use the **aggregate-address** command with no keywords to create an aggregate entry if any more-specific BGP routes are available that fall in the specified range.
- Use the **as-set** keyword to specify that the path advertised for this route is an AS_SET. Do not use the **as-set** keyword when aggregating many paths because this route is withdrawn and updated every time the reachability information for the aggregated route changes.

Note Only partial syntax is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.

Step 5 **end**

Example:

```
Device(config-router)# end
```

Exits router configuration mode and enters privileged EXEC mode.

Suppressing and Unsuppressing the Advertisement of Aggregated Routes Using BGP

Use this task to create an aggregate route, suppress the advertisement of routes using BGP, and subsequently unsuppress the advertisement of routes. Routes that are suppressed are not advertised to any neighbors, but it is possible to unsuppress routes that were previously suppressed to specific neighbors.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** *ip-address* **remote-as** *autonomous-system-number*
5. Do one of the following:
   - **aggregate-address** *address mask* [**summary-only**]
   - **aggregate-address** *address mask* [**suppress-map** *map-name*]
6. **neighbor** {*ip-address* | *peer-group-name*} **unsuppress-map** *map-name*
7. **end**

DETAILED STEPS

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **router bgp** *autonomous-system-number*

Example:

```
Device(config)# router bgp 45000
```

Enters router configuration mode for the specified routing process.

Step 4 **neighbor** *ip-address* **remote-as** *autonomous-system-number*

Example:

```
Device(config-router)# neighbor 192.168.1.2 remote-as 40000
```

Adds the IP address of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local device.

Step 5 Do one of the following:

- **aggregate-address** *address mask* [**summary-only**]
- **aggregate-address** *address mask* [**suppress-map** *map-name*]

Example:

```
Device(config-router)# aggregate-address 172.17.0.0 255.0.0.0 summary-only
```

Example:

```
Device(config-router)# aggregate-address 172.17.0.0 255.0.0.0 suppress-map map1
```

Creates an aggregate route.

- Use the optional **summary-only** keyword to create the aggregate route (for example, 10.*.*.*) and also suppresses advertisements of more-specific routes to all neighbors.
- Use the optional **suppress-map** keyword to create the aggregate route but suppress advertisement of specified routes. Routes that are suppressed are not advertised to any neighbors. You can use the **match** clauses of route maps to selectively suppress some more-specific routes of the aggregate and leave others unsuppressed. IP access lists and autonomous system path access lists **match** clauses are supported.

Note Only partial syntax is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.

Step 6 **neighbor** {*ip-address* | *peer-group-name*} **unsuppress-map** *map-name*

Example:

```
Device(config-router)# neighbor 192.168.1.2 unsuppress map1
```

(Optional) Selectively advertises routes previously suppressed by the **aggregate-address** command.

- In this example, the routes previously suppressed in Step 5 are advertised to neighbor 192.168.1.2.

Step 7 **end**

Example:

```
Device(config-router)# end
```

Exits router configuration mode and enters privileged EXEC mode.

Conditionally Advertising BGP Routes

Perform this task to conditionally advertise selected BGP routes. The routes or prefixes that will be conditionally advertised are defined in two route maps: an advertise map and either an exist map or nonexist map. The route map associated with the exist map or nonexist map specifies the prefix that the BGP speaker will track. The route map associated with the advertise map specifies the prefix that will be advertised to the specified neighbor when the condition is met.

- If a prefix is found to be present in the exist map by the BGP speaker, the prefix specified by the advertise map is advertised.
- If a prefix is found not to be present in the nonexist map by the BGP speaker, the prefix specified by the advertise map is advertised.

If the condition is not met, the route is withdrawn and conditional advertisement does not occur. All routes that may be dynamically advertised or not advertised must exist in the BGP routing table in order for conditional advertisement to occur. These routes are referenced from an access list or an IP prefix list.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*
5. **neighbor** *ip-address* **advertise-map** *map-name* {**exist-map** *map-name* | **non-exist-map** *map-name*}
6. **exit**
7. **route-map** *map-tag* [**permit** | **deny**] [**sequence-number**]
8. **match ip address** {*access-list-number* [*access-list-number...* | *access-list-name...*] | *access-list-name* [*access-list-number...* | *access-list-name*] | **prefix-list** *prefix-list-name* [*prefix-list-name...*]}
9. **exit**
10. **route-map** *map-tag* [**permit** | **deny**] [*sequence-number*]
11. **match ip address** {*access-list-number* [*access-list-number...* | *access-list-name...*] | *access-list-name* [*access-list-number...* | *access-list-name*] | **prefix-list** *prefix-list-name* [*prefix-list-name...*]}
12. **exit**
13. **access-list** *access-list-number* {**deny** | **permit**} *source* [*source-wildcard*] [**log**]
14. **access-list** *access-list-number* {**deny** | **permit**} *source* [*source-wildcard*] [**log**]
15. **exit**

DETAILED STEPS

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **router bgp** *autonomous-system-number*

Example:

```
Device(config)# router bgp 45000
```

Enters router configuration mode for the specified routing process.

Step 4 **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*

Example:

```
Device(config-router)# neighbor 192.168.1.2 remote-as 40000
```

Adds the IP address of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local device.

Step 5 **neighbor** *ip-address* **advertise-map** *map-name* {**exist-map** *map-name* | **non-exist-map** *map-name*}

Example:

```
Device(config-router)# neighbor 192.168.1.2 advertise-map map1 exist-map map2
```

Adds the IP address of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local device.

- In this example, the prefix (172.17.0.0) matching the ACL in the advertise map (the route map named map1) will be advertised to the neighbor only when a prefix (192.168.50.0) matching the ACL in exist map (the route-map named map2) is in the local BGP table.

Step 6 **exit**

Example:

```
Device(config-router)# exit
```

Exits router configuration mode and enters global configuration mode.

Step 7 **route-map** *map-tag* [**permit** | **deny**] [**sequence-number**]

Example:

```
Device(config)# route-map map1 permit 10
```

Configures a route map and enters route map configuration mode.

- In this example, a route map named map1 is created.

Step 8 **match ip address** {*access-list-number* [*access-list-number...* | *access-list-name...*] | *access-list-name* [*access-list-number...* | *access-list-name*] | **prefix-list** *prefix-list-name* [*prefix-list-name...*]}

Example:

```
Device(config-route-map)# match ip address 1
```

Configures the route map to match a prefix that is permitted by a standard access list, an extended access list, or a prefix list.

- In this example, the route map is configured to match a prefix permitted by access list 1.

Step 9 **exit**

Example:

```
Device(config-route-map)# exit
```

Exits route map configuration mode and enters global configuration mode.

Step 10 **route-map** *map-tag* [**permit** | **deny**] [*sequence-number*]

Example:

```
Device(config)# route-map map2 permit 10
```

Configures a route map and enters route map configuration mode.

- In this example, a route map named map2 is created.

Step 11 **match ip address** {*access-list-number* [*access-list-number...* | *access-list-name...*] | *access-list-name* [*access-list-number...* | *access-list-name*] | **prefix-list** *prefix-list-name* [*prefix-list-name...*]}

Example:

```
Device(config-route-map)# match ip address 2
```

Configures the route map to match a prefix that is permitted by a standard access list, an extended access list, or a prefix list.

- In this example, the route map is configured to match a prefix permitted by access list 2.

Step 12 **exit**

Example:

```
Device(config-route-map)# exit
```

Exits route map configuration mode and enters global configuration mode.

Step 13 **access-list** *access-list-number* {**deny** | **permit**} *source* [*source-wildcard*] [**log**]

Example:

```
Device(config)# access-list 1 permit 172.17.0.0
```

Configures a standard access list.

- In this example, access list 1 permits advertising of the 172.17.0.0 prefix, depending on other conditions set by the **neighbor advertise-map** command.

Step 14 **access-list** *access-list-number* {**deny** | **permit**} *source* [*source-wildcard*] [**log**]

Example:

```
Device(config)# access-list 2 permit 192.168.50.0
```

Configures a standard access list.

- In this example, access list 2 permits the 192.168.50.0 to be the prefix of the exist-map.

Step 15 **exit**

Example:

```
Device(config)# exit
```

Exits global configuration mode and returns to privileged EXEC mode.

Originating BGP Routes

Route aggregation is useful to minimize the size of the BGP table, but there are situations when you want to add more specific prefixes to the BGP table. Route aggregation can hide more specific routes. Using the **network** command as shown in the "Configuring a BGP Routing Process" section originates routes, and the following optional tasks originate BGP routes for the BGP table for different situations.

Advertising a Default Route Using BGP

Perform this task to advertise a default route to BGP peers. The default route is locally originated. A default route can be useful to simplify configuration or to prevent the device from using too many system resources. If the device is peered with an Internet service provider (ISP), the ISP will carry full routing tables, so configuring a default route into the ISP network saves resources at the local device.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip prefix-list** *list-name* [**seq** *seq-value*] {**deny** *network / length* | **permit** *network / length*} [**ge** *ge-value*] [**le** *le-value*]
4. **route-map** *map-tag* [**permit** | **deny**] [*sequence-number*]
5. **match ip address** {*access-list-number* [*access-list-number...* | *access-list-name...*] | *access-list-name* [*access-list-number...* | *access-list-name*] | **prefix-list** *prefix-list-name* [*prefix-list-name...*]}
6. **exit**
7. **router bgp** *autonomous-system-number*
8. **neighbor** {*ip-address* | *peer-group-name*} **default-originate** [**route-map** *map-name*]
9. **end**

DETAILED STEPS

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **ip prefix-list** *list-name* [**seq** *seq-value*] {**deny** *network* / *length* | **permit** *network* / *length*} [**ge** *ge-value*] [**le** *le-value*]

Example:

```
Device(config)# ip prefix-list DEFAULT permit 10.1.1.0/24
```

Configures an IP prefix list.

- In this example, prefix list DEFAULT permits advertising of the 10.1.1.0/24. prefix depending on a match set by the **match ip address** command.

Step 4 **route-map** *map-tag* [**permit** | **deny**] [*sequence-number*]

Example:

```
Device(config)# route-map ROUTE
```

Configures a route map and enters route map configuration mode.

- In this example, a route map named ROUTE is created.

Step 5 **match ip address** {*access-list-number* [*access-list-number...* | *access-list-name...*] | *access-list-name* [*access-list-number...* | *access-list-name*] | **prefix-list** *prefix-list-name* [*prefix-list-name...*]}

Example:

```
Device(config-route-map)# match ip address prefix-list DEFAULT
```

Configures the route map to match a prefix that is permitted by a standard access list, an extended access list, or a prefix list.

- In this example, the route map is configured to match a prefix permitted by prefix list DEFAULT.

Step 6 **exit**

Example:

```
Device(config-route-map)# exit
```

Exits route map configuration mode and enters global configuration mode.

Step 7 **router bgp** *autonomous-system-number*

Example:

```
Device(config)# router bgp 40000
```

Enters router configuration mode for the specified routing process.

Step 8 **neighbor** {*ip-address* | *peer-group-name*} **default-originate** [**route-map** *map-name*]

Example:

```
Device(config-router)# neighbor 192.168.3.2 default-originate
```

(Optional) Permits a BGP speaker--the local device--to send the default route 0.0.0.0 to a peer for use as a default route.

Step 9 **end**

Example:

```
Device(config-router)# end
```

Exits router configuration mode and enters privileged EXEC mode.

Originating BGP Routes Using Backdoor Routes

Use this task to indicate to border devices which networks are reachable using a backdoor route. A backdoor network is treated the same as a local network, except that it is not advertised. For more information, see the BGP Backdoor Routes section.

Before you begin

This task assumes that the IGP (EIGRP, in this example) is already configured for the BGP peers. The configuration is done at Router B in the figure 1 in the "BGP Backdoor Routes" section, and the BGP peer is Router D.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** *ip-address* **remote-as** *autonomous-system-number*
5. **network** *ip-address* **backdoor**
6. **end**

DETAILED STEPS

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **router bgp** *autonomous-system-number*

Example:

```
Device(config)# router bgp 45000
```

Enters router configuration mode for the specified routing process.

Step 4 **neighbor** *ip-address* **remote-as** *autonomous-system-number*

Example:

```
Device(config-router)# neighbor 172.22.1.2 remote-as 45000
```

Adds the IP address of the neighbor in the specified autonomous system to the multiprotocol BGP neighbor table of the local device.

- In this example, the peer is an internal peer as the autonomous system number specified for the peer is the same number specified in Step 3.

Step 5 **network** *ip-address* **backdoor**

Example:

```
Device(config-router)# network 172.21.1.0 backdoor
```

Indicates a network that is reachable through a backdoor route.

Step 6 **end**

Example:

```
Device(config-router)# end
```

Exits router configuration mode and returns to privileged EXEC mode.

Configuring a BGP Peer Group

This task explains how to configure a BGP peer group. Often, in a BGP speaker, many neighbors are configured with the same update policies (that is, the same outbound route maps, distribute lists, filter lists, update source, and so on). Neighbors with the same update policies can be grouped into peer groups to simplify configuration and, more importantly, to make updating more efficient. When you have many peers, this approach is highly recommended.

The three steps to configure a BGP peer group, described in the following task, are as follows:

- Creating the peer group
- Assigning options to the peer group
- Making neighbors members of the peer group

You can disable a BGP peer or peer group without removing all the configuration information using the **neighbor shutdown** router configuration command.

Note By default, neighbors that are defined using the **neighbor remote-as** command in router configuration mode exchange only IPv4 unicast address prefixes. To exchange other address prefix types, such as IPv6 prefixes, neighbors must also be activated using the **neighbor activate** command in address family configuration mode for the other prefix types.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** *peer-group-name* **peer-group**
5. **neighbor** *ip-address* **remote-as** *autonomous-system-number*
6. **neighbor** *ip-address* **peer-group** *peer-group-name*
7. **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]
8. **neighbor** *peer-group-name* **activate**
9. **neighbor** *ip-address* **peer-group** *peer-group-name*
10. **end**

DETAILED STEPS

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **router bgp** *autonomous-system-number*

Example:

```
Device(config)# router bgp 40000
```

Enters router configuration mode for the specified routing process.

Step 4 **neighbor** *peer-group-name* **peer-group**

Example:

```
Device(config-router)# neighbor fingroup peer-group
```

Creates a BGP peer group.

Step 5 **neighbor** *ip-address* **remote-as** *autonomous-system-number*

Example:

```
Device(config-router)# neighbor 192.168.1.1 remote-as 45000
```

Adds the IP address of the neighbor in the specified autonomous system to the multiprotocol BGP neighbor table of the local device.

Step 6 **neighbor** *ip-address* **peer-group** *peer-group-name*

Example:

```
Device(config-router)# neighbor 192.168.1.1 peer-group fingroup
```

Assigns the IP address of a BGP neighbor to a peer group.

Step 7 **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]

Example:

```
Device(config-router)# address-family ipv4 multicast
```

Specifies the IPv4 address family and enters address family configuration mode.

- The **unicast** keyword specifies the IPv4 unicast address family. This is the default.
- The **multicast** keyword specifies that IPv4 multicast address prefixes will be exchanged.
- The **vrf** keyword and *vrf-name* argument specify that IPv4 VRF instance information will be exchanged.

Step 8 **neighbor** *peer-group-name* **activate**

Example:

```
Device(config-router-af)# neighbor fingroup activate
```

Enables the neighbor to exchange prefixes for the IPv4 address family with the local device.

Note By default, neighbors that are defined using the **neighbor remote-as** command in router configuration mode exchange only unicast address prefixes. To allow BGP to exchange other address prefix types, such as multicast that is configured in this example, neighbors must also be activated using the **neighbor activate** command.

Step 9 **neighbor** *ip-address* **peer-group** *peer-group-name*

Example:

```
Device(config-router-af)# neighbor 192.168.1.1 peer-group fingroup
```

Assigns the IP address of a BGP neighbor to a peer group.

Step 10 **end**

Example:

```
Device(config-router-af)# end
```

Exits address family configuration mode and returns to privileged EXEC mode.

Configuration Examples for BGP 4

Example: Configuring a BGP Process and Customizing Peers

The following example shows the configuration for Router B in the figure 4 above (in the "Customizing a BGP Peer" section) with a BGP process configured with two neighbor peers (at Router A and at Router E) in separate autonomous systems. IPv4 unicast routes are exchanged with both peers and IPv4 multicast routes are exchanged with the BGP peer at Router E.

Router B

```
router bgp 45000
 bgp router-id 172.17.1.99
 no bgp default ipv4-unicast
 bgp log-neighbor-changes
 timers bgp 70 120
 neighbor 192.168.1.2 remote-as 40000
 neighbor 192.168.3.2 remote-as 50000
 neighbor 192.168.3.2 description finance
 !
 address-family ipv4
  neighbor 192.168.1.2 activate
  neighbor 192.168.3.2 activate
  no auto-summary
  no synchronization
  network 172.17.1.0 mask 255.255.255.0
  exit-address-family
 !
 address-family ipv4 multicast
  neighbor 192.168.3.2 activate
  neighbor 192.168.3.2 advertisement-interval 25
  no auto-summary
  no synchronization
  network 172.17.1.0 mask 255.255.255.0
  exit-address-family
```

Examples: Removing BGP Configuration Commands Using a Redistribution Example

The following examples show first the CLI configuration to enable the redistribution of BGP routes into EIGRP using a route map and then the CLI configuration to remove the redistribution and route map. Some BGP configuration commands can affect other CLI commands and this example demonstrates how the removal of one command affects another command.

In the first configuration example, a route map is configured to match and set autonomous system numbers. BGP neighbors in three different autonomous systems are configured and activated. An EIGRP routing process is started, and the redistribution of BGP routes into EIGRP using the route map is configured.

CLI to Enable BGP Route Redistribution Into EIGRP

```
route-map bgp-to-eigrp permit 10
 match tag 50000
 set tag 65000
 exit
router bgp 45000
 bgp log-neighbor-changes
 address-family ipv4
  neighbor 172.16.1.2 remote-as 45000
  neighbor 172.21.1.2 remote-as 45000
  neighbor 192.168.1.2 remote-as 40000
  neighbor 192.168.3.2 remote-as 50000
  neighbor 172.16.1.2 activate
  neighbor 172.21.1.2 activate
  neighbor 192.168.1.2 activate
  neighbor 192.168.3.2 activate
  network 172.17.1.0 mask 255.255.255.0
  exit-address-family
 exit
router eigrp 100
 redistribute bgp 45000 metric 10000 100 255 1 1500 route-map bgp-to-eigrp
 no auto-summary
 exit
```

In the second configuration example, both the **route-map** command and the **redistribute** command are disabled. If only the route-map command is removed, it does not automatically disable the redistribution. The redistribution will now occur without any matching or filtering. To remove the redistribution configuration, the **redistribute** command must also be disabled.

CLI to Remove BGP Route Redistribution Into EIGRP

```
configure terminal
 no route-map bgp-to-eigrp
 router eigrp 100
  no redistribute bgp 45000
  end
```

Examples: BGP Soft Reset

The following examples show two ways to reset the connection for BGP peer 192.168.1.1.

Example: Dynamic Inbound Soft Reset

The following example shows the command used to initiate a dynamic soft reconfiguration in the BGP peer 192.168.1.1. This command requires that the peer support the route refresh capability.

```
clear ip bgp 192.168.1.1 soft in
```

Example: Inbound Soft Reset Using Stored Information

The following example shows how to enable inbound soft reconfiguration for the neighbor 192.168.1.1. All the updates received from this neighbor will be stored unmodified, regardless of the inbound policy. When inbound soft reconfiguration is performed later, the stored information will be used to generate a new set of inbound updates.

```
router bgp 100
 neighbor 192.168.1.1 remote-as 200
 neighbor 192.168.1.1 soft-reconfiguration inbound
```

The following example clears the session with the neighbor 192.168.1.1:

```
clear ip bgp 192.168.1.1 soft in
```

Example: Resetting and Displaying Basic BGP Information

The following example shows how to reset and display basic BGP information.

The **clear ip bgp *** command clears and resets all the BGP neighbor sessions. Specific neighbors or all peers in an autonomous system can be cleared by using the *neighbor-address* and *autonomous-system-number* arguments. If no argument is specified, this command will clear and reset all BGP neighbor sessions.

Note The **clear ip bgp *** command also clears all the internal BGP structures, which makes it useful as a troubleshooting tool.

```
Device# clear ip bgp *
```

The **show ip bgp** command is used to display all the entries in the BGP routing table. The following example displays BGP routing table information for the 10.1.1.0 network:

```
Device# show ip bgp 10.1.1.0 255.255.255.0

BGP routing table entry for 10.1.1.0/24, version 2
Paths: (1 available, best #1, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1
  40000
    192.168.1.2 from 192.168.1.2 (10.1.1.99)
      Origin IGP, metric 0, localpref 100, valid, external, best
```

The **show ip bgp neighbors** command is used to display information about the TCP and BGP connections to neighbors. The following example displays the routes that were advertised from Router B in the figure above (in the “Configuring a BGP Peer for the IPv4 VRF Address Family” section) to its BGP neighbor 192.168.3.2 on Router E:

```
Device# show ip bgp neighbors 192.168.3.2 advertised-routes

BGP table version is 3, local router ID is 172.17.1.99
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network          Next Hop            Metric LocPrf Weight Path
*> 10.1.1.0/24      192.168.1.2              0             0 40000 i
*> 172.17.1.0/24    0.0.0.0                  0         32768 i
Total number of prefixes 2
```

The **show ip bgp paths** command is used to display all the BGP paths in the database. The following example displays BGP path information for Router B in the figure above (in the “Customizing a BGP Peer” section):

```
Device# show ip bgp paths

Address    Hash Refcount Metric Path
0x2FB5DB0     0        5      0 i
0x2FB5C90     1        4      0 i
0x2FB5C00  1361        2      0 50000 i
0x2FB5D20  2625        2      0 40000 i
```

The **show ip bgp summary** command is used to display the status of all BGP connections. The following example displays BGP routing table information for Router B in the figure above (in the "Customizing a BGP Peer" section):

```
Device# show ip bgp summary

BGP router identifier 172.17.1.99, local AS number 45000
BGP table version is 3, main routing table version 3
2 network entries using 234 bytes of memory
2 path entries using 104 bytes of memory
4/2 BGP path/bestpath attribute entries using 496 bytes of memory
2 BGP AS-PATH entries using 48 bytes of memory
0 BGP route-map cache entries using 0 bytes of memory
0 BGP filter-list cache entries using 0 bytes of memory
BGP using 882 total bytes of memory
BGP activity 14/10 prefixes, 16/12 paths, scan interval 60 secs
Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
192.168.1.2     4 40000     667     672        3    0    0 00:03:49        1
192.168.3.2     4 50000     468     467        0    0    0 00:03:49 (NoNeg)
```

Examples: Aggregating Prefixes Using BGP

The following examples show how you can use aggregate routes in BGP either by redistributing an aggregate route into BGP or by using the BGP conditional aggregation routing feature.

In the following example, the **redistribute static** router configuration command is used to redistribute aggregate route 10.0.0.0:

```
ip route 10.1.0.0 255.0.0.0 null 0
!
router bgp 100
 redistribute static
```

The following configuration shows how to create an aggregate entry in the BGP routing table when at least one specific route falls into the specified range. The aggregate route will be advertised as coming from your autonomous system and has the atomic aggregate attribute set to show that information might be missing. (By default, atomic aggregate is set unless you use the **as-set** keyword in the **aggregate-address** router configuration command.)

```
router bgp 100
 aggregate-address 10.1.0.0 255.0.0.0
```

The following example shows how to create an aggregate entry using the same rules as in the previous example, but the path advertised for this route will be an AS_SET consisting of all elements contained in all paths that are being summarized:

```
router bgp 100
 aggregate-address 10.1.0.0 255.0.0.0 as-set
```

The following example shows how to create the aggregate route for 10.0.0.0 and also suppress advertisements of more specific routes to all neighbors:

```
router bgp 100
 aggregate-address 10.1.0.0 255.0.0.0 summary-only
```

The following example configures BGP to not advertise inactive routes:

```
Device(config)# router bgp 50000
Device(config-router)# address-family ipv4 unicast
Device(config-router-af)# bgp suppress-inactive
Device(config-router-af)# end
```

The following example configures a maximum route limit in the VRF named RED and configures BGP to not advertise inactive routes through the VRF named RED:

```
Device(config)# ip vrf RED
Device(config-vrf)# rd 50000:10
Device(config-vrf)# maximum routes 1000 10
Device(config-vrf)# exit
Device(config)# router bgp 50000
Device(config-router)# address-family ipv4 vrf RED
Device(config-router-af)# bgp suppress-inactive
Device(config-router-af)# end
```

Example: Configuring a BGP Peer Group

The following example shows how to use an address family to configure a peer group so that all members of the peer group are both unicast- and multicast-capable:

```
router bgp 45000
 neighbor 192.168.1.2 remote-as 40000
 neighbor 192.168.3.2 remote-as 50000
 address-family ipv4 unicast
  neighbor mygroup peer-group
  neighbor 192.168.1.2 peer-group mygroup
  neighbor 192.168.3.2 peer-group mygroup
router bgp 45000
 neighbor 192.168.1.2 remote-as 40000
 neighbor 192.168.3.2 remote-as 50000
 address-family ipv4 multicast
  neighbor mygroup peer-group
  neighbor 192.168.1.2 peer-group mygroup
  neighbor 192.168.3.2 peer-group mygroup
  neighbor 192.168.1.2 activate
  neighbor 192.168.3.2 activate
```