Connecting to a Service Provider Using External BGP

This module describes configuration tasks that will enable your Border Gateway Protocol (BGP) network to access peer devices in external networks such as those from Internet service providers (ISPs). BGP is an interdomain routing protocol that is designed to provide loop-free routing between organizations. External BGP (eBGP) peering sessions are configured to allow peers from different autonomous systems to exchange routing updates. Tasks to help manage the traffic that is flowing inbound and outbound are described, as are tasks to configure BGP policies to filter the traffic. Multihoming techniques that provide redundancy for connections to a service provider are also described.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Connecting to a Service Provider Using External BGP

- Before connecting to a service provider you need to understand how to configure the basic BGP process and peers. See the "Cisco BGP Overview" and "Configuring a Basic BGP Network" modules for more details.

- The tasks and concepts in this chapter will help you configure BGP features that would be useful if you are connecting your network to a service provider. For each connection to the Internet, you must have an assigned autonomous system number from the Internet Assigned Numbers Authority (IANA).

Restrictions for Connecting to a Service Provider Using External BGP

- A router that runs Cisco IOS 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 address family configurations.

- Policy lists are not supported in versions of Cisco IOS software prior to Cisco IOS Release 12.0(22)S and 12.2(15)T. Reloading a router that is running an older version of Cisco IOS software may cause some routing policy configurations to be lost.

Information About Connecting to a Service Provider Using External BGP

External BGP Peering

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 and it uses TCP (port 179) as the transport protocol. The destination TCP port is assigned 179, and the local port is assigned a random port number. Cisco IOS software supports BGP version 4, which has been used by 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.

External BGP peering sessions are configured to allow BGP peers from different autonomous systems to exchange routing updates. By design, a BGP routing process expects eBGP peers to be directly connected, for example, over a WAN connection. However, there are many real-world scenarios where this rule would prevent routing from occurring. Peering sessions for multihop neighbors are configured with the **neighbor ebgp-multihop** command. The figure below shows simple eBGP peering between three routers. Router B peers with Router A and Router E. In the figure below, the **neighbor ebgp-multihop** command could be used to establish peering between Router A and Router E although this is a very simple network design. BGP forwards information about the next hop in the network using the NEXT_HOP attribute, which is set to the IP address of the interface that advertises a route in an eBGP peering session by default. The source interface can be a physical interface or a loopback interface.

Figure 1: BGP Peers in Different Autonomous Systems
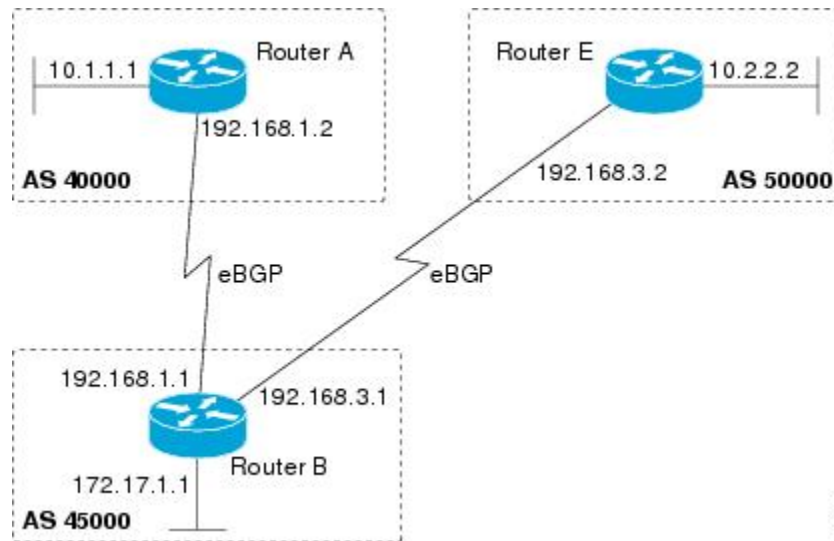
Loopback interfaces are preferred for establishing eBGP peering sessions because loopback interfaces are less susceptible to interface flapping. Interfaces on networking devices can fail, and they can also be taken out of service for maintenance. When an interface is administratively brought up or down, due to failure or maintenance, it is referred to as a flap. Loopback interfaces provide a stable source interface to ensure that the IP address assigned to the interface is always reachable as long as the IP routing protocols continue to advertise the subnet assigned to the loopback interface. Loopback interfaces allow you to conserve address space by configuring a single address with /32 bit mask. Before a loopback interface is configured for an eBGP peering session, you must configure the **neighbor update-source** command and specify the loopback interface. With this configuration, the loopback interface becomes the source interface and its IP address is advertised as the next hop for routes that are advertised through this loopback. If loopback interfaces are used to connect single-hop eBGP peers, you must configure the **neighbor disable-connected-check** command before you can establish the eBGP peering session.

Connecting to external networks enables traffic from your network to be forwarded to other networks and across the Internet. Traffic will also be flowing into, and possibly through, your network. BGP contains various techniques to influence how the traffic flows into and out of your network, and to create BGP policies that filter the traffic, inbound and outbound. To influence the traffic flow, BGP uses certain BGP attributes that can be included in update messages or used by the BGP routing algorithm. BGP policies to filter traffic also use some of the BGP attributes with route maps, access lists including AS-path access lists, filter lists, policy lists, and distribute lists. Managing your external connections may involve multihoming techniques where there is more than one connection to an ISP or connections to more than one ISP for backup or performance purposes. Tagging BGP routes with different community attributes across autonomous system or physical boundaries can prevent the need to configure long lists of individual permit or deny statements.

BGP Autonomous System Number Formats

Prior to January 2009, BGP autonomous system numbers that were allocated to companies were 2-octet numbers in the range from 1 to 65535 as described in RFC 4271, *A Border Gateway Protocol 4 (BGP-4)*. Due to increased demand for autonomous system numbers, the Internet Assigned Number Authority (IANA) will start in January 2009 to allocate four-octet autonomous system numbers in the range from 65536 to 4294967295. RFC 5396, *Textual Representation of Autonomous System (AS) Numbers* , documents three methods of representing autonomous system numbers. Cisco has implemented the following two methods:

- Asplain--Decimal value notation where both 2-byte and 4-byte autonomous system numbers are represented by their decimal value. For example, 65526 is a 2-byte autonomous system number and 234567 is a 4-byte autonomous system number.

- Asdot--Autonomous system dot notation where 2-byte autonomous system numbers are represented by their decimal value and 4-byte autonomous system numbers are represented by a dot notation. For example, 65526 is a 2-byte autonomous system number and 1.169031 is a 4-byte autonomous system number (this is dot notation for the 234567 decimal number).

For details about the third method of representing autonomous system numbers, see RFC 5396.

Asdot Only Autonomous System Number Formatting

In Cisco IOS Release 12.0(32)S12, 12.4(24)T, and later releases, the 4-octet (4-byte) autonomous system numbers are entered and displayed only in asdot notation, for example, 1.10 or 45000.64000. When using regular expressions to match 4-byte autonomous system numbers the asdot format includes a period which is a special character in regular expressions. A backslash must be entered before the period for example, 1\.14, to ensure the regular expression match does not fail. The table below shows the format in which 2-byte and 4-byte autonomous system numbers are configured, matched in regular expressions, and displayed in **show** command output in Cisco IOS images where only asdot formatting is available.

Table 1: Asdot Only 4-Byte Autonomous System Number Format

Format	Configuration Format	Show Command Output and Regular Expression Match Format
asdot	2-byte: 1 to 65535 4-byte: 1.0 to 65535.65535	2-byte: 1 to 65535 4-byte: 1.0 to 65535.65535

Asplain as Default Autonomous System Number Formatting

In Cisco IOS Release 12.0(32)SY8, 12.0(33)S3, 12.2(33)SRE, 12.2(33)XNE, 12.2(33)SXI1, and later releases, the Cisco implementation of 4-byte autonomous system numbers uses asplain as the default display format for autonomous system numbers, but you can configure 4-byte autonomous system numbers in both the asplain and asdot format. In addition, the default format for matching 4-byte autonomous system numbers in regular expressions is asplain, so you must ensure that any regular expressions to match 4-byte autonomous system numbers are written in the asplain format. If you want to change the default **show** command output to display 4-byte autonomous system numbers in the asdot format, use the **bgp asnotation dot** command under router configuration mode. When the asdot format is enabled as the default, any regular expressions to match 4-byte autonomous system numbers must be written using the asdot format, or the regular expression match will fail. The tables below show that although you can configure 4-byte autonomous system numbers in either asplain or asdot format, only one format is used to display **show** command output and control 4-byte autonomous system number matching for regular expressions, and the default is asplain format. To display 4-byte autonomous system numbers in **show** command output and to control matching for regular expressions in the asdot format, you must configure the **bgp asnotation dot** command. After enabling the **bgp asnotation dot** command, a hard reset must be initiated for all BGP sessions by entering the **clear ip bgp *** command.

Note If you are upgrading to an image that supports 4-byte autonomous system numbers, you can still use 2-byte autonomous system numbers. The **show** command output and regular expression match are not changed and remain in asplain (decimal value) format for 2-byte autonomous system numbers regardless of the format configured for 4-byte autonomous system numbers.

Table 2: Default Asplain 4-Byte Autonomous System Number Format

Format	Configuration Format	Show Command Output and Regular Expression Match Format
asplain	2-byte: 1 to 65535 4-byte: 65536 to 4294967295	2-byte: 1 to 65535 4-byte: 65536 to 4294967295
asdot	2-byte: 1 to 65535 4-byte: 1.0 to 65535.65535	2-byte: 1 to 65535 4-byte: 65536 to 4294967295

Table 3: Asdot 4-Byte Autonomous System Number Format

Format	Configuration Format	Show Command Output and Regular Expression Match Format
asplain	2-byte: 1 to 65535 4-byte: 65536 to 4294967295	2-byte: 1 to 65535 4-byte: 1.0 to 65535.65535
asdot	2-byte: 1 to 65535 4-byte: 1.0 to 65535.65535	2-byte: 1 to 65535 4-byte: 1.0 to 65535.65535

Reserved and Private Autonomous System Numbers

In Cisco IOS Release 12.0(32)S12, 12.0(32)SY8, 12.0(33)S3, 12.2(33)SRE, 12.2(33)XNE, 12.2(33)SXI1, 12.4(24)T, and later releases, the Cisco implementation of BGP supports RFC 4893. RFC 4893 was developed to allow BGP to support a gradual transition from 2-byte autonomous system numbers to 4-byte autonomous system numbers. A new reserved (private) autonomous system number, 23456, was created by RFC 4893 and this number cannot be configured as an autonomous system number in the Cisco IOS CLI.

RFC 5398, *Autonomous System (AS) Number Reservation for Documentation Use*, describes new reserved autonomous system numbers for documentation purposes. Use of the reserved numbers allow configuration examples to be accurately documented and avoids conflict with production networks if these configurations are literally copied. The reserved numbers are documented in the IANA autonomous system number registry. Reserved 2-byte autonomous system numbers are in the contiguous block, 64496 to 64511 and reserved 4-byte autonomous system numbers are from 65536 to 65551 inclusive.

Private 2-byte autonomous system numbers are still valid in the range from 64512 to 65534 with 65535 being reserved for special use. Private autonomous system numbers can be used for internal routing domains but must be translated for traffic that is routed out to the Internet. BGP should not be configured to advertise private autonomous system numbers to external networks. Cisco IOS software does not remove private autonomous system numbers from routing updates by default. We recommend that ISPs filter private autonomous system numbers.

Note   Autonomous system number assignment for public and private networks is governed by the IANA. For information about autonomous-system numbers, including reserved number assignment, or to apply to register an autonomous system number, see the following URL: http://www.iana.org/.

BGP Attributes

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 various attributes that are used in BGP best-path analysis. Cisco IOS software provides the ability to influence BGP path selection by altering these attributes via the command-line interface (CLI). BGP path selection can also be influenced through standard BGP policy configuration.

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, then the oldest paths are selected as multipaths.

BGP can include path attribute information in update messages. BGP attributes describe the characteristic of the route, and the software uses these attributes to help make decisions about which routes to advertise. Some of this attribute information can be configured at a BGP-speaking networking device. There are some mandatory attributes that are always included in the update message and some discretionary attributes. The following BGP attributes can be configured:

- AS_Path

- Community

- Local_Pref

- Multi_Exit_Discriminator (MED)

- Next_Hop

- Origin

AS_Path

This attribute contains a list or set of the autonomous system numbers through which routing information has passed. The BGP speaker adds its own autonomous system number to the list when it forwards the update message to external peers.

Community

BGP communities are used to group networking devices that share common properties, regardless of network, autonomous system, or any physical boundaries. In large networks applying a common routing policy through prefix lists or access lists requires individual peer statements on each networking device. Using the BGP community attribute BGP neighbors, with common routing policies, can implement inbound or outbound route filters based on the community tag rather than consult large lists of individual permit or deny statements.

Local_Pref

Within an autonomous system, the Local_Pref attribute is included in all update messages between BGP peers. If there are several paths to the same destination, the local preference attribute with the highest value indicates the preferred outbound path from the local autonomous system. The highest ranking route is advertised to internal peers. The Local_Pref value is not forwarded to external peers.

Multi_Exit_Discriminator

The MED attribute indicates (to an external peer) a preferred path into an autonomous system. If there are multiple entry points into an autonomous system, the MED can be used to influence another autonomous system to choose one particular entry point. A metric is assigned where a lower MED metric is preferred by the software over a higher MED metric. The MED metric is exchanged between autonomous systems, but after a MED is forwarded into an autonomous system, the MED metric is reset to the default value of 0. When an update is sent to an internal BGP (iBGP) peer, the MED is passed along without any change, allowing all the peers in the same autonomous system to make a consistent path selection.

By default, a router will compare the MED attribute for paths only from BGP peers that reside in the same autonomous system. The **bgp always-compare-med** command can be configured to allow the router to compare metrics from peers in different autonomous systems.

Note

The Internet Engineering Task Force (IETF) decision regarding BGP MED assigns a value of infinity to the missing MED, making the route that lacks the MED variable the least preferred. The default behavior of BGP routers that run Cisco software is to treat routes without the MED attribute as having a MED of 0, making the route that lacks the MED variable the most preferred. To configure the router to conform to the IETF standard, use the **bgp bestpath med missing-as-worst** router configuration command.

Next_Hop

The Next_Hop attribute identifies the next-hop IP address to be used as the BGP next hop to the destination. The router makes a recursive lookup to find the BGP next hop in the routing table. In external BGP (eBGP), the next hop is the IP address of the peer that sent the update. Internal BGP (iBGP) sets the next-hop address to the IP address of the peer that advertised the prefix for routes that originate internally. When any routes to iBGP that are learned from eBGP are advertised, the Next_Hop attribute is unchanged.

A BGP next-hop IP address must be reachable in order for the router to use a BGP route. Reachability information is usually provided by the IGP, and changes in the IGP can influence the forwarding of the next-hop address over a network backbone.

Origin

This attribute indicates how the route was included in a BGP routing table. In Cisco software, a route defined using the BGP **network** command is given an origin code of Interior Gateway Protocol (IGP). Routes distributed from an Exterior Gateway Protocol (EGP) are coded with an origin of EGP, and routes redistributed from other protocols are defined as Incomplete. BGP decision policy for origin prefers IGP over EGP, and then EGP over Incomplete.

Multihoming

Multihoming is defined as connecting an autonomous system with more than one service provider. If you have any reliability issues with one service provider, then you have a backup connection. Performance issues can also be addressed by multihoming because better paths to the destination network can be utilized.

Unless you are a service provider, you must plan your routing configuration carefully to avoid Internet traffic traveling through your autonomous system and consuming all your bandwidth. The figure below shows that autonomous system 45000 is multihomed to autonomous system 40000 and autonomous system 50000. Assuming autonomous system 45000 is not a service provider, then several techniques such as load balancing or some form of routing policy must be configured to allow traffic from autonomous system 45000 to reach either autonomous system 40000 or autonomous system 50000 but not allow much, if any, transit traffic.

Figure 2: Multihoming Topology
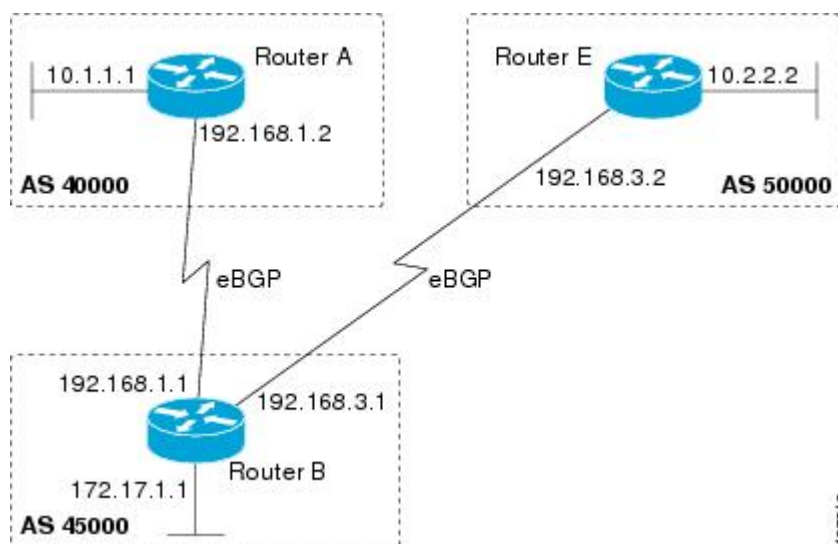
MED Attribute

Configuring the MED attribute is another method that BGP can use to influence the choice of paths into another autonomous system. The MED attribute indicates (to an external peer) a preferred path into an autonomous system. If there are multiple entry points into an autonomous system, the MED can be used to influence another autonomous system to choose one particular entry point. A metric is assigned using route maps where a lower MED metric is preferred by the software over a higher MED metric.

Transit Versus Nontransit Traffic

Most of the traffic within an autonomous system contains a source or destination IP address residing within the autonomous system, and this traffic is referred to as nontransit (or local) traffic. Other traffic is defined as transit traffic. As traffic across the Internet increases, controlling transit traffic becomes more important.

A service provider is considered to be a transit autonomous system and must provide connectivity to all other transit providers. In reality, few service providers actually have enough bandwidth to allow all transit traffic, and most service providers have to purchase such connectivity from Tier 1 service providers.

An autonomous system that does not usually allow transit traffic is called a stub autonomous system and will link to the Internet through one service provider.

BGP Policy Configuration

BGP policy configuration is used to control prefix processing by the BGP routing process and to filter routes from inbound and outbound advertisements. Prefix processing can be controlled by adjusting BGP timers, altering how BGP handles path attributes, limiting the number of prefixes that the routing process will accept, and configuring BGP prefix dampening. Prefixes in inbound and outbound advertisements are filtered using route maps, filter lists, IP prefix lists, autonomous-system-path access lists, IP policy lists, and distribute lists. The table below shows the processing order of BGP policy filters.

Table 4: BGP Policy Processing Order

Inbound	Outbound
Route map	Distribute list
Filter list, AS-path access list, or IP policy	IP prefix list
IP prefix list	Filter list, AS-path access list, or IP policy
Distribute list	Route map

Note   In Cisco IOS Releases 12.0(22)S, 12.2(15)T, 12.2(18)S, and later releases, the maximum number of autonomous system access lists that can be configured with the **ip as-path access-list** command is increased from 199 to 500.

Whenever there is a change in the routing policy due to a configuration change, BGP peering sessions must be reset using the **clear ip bgp** command. Cisco IOS 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 reset uses stored update information, at the cost of additional memory for storing the updates, to allow you to apply a new BGP policy without disrupting the network. Soft reset can be configured for inbound or outbound sessions.

- Dynamic inbound soft reset--The route refresh capability, as defined in RFC 2918, allows the local router 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 routers must support the route refresh capability.

To determine if a BGP router supports this capability, use the **show ip bgp neighbors**command. The following message is displayed in the output when the router supports the route refresh capability:

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

BGP COMMUNITIES Attribute

A BGP community is a group of routes that share a common property, regardless of their network, autonomous system, or any physical boundaries. In large networks, applying a common routing policy by using prefix lists or access lists requires individual peer statements on each networking device. Using the BGP COMMUNITIES attribute, BGP speakers with common routing policies can implement inbound or outbound route filters based on the community tag, rather than consult long lists of individual permit or deny statements. A COMMUNITIES attribute can contain multiple communities.

A route can belong to multiple communities. The network administrator defines the communities to which a route belongs. By default, all routes belong to the general Internet community.

In addition to numbered communities, there are several predefined (well-known) communities:

- no-export—Do not advertise this route to external BGP peers.

- no-advertise—Do not advertise this route to any peer.

- internet—Advertise this route to the Internet community. All BGP-speaking networking devices belong to this community.

- local-as—Do not send this route outside the local autonomous system.

- gshut—Community of routes gracefully shut down.

The COMMUNITIES attribute is optional, which means that it will not be passed on by networking devices that do not understand communities. Networking devices that understand communities must be configured to handle the communities or else the COMMUNITIES attribute will be discarded. By default, no COMMUNITIES attribute is sent to a neighbor. In order for a COMMUNITIES attribute to be sent to a neighbor, use the **neighbor send-community** command.

Extended Communities

Extended community attributes are used to configure, filter, and identify routes for virtual routing and forwarding (VRF) instances and Multiprotocol Label Switching (MPLS) Virtual Private Networks (VPNs). All of the standard rules of access lists apply to the configuration of extended community lists. Regular expressions are supported by the expanded range of extended community list numbers. All regular expression configuration options are supported. The route target (RT) and site of origin (SoO) extended community attributes are supported by the standard range of extended community lists.

Route Target Extended Community Attribute

The RT extended community attribute is configured with the **rt** keyword of the **ip extcommunity-list** command. This attribute is used to identify a set of sites and VRFs that may receive routes that are tagged with the configured route target. Configuring the route target extended community attribute with a route allows that route to be placed in the per-site forwarding tables that are used for routing traffic that is received from corresponding sites.

Site of Origin Extended Community Attribute

The SoO extended community attribute is configured with the **soo** keyword of the **ip extcommunity-list** command. This attribute uniquely identifies the site from which the provider edge (PE) router learned the route. All routes learned from a particular site must be assigned the same SoO extended community attribute, regardless if a site is connected to a single PE router or multiple PE routers. Configuring this attribute prevents

routing loops from occurring when a site is multihomed. The SoO extended community attribute is configured on the interface and is propagated into BGP through redistribution. The SoO extended community attribute can be applied to routes that are learned from VRFs. The SoO extended community attribute should not be configured for stub sites or sites that are not multihomed.

IP Extended Community-List Configuration Mode

Named and numbered extended community lists can be configured in IP extended community-list configuration mode. The IP extended community-list configuration mode supports all of the functions that are available in global configuration mode. In addition, the following operations can be performed:

- Configure sequence numbers for extended community list entries.
- Resequence existing sequence numbers for extended community list entries.
- Configure an extended community list to use default values.

Default Sequence Numbering

Extended community list entries start with the number 10 and increment by 10 for each subsequent entry when no sequence number is specified, when default behavior is configured, and when an extended community list is resequenced without specifying the first entry number or the increment range for subsequent entries.

Resequencing Extended Community Lists

Extended community-list entries are sequenced and resequenced on a per-extended community list basis. The **resequence** command can be used without any arguments to set all entries in a list to default sequence numbering. The **resequence** command also allows the sequence number of the first entry and increment range to be set for each subsequent entry. The range of configurable sequence numbers is from 1 to 2147483647.

Extended Community Lists

Extended community attributes are used to configure, filter, and identify routes for VRF instances and MPLS VPNs. The **ip extcommunity-list** command is used to configure named or numbered extended community lists. All of the standard rules of access lists apply to the configuration of extended community lists. Regular expressions are supported by the expanded range of extended community list numbers.

Administrative Distance

Administrative distance is a measure of the preference of different routing protocols. BGP has a **distance bgp** command that allows you to set different administrative distances for three route types: external, internal, and local. BGP, like other protocols, prefers the route with the lowest administrative distance.

BGP Route Map Policy Lists

BGP route map policy lists allow a network operator to group route map match clauses into named lists called policy lists. A policy list functions like a macro. When a policy list is referenced in a route map, all of the match clauses are evaluated and processed as if they had been configured directly in the route map. This enhancement simplifies the configuration of BGP routing policy in medium-size and large networks because a network operator can preconfigure policy lists with groups of match clauses and then reference these policy

lists within different route maps. The network operator no longer needs to manually reconfigure each recurring group of match clauses that occur in multiple route map entries.

A policy lists functions like a macro when it is configured in a route map and has the following capabilities and characteristics:

- When a policy list is referenced within a route map, all the match statements within the policy list are evaluated and processed.

- Two or more policy lists can be configured with a route map. Policy lists can be configured within a route map to be evaluated with AND or OR semantics.

- Policy lists can coexist with any other preexisting match and set statements that are configured within the same route map but outside of the policy lists.

- When multiple policy lists perform matching within a route map entry, all policy lists match on the incoming attribute only.

Policy lists support only match clauses and do not support set clauses. Policy lists can be configured for all applications of route maps, including redistribution, and can also coexist, within the same route map entry, with match and set clauses that are configured separately from the policy lists.

Note Policy lists are supported only by BGP and are not supported by other IP routing protocols.

How to Connect to a Service Provider Using External BGP

Influencing Inbound Path Selection

BGP can be used to influence the choice of paths in another autonomous system. There may be several reasons for wanting BGP to choose a path that is not the obvious best route, for example, to avoid some types of transit traffic passing through an autonomous system or perhaps to avoid a very slow or congested link. BGP can influence inbound path selection using one of the following BGP attributes:

- AS-path

- Multi-Exit Discriminator (MED)

Perform one of the following tasks to influence inbound path selection:

Influencing Inbound Path Selection by Modifying the AS_PATH Attribute

Perform this task to influence the inbound path selection for traffic destined for the 172.17.1.0 network by modifying the AS_PATH attribute. The configuration is performed at Router A in the figure below. For a configuration example of this task using 4-byte autonomous system numbers in asplain format, see the "Example: Influencing Inbound Path Selection by Modifying the AS_PATH Attribute Using 4-Byte AS Numbers".

One of the methods that BGP can use to influence the choice of paths in another autonomous system is to modify the AS_PATH attribute. For example, in the figure below, Router A advertises its own network, 172.17.1.0, to its BGP peers in autonomous system 45000 and autonomous system 60000. When the routing

information is propagated to autonomous system 50000, the routers in autonomous system 50000 have network reachability information about network 172.17.1.0 from two different routes. The first route is from autonomous system 45000 with an AS_PATH consisting of 45000, 40000, the second route is through autonomous system 55000 with an AS-path of 55000, 60000, 40000. If all other BGP attribute values are the same, Router C in autonomous system 50000 would choose the route through autonomous system 45000 for traffic destined for network 172.17.1.0 because it is the shortest route in terms of autonomous systems traversed.

Autonomous system 40000 now receives all traffic from autonomous system 50000 for the 172.17.1.0 network through autonomous system 45000. If, however, the link between autonomous system 45000 and autonomous system 40000 is a really slow and congested link, the **set as-path prepend** command can be used at Router A to influence inbound path selection for the 172.17.1.0 network by making the route through autonomous system 45000 appear to be longer than the path through autonomous system 60000. The configuration is done at Router A in the figure below by applying a route map to the outbound BGP updates to Router B. Using the **set as-path prepend** command, all the outbound BGP updates from Router A to Router B will have their AS_PATH attribute modified to add the local autonomous system number 40000 twice. After the configuration, autonomous system 50000 receives updates about the 172.17.1.0 network through autonomous system 45000. The new AS_PATH is 45000, 40000, 40000, and 40000, which is now longer than the AS-path from autonomous system 55000 (unchanged at a value of 55000, 60000, 40000). Networking devices in autonomous system 50000 will now prefer the route through autonomous system 55000 to forward packets with a destination address in the 172.17.1.0 network.

Figure 3: Network Topology for Modifying the AS_PATH Attribute
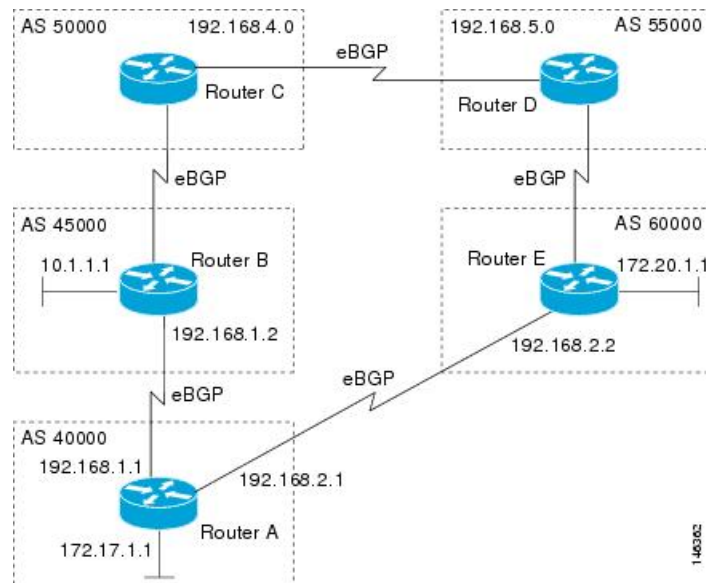
SUMMARY STEPS

1. **enable**
2. **configure   terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*
5. **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]
6. **network** *network-number* [**mask** *network-mask*] [**route-map** *route-map-name*]
7. **neighbor** {*ip-address* | *peer-group-name*} **activate**

8.   **neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**}
9.   **exit-address-family**
10. **exit**
11. **route-map** *map-name* [**permit** | **deny**] [*sequence-number*]
12. **set as-path** {**tag** | **prepend** *as-path-string*}
13. **end**
14. **show running-config**

DETAILED STEPS

	Command or Action	**Purpose**
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*	*peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 192.168.1.2 remote-as 45000`	Adds the IP address or peer group name of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local router. • In this example, the BGP peer on Router B at 192.168.1.2 is added to the IPv4 multiprotocol BGP neighbor table and will receive BGP updates.
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 router is placed in address family 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.

	Command or Action	**Purpose**
Step 6	**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`	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 7	**neighbor** {*ip-address* | *peer-group-name*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.1.2 activate`	Enables address exchange for address family IPv4 unicast for the BGP neighbor at 192.168.1.2 on Router B.
Step 8	**neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**} **Example:** `Device(config-router-af)# neighbor 192.168.1.2 route-map PREPEND out`	Applies a route map to incoming or outgoing routes. • In this example, the route map named PREPEND is applied to outbound routes to Router B.
Step 9	**exit-address-family** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode and enters router configuration mode.
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 PREPEND permit 10`	Configures a route map and enters route map configuration mode. • In this example, a route map named PREPEND is created with a permit clause.
Step 12	**set as-path** {**tag** | **prepend** *as-path-string*} **Example:** `Device(config-route-map)# set as-path prepend 40000 40000`	Modifies an autonomous system path for BGP routes. • Use the **prepend** keyword to prepend an arbitrary autonomous system path string to BGP routes. Usually the local autonomous system number is prepended multiple times, increasing the autonomous system path length. • In this example, two additional autonomous system entries are added to the autonomous system path for outbound routes to Router B.

	Command or Action	**Purpose**
Step 13	**end** **Example:** `Device(config-route-map)# end`	Exits route map configuration mode and returns to privileged EXEC mode.
Step 14	**show running-config** **Example:** `Device# show running-config`	Displays the running configuration file.

Examples

Router A

The following partial output of the **show running-config** command shows the configuration from this task.

```
Device# show running-config
.
.
.
router bgp 40000
 neighbor 192.168.1.2 remote-as 45000
 !
 address-family ipv4
 neighbor 192.168.1.2 activate
 neighbor 192.168.1.2 route-map PREPEND out
 no auto-summary
 no synchronization
 network 172.17.1.0 mask 255.255.255.0
 exit-address-family
!
route-map PREPEND permit 10
 set as-path prepend 40000 40000
.
.
.
```

Influencing Inbound Path Selection by Setting the MED Attribute

One of the methods that BGP can use to influence the choice of paths into another autonomous system is to set the Multi-Exit Discriminator (MED) attribute. The MED attribute indicates (to an external peer) a preferred path to an autonomous system. If there are multiple entry points to an autonomous system, the MED can be used to influence another autonomous system to choose one particular entry point. A metric is assigned using route maps where a lower MED metric is preferred by the software over a higher MED metric.

Perform this task to influence inbound path selection by setting the MED metric attribute. The configuration is performed at Router B and Router D in the figure below. Router B advertises the network 172.16.1.0. to its BGP peer, Router E in autonomous system 50000. Using a simple route map Router B sets the MED metric to 50 for outbound updates. The task is repeated at Router D but the MED metric is set to 120. When Router E receives the updates from both Router B and Router D the MED metric is stored in the BGP routing table.

Before forwarding packets to network 172.16.1.0, Router E compares the attributes from peers in the same autonomous system (both Router B and Router D are in autonomous system 45000). The MED metric for Router B is less than the MED for Router D, so Router E will forward the packets through Router B.

Figure 4: Network Topology for Setting the MED Attribute
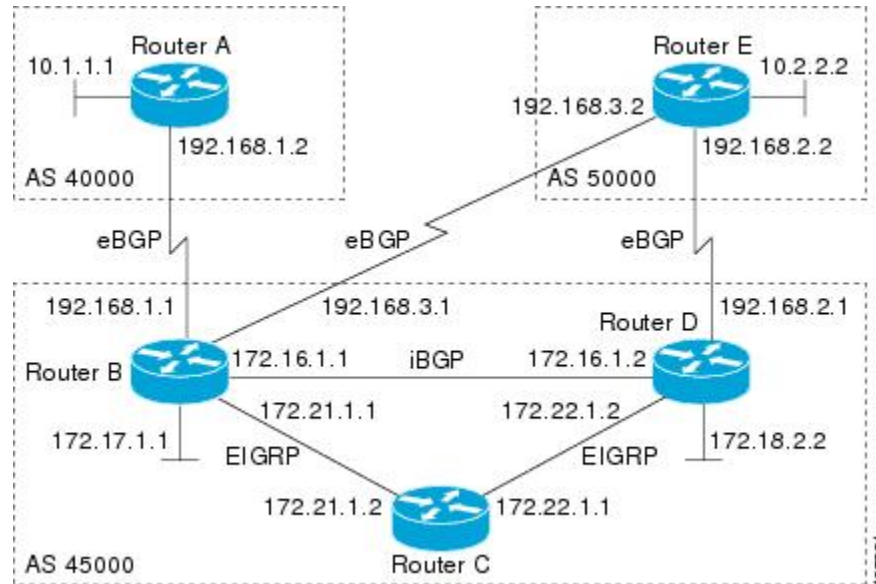
Use the **bgp always-compare-med** command to compare MED attributes from peers in other autonomous systems.

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. **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]
6. **network**   *network-number*   [**mask** *network-mask*] [**route-map** *route-map-name*]
7. **neighbor**   {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**}
8. **exit**
9. **exit**
10. **route-map**   *map-name* [**permit** | **deny**] [*sequence-number*]
11. **set metric**   *value*
12. **end**
13. Repeat Step 1 through Step 12 at Router D.
14. **show ip bgp** [*network*] [*network-mask*]

DETAILED STEPS

	Command or Action	**Purpose**
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	• 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 192.168.3.2 remote-as 50000`	Adds the IP address or peer group name of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local router.
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 router is placed in address family 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 6	**network** *network-number* [**mask** *network-mask*] [**route-map** *route-map-name*] **Example:** `Device(config-router-af)# network 172.16.1.0 mask 255.255.255.0`	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 7	**neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**} **Example:**	Applies a route map to incoming or outgoing routes. • In this example, the route map named MED is applied to outbound routes to the BGP peer at Router E.

	Command or Action	Purpose
	`Device(config-router-af)# neighbor 192.168.3.2 route-map MED out`	
Step 8	**exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode and enters router configuration mode.
Step 9	**exit** **Example:** `Device(config-router)# exit`	Exits router configuration mode and enters global configuration mode.
Step 10	**route-map** *map-name* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map MED permit 10`	Configures a route map and enters route map configuration mode. • In this example, a route map named MED is created.
Step 11	**set metric** *value* **Example:** `Device(config-route-map)# set metric 50`	Sets the MED metric value.
Step 12	**end** **Example:** `Device(config-route-map)# end`	Exits route map configuration mode and enters privileged EXEC mode.
Step 13	Repeat Step 1 through Step 12 at Router D.	—
Step 14	**show ip bgp** [*network*] [*network-mask*] **Example:** `Device# show ip bgp 172.17.1.0 255.255.255.0`	(Optional) Displays the entries in the BGP routing table. • Use this command at Router E in the figure above when both Router B and Router D have configured the MED attribute. • 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 output is from Router E in the figure above after this task has been performed at both Router B and Router D. Note the metric (MED) values for the two routes to network 172.16.1.0. The peer 192.168.2.1 at Router D has a metric of 120 for the path to network 172.16.1.0, whereas the peer 192.168.3.1 at Router B has a metric of 50. The entry for the peer 192.168.3.1 at Router B has

the word best at the end of the entry to show that Router E will choose to send packets destined for network 172.16.1.0 via Router B because the MED metric is lower.

```
Device# show ip bgp 172.16.1.0

BGP routing table entry for 172.16.1.0/24, version 10
Paths: (2 available, best #2, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1
  45000
    192.168.2.1 from 192.168.2.1 (192.168.2.1)
      Origin IGP, metric 120, localpref 100, valid, external
  45000
    192.168.3.1 from 192.168.3.1 (172.17.1.99)
      Origin IGP, metric 50, localpref 100, valid, external, best
```

Influencing Outbound Path Selection

BGP can be used to influence the choice of paths for outbound traffic from the local autonomous system. This section contains two methods that BGP can use to influence outbound path selection:

- Using the Local_Pref attribute

- Using the BGP outbound route filter (ORF) capability

Perform one of the following tasks to influence outbound path selection:

Influencing Outbound Path Selection Using the Local_Pref Attribute

One of the methods to influence outbound path selection is to use the BGP Local-Pref attribute. Perform this task using the local preference attribute to influence outbound path selection. If there are several paths to the same destination the local preference attribute with the highest value indicates the preferred path.

Refer to the figure below for the network topology used in this task. Both Router B and Router C are configured. autonomous system 45000 receives updates for network 192.168.3.0 via autonomous system 40000 and autonomous system 50000. Router B is configured to set the local preference value to 150 for all updates to autonomous system 40000. Router C is configured to set the local preference value for all updates to autonomous system 50000 to 200. After the configuration, local preference information is exchanged within autonomous system 45000. Router B and Router C now see that updates for network 192.168.3.0 have a higher preference value from autonomous system 50000 so all traffic in autonomous system 45000 with a destination network of 192.168.3.0 is sent out via Router C.

Figure 5: Network Topology for Outbound Path Selection
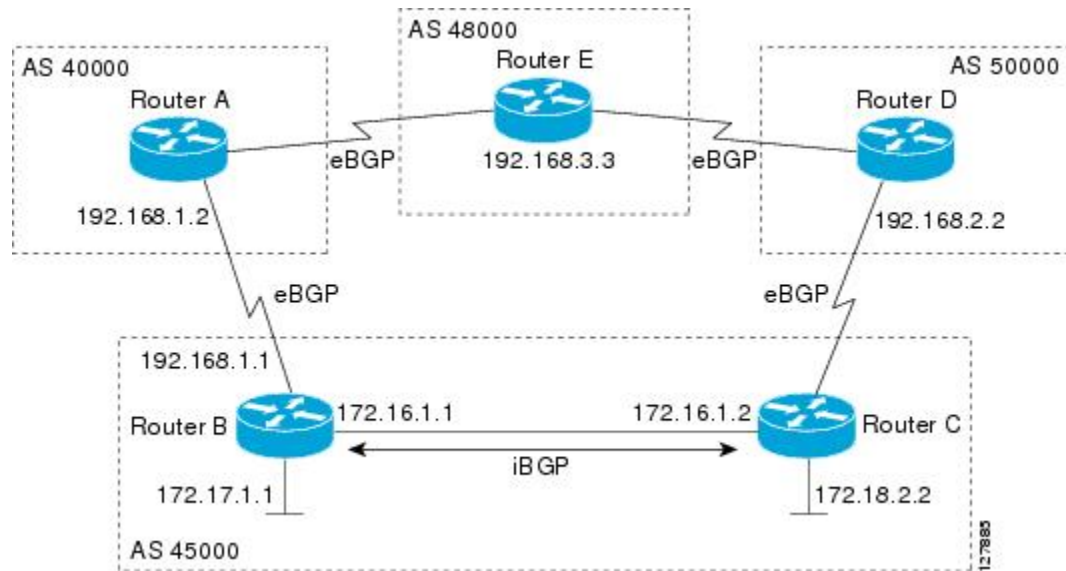
SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ip-address*| *peer-group-name*} **remote-as** *autonomous-system-number*
5. **bgp default local-preference** *value*
6. **address-family ipv4** [**unicast** | **multicast**| **vrf** *vrf-name*]
7. **network** *network-number* [**mask** *network-mask*][**route-map** *route-map-name*]
8. **neighbor** {*ip-address*| *peer-group-name*} **activate**
9. **end**
10. Repeat Step 1 through Step 9 at Router C but change the IP address of the peer, the autonomous system number, and set the local preference value to 200.
11. **show ip bgp** [*network*] [*network-mask*]

DETAILED STEPS

	Command or Action	**Purpose**
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(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:
	
	 `Router(config-router)# neighbor 192.168.1.2`
	 `remote-as 40000`
		Adds the IP address or peer group name of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local router.

Step 5	**bgp default local-preference** *value*
	
	 Example:
	
	 `Router(config-router)# bgp default`
	 `local-preference 150`
		Changes the default local preference value.
		
		 • In this example, the local preference is changed to 150 for all updates from autonomous system 40000 to autonomous system 45000.
		
		 • By default, the local preference value is 100.

Step 6	**address-family ipv4** [**unicast** | **multicast**| **vrf** *vrf-name*] **Example:** `Router(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 router is placed in address family 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 7	**network** *network-number* [**mask** *network-mask*][**route-map** *route-map-name*]
	
	 Example:
	
	 `Router(config-router-af)# network 172.17.1.0 mask`
	 ` 255.255.255.0`
		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 8	**neighbor** {*ip-address*| *peer-group-name*} **activate**
	
	 Example:
	
	 `Router(config-router-af)# neighbor 192.168.1.2`
	 `activate`
		Adds the IP address or peer group name of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local router.

	Command or Action	**Purpose**
Step 9	**end** **Example:** `Router(config-router-af)# end`	Exits route map configuration mode and enters privileged EXEC mode.
Step 10	Repeat Step 1 through Step 9 at Router C but change the IP address of the peer, the autonomous system number, and set the local preference value to 200.	--
Step 11	**show ip bgp** [*network*] [*network-mask*] **Example:** `Router# show ip bgp 192.168.3.0 255.255.255.0`	Displays the entries in the BGP routing table. • Enter this command at both Router B and Router C and note the Local_Pref value. The route with the highest preference value will be the preferred route to network 192.168.3.0. **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*.

Filtering Outbound BGP Route Prefixes

Perform this task to use BGP prefix-based outbound route filtering to influence outbound path selection.

Before you begin

BGP peering sessions must be established, and BGP ORF capabilities must be enabled on each participating router before prefix-based ORF announcements can be received.

Note

• BGP prefix-based outbound route filtering does not support multicast.
• IP addresses that are used for outbound route filtering must be defined in an IP prefix list. BGP distribute lists and IP access lists are not supported.
• Outbound route filtering is configured on only a per-address family basis and cannot be configured under the general session or BGP routing process.
• Outbound route filtering is configured for external peering sessions only.

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. **router bgp** autonomous-system-*number*
5. **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*
6. **neighbor** *ip-address* **ebgp-multihop** [*hop-count*]
7. **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]

8.　**neighbor** *ip-address* **capability orf prefix-list** [**send** | **receive** | **both**]

9.　**neighbor** {*ip-address* | *peer-group-name*} **prefix-list** *prefix-list-name* {**in** | **out**}

10.　**end**

11.　**clear ip bgp** {*ip-address* | *****} **in prefix-filter**

DETAILED STEPS

	Command or Action	**Purpose**
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# 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:** `Router(config)# ip prefix-list FILTER seq 10 permit 192.168.1.0/24`	Creates a prefix list for prefix-based outbound route filtering. • Outbound route filtering supports prefix length matching, wildcard-based prefix matching, and exact address prefix matching on a per address-family basis. • The prefix list is created to define the outbound route filter. The filter must be created when the outbound route filtering capability is configured to be advertised in send mode or both mode. It is not required when a peer is configured to advertise receive mode only. • The example creates a prefix list named FILTER that defines the 192.168.1.0/24 subnet for outbound route filtering.
Step 4	**router bgp** autonomous-system-*number* **Example:** `Router(config)# router bgp 100`	Enters router configuration mode, and creates a BGP routing process.
Step 5	**neighbor** {*ip-address*	*peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Router(config-router)# neighbor 10.1.1.1 remote-as 200`	Establishes peering with the specified neighbor or peer group. BGP peering must be established before ORF capabilities can be exchanged. • The example establishes peering with the 10.1.1.1 neighbor.
Step 6	**neighbor** *ip-address* **ebgp-multihop** [*hop-count*] **Example:**	Accepts or initiates BGP connections to external peers residing on networks that are not directly connected.

Command or Action	Purpose
`Router(config-router)# neighbor 10.1.1.1 ebgp-multihop`	
Step 7 **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*] **Example:** `Router(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 router is placed in address family 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. **Note** Outbound route filtering is configured on a per-address family basis.
Step 8 **neighbor** *ip-address* **capability orf prefix-list** [**send** | **receive** | **both**] **Example:** `Router(config-router-af)# neighbor 10.1.1.1 capability orf prefix-list both`	Enables the ORF capability on the local router, and enables ORF capability advertisement to the BGP peer specified with the *ip-address* argument. • The **send** keyword configures a router to advertise ORF send capabilities. • The **receive** keyword configures a router to advertise ORF receive capabilities. • The **both** keyword configures a router to advertise send and receive capabilities. • The remote peer must be configured to either send or receive ORF capabilities before outbound route filtering is enabled. • The example configures the router to advertise send and receive capabilities to the 10.1.1.1 neighbor.
Step 9 **neighbor** {*ip-address* | *peer-group-name*} **prefix-list** *prefix-list-name* {**in** | **out**} **Example:** `Router(config-router-af)# neighbor 10.1.1.1 prefix-list FILTER in`	Applies an inbound prefix-list filter to prevent distribution of BGP neighbor information. • In this example, the prefix list named FILTER is applied to incoming advertisements from the 10.1.1.1 neighbor, which prevents distribution of the 192.168.1.0/24 subnet.

	Command or Action	Purpose
Step 10	**end** **Example:** `Router(config-router-af)# end`	Exits address family configuration mode, and enters privileged EXEC mode.
Step 11	**clear ip bgp** {*ip-address* | *****} **in prefix-filter** **Example:** `Router# clear ip bgp 10.1.1.1 in prefix-filter`	Clears BGP outbound route filters and initiates an inbound soft reset. • A single neighbor or all neighbors can be specified. **Note** The inbound soft refresh must be initiated with the **clear ip bgp** command in order for this feature to function.

Configuring BGP Peering with ISPs

BGP was developed as an interdomain routing protocol and connecting to ISPs is one of the main functions of BGP. Depending on the size of your network and the purpose of your business, there are many different ways to connect to your ISP. Multihoming to one or more ISPs provides redundancy in case an external link to an ISP fails. This section introduces some optional tasks that can be used to connect to a service provider using multihoming techniques. Smaller companies may use just one ISP but require a backup route to the ISP. Larger companies may have access to two ISPs, using one of the connections as a backup, or may need to configure a transit autonomous system.

Perform one of the following optional tasks to connect to one or more ISPs:

Configuring Multihoming with Two ISPs

Perform this task to configure your network to access two ISPs where one ISP is the preferred route and the second ISP is a backup route. In the figure below Router B in autonomous system 45000 has BGP peers in two ISPs, autonomous system 40000 and autonomous system 50000. Using this task, Router B will be configured to prefer the route to the BGP peer at Router A in autonomous system 40000.

All routes learned from this neighbor will have an assigned weight. The route with the highest weight will be chosen as the preferred route when multiple routes are available to a particular network.

Note The weights assigned with the **set weight** route-map configuration command override the weights assigned using the **neighbor weight** command.

Figure 6: Multihoming with Two ISPs
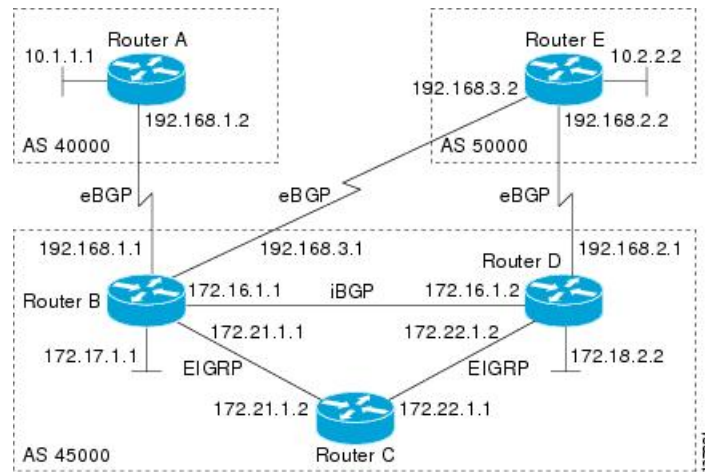
SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*
5. **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]
6. **network** *network-number* [**mask** *network-mask*]
7. **neighbor** {*ip-address* | *peer-group-name*} **weight** *number*
8. **exit**
9. **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*
10. **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]
11. **neighbor** {*ip-address* | *peer-group-name*} **weight** *number*
12. **end**
13. **clear ip bgp** {***** | *ip-address* | *peer-group-name*} [**soft** [**in** | **out**]]
14. **show ip bgp** [*network*] [*network-mask*]

DETAILED STEPS

	Command or Action	**Purpose**
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	**Purpose**
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 45000`	Enters router configuration mode, and creates a BGP routing process.

Step 4	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*
	
	Example:
	
	`Router(config-router)# neighbor 192.168.1.2`
	`remote-as 40000`
		Adds the IP address or peer group name of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local router.

Step 5	**address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*] **Example:** `Router(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 router 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 6	**network** *network-number* [**mask** *network-mask*]
	
	Example:
	
	`Router(config-router-af)# network 172.17.1.0 mask`
	` 255.255.255.0`
		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 7	**neighbor** {*ip-address* | *peer-group-name*} **weight** *number*
	
	Example:
	
	`Router(config-router-af)# neighbor 192.168.1.2`
	`weight 150`
		Assigns a weight to a BGP peer connection.
		
		• In this example, the weight attribute for routes received from the BGP peer 192.168.1.2 is set to 150.

Step 8	**exit** **Example:** `Router(config-router-af)# exit`	Exits address family configuration mode and enters router configuration mode.

	Command or Action	Purpose
Step 9	**neighbor** {*ip-address*	*peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Router(config-router)# neighbor 192.168.3.2 remote-as 50000`	Adds the IP address or peer group name of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local router.
Step 10	**address-family ipv4** [**unicast**	**multicast**	**vrf** *vrf-name*] **Example:** `Router(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 router 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 11	**neighbor** {*ip-address*	*peer-group-name*} **weight** *number* **Example:** `Router(config-router-af)# neighbor 192.168.3.2 weight 100`	Assigns a weight to a BGP peer connection. • In this example, the weight attribute for routes received from the BGP peer 192.168.3.2 is set to 100.
Step 12	**end** **Example:** `Router(config-router-af)# end`	Exits address family configuration mode and enters privileged EXEC mode.
Step 13	**clear ip bgp** {*****	*ip-address*	*peer-group-name*} [**soft** [**in**	**out**]] **Example:** `Router# clear ip bgp *`	(Optional) Clears BGP outbound route filters and initiates an outbound soft reset. A single neighbor or all neighbors can be specified.
Step 14	**show ip bgp** [*network*] [*network-mask*] **Example:** `Router# show ip bgp`	Displays the entries in the BGP routing table. • Enter this command at Router B to see the weight attribute for each route to a BGP peer. The route with the highest weight attribute will be the preferred route to network 172.17.1.0. **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 example shows the BGP routing table at Router B with the weight attributes assigned to routes. The route through 192.168.1.2 (Router A in the figure above) has the highest weight attribute and will be the preferred route to network 10.3.0.0, wherein the network 10.3.0.0 is accessible through Router A and Router E. If this route (through Router B) fails for some reason, the route through 192.168.3.2 (Router E) will be used to reach network 10.3.0.0. This way, redundancy is provided for reaching Router B.

```
BGP table version is 8, 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                150       40000 i
*> 10.2.2.0/24        192.168.3.2          0                100       50000 i

*> 10.3.0.0/16        192.168.1.2          0                150       40000 i
*                     192.168.3.2          0                100       50000 i

*> 172.17.1.0/24      0.0.0.0              0                32768          i
```

Multihoming with a Single ISP

Perform this task to configure your network to access one of two connections to a single ISP, where one of the connections is the preferred route and the second connection is a backup route. In the figure above Router E in autonomous system 50000 has two BGP peers in a single autonomous system, autonomous system 45000. Using this task, autonomous system 50000 does not learn any routes from autonomous system 45000 and is sending its own routes using BGP. This task is configured at Router E in the figure above and covers three features about multihoming to a single ISP:

- Outbound traffic—Router E will forward default routes and traffic to autonomous system 45000 with Router B as the primary link and Router D as the backup link. Static routes are configured to both Router B and Router D with a lower distance configured for the link to Router B.

- Inbound traffic—Inbound traffic from autonomous system 45000 is configured to be sent from Router B unless the link fails when the backup route is to send traffic from Router D. To achieve this, outbound filters are set using the MED metric.

- Prevention of transit traffic—A route map is configured at Router E in autonomous system 50000 to block all incoming BGP routing updates to prevent autonomous system 50000 from receiving transit traffic from the ISP in autonomous system 45000.

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

6. **network** *network-number* [**mask** *network-mask*] [**route-map** *route-map-name*]
7. **neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**}
8. Repeat Step 7 to apply another route map to the neighbor specified in Step 7.
9. **exit**
10. **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*
11. **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]
12. **neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**}
13. Repeat Step 10 to apply another route map to the neighbor specified in Step 10.
14. **exit**
15. **exit**
16. **ip route** *prefix mask* {*ip-address* | *interface-type interface-number* [*ip-address*]} [*distance*] [*name*] [**permanent** | **track** *number*] [**tag** *tag*]
17. Repeat Step 14 to establish another static route.
18. **route-map** *map-name* [**permit** | **deny**] [*sequence-number*]
19. **set metric** *value*
20. **exit**
21. **route-map** *map-name* [**permit** | **deny**] [*sequence-number*]
22. **set metric** *value*
23. **exit**
24. **route-map** *map-name* [**permit** | **deny**] [*sequence-number*]
25. **end**
26. **show ip route** [*ip-address*] [*mask*] [**longer-prefixes**]
27. **show ip bgp** [*network*] [*network-mask*]

DETAILED STEPS

	Command or Action	**Purpose**
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(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:**	Adds the IP address or peer group name of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local router.

	Command or Action	**Purpose**
	Router(config-router)# neighbor 192.168.2.1 remote-as 45000	• In this example, the BGP peer at Router D is added to the BGP routing table.
Step 5	**address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*] **Example:** Router(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 router is placed in address family 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 6	**network** *network-number* [**mask** *network-mask*] [**route-map** *route-map-name*] **Example:** Router(config-router-af)# network 10.2.2.0 mask 255.255.255.0	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 7	**neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**} **Example:** Router(config-router-af)# neighbor 192.168.2.1 route-map BLOCK in **Example:** Router(config-router-af)# neighbor 192.168.2.1 route-map SETMETRIC1 out	Applies a route map to incoming or outgoing routes. • In the first example, the route map named BLOCK is applied to inbound routes at Router E. • In the second example, the route map named SETMETRIC1 is applied to outbound routes to Router D. **Note** Two examples are shown here because the task example requires both these statements to be configured.
Step 8	Repeat Step 7 to apply another route map to the neighbor specified in Step 7.	--
Step 9	**exit** **Example:** Router(config-router-af)# exit	Exits address family configuration mode and enters router configuration mode.

	Command or Action	**Purpose**
Step 10	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Router(config-router)# neighbor 192.168.3.1 remote-as 45000`	Adds the IP address or peer group name of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local router. • In this example, the BGP peer at Router D is added to the BGP routing table.
Step 11	**address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*] **Example:** `Router(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 router is placed in address family 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 12	**neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**}
	
	Example:
	
	`Router(config-router-af)# neighbor 192.168.3.1 route-map BLOCK in`
	
	Example:
	
	`Router(config-router-af)# neighbor 192.168.3.1 route-map SETMETRIC2 out`
		Applies a route map to incoming or outgoing routes.
		
		• In the first example, the route map named BLOCK is applied to inbound routes at Router E.
		
		• In the second example, the route map named SETMETRIC2 is applied to outbound routes to Router D.
		
		Note Two examples are shown here because the task example requires both these statements to be configured.

Step 13	Repeat Step 10 to apply another route map to the neighbor specified in Step 10.	--
Step 14	**exit** **Example:** `Router(config-router-af)# exit`	Exits address family configuration mode and enters router configuration mode.
Step 15	**exit** **Example:** `Router(config-router)# exit`	Exits router configuration mode and enters global configuration mode.

	Command or Action	**Purpose**

Step 16	**ip route** *prefix mask* {*ip-address* | *interface-type interface-number* [*ip-address*]} [*distance*] [*name*] [**permanent** | **track** *number*] [**tag** *tag*]
	
	Example:
	
	`Router(config)# ip route 0.0.0.0 0.0.0.0 192.168.2.1 50`
	
	Example:
	
	`Router(config)# ip route 0.0.0.0 0.0.0.0 192.168.2.1 50`
	
	Example:
	
	and
	
	Example:
	
	`Router(config)# ip route 0.0.0.0 0.0.0.0 192.168.3.1 40`
		Establishes a static route.
		
		• In the first example, a static route to BGP peer 192.168.2.1 is established and given an administrative distance of 50.
		
		• In the second example, a static route to BGP peer 192.168.3.1 is established and given an administrative distance of 40. The lower administrative distance makes this route via Router B the preferred route.
		
		Note Two examples are shown here because the task example requires both these statements to be configured.

Step 17	Repeat Step 14 to establish another static route.	--
Step 18	**route-map** *map-name* [**permit** | **deny**] [*sequence-number*] **Example:** `Router(config)# route-map SETMETRIC1 permit 10`	Configures a route map and enters route map configuration mode. • In this example, a route map named SETMETRIC1 is created.
Step 19	**set metric** *value* **Example:** `Router(config-route-map)# set metric 100`	Sets the MED metric value.
Step 20	**exit** **Example:** `Router(config-route-map)# exit`	Exits route map configuration mode and enters global configuration mode.
Step 21	**route-map** *map-name* [**permit** | **deny**] [*sequence-number*] **Example:** `Router(config)# route-map SETMETRIC2 permit 10`	Configures a route map and enters route map configuration mode. • In this example, a route map named SETMETRIC2 is created.
Step 22	**set metric** *value* **Example:** `Router(config-route-map)# set metric 50`	Sets the MED metric value.

	Command or Action	**Purpose**
Step 23	**exit** **Example:** `Router(config-route-map)# exit`	Exits route map configuration mode and enters global configuration mode.
Step 24	**route-map** *map-name* [**permit** | **deny**] [*sequence-number*] **Example:** `Router(config)# route-map BLOCK deny 10`	Configures a route map and enters route map configuration mode. • In this example, a route map named BLOCK is created to block all incoming routes from autonomous system 45000.
Step 25	**end** **Example:** `Router(config-route-map)# end`	Exits route map configuration mode and enters privileged EXEC mode.
Step 26	**show ip route** [*ip-address*] [*mask*] [**longer-prefixes**] **Example:** `Router# show ip route`	(Optional) Displays route information from the routing tables. • Use this command at Router E in the figure above after Router B and Router D have received update information containing the MED metric from Router E. • 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 27	**show ip bgp** [*network*] [*network-mask*] **Example:** `Router# show ip bgp 172.17.1.0 255.255.255.0`	(Optional) Displays the entries in the BGP routing table. • Use this command at Router E in the figure above after Router B and Router D have received update information containing the MED metric from Router E. • 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 example shows output from the **show ip route** command entered at Router E after this task has been configured and Router B and Router D have received update information containing the MED metric. Note that the gateway of last resort is set as 192.168.3.1, which is the route to Router B.

```
Router# show ip route
```

```
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route
Gateway of last resort is 192.168.3.1 to network 0.0.0.0
     10.0.0.0/24 is subnetted, 1 subnets
C       10.2.2.0 is directly connected, Ethernet0/0
C     192.168.2.0/24 is directly connected, Serial3/0
C     192.168.3.0/24 is directly connected, Serial2/0
S*    0.0.0.0/0 [40/0] via 192.168.3.1
```

The following example shows output from the **show ip bgp** command entered at Router E after this task has been configured and Router B and Router D have received routing updates. The route map BLOCK has denied all routes coming in from autonomous system 45000 so the only network shown is the local network.

```
Router# show ip bgp

BGP table version is 2, local router ID is 10.2.2.99
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      0.0.0.0                  0          32768 i
```

The following example shows output from the **show ip bgp** command entered at Router B after this task has been configured at Router E and Router B has received routing updates. Note the metric of 50 for network 10.2.2.0.

```
Router# show ip bgp

BGP table version is 7, 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
*> 10.2.2.0/24      192.168.3.2             50              0 50000 i
*> 172.16.1.0/24    0.0.0.0                  0          32768 i
*> 172.17.1.0/24    0.0.0.0                  0          32768 i
```

The following example shows output from the **show ip bgp** command entered at Router D after this task has been configured at Router E and Router D has received routing updates. Note the metric of 100 for network 10.2.2.0.

```
Router# show ip bgp

BGP table version is 3, local router ID is 192.168.2.1
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      192.168.2.2            100              0 50000 i
*> 172.16.1.0/24    0.0.0.0                  0          32768 i
```

Configuring Multihoming to Receive the Full Internet Routing Table

Perform this task to configure your network to build neighbor relationships with other routers in other autonomous systems while filtering outbound routes. In this task the full Internet routing table will be received from the service providers in the neighboring autonomous systems but only locally originated routes will be advertised to the service providers. This task is configured at Router B in the figure above and uses an access list to permit only locally originated routes and a route map to ensure that only the locally originated routes are advertised outbound to other autonomous systems.

> **Note**   Be aware that receiving the full Internet routing table from two ISPs may use all the memory in smaller routers.

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. **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]
6. **network** *network-number* [**mask** *network-mask*]
7. **neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**}
8. **exit**
9. **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*
10. **address-family ipv4** [**unicast** | **multicast**| **vrf** *vrf-name*]
11. **neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**}
12. **exit**
13. **exit**
14. **ip as-path access-list** *access-list-number* {**deny** | **permit**} *as-regular-expression*
15. **route-map** *map-name* [**permit** | **deny**] [*sequence-number*]
16. **match as-path** *path-list-number*
17. **end**
18. **show ip bgp** [*network*] [*network-mask*]

DETAILED STEPS

	Command or Action	**Purpose**
Step 1	**enable**	Enables privileged EXEC mode.
	Example:	• Enter your password if prompted.
	Router> enable	
Step 2	**configure terminal**	Enters global configuration mode.
	Example:	
	Router# configure terminal	

	Command or Action	**Purpose**
Step 3	**router bgp** *autonomous-system-number* **Example:** Router(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:** Router(config-router)# neighbor 192.168.1.2 remote-as 40000	Adds the IP address or peer group name of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local router.
Step 5	**address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*] **Example:** Router(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 router is placed in address family 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 6	**network** *network-number* [**mask** *network-mask*] **Example:** Router(config-router-af)# network 172.17.1.0 mask 255.255.255.0	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 7	**neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**} **Example:** Router(config-router-af)# neighbor 192.168.1.2 route-map localonly out	Applies a route map to incoming or outgoing routes. • In this example, the route map named localonly is applied to outbound routes to Router A.
Step 8	**exit** **Example:** Router(config-router-af)# exit	Exits address family configuration mode and enters router configuration mode.

	Command or Action	**Purpose**
Step 9	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Router(config-router)# neighbor 192.168.3.2 remote-as 50000`	Adds the IP address or peer group name of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local router.
Step 10	**address-family ipv4** [**unicast** | **multicast**| **vrf** *vrf-name*] **Example:** `Router(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 router is placed in address family 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 11	**neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**} **Example:** `Router(config-router-af)# neighbor 192.168.3.2 route-map localonly out`	Applies a route map to incoming or outgoing routes. • In this example, the route map named localonly is applied to outbound routes to Router E.
Step 12	**exit** **Example:** `Router(config-router-af)# exit`	Exits address family configuration mode and enters router configuration mode.
Step 13	**exit** **Example:** `Router(config-router)# exit`	Exits router configuration mode and enters global configuration mode.
Step 14	**ip as-path access-list** *access-list-number* {**deny** | **permit**} *as-regular-expression* **Example:** `Router(config)# ip as-path access-list 10 permit ^$`	Defines a BGP-related access list. • In this example, the access list number 10 is defined to permit only locally originated BGP routes.
Step 15	**route-map** *map-name* [**permit** | **deny**] [*sequence-number*]	Configures a route map and enters route map configuration mode.

	Command or Action	Purpose
	Example: Router(config)# route-map localonly permit 10	• In this example, a route map named localonly is created.
Step 16	**match as-path** *path-list-number* **Example:** Router(config-route-map)# match as-path 10	Matches a BGP autonomous system path access list. • In this example, the BGP autonomous system path access list created in Step 12 is used for the match clause.
Step 17	**end** **Example:** Router(config-route-map)# end	Exits route map configuration mode and enters privileged EXEC mode.
Step 18	**show ip bgp** [*network*] [*network-mask*] **Example:** Router# show ip bgp	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 example shows the BGP routing table for Router B in the figure above after this task has been configured. Note that the routing table contains the information about the networks in the autonomous systems 40000 and 50000.

```
BGP table version is 5, 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
*> 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
```

Configuring BGP Policies

The tasks in this section help you configure BGP policies that filter the traffic in your BGP network. The following optional tasks demonstrate some of the various methods by which traffic can be filtered in your BGP network:

Filtering BGP Prefixes with Prefix Lists

Perform this task to use prefix lists to filter BGP route information. The task is configured at Router B in the figure below where both Router A and Router E are set up as BGP peers. A prefix list is configured to permit only routes from the network 10.2.2.0/24 to be outbound. In effect, this will restrict the information that is received from Router E to be forwarded to Router A. Optional steps are included to display the prefix list information and to reset the hit count.

Figure 7: BGP Topology for Configuring BGP Policies Tasks

![Note icon]
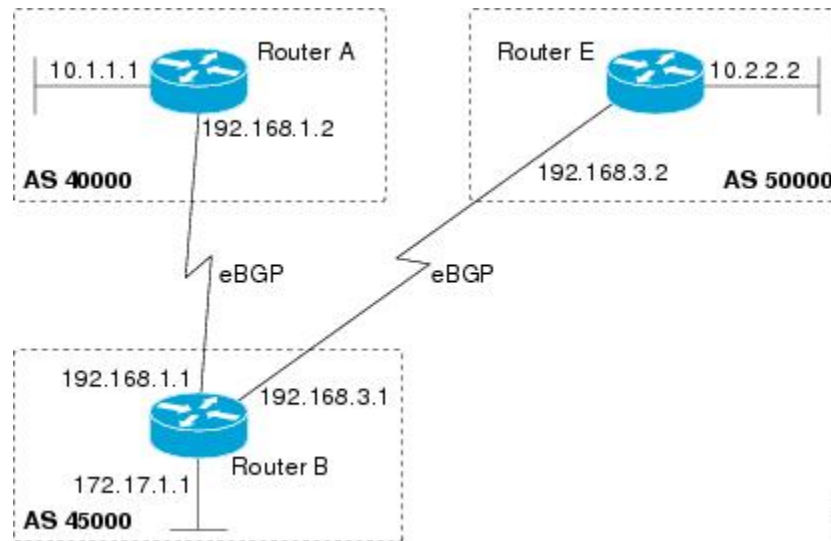
Note	The **neighbor prefix-list** and the **neighbor distribute-list** commands are mutually exclusive for a BGP peer.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** *ip-address* **remote-as** *autonomous-system-number*
5. Repeat Step 5 for all BGP peers.
6. **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]
7. **network** *network-number* [**mask** *network-mask*]
8. **aggregate-address** *address mask* [**as-set**]
9. **neighbor** *ip-address* **prefix-list** *list-name* {**in** | **out**}
10. **exit**
11. **exit**
12. **ip prefix-list** *list-name* [**seq** *seq-number*] {**deny** *network/length* | **permit** *network/length*} [**ge** *ge-value*] [**le** *le-value*] [**eq** *eq-value*]
13. **end**
14. **show ip prefix-list** [**detail** | **summary**] [*prefix-list-name* [**seq** *seq-number* | *network/length* [**longer** | **first-match**]]]
15. **clear ip prefix-list** {***** | *ip-address* | *peer-group-name*} **out**

DETAILED STEPS

	Command or Action	**Purpose**
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Router> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 45000`	Enters router configuration mode for the specified routing process.

Step 4	**neighbor** *ip-address* **remote-as** *autonomous-system-number*
	
	Example:
	
	`Router(config-router)# neighbor 192.168.1.2`
	`remote-as 40000`
		Adds the IP address of the neighbor in the specified autonomous system to the BGP neighbor table of the local router.

Step 5	Repeat Step 5 for all BGP peers.	--
Step 6	**address-family ipv4** [**unicast**	**multicast**	**vrf** *vrf-name*] **Example:** `Router(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 router is placed in address family 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 7	**network** *network-number* [**mask** *network-mask*]
	
	Example:
	
	`Router(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 8	**aggregate-address** *address mask* [**as-set**] **Example:**	Creates an aggregate entry in a BGP routing table. • A specified route must exist in the BGP table.

	Command or Action	**Purpose**
	`Router(config-router-af)# aggregate-address` `172.0.0.0 255.0.0.0`	• 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. **Note** Only partial syntax is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.

Step 9	**neighbor** *ip-address* **prefix-list** *list-name* {**in** | **out**}
	Example:
	`Router(config-router-af)# neighbor 192.168.1.2`
	`prefix-list super172 out`
		Distributes BGP neighbor information as specified in a prefix list.
		
		• In this example, a prefix list called super172 is set for outgoing routes to Router A.

Step 10	**exit** **Example:** `Router(config-router-af)# exit`	Exits address family configuration mode and enters router configuration mode.
Step 11	**exit** **Example:** `Router(config-router) exit`	Exits router configuration mode and enters global configuration mode.

Step 12	**ip prefix-list** *list-name* [**seq** *seq-number*] {**deny** *network/length* | **permit** *network/length*} [**ge** *ge-value*] [**le** *le-value*] [**eq** *eq-value*]
	Example:
	`Router(config)# ip prefix-list super172 permit`
	`172.0.0.0/8`
		Defines a BGP-related prefix list and enters access list configuration mode.
		
		• In this example, the prefix list called super172 is defined to permit only route 172.0.0.0/8 to be forwarded.
		
		• All other routes will be denied because there is an implicit deny at the end of all prefix lists.

Step 13	**end** **Example:** `Router(config-access-list)# end`	Exits access list configuration mode and enters privileged EXEC mode.
Step 14	**show ip prefix-list** [**detail** | **summary**] [*prefix-list-name* [**seq** *seq-number* | *network/length* [**longer** | **first-match**]]] **Example:** `Router# show ip prefix-list detail super172`	Displays information about prefix lists. • In this example, details of the prefix list named super172 will be displayed, including the hit count. Hit count is the number of times the entry has matched a route.
Step 15	**clear ip prefix-list** {***** | *ip-address* | *peer-group-name*} **out**	Resets the hit count of the prefix list entries.

Command or Action	Purpose
Example: `Router# clear ip prefix-list super172 out`	• In this example, the hit count for the prefix list called super172 will be reset.

Examples

The following output from the **show ip prefix-list** command shows details of the prefix list named super172, including the hit count. The **clear ip prefix-list** command is entered to reset the hit count and the **show ip prefix-list** command is entered again to show the hit count reset to 0.

```
Router# show ip prefix-list detail super172

ip prefix-list super172:
   count: 1, range entries: 0, sequences: 5 - 5, refcount: 4
   seq 5 permit 172.0.0.0/8 (hit count: 1, refcount: 1)

Router# clear ip prefix-list super172

Router# show ip prefix-list detail super172

ip prefix-list super172:
   count: 1, range entries: 0, sequences: 5 - 5, refcount: 4
   seq 5 permit 172.0.0.0/8 (hit count: 0, refcount: 1)
```

Filtering BGP Prefixes with AS-Path Filters

Perform this task to filter BGP prefixes using AS-path filters with an access list based on the value of the AS-path attribute to filter route information. An AS-path access list is configured at Router B in the figure above. The first line of the access list denies all matches to AS-path 50000, and the second line allows all other paths. The router uses the **neighbor filter-list** command to specify the AS-path access list as an outbound filter. After the filter is enabled, traffic can be received from both Router A and Router C, but updates originating from autonomous system 50000 (Router C) are not forwarded by Router B to Router A. If any updates from Router C originated from another autonomous system, they would be forwarded because they would contain both autonomous system 50000 and another autonomous system number, and that would not match the AS-path access list.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip as-path access-list** *access-list-number* {**deny** | **permit**} *as-regular-expression*
4. Repeat Step 3 for all entries required in the AS-path access list.
5. **router bgp** *autonomous-system-number*
6. **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*
7. Repeat Step 6 for all BGP peers.
8. **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]
9. **neighbor** {*ip-address* | *peer-group-name*} **filter-list** *access-list-number* {**in** | **out**}
10. **end**
11. **show ip bgp regexp** *as-regular-expression*

DETAILED STEPS

	Command or Action	Purpose
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 as-path access-list** *access-list-number* {**deny** | **permit**} *as-regular-expression*
	
	Example:
	
	Device(config)# ip as-path access-list 100 deny ^50000$
	
	Example:
	
	Device(config)# ip as-path access-list 100 permit .*
		Defines a BGP-related access list and enters access list configuration mode.
		
		    • In the first example, access list number 100 is defined to deny any AS-path that starts and ends with 50000.
		
		    • In the second example, all routes that do not match the criteria in the first example of the AS-path access list will be permitted. The period and asterisk symbols imply that all characters in the AS-path will match, so Router B will forward those updates to Router A.
		
		Note Two examples are shown here because the task example requires both these statements to be configured.

Step 4	Repeat Step 3 for all entries required in the AS-path access list.	—
Step 5	**router bgp** *autonomous-system-number* **Example:** Device(config)# router bgp 45000	Enters router configuration mode for the specified routing process.
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 or peer group name of the neighbor in the specified autonomous system BGP neighbor table of the local router.
Step 7	Repeat Step 6 for all BGP peers.	—
Step 8	**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 router is placed in address family configuration mode for the IPv4

	Command or Action	Purpose
		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 9	**neighbor** {*ip-address* | *peer-group-name*} **filter-list** *access-list-number* {**in** | **out**} **Example:** `Device(config-router-af)# neighbor 192.168.1.2 filter-list 100 out`	Distributes BGP neighbor information as specified in a prefix list. • In this example, an access list number 100 is set for outgoing routes to Router A.
Step 10	**end** **Example:** `Device(config-router-af)# end`	Exits address family configuration mode and returns to privileged EXEC mode.
Step 11	**show ip bgp regexp** *as-regular-expression* **Example:** `Device# show ip bgp regexp ^50000$`	Displays routes that match the regular expression. • To verify the regular expression, you can use this command. • In this example, all paths that match the expression "starts and ends with 50000" will be displayed.

Examples

The following output from the **show ip bgp regexp** command shows the autonomous system paths that match the regular expression—start and end with AS-path 50000:

```
Device# show ip bgp regexp ^50000$

BGP table version is 9, 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            150 50000 i
```

Filtering BGP Prefixes with AS-path Filters Using 4-Byte Autonomous System Numbers

In Cisco IOS Release 12.0(32)SY8, 12.0(33)S3, 12.2(33)SRE, 12.2(33)SXI1, and later releases, BGP support for 4-octet (4-byte) autonomous system numbers was introduced. The 4-byte autonomous system numbers in this task are formatted in the default asplain (decimal value) format, for example, Router B is in autonomous

system number 65538 in the figure below. For more details about the introduction of 4-byte autonomous system numbers, see the "BGP Autonomous System Number Formats" section.

Perform this task to filter BGP prefixes with AS-path filters using 4-byte autonomous system numbers with an access list based on the value of the AS-path attribute to filter route information. An AS-path access list is configured at Router B in the figure below. The first line of the access list denies all matches to the AS-path 65550 and the second line allows all other paths. The router uses the **neighbor filter-list** command to specify the AS-path access list as an outbound filter. After the filtering is enabled, traffic can be received from both Router A and Router E but updates originating from autonomous system 65550 (Router E) are not forwarded by Router B to Router A. If any updates from Router E originated from another autonomous system, they would be forwarded because they would contain both autonomous system 65550 plus another autonomous system number, and that would not match the AS-path access list.

Note   In Cisco IOS Releases 12.0(22)S, 12.2(15)T, 12.2(18)S, and later releases, the maximum number of autonomous system access lists that can be configured with the **ip as-path access-list** command is increased from 199 to 500.

Figure 8: BGP Topology for Filtering BGP Prefixes with AS-path Filters Using 4-Byte Autonomous System Numbers
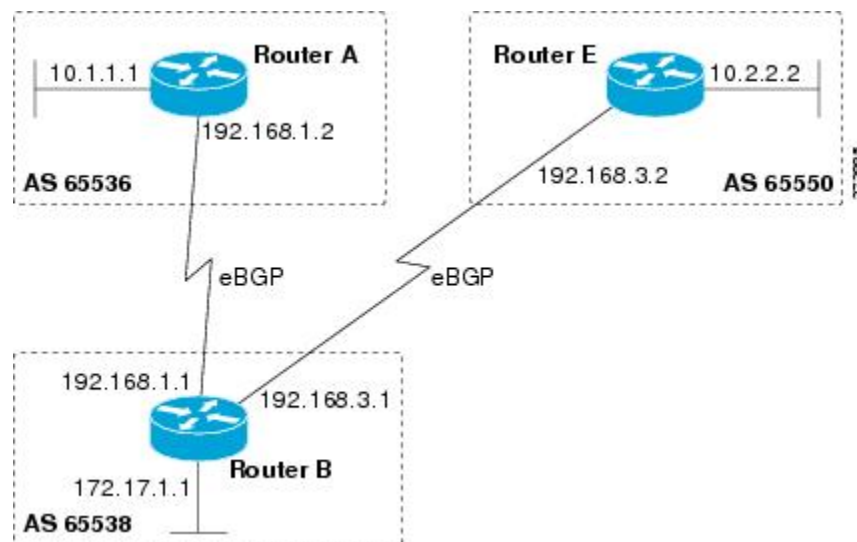
SUMMARY STEPS

1. **enable**
2. **configure   terminal**
3. **router bgp**   *autonomous-system-number*
4. **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*
5. Repeat Step 4 for all BGP peers.
6. **address-family ipv4** [**unicast** | **multicast**| **vrf** *vrf-name*]
7. **network**   *network-number*   [**mask**   *network-mask*]
8. **neighbor** {*ip-address* | *peer-group-name*} **filter-list** *access-list-number*{**in** | **out**}
9. **exit**
10. **exit**

11. **ip as-path access-list** *access-list-number* {**deny** | **permit**} *as-regular-expression*
12. Repeat Step 11 for all entries required in the AS-path access list.
13. **end**
14. **show ip bgp regexp** *as-regular-expression*

DETAILED STEPS

	Command or Action	**Purpose**
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 65538`	Enters router configuration mode for the specified routing process.
Step 4	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Router(config-router-af)# neighbor 192.168.1.2 remote-as 65536`	Adds the IP address or peer group name of the neighbor in the specified autonomous system BGP neighbor table of the local router. • In this example, the IP address for the neighbor at Router A is added.
Step 5	Repeat Step 4 for all BGP peers.	--
Step 6	**address-family ipv4** [**unicast** | **multicast**| **vrf** *vrf-name*] **Example:** `Router(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 router is placed in address family 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.

	Command or Action	**Purpose**
Step 7	**network** *network-number* [**mask** *network-mask*] **Example:** `Router(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. **Note** Only partial syntax is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.
Step 8	**neighbor** {*ip-address* | *peer-group-name*} **filter-list** *access-list-number*{**in** | **out**} **Example:** `Router(config-router-af)# neighbor 192.168.1.2 filter-list 99 out`	Distributes BGP neighbor information as specified in a prefix list. • In this example, an access list number 99 is set for outgoing routes to Router A.
Step 9	**exit** **Example:** `Router(config-router-af)# exit`	Exits address family configuration mode and returns to router configuration mode.
Step 10	**exit** **Example:** `Router(config-router)# exit`	Exits router configuration mode and returns to global configuration mode.

Step 11	**ip as-path access-list** *access-list-number* {**deny** | **permit**} *as-regular-expression*
	
	Example:
	
	`Router(config)# ip as-path access-list 99 deny ^65550$`
	
	Example:
	
	`and`
	
	Example:
	
	`Router(config)# ip as-path access-list 99 permit .*`
		Defines a BGP-related access list and enters access list configuration mode.
		
		• In the first example, access list number 99 is defined to deny any AS-path that starts and ends with 65550.
		
		• In the second example, all routes that do not match the criteria in the first example of the AS-path access list will be permitted. The period and asterisk symbols imply that all characters in the AS-path will match, so Router B will forward those updates to Router A.
		
		Note Two examples are shown here because the task example requires both these statements to be configured.

Step 12	Repeat Step 11 for all entries required in the AS-path access list.	--
Step 13	**end** **Example:**	Exits access list configuration mode and returns to privileged EXEC mode.

	Command or Action	Purpose
	`Router(config-access-list)# end`	
Step 14	**show ip bgp regexp** *as-regular-expression* **Example:** `Router# show ip bgp regexp ^65550$`	Displays routes that match the regular expression. • To verify the regular expression, you can use this command. • In this example, all paths that match the expression "starts and ends with 65550" will be displayed.

Examples

The following output from the **show ip bgp regexp** command shows the autonomous system paths that match the regular expression--start and end with AS-path 65550:

```
RouterB# show ip bgp regexp ^65550$
BGP table version is 4, 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 65550 i
```

Filtering Traffic Using Community Lists

Perform this task to filter traffic by creating a BGP community list, referencing the community list within a route map, and then applying the route map to a neighbor.

In this task, Router B in the figure below is configured with route maps and a community list to control incoming routes.

Figure 9: Topology for Which a Community List Is Configured
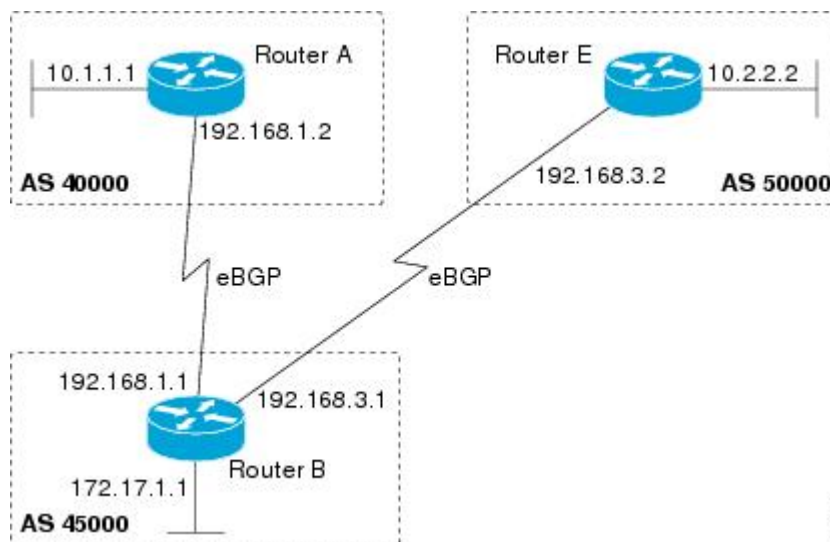
SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*
5. **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]
6. **neighbor** {*ip-address* | *peer-group-name*} **route-map** *route-map-name* {**in** | **out**}
7. **exit**
8. **exit**
9. **route-map** *map-name* [**permit** | **deny**] [*sequence-number*]
10. **match community** {*standard-list-number* | *expanded-list-number* | *community-list-name* [**exact**]}
11. **set weight** *weight*
12. **exit**
13. **route-map** *map-name* [**permit** | **deny**] [*sequence-number*]
14. **match community** {*standard-list-number* | *expanded-list-number* | *community-list-name* [**exact**]}
15. **set community** *community-number*
16. **exit**
17. **ip community-list** {*standard-list-number* | **standard** *list-name* {**deny** | **permit**} [*community-number*] [*AA:NN*] [**internet**] [**local-AS**] [**no-advertise**] [**no-export**]} | {*expanded-list-number* | **expanded** *list-name* {**deny** | **permit**} *regular-expression*}
18. Repeat Step 17 to create all the required community lists.
19. **exit**
20. **show ip community-list** [*standard-list-number* | *expanded-list-number* | *community-list-name*] [**exact-match**]

DETAILED STEPS

	Command or Action	**Purpose**
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:**	Adds the IP address or peer group name of the neighbor to the specified autonomous system BGP neighbor table of the local router.

	Command or Action	Purpose
	`Device(config-router)# neighbor 192.168.3.2 remote-as 50000`	
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 router is placed in address family 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 6	**neighbor** {*ip-address* | *peer-group-name*} **route-map** *route-map-name* {**in** | **out**} **Example:** `Device(config-router-af)# neighbor 192.168.3.2 route-map 2000 in`	Applies a route map to inbound or outbound routes. • In this example, the route map called 2000 is applied to inbound routes from the BGP peer at 192.168.3.2.
Step 7	**exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode and enters router configuration mode.
Step 8	**exit** **Example:** `Device(config-router)# exit`	Exits router configuration mode and enters global configuration mode.
Step 9	**route-map** *map-name* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map 2000 permit 10`	Creates a route map and enters route map configuration mode. • In this example, the route map called 2000 is defined.
Step 10	**match community** {*standard-list-number* | *expanded-list-number* | *community-list-name* [**exact**]} **Example:** `Device(config-route-map)# match community 1`	Matches on the communities in a BGP community list. • In this example, the route's community attribute is matched to communities in community list 1.

	Command or Action	**Purpose**
Step 11	**set weight** *weight* **Example:** `Device(config-route-map)# set weight 30`	Sets the weight of BGP routes that match the community list. • In this example, any route that matches community list 1 will have its weight set to 30.
Step 12	**exit** **Example:** `Device(config-route-map)# exit`	Exits route map configuration mode and enters global configuration mode.
Step 13	**route-map** *map-name* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map 3000 permit 10`	Creates a route map and enters route map configuration mode. • In this example, the route map called 3000 is defined.
Step 14	**match community** {*standard-list-number* | *expanded-list-number* | *community-list-name* [**exact**]} **Example:** `Device(config-route-map)# match community 2`	Matches on the communities in a BGP community list. • In this example, the route's COMMUNITIES attribute is matched to communities in community list 2.
Step 15	**set community** *community-number* **Example:** `Device(config-route-map)# set community 99`	Sets the BGP communities attribute. • In this example, any route that matches community list 2 will have the COMMUNITIES attribute set to 99.
Step 16	**exit** **Example:** `Device(config-route-map)# exit`	Exits route map configuration mode and enters global configuration mode.

Step 17	**ip community-list** {*standard-list-number* | **standard** *list-name* {**deny** | **permit**} [*community-number*] [*AA:NN*] [**internet**] [**local-AS**] [**no-advertise**] [**no-export**]} | {*expanded-list-number* | **expanded** *list-name* {**deny** | **permit**} *regular-expression*}
	
	Example:
	
	`Device(config)# ip community-list 1 permit 100`
	
	Example:
	
	`Device(config)# ip community-list 2 permit internet`
		Creates a community list for BGP and controls access to it.
		
		• In the first example, community list 1 permits routes with a COMMUNITIES attribute of 100. Router E routes all have a COMMUNITIES attribute of 100, so their weight will be set to 30.
		
		• In the second example, community list 2 effectively permits all routes by specifying the **internet** community. Any routes that did not match community list 1 are checked against community list 2. All routes are permitted, but no changes are made to the route attributes.

	Command or Action	Purpose
		Note Two examples are shown here because the task example requires both of these statements to be configured.
Step 18	Repeat Step 17 to create all the required community lists.	—
Step 19	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and enters privileged EXEC mode.
Step 20	**show ip community-list** [*standard-list-number* | *expanded-list-number* | *community-list-name*] [**exact-match**] **Example:** `Device# show ip community-list 1`	Displays configured BGP community list entries.

Examples

The following sample output verifies that community list 1 has been created and it permits routes that have a community attribute of 100:

```
Device# show ip community-list 1

Community standard list 1
    permit 100
```

The following sample output verifies that community list 2 has been created and it effectively permits all routes by specifying the **internet** community:

```
Device# show ip community-list 2

Community standard list 2
    permit internet
```

Filtering Traffic Using Extended Community Lists

Perform this task to filter traffic by creating an extended BGP community list to control outbound routes.

Figure 10: Topology for Which a Community List Is Configured
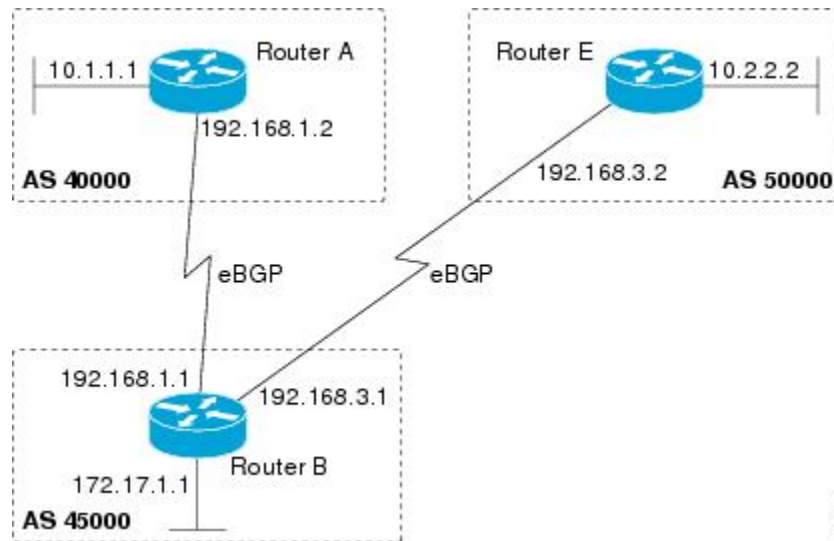
In this task, Router B in the figure above is configured with an extended named community list to specify that the BGP peer at 192.168.1.2 is not sent advertisements about any path through or from autonomous system 50000. The IP extended community-list configuration mode is used and the ability to resequence entries is shown.

Note A sequence number is applied to all extended community list entries by default, regardless of the configuration mode. Explicit sequencing and resequencing of extended community list entries can be configured only in IP extended community-list configuration mode, not in global configuration mode.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip extcommunity-list** {*expanded-list-number* | **expanded** *list-name* | *standard-list-number* | **standard** *list-name*}
4. [*sequence-number*] {**deny** [*regular-expression*] | **exit** | **permit** [*regular-expression*]}
5. Repeat Step 4 for all the required permit or deny entries in the extended community list.
6. **resequence** [*starting-sequence*] [*sequence-increment*]
7. **exit**
8. **router bgp** *autonomous-system-number*
9. **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*
10. Repeat the prior step for all of the required BGP peers.
11. **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]
12. **network** *network-number* [**mask** *network-mask*]
13. **end**
14. **show ip extcommunity-list** [*list-name*]

DETAILED STEPS

	Command or Action	**Purpose**
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 extcommunity-list** {*expanded-list-number*	**expanded** *list-name*	*standard-list-number*	**standard** *list-name*} **Example:** Device(config)# ip extcommunity-list expanded DENY50000	Enters IP extended community-list configuration mode to create or configure an extended community list. • In this example, the expanded community list DENY50000 is created.
Step 4	[*sequence-number*] {**deny** [*regular-expression*]	**exit**	**permit** [*regular-expression*]} **Example:** Device(config-extcomm-list)# 10 deny _50000_ **Example:** Device(config-extcomm-list)# 20 deny ^50000 .*	Configures an expanded community list entry. • In the first example, an expanded community list entry with the sequence number 10 is configured to deny advertisements about paths from autonomous system 50000. • In the second example, an expanded community list entry with the sequence number 20 is configured to deny advertisements about paths through autonomous system 50000. **Note** Two examples are shown here because the task example requires both these statements to be configured. **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 5	Repeat Step 4 for all the required permit or deny entries in the extended community list.	—
Step 6	**resequence** [*starting-sequence*] [*sequence-increment*] **Example:** Device(config-extcomm-list)# resequence 50 100	Resequences expanded community list entries. • In this example, the sequence number of the first expanded community list entry is set to 50 and subsequent entries are set to increment by 100. The second expanded community list entry is therefore set to 150.

	Command or Action	**Purpose**
	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 7	**exit** **Example:** `Device(config-extcomm-list)# exit`	Exits expanded community-list configuration mode and enters global configuration mode.
Step 8	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 45000`	Enters router configuration mode for the specified routing process.

Step 9	**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 or peer group name of the neighbor to the specified autonomous system BGP neighbor table of the local router.

Step 10	Repeat the prior step for all of the required BGP peers.	—
Step 11	**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 router is placed in address family configuration mode for the IPv4 unicast address family if the **unicast** keyword is not specified in the **address-family ipv4** command. • The **multicast** keyword specifies IPv4 multicast address prefixes. Note 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 12	**network** *network-number* [**mask** *network-mask*]
	
	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.

	Command or Action	Purpose
		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 13	**end** **Example:** Device(config-router-af)# end	Exits address family configuration mode and enters privileged EXEC mode.
Step 14	**show ip extcommunity-list** [*list-name*] **Example:** Device# show ip extcommunity-list DENY50000	Displays configured BGP expanded community list entries.

Examples

The following sample output verifies that the BGP expanded community list DENY50000 has been created, with the output showing that the entries to deny advertisements about autonomous system 50000 have been resequenced from 10 and 20 to 50 and 150:

```
Device# show ip extcommunity-list DENY50000

Expanded extended community-list DENY50000
    50 deny _50000_
    150 deny ^50000 .*
```

Filtering Traffic Using a BGP Route Map Policy List

Perform this task to create a BGP policy list and then reference it within a route map.

A policy list is like a route map that contains only match clauses. With policy lists there are no changes to match clause semantics and route map functions. The match clauses are configured in policy lists with permit and deny statements and the route map evaluates and processes each match clause to permit or deny routes based on the configuration. AND and OR semantics in the route map function the same way for policy lists as they do for match clauses.

Policy lists simplify the configuration of BGP routing policy in medium-size and large networks. The network operator can reference preconfigured policy lists with groups of match clauses in route maps and easily apply general changes to BGP routing policy. The network operator no longer needs to manually reconfigure each recurring group of match clauses that occur in multiple route map entries.

Perform this task to create a BGP policy list to filter traffic that matches the autonomous system path and MED of a router and then create a route map to reference the policy list.

Before you begin

BGP routing must be configured in your network and BGP neighbors must be established.

Note

- BGP route map policy lists do not support the configuration of IPv6 match clauses in policy lists.
- Policy lists are not supported in versions of Cisco IOS software prior to Cisco IOS Releases 12.0(22)S and 12.2(15)T. Reloading a router that is running an older version of Cisco IOS software may cause some routing policy configurations to be lost.
- Policy lists support only match clauses and do not support set clauses. However, policy lists can coexist, within the same route map entry, with match and set clauses that are configured separately from the policy lists.
- Policy lists are supported only by BGP. They are not supported by other IP routing protocols. This limitation does not interfere with normal operations of a route map, including redistribution, because policy list functions operate transparently within BGP and are not visible to other IP routing protocols.
- Policy lists support only match clauses and do not support set clauses. However, policy lists can coexist, within the same route map entry, with match and set clauses that are configured separately from the policy lists. The first route map example configures AND semantics, and the second route map configuration example configures semantics. Both examples in this section show sample route map configurations that reference policy lists and separate match and set clauses in the same configuration.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip policy-list** *policy-list-name* {**permit** | **deny**}
4. **match as-path** *as-number*
5. **match metric** *metric*
6. **exit**
7. **route-map** *map-name* [**permit** | **deny**] [*sequence-number*]
8. **match ip address** {*access-list-number* | *access-list-name*} [*... access-list-number* | *... access-list-name*]
9. **match policy-list** *policy-list-name*
10. **set community** *community-number* [**additive**] [*well-known-community*] | **none**}
11. **set local-preference** *preference-value*
12. **end**
13. **show ip policy-list** [*policy-list-name*]
14. **show route-map** [*route-map-name*]

DETAILED STEPS

	Command or Action	**Purpose**
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**ip policy-list** *policy-list-name* {**permit** | **deny**} **Example:** `Router(config)# ip policy-list POLICY-LIST-NAME-1 permit`	Enters policy list configuration mode and creates a BGP policy list that will permit routes that are allowed by the match clauses that follow.
Step 4	**match as-path** *as-number* **Example:** `Router(config-policy-list)# match as-path 500`	Creates a match clause to permit routes from the specified autonomous system path.
Step 5	**match metric** *metric* **Example:** `Router(config-policy-list)# match metric 10`	Creates a match clause to permit routes with the specified metric.
Step 6	**exit** **Example:** `Router(config-policy-list)# exit`	Exits policy list configuration mode and enters global configuration mode.
Step 7	**route-map** *map-name* [**permit** | **deny**] [*sequence-number*] **Example:** `Router(config)# route-map MAP-NAME-1 permit 10`	Creates a route map and enters route map configuration mode.
Step 8	**match ip address** {*access-list-number* | *access-list-name*} [... *access-list-number* | ... *access-list-name*] **Example:** `Router(config-route-map)# match ip address 1`	Creates a match clause to permit routes that match the specified *access-list-number* or *access-list-name* argument.
Step 9	**match policy-list** *policy-list-name* **Example:** `Router(config-route-map)# match policy-list POLICY-LIST-NAME-1`	Creates a clause that will match the specified policy list. • All match clauses within the policy list will be evaluated and processed. Multiple policy lists can referenced with this command. • This command also supports AND or OR semantics like a standard match clause.
Step 10	**set community** *community-number* [**additive**] [*well-known-community*] | **none**} **Example:** `Router(config-route-map)# set community 10:1`	Creates a clause to set or remove the specified community.

	Command or Action	**Purpose**
Step 11	**set local-preference** *preference-value* **Example:** `Router(config-route-map)# set local-preference 140`	Creates a clause to set the specified local preference value.
Step 12	**end** **Example:** `Router(config-route-map)# end`	Exits route map configuration mode and enters privileged EXEC mode.
Step 13	**show ip policy-list** [*policy-list-name*] **Example:** `Router# show ip policy-list POLICY-LIST-NAME-1`	Display information about configured policy lists and policy list entries.
Step 14	**show route-map** [*route-map-name*] **Example:** `Router# show route-map`	Displays locally configured route maps and route map entries.

Examples

The following sample output verifies that a policy list has been created, with the output displaying the policy list name and configured match clauses:

```
Router# show ip policy-list
 POLICY-LIST-NAME-1

policy-list POLICY-LIST-NAME-1 permit
  Match clauses:
    metric 20
    as-path (as-path filter): 1
```

Note A policy list name can be specified when the **show ip policy-list** command is entered. This option can be useful for filtering the output of this command and verifying a single policy list.

The following sample output from the **show route-map** command verifies that a route map has been created and a policy list is referenced. The output of this command displays the route map name and policy lists that are referenced by the configured route maps.

```
Router# show route-map

route-map ROUTE-MAP-NAME-1, deny, sequence 10
  Match clauses:
  Set clauses:
  Policy routing matches: 0 packets, 0 bytes
route-map ROUTE-MAP-NAME-1, permit, sequence 10
```

```
Match clauses:
  IP Policy lists:
      POLICY-LIST-NAME-1
Set clauses:
Policy routing matches: 0 packets, 0 bytes
```

Filtering Traffic Using Continue Clauses in a BGP Route Map

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. **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]
6. **neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**}
7. **exit**
8. **exit**
9. **route-map** *map-name* {**permit** | **deny**} [*sequence-number*]
10. **match ip address** {*access-list-number* | *access-list-name*} [*... access-list-number* | *... access-list-name*]
11. **set community** { { [*community-number*] [*well-known-community*] [**additive**]} | **none**}
12. **continue** [*sequence-number*]
13. **end**
14. **show route-map** [*map-name*]

DETAILED STEPS

	Command or Action	**Purpose**
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 50000	Enters router configuration mode, and creates a BGP routing process.
Step 4	**neighbor** {*ip-address*	*peer-group-name*} **remote-as** *autonomous-system-number* **Example:**	Adds the IP address or peer group name of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local device.

	Command or Action	Purpose
	Device(config-router)# neighbor 10.0.0.1 remote-as 50000	
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 address family configuration mode for the IPv4 unicast address family if the **unicast** keyword is not specified. • 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 6	**neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**} **Example:** Device(config-router-af)# neighbor 10.0.0.1 route-map ROUTE-MAP-NAME in	Applies the inbound route map to routes received from the specified neighbor, or applies an outbound route map to routes advertised to the specified neighbor.
Step 7	**exit** **Example:** Device(config-router-af)# exit	Exits address family configuration mode and enters router configuration mode.
Step 8	**exit** **Example:** Device(config-router)# exit	Exits router configuration mode and enters global configuration mode.
Step 9	**route-map** *map-name* {**permit** | **deny**} [*sequence-number*] **Example:** Device(config)# route-map ROUTE-MAP-NAME permit 10	Enters route-map configuration mode to create or configure a route map.
Step 10	**match ip address** {*access-list-number* | *access-list-name*} [... *access-list-number* | ... *access-list-name*] **Example:** Device(config-route-map)# match ip address 1	Configures a **match** command that specifies the conditions under which policy routing and route filtering occur. • Multiple **match** commands can be configured. If a **match** command is configured, a match must occur in order for the continue statement to be executed. If

Command or Action	Purpose
	a **match** command is not configured, set and continue clauses will be executed. **Note** The **match** and **set** commands used in this task are examples that are used to help describe the operation of the **continue** command. For a list of specific **match** and **set** commands, see the **continue** command in the *Cisco IOS IP Routing: BGP Command Reference*.
Step 11 **set community** { { [*community-number*] [*well-known-community*] [**additive**]} | **none**} **Example:** Device(config-route-map)# set community 10:1	Configures a **set** command that specifies the routing action to perform if the criteria enforced by the **match** commands are met. • Multiple **set** commands can be configured. • In this example, a clause is created to set the specified community number in aa:nn format.
Step 12 **continue** [*sequence-number*] **Example:** Device(config-route-map)# continue	Configures a route map to continue to evaluate and execute match statements after a successful match occurs. • If a sequence number is configured, the continue clause will go to the route map with the specified sequence number. • If no sequence number is specified, the continue clause will go to the route map with the next sequence number. This behavior is called an "implied continue."
Step 13 **end** **Example:** Device(config-route-map)# end	Exits route-map configuration mode and enters privileged EXEC mode.
Step 14 **show route-map** [*map-name*] **Example:** Device# show route-map	(Optional) Displays locally configured route maps. The name of the route map can be specified in the syntax of this command to filter the output.

Examples

The following sample output shows how to verify the configuration of continue clauses using the **show route-map** command. The output displays configured route maps including the match, set, and continue clauses.

```
Device# show route-map
```

```
route-map MARKETING, permit, sequence 10
  Match clauses:
    ip address (access-lists): 1
    metric 10
  Continue: sequence 40
  Set clauses:
    as-path prepend 10
  Policy routing matches: 0 packets, 0 bytes
route-map MARKETING, permit, sequence 20
  Match clauses:
    ip address (access-lists): 2
    metric 20
  Set clauses:
    as-path prepend 10 10
  Policy routing matches: 0 packets, 0 bytes
route-map MARKETING, permit, sequence 30
  Match clauses:
  Continue: to next entry 40
  Set clauses:
    as-path prepend 10 10 10
  Policy routing matches: 0 packets, 0 bytes
route-map MARKETING, permit, sequence 40
  Match clauses:
    community (community-list filter): 10:1
  Set clauses:
    local-preference 104
  Policy routing matches: 0 packets, 0 bytes
route-map MKTG-POLICY-MAP, permit, sequence 10
  Match clauses:
  Set clauses:
    community 655370
  Policy routing matches: 0 packets, 0 bytes
```

Configuration Examples for Connecting to a Service Provider Using External BGP

Example: Influencing Inbound Path Selection

The following example shows how you can use route maps to modify incoming data from a neighbor. Any route received from 10.222.1.1 that matches the filter parameters set in autonomous system access list 200 will have its weight set to 200 and its local preference set to 250, and it will be accepted.

```
router bgp 100
!
 neighbor 10.222.1.1 route-map FIX-WEIGHT in
 neighbor 10.222.1.1 remote-as 1
!
ip as-path access-list 200 permit ^690$
ip as-path access-list 200 permit ^1800
!
route-map FIX-WEIGHT permit 10
 match as-path 200
 set local-preference 250
 set weight 200
```

In the following example, the route map named FINANCE marks all paths originating from autonomous system 690 with an MED metric attribute of 127. The second permit clause is required so that routes not matching autonomous system path list 1 will still be sent to neighbor 10.1.1.1.

```
router bgp 65000
 neighbor 10.1.1.1 route-map FINANCE out
!
ip as-path access-list 1 permit ^690_
ip as-path access-list 2 permit .*
!
route-map FINANCE permit 10
 match as-path 1
 set metric 127
!
route-map FINANCE permit 20
 match as-path 2
```

Inbound route maps could perform prefix-based matching and set various parameters of the update. Inbound prefix matching is available in addition to autonomous system path and community list matching. The following example shows how the route map named SET-LOCAL-PREF sets the local preference of the inbound prefix 172.20.0.0/16 to 120:

```
!
router bgp 65100
 network 10.108.0.0
 neighbor 10.108.1.1 remote-as 65200
 neighbor 10.108.1.1 route-map SET-LOCAL-PREF in
!
route-map SET-LOCAL-PREF permit 10
 match ip address 2
 set local-preference 120
!
route-map SET-LOCAL-PREF permit 20
!
access-list 2 permit 172.20.0.0 0.0.255.255
access-list 2 deny any
```

Example: Influencing Inbound Path Selection by Modifying the AS-path Attribute Using 4-Byte AS Numbers

This example shows how to configure BGP to influence the inbound path selection for traffic destined for the 172.17.1.0 network by modifying the AS-path attribute. In Cisco IOS Release 12.0(32)SY8, 12.0(33)S3, 12.2(33)SXI1, and later releases, BGP support for 4-octet (4-byte) autonomous system numbers was introduced. The 4-byte autonomous system numbers in this example are formatted in the default asplain (decimal value) format; for example, Router B is in autonomous system number 65538 in the figure below. For more details about the introduction of 4-byte autonomous system numbers, see the "BGP Autonomous System Number Formats" section.

One of the methods that BGP can use to influence the choice of paths in another autonomous system is to modify the AS-path attribute. For example, in the figure below, Router A advertises its own network, 172.17.1.0, to its BGP peers in autonomous system 65538 and autonomous system 65550. When the routing information is propagated to autonomous system 65545, the routers in autonomous system 65545 have network reachability information about network 172.17.1.0 from two different routes. The first route is from autonomous system 65538 with an AS-path consisting of 65538, 65536. The second route is through autonomous system 65547 with an AS-path of 65547, 65550, 65536. If all other BGP attribute values are the same, Router C in autonomous

system 65545 would choose the route through autonomous system 65538 for traffic destined for network 172.17.1.0 because it is the shortest route in terms of autonomous systems traversed.

Autonomous system 65536 now receives all traffic from autonomous system 65545 for the 172.17.1.0 network through Router B in autonomous system 65538. If, however, the link between autonomous system 65538 and autonomous system 65536 is a really slow and congested link, the **set as-path prepend**command can be used at Router A to influence inbound path selection for the 172.17.1.0 network by making the route through autonomous system 65538 appear to be longer than the path through autonomous system 65550. The configuration is done at Router A in the figure below by applying a route map to the outbound BGP updates to Router B. Using the **set as-path prepend**command, all the outbound BGP updates from Router A to Router B will have their AS-path attribute modified to add the local autonomous system number 65536 twice. After the configuration, autonomous system 65545 receives updates about the 172.17.1.0 network through autonomous system 65538. The new AS-path is 65538, 65536, 65536, 65536, which is now longer than the AS-path from autonomous system 65547 (unchanged at a value of 65547, 65550, 65536). Networking devices in autonomous system 65545 will now prefer the route through autonomous system 65547 to forward packets with a destination address in the 172.17.1.0 network.

Figure 11: Network Topology for Modifying the AS-path Attribute
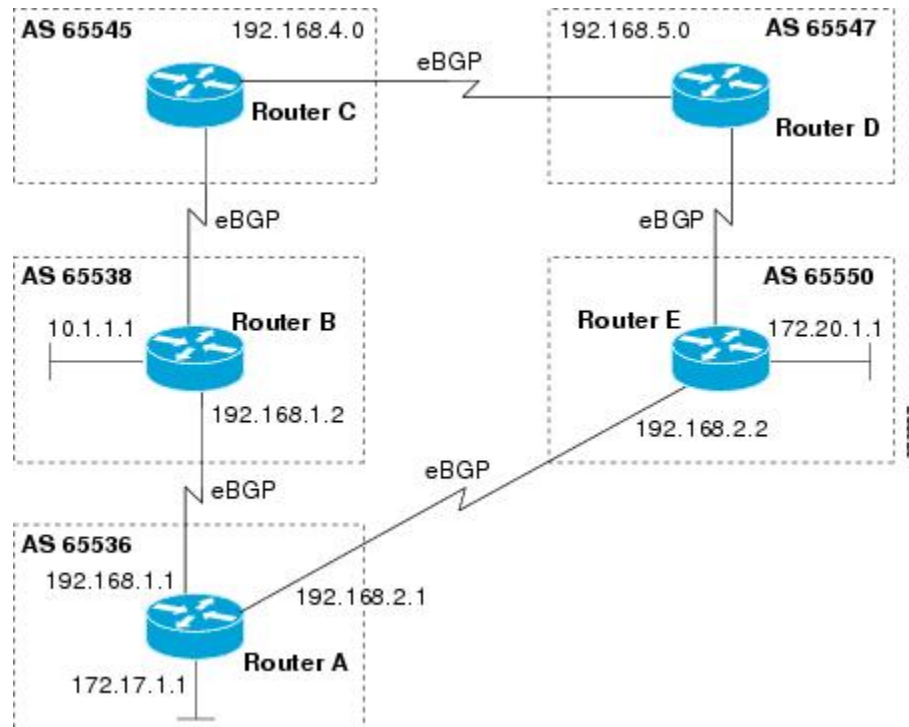
The configuration for this example is performed at Router A in the figure above.

```
router bgp 65536
 address-family ipv4 unicast
  network 172.17.1.0 mask 255.255.255.0
  neighbor 192.168.1.2 remote-as 65538
  neighbor 192.168.1.2 activate
  neighbor 192.168.1.2 route-map PREPEND out
  exit-address-family
 exit
route-map PREPEND permit 10
set as-path prepend 65536 65536
```

Example: Filtering BGP Prefixes with Prefix Lists

This section contains the following examples:

Example: Filtering BGP Prefixes Using a Single Prefix List

The following example shows how a prefix list denies the default route 0.0.0.0/0:

```
ip prefix-list abc deny 0.0.0.0/0
```

The following example shows how a prefix list permits a route that matches the prefix 10.0.0.0/8:

```
ip prefix-list abc permit 10.0.0.0/8
```

The following example shows how to configure the BGP process so that it accepts only prefixes with a prefix length of /8 to /24:

```
router bgp 40000
 network 10.20.20.0
 distribute-list prefix max24 in
!
ip prefix-list max24 seq 5 permit 0.0.0.0/0 ge 8 le 24
```

The following example configuration shows how to conditionally originate a default route (0.0.0.0/0) in RIP when a prefix 10.1.1.0/24 exists in the routing table:

```
ip prefix-list cond permit 10.1.1.0/24
!
route-map default-condition permit 10
 match ip address prefix-list cond
!
router rip
 default-information originate route-map default-condition
```

The following example shows how to configure BGP to accept routing updates from 192.168.1.1 only, besides filtering on the prefix length:

```
router bgp 40000
 distribute-list prefix max24 gateway allowlist in
!
ip prefix-list allowlist seq 5 permit 192.168.1.1/32
!
```

The following example shows how to direct the BGP process to filter incoming updates to the prefix using name1, and match the gateway (next hop) of the prefix being updated to the prefix list name2, on Gigabit Ethernet interface 0/0/0:

```
router bgp 103
 distribute-list prefix name1 gateway name2 in gigabitethernet 0/0/0
```

Example: Filtering BGP Prefixes Using a Group of Prefixes

The following example shows how to configure BGP to permit routes with a prefix length up to 24 in network 192/8:

```
ip prefix-list abc permit 192.0.0.0/8 le 24
```

The following example shows how to configure BGP to deny routes with a prefix length greater than 25 in 192/8:

```
ip prefix-list abc deny 192.0.0.0/8 ge 25
```

The following example shows how to configure BGP to permit routes with a prefix length greater than 8 and less than 24 in all address space:

```
ip prefix-list abc permit 0.0.0.0/0 ge 8 le 24
```

The following example shows how to configure BGP to deny routes with a prefix length greater than 25 in all address space:

```
ip prefix-list abc deny 0.0.0.0/0 ge 25
```

The following example shows how to configure BGP to deny all routes in network 10/8, because any route in the Class A network 10.0.0.0/8 is denied if its mask is less than or equal to 32 bits:

```
ip prefix-list abc deny 10.0.0.0/8 le 32
```

The following example shows how to configure BGP to deny routes with a mask greater than 25 in 192.168.1.0/24:

```
ip prefix-list abc deny 192.168.1.0/24 ge 25
```

The following example shows how to configure BGP to permit all routes:

```
ip prefix-list abc permit 0.0.0.0/0 le 32
```

Example: Adding or Deleting Prefix List Entries

You can add or delete individual entries in a prefix list if a prefix list has the following initial configuration:

```
ip prefix-list abc deny 0.0.0.0/0 le 7
ip prefix-list abc deny 0.0.0.0/0 ge 25
ip prefix-list abc permit 192.168.0.0/15
```

The following example shows how to delete an entry from the prefix list so that 192.168.0.0 is not permitted, and add a new entry that permits 10.0.0.0/8:

```
no ip prefix-list abc permit 192.168.0.0/15
ip prefix-list abc permit 10.0.0.0/8
```

The new configuration is as follows:

```
ip prefix-list abc deny 0.0.0.0/0 le 7
ip prefix-list abc deny 0.0.0.0/0 ge 25
ip prefix-list abc permit 10.0.0.0/8
```

Example: Filtering Traffic Using COMMUNITIES Attributes

This section contains two examples of the use of BGP COMMUNITIES attributes with route maps.

The first example configures a route map named *set-community*, which is applied to the outbound updates to the neighbor 172.16.232.50. The routes that pass access list 1 are given the well-known COMMUNITIES

attribute value **no-export**. The remaining routes are advertised normally. The **no-export** community value automatically prevents the advertisement of those routes by the BGP speakers in autonomous system 200.

```
router bgp 100
 neighbor 172.16.232.50 remote-as 200
 neighbor 172.16.232.50 send-community
 neighbor 172.16.232.50 route-map set-community out
!
route-map set-community permit 10
 match address 1
 set community no-export
!
route-map set-community permit 20
 match address 2
```

The second example configures a route map named *set-community*, which is applied to the outbound updates to neighbor 172.16.232.90. All the routes that originate from autonomous system 70 have the COMMUNITIES attribute values 200 200 added to their already existing communities. All other routes are advertised as normal.

```
route-map bgp 200
 neighbor 172.16.232.90 remote-as 100
 neighbor 172.16.232.90 send-community
 neighbor 172.16.232.90 route-map set-community out
!
route-map set-community permit 10
 match as-path 1
 set community 200 200 additive
!
route-map set-community permit 20
!
ip as-path access-list 1 permit 70$
ip as-path access-list 2 permit .*
```

Example: Filtering Traffic Using AS-Path Filters

The following example shows BGP path filtering by neighbor. Only the routes that pass autonomous system path access list 2 will be sent to 192.168.12.10. Similarly, only routes passing access list 3 will be accepted from 192.168.12.10.

```
router bgp 200
 neighbor 192.168.12.10 remote-as 100
 neighbor 192.168.12.10 filter-list 1 out
 neighbor 192.168.12.10 filter-list 2 in
 exit
ip as-path access-list 1 permit _109_
ip as-path access-list 2 permit _200$
ip as-path access-list 2 permit ^100$
ip as-path access-list 3 deny _690$
ip as-path access-list 3 permit .*
```

Example: Filtering Traffic with AS-path Filters Using 4-Byte Autonomous System Numbers

Asplain Default Format in Cisco IOS Release 12.0(32)SY8, 12.0(33)S3, 12.2(33)SRE, 12.2(33)SXI1, and Later Releases

The following example is available in Cisco IOS Release 12.0(32)SY8, 12.0(33)S3, 12.2(33)SRE, 12.2(33)XNE, 12.2(33)SXI1, and later releases and shows BGP path filtering by neighbor using 4-byte autonomous system numbers in asplain format. Only the routes that pass autonomous system path access list 2 will be sent to 192.168.3.2.

```
ip as-path access-list 2 permit ^65536$
router bgp 65538
 address-family ipv4 unicast
  neighbor 192.168.3.2 remote-as 65550
  neighbor 192.168.3.2 activate
  neighbor 192.168.3.2 filter-list 2 in
  end
```

Asdot Default Format in Cisco IOS Release 12.0(32)S12, and 12.4(24)T

The following example available in Cisco IOS Release 12.0(32)S12, 12.4(24)T, and later releases shows BGP path filtering by neighbor using 4-byte autonomous system numbers in asdot format. Only the routes that pass autonomous system path access list 2 will be sent to 192.168.3.2.

Note   In Cisco IOS Release 12.0(32)SY8, 12.0(33)S3, 12.2(33)SRE, 12.2(33)XNE, 12.2(33)SXI1, and later releases, this example works if you have configured asdot as the default display format using the **bgp asnotation dot** command.

```
ip as-path access-list 2 permit ^1\.0$
router bgp 1.2
 address-family ipv4 unicast
  neighbor 192.168.3.2 remote-as 1.14
  neighbor 192.168.3.2 filter-list 2 in
  end
```
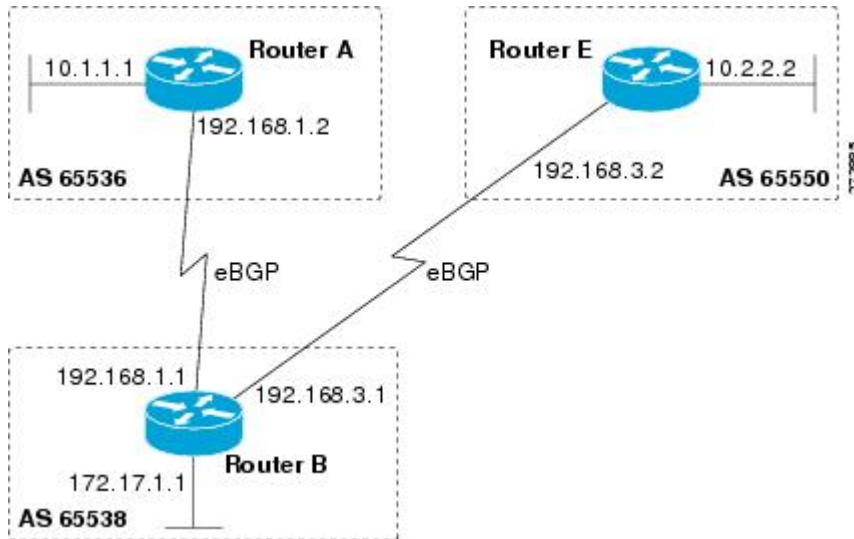
Example: Filtering Traffic Using Extended Community Lists with 4-Byte Autonomous System Numbers

Asplain Default Format in Cisco IOS Release 12.0(32)SY8, 12.0(33)S3, 12.2(33)SRE, 12.2(33)SXI1, and Later Releases

The following example shows how to filter traffic by creating an extended BGP community list to control outbound routes. In Cisco IOS Release 12.0(32)SY8, 12.0(33)S3, 12.2(33)SRE, 12.2(33)XNE, 12.2(33)SXI1, and later releases, extended BGP communities support 4-byte autonomous system numbers in the regular expressions in asplain by default. Extended community attributes are used to configure, filter, and identify routes for VRF instances and MPLS VPNs. The **ip extcommunity-list**command is used to configure named or numbered extended community lists. All of the standard rules of access lists apply to the configuration of

extended community lists. Regular expressions are supported by the expanded range of extended community list numbers.

Figure 12: BGP Topology for Filtering Traffic Using Extended Community Lists with 4-Byte Autonomous System Numbers in Asplain Format

Note A sequence number is applied to all extended community list entries by default regardless of the configuration mode. Explicit sequencing and resequencing of extended community list entries can be configured only in IP extended community-list configuration mode and not in global configuration mode.

In this exam the figure above is configured with an extended named community list to specify that the BGP peer at 192.1681.2 is not sent advertisements about any path through or from the 4-byte autonomous system 65550. The IP extended community-list configuration mode is used, and the ability to resequence entries is shown.

```
ip extcommunity-list expanded DENY65550
 10 deny _65550_
 20 deny ^65550 .*
 resequence 50 100
 exit
router bgp 65538
 network 172.17.1.0 mask 255.255.255.0
 address-family ipv4 unicast
  neighbor 192.168.3.2 remote-as 65550
  neighbor 192.168.1.2 remote-as 65536
  neighbor 192.168.3.2 activate
  neighbor 192.168.1.2 activate
  end
show ip extcommunity-list DENY65550
```

Asdot Default Format in Cisco IOS Release 12.0(32)S12, and 12.4(24)T
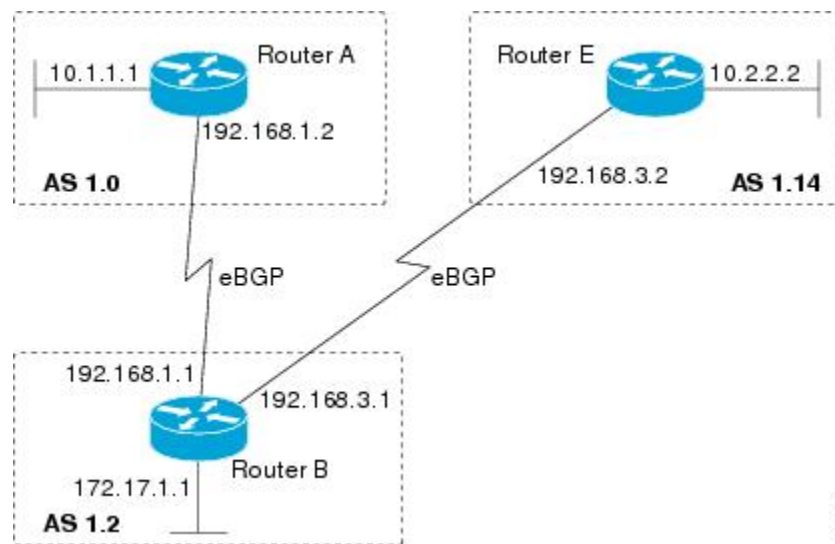
The following example shows how to filter traffic by creating an extended BGP community list to control outbound routes. In Cisco IOS Release 12.0(32)S12, 12.4(24)T, and later releases, extended BGP communities support 4-byte autonomous system numbers in the regular expressions in asdot format only. Extended

community attributes are used to configure, filter, and identify routes for VRF instances and MPLS VPNs. The **ip extcommunity-list**command is used to configure named or numbered extended community lists. All of the standard rules of access lists apply to the configuration of extended community lists. Regular expressions are supported by the expanded range of extended community list numbers.

Note In Cisco IOS Release 12.0(32)SY8, 12.0(33)S3, 12.2(33)SXI1, and later releases, this example works if you have configured asdot as the default display format using the **bgp asnotation dot** command.

Figure 13: BGP Topology for Filtering Traffic Using Extended Community Lists with 4-Byte Autonomous System Numbers in Asdot Format

Note A sequence number is applied to all extended community list entries by default regardless of the configuration mode. Explicit sequencing and resequencing of extended community list entries can be configured only in IP extended community-list configuration mode and not in global configuration mode.

In this exam the figure above is configured with an extended named community list to specify that the BGP peer at 192.1681.2 is not sent advertisements about any path through or from the 4-byte autonomous system 65550. The IP extended community-list configuration mode is used, and the ability to resequence entries is shown.

```
ip extcommunity-list expanded DENY114
 10 deny _1\.14_
 20 deny ^1\.14_.*
 resequence 50 100
 exit
router bgp 1.2
 network 172.17.1.0 mask 255.255.255.0
 address-family ipv4 unicast
  neighbor 192.168.3.2 remote-as 1.14
  neighbor 192.168.1.2 remote-as 1.0
  neighbor 192.168.3.2 activate
  neighbor 192.168.1.2 activate
```

```
    end
show ip extcommunity-list DENY114
```

Example: Filtering Traffic Using a BGP Route Map

The following example shows how to use an address family to configure BGP so that any unicast and multicast routes from neighbor 10.1.1.1 are accepted if they match access list 1:

```
route-map filter-some-multicast
 match ip address 1
 exit
router bgp 65538
 neighbor 10.1.1.1 remote-as 65537
 address-family ipv4 unicast
  neighbor 10.1.1.1 activate
  neighbor 10.1.1.1 route-map filter-some-multicast in
  exit
 exit
router bgp 65538
 neighbor 10.1.1.1 remote-as 65537
 address-family ipv4 multicast
  neighbor 10.1.1.1 activate
  neighbor 10.1.1.1 route-map filter-some-multicast in
  end
```

Where to Go Next

- To configure advanced BGP feature tasks, proceed to the "Configuring Advanced BGP Features" module.

- To configure BGP neighbor session options, proceed to the "Configuring BGP Neighbor Session Options" module.

- To configure internal BGP tasks, proceed to the "Configuring Internal BGP Features" module.

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
BGP commands: complete command syntax, command mode, defaults, command history, usage guidelines, and examples	Cisco IOS IP Routing: BGP Command Reference
BGP overview	"Cisco BGP Overview" module
Configuring basic BGP tasks	"Configuring a Basic BGP Network" module
BGP fundamentals and description	*Large-Scale IP Network Solutions*, Khalid Raza and Mark Turner, Cisco Press, 2000

Related Topic	Document Title
Implementing and controlling BGP in scalable networks	*Building Scalable Cisco Networks*, Catherine Paquet and Diane Teare, Cisco Press, 2001
Interdomain routing basics	*Internet Routing Architectures*, Bassam Halabi, Cisco Press, 1997

Standards

Standard	Title
MDT SAFI	MDT SAFI

MIBs

MIB	MIBs Link
CISCO-BGP4-MIB	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 1772	*Application of the Border Gateway Protocol in the Internet*
RFC 1773	*Experience with the BGP Protocol*
RFC 1774	*BGP-4 Protocol Analysis*
RFC 1930	*Guidelines for Creation, Selection, and Registration of an Autonomous System (AS)*
RFC 2519	*A Framework for Inter-Domain Route Aggregation*
RFC 2858	*Multiprotocol Extensions for BGP-4*
RFC 2918	*Route Refresh Capability for BGP-4*
RFC 3392	*Capabilities Advertisement with BGP-4*
RFC 4271	*A Border Gateway Protocol 4 (BGP-4)*
RFC 4684	*Constrained Route Distribution for Border Gateway Protocol/MultiProtocol Label Switching (BGP/MPLS) Internet Protocol (IP) Virtual Private Networks (VPNs)*
RFC 4893	*BGP Support for Four-Octet AS Number Space*
RFC 5291	*Outbound Route Filtering Capability for BGP-4*
RFC 5396	*Textual Representation of Autonomous system (AS) Numbers*

RFC	Title
RFC 5398	*Autonomous System (AS) Number Reservation for Documentation Use*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Connecting to a Service Provider Using External BGP

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 5: Feature Information for Connecting to a Service Provider Using External BGP

Feature Name	Releases	Feature Configuration Information
BGP Increased Support of Numbered AS-Path Access Lists to 500	12.0(22)S 12.2(15)T 12.2(18)S 12.2(18)SXD 12.2(27)SBC 15.0(1)S	The BGP Increased Support of Numbered AS-Path Access Lists to 500 feature increases the maximum number of autonomous systems access lists that can be configured using the **ip as-path access-list** command from 199 to 500.

Feature Name	Releases	Feature Configuration Information
BGP Named Community Lists	12.2(8)T 12.2(14)S 15.0(1)S	The BGP Named Community Lists feature introduces a new type of community list called the named community list. The BGP Named Community Lists feature allows the network operator to assign meaningful names to community lists and increases the number of community lists that can be configured. A named community list can be configured with regular expressions and with numbered community lists. All rules of numbered communities apply to named community lists except that there is no limitation on the number of community attributes that can be configured for a named community list.
BGP Route-Map Policy List Support	12.0(22)S 12.2(15)T 12.2(18)S 12.2(18)SXD 12.2(27)SBC 15.0(1)S	The BGP Route-Map Policy List Support feature introduces new functionality to BGP route maps. This feature adds the capability for a network operator to group route map match clauses into named lists called policy lists. A policy list functions like a macro. When a policy list is referenced in a route map, all of the match clauses are evaluated and processed as if they had been configured directly in the route map. This enhancement simplifies the configuration of BGP routing policy in medium-size and large networks because a network operator can preconfigure policy lists with groups of match clauses and then reference these policy lists within different route maps. The network operator no longer needs to manually reconfigure each recurring group of match clauses that occur in multiple route map entries.
BGP Support for Named Extended Community Lists	12.2(25)S 12.2(27)SBC 12.2(33)SRA 12.2(33)SXH 12.3(11)T 15.0(1)S	The BGP Support for Named Extended Community Lists feature introduces the ability to configure extended community lists using names in addition to the existing numbered format.
BGP Support for Sequenced Entries in Extended Community Lists	12.2(25)S 12.2(27)SBC 12.2(33)SRA 12.2(33)SXH 12.3(11)T 15.0(1)S	The BGP Support for Sequenced Entries in Extended Community Lists feature introduces automatic sequencing of individual entries in BGP extended community lists. This feature also introduces the ability to remove or resequence extended community list entries without deleting the entire existing extended community list.
BGP 4 Prefix Filter and Inbound Route Maps	Cisco IOS XE 3.1.0SG	