

Configuration Guide

Configuring Border Gateway Protocol in AOS for Releases Prior to 18.03.00/R10.1.0



*This guide only addresses BGP in AOS **data** products using AOS firmware prior to **18.03.00** and AOS **voice** products using AOS firmware prior to **R10.1.0**. For information about BGP configuration for products using AOS firmware later than these releases, refer to [Configuring BGP in AOS for Releases 18.03.00/R10.1.0 or Later](#), available online at <https://supportforums.adtran.com>.*

This configuration and troubleshooting guide will aid in the setup of Border Gateway Protocol (BGP) for ADTRAN Operating System (AOS) products. An overview of BGP general concepts combined with detailed command descriptions provide step-by-step assistance for the most common BGP configurations. The troubleshooting section outlines proper use of **show** and **debug** commands to verify that BGP has been configured properly on the AOS product(s).

This guide consists of the following sections:

- [BGP Overview on page 2](#)
- [Hardware and Software Requirements and Limitations on page 8](#)
- [Basic BGP Configuration Using the CLI on page 9](#)
- [Additional BGP Configuration on page 12](#)
- [Example Configurations on page 35](#)
- [Configuration Command Summary on page 58](#)
- [Troubleshooting on page 66](#)

BGP Overview

BGP is an Exterior Gateway Protocol (EGP) that is used within the Internet and multinational organizations. EGP is one of two different types of dynamic routing protocols. The other protocol is Interior Gateway Protocol (IGP). The difference between the two protocols is that IGPs (for example, Routing Information Protocol (RIP), Open Shortest Path First (OSPF)) operate within an autonomous system (AS), whereas EGPs allow routes to be exchanged between different autonomous systems. Typically, an AS is defined by the boundaries of an organization. As an EGP, BGP routers must regulate traffic between networks controlled by organizations with different policies. BGP is designed to allow administrators to customize policies for route exchange. The following are some characteristics of BGP that make it an appropriate protocol for connecting different autonomous systems:

- BGP can filter both the routes it receives and those that it sends according to bit length, thereby minimizing the number of routes exchanged.
- BGP uses policies to determine best routes rather than per-hop counts used in RIP or link states used in OSPF. Each AS can set their own policy.
- BGP routers communicate only with manually configured neighbors.
- You can configure different policies for route exchange with different neighbors.

BGP Advantages

Static routing, OSPF, and RIP are simple to configure, have low overhead, and are well suited for medium-to-small networks. However, BGP offers several advantages, particularly in more complex environments:

- Unlike routers using static routing, routers running BGP can automatically respond to connections that are down and changes in network topology. Multiple protocol layer switching (MPLS) networks allow an organization to change its IP addressing scheme without notifying the service provider.
- BGP can handle complex applications in which the private network connects to multiple service provider routers or multiple service providers. BGP can be configured to balance loads among these connections.
- BGP is the native protocol implemented by service providers, which decreases problems caused by redistributing other routing protocols into BGP.
- BGP is policy based; therefore, organizations can maintain tight control over the routes transmitted and accepted.

This configuration guide focuses on the common BGP applications of CE routers. The most popular scenarios are illustrated in [Example Configurations on page 35](#).

Autonomous Systems

An AS is a group of networks administered by the same authority. Usually, an AS is the same as an organization. If the AS connects directly to the Internet, then the organization must acquire a unique number from the Internet Assigned Numbers Authority (IANA). However, many organizations connect to the Internet through a service provider who has already been assigned an AS number for connecting to the Internet. The service provider's AS is subdivided into areas and includes any organizations connecting to the Internet through them.

By defining autonomous systems, a demarcation point is created between organizations and the Internet. Within the AS, information about all networks can be transmitted to every other router using some type of IGP, such as OSPF or RIP. However, since the Internet is so vast, it would be impractical for all routers to hold routes to all networks in all of the autonomous systems. The capability of the Internet to identify an entire organization by means of a unique integer allows a large reduction in the amount of information that needs to be held in routing tables or transmitted in routing updates between autonomous systems. The result is a considerably smaller amount of summarized information exchanged using BGP between different autonomous systems. This level of hierarchy is essential to the successful operation and maintenance of the Internet.

Initially, AS numbers were only 16-bit integers. However, as the Internet continues to experience massive growth, engineers have expanded the AS number space from 2 bytes to 4 bytes, providing over 4 billion AS numbers. January 2009 marked the transition to allocate 32-bit AS numbers by default.



As of release 18.1, AOS supports 32-bit AS numbers.

External BGP (eBGP) and Internal BGP (iBGP)

eBGP uses BGP between peers in different autonomous systems, and iBGP uses BGP between peers within the same AS.

Do not confuse eBGP with EGP, a nearly obsolete protocol once used on the Internet. Also, do not confuse iBGP with an IGP, such as OSPF. Service providers use iBGP to distribute BGP routes between routers within an AS. However, service providers usually still need to run an IGP to generate routes for traffic within the AS.

eBGP allows an organization to peer and exchange routes with a service provider. MPLS implementations by a service provider allow the exchange of private subnets between remote sites through the provider's network. eBGP is also used between different service providers to facilitate the Internet backbone. Customers can peer using eBGP with service providers to exchange public IP addresses that they have purchased from IANA or to exchange private routes to other remote sites over an MPLS circuit.

iBGP is used between BGP routers within the same AS. iBGP routers prevent routing loops by following a rule where no updates learned from internal peers are sent to other internal peers. This means that iBGP routers will only propagate a route if the route originates in the transmitting router or if the route is directly connected to the transmitting router. As a result, iBGP routers must be fully meshed to have a complete knowledge of the network.



AOS devices have a 16 BGP neighbor limit. Since iBGP routers must be fully meshed, AOS devices are only suitable for small scale iBGP applications.

Route reflectors are sometimes identified and used in an iBGP network to reduce the number of peering sessions required.



AOS devices cannot act as a route reflector for iBGP setups. AOS devices can peer with a route reflector acting as a client, however, the connection is no different than a regular iBGP peer.

BGP Neighbors and Messages

Unlike other routing protocols, BGP does not automatically discover its neighbors. The transport medium for BGP is Transmission Control Protocol (TCP), port 179. TCP is a connection-oriented protocol; thus, providing an extra layer of reliability for BGP communication.

BGP neighbors must be manually configured. After the three-way TCP handshake has been established between two BGP neighbors, a peering session is established through an open message that contains a hold time and BGP router ID. During the exchange of the open message, a router will decide if their neighbor is in the same AS (iBGP) or a different AS (eBGP). Once a session is established, keepalive messages are periodically exchanged between the peers to maintain connections and verify paths held by the router sending the keepalive.

BGP routers send update messages to their neighbors whenever a path changes. There is only one path per update. Each update message contains information on the path to a destination network and the path attributes, such as origin, AS path, and neighbor. Routes that are no longer available or withdrawn routes are included in updates.

Notification messages are sent when an error has been received. The BGP connection is closed after the notification message has been sent.

BGP Attributes

BGP attributes are properties that are used to determine the best route to a destination when multiple paths exist to a single destination. An understanding of how BGP attributes influence route selection is important when designing networks.

The following BGP attributes are supported by AOS:

- LOCAL_PREF
- MULTI_EXIT_DISC or MED
- Origin
- AS_PATH
- NEXT_HOP
- Community

Local Preference

The local preference (LOCAL_PREF) attribute is used to choose a path when there are multiple exit points from the local AS. Adjusting the local preference value can affect the local router's decision, as well as the decision of other iBGP routers in the same AS when determining the best route to a destination.



The higher the local preference, the more desirable a route becomes.

Multi-Exit Discriminator

The multi-exit discriminator (MULTI_EXIT_DISC or MED) attribute is used to influence eBGP neighbors to select a certain path for inbound traffic into the AS that is advertising the metric.



The lower the MED metric, the more desirable a route becomes.



The MED metric is merely a suggestion to the external neighbor as to which path should be used inbound to the local AS. This is because the external AS that is receiving the MED metric might be using other BGP attributes for route selection.

Origin

The origin attribute identifies the source of a learned route. There are three possible values for this attribute:

- **IGP** - The route has been learned by an iBGP neighbor that is internal to the local AS. This value is set when the **network** command is issued from BGP configuration mode to inject a route into BGP.
- **EGP** - The route has been learned from an eBGP neighbor.
- **Incomplete** - The origin of the route is not known or learned in some other way. A route that has been redistributed into BGP or specified using the **network** command is set to this value.

AS Path

The AS path (AS_PATH) attribute consists of an ordered list of AS numbers that the route advertisement has crossed. Each time a router advertisement passes through an AS, the AS number is added to the list. The AS_PATH attribute is used by BGP as a loop avoidance mechanism. A route is rejected anytime a router detects its own AS number in a route advertisement.

The AS path can also be used to determine the best route to a given destination. If two identical routes are learned and all other attributes are equal, the one with the fewest number of traversed autonomous systems is preferred.

Next Hop

The next hop (NEXT_HOP) attribute is the IPv4 address that is used to reach the advertising router. Since BGP routes traffic from AS to AS, the default next hop that is advertised is the next AS.

Community

The community attribute provides a way to group routes together in communities and apply a consistent policy to the group. The policies can be set to control routing decisions, such as acceptance, preference, and redistribution.



For more detailed information on many of these BGP attributes and the related commands used to influence path selection, refer to [Additional BGP Configuration on page 12](#).

BGP Path Selection

When BGP receives advertisements for the same route from multiple sources, one path is selected as the best path and stored in the routing table. The decision logic used by BGP to determine the best path is fairly extensive. The following BGP criteria are used in AOS to select the best path to a destination:

1. Prefer the path with the higher LOCAL_PREF value.
2. If the LOCAL_PREF value is identical, compare local-origination status. Prefer a route injected into BGP via the **network** <ipv4 address> **mask** <subnet mask> command issued from BGP configuration mode over a redistributed route.
3. If the local origination status is identical, prefer the shortest AS_PATH distance.
4. If the AS_PATH distance is identical, prefer lower origin type (where routes originally injected via the **network** <ipv4 address> **mask** <subnet mask> command issued from BGP configuration mode or aggregation (IGP) are lower in origin than routes learned from a neighbor using eBGP. Routes originally injected by redistribution into BGP (incomplete) have the highest origin value).
5. If the origin type is identical, prefer the route with the lowest MULTI_EXIT_DISC value.
6. If the MULTI_EXIT_DISC value is identical, prefer eBGP paths over iBGP paths.
7. If the paths are still identical, prefer the path through the closest IGP neighbor.
8. Compare and prefer lower value for any other metrics on the route.
9. Compare and prefer the route from the router with the lowest router ID.
10. Compare and prefer the route that came from the lowest neighbor IPv4 address.

VRF and MPLS

The following information is provided to enhance the understanding of how service providers are able to maintain separation of private routes that belong to different customers. All BGP applications discussed in this configuration guide simply require the AOS device to be configured for eBGP when connecting to a service provider. AOS devices support an implementation of multi-VRF, but this functionality is not used or needed in the BGP applications discussed in this configuration guide.



As of AOS firmware release 18.3, Multi-VRF BGP support is included in BGP. For more information about this feature, refer to the configuration guide [Configuring BGP in AOS](#), (for AOS data products using AOS firmware 18.03.00 or later, and AOS voice products using AOS firmware R10.1.0 or later). The guide is available online at <https://supportforums.adtran.com>.

A service provider uses virtual routing and forwarding (VRF) to separate one customer's routes from another's and MPLS to ensure that the routes reach only the authorized remote sites. Without VRF, customers could not transmit private network routes between remote sites; the service provider's routers would have no way of knowing which route belonged to which customer.

For example, in [Figure 1 on page 7](#) the provider's edge routers connect to two independent customers, Customer A and Customer B. Each customer would like to communicate private information between their own respective sites. Customer A Site 1 uses an IPv4 network address of 192.168.1.0 /24 and Customer A Site 2 uses an IPv4 network address of 192.168.2.0 /24. These identical network addresses are also used for Customer B Sites 1 and 2, respectively. The provider's router must be able to associate one 192.168.1.0 /24 with the public address for Customer A Site 1 and the other with Customer B Site 1. Likewise, separate associations for 192.168.2.0 /24 for Customer A and B Site 2 must be maintained.

The provider's edge router separates routes by the physical or logical interface on which they arrive. The router then stores routes from each customer in a separate VRF routing table. Different customers' routing tables cannot be combined.

The service provider edge router connecting to the local site forms an MPLS label switched path (LSP) with the service provider edge router connecting to the authorized remote site. (An LSP resembles a dynamic permanent virtual circuit (PVC).) The edge routers mark packets with an MPLS label that directs them toward the other router through the LSP so that only Customer A sites receive Customer A routes.

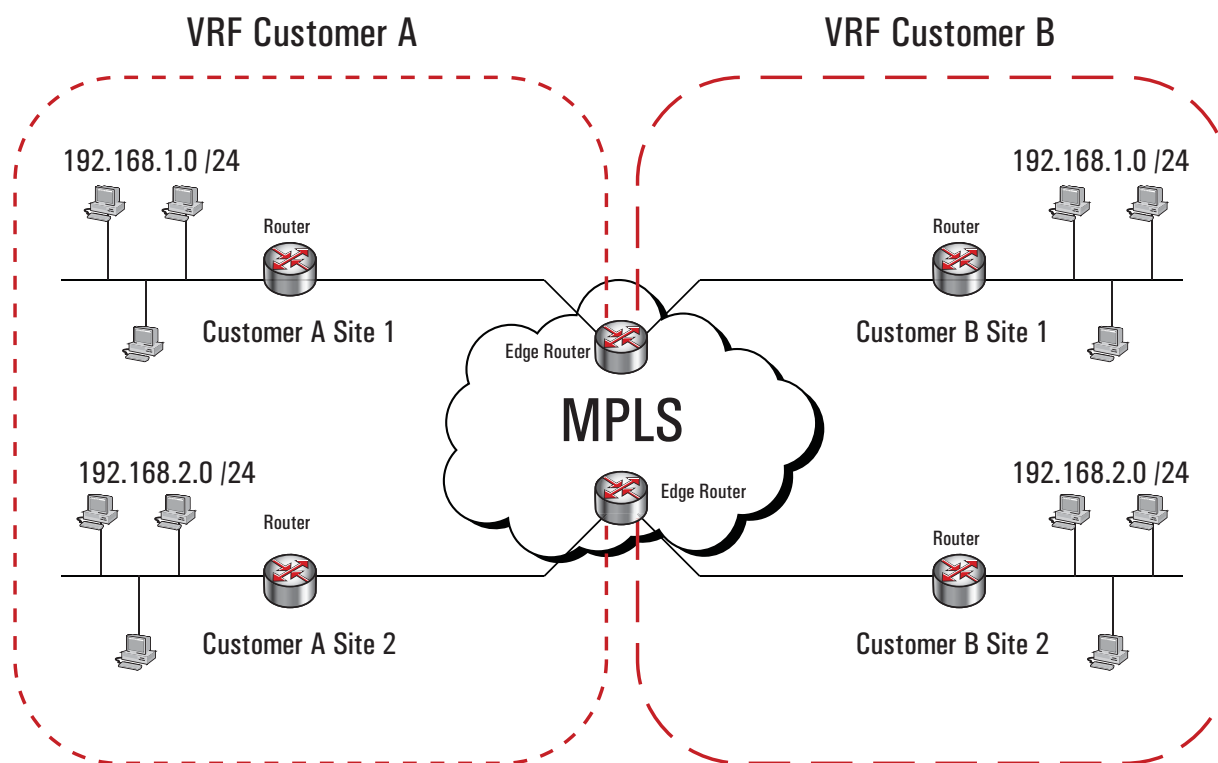


Figure 1. Traffic Separation Using VRF

Multihoming

Multihoming is when a router has more than one connection to the Internet. BGP is especially useful for this type of application. When a router has only one Internet connection, static routing is normally implemented. The lower overhead and simpler configuration outweigh the loss in control. It does not matter that the router cannot balance loads because all external traffic has the same destination.

The Internet connections for a multihomed router can be to the same provider or to two different providers. Regardless, when a router has two Internet connections, the router must decide which connection to use for certain traffic. BGP allows a router to receive different routes from the various service providers to which it connects. Based on this information, traffic can be routed accordingly. BGP also allows a router to advertise different networks to different neighbors. This practice keeps one link from being overused while leaving the other idle and, therefore, ensures that your organization actually receives the benefit of the connections for which it has paid.



When multihoming to two different service providers, it is good practice to advertise only intended networks to prevent becoming a transit AS. Without route filtering, BGP will advertise any BGP routes learned - including routes to service provider B, the router learned from service provider A.

Hardware and Software Requirements and Limitations

eBGP is supported in AOS products running version 8.1 or later.

iBGP is supported in AOS products running version 10.1 or later.

eBGP and iBGP are available on AOS data products as outlined in the [AOS Product Feature Matrix](#) (available online at <https://supportforums.adtran.com>).



AOS devices have a 16 BGP neighbor limit. Since iBGP routers must be fully meshed, AOS devices are only suitable for small scale iBGP applications.



Support for the full Internet forwarding information base (FIB) from two peers requires either a NetVanta 4430 or NetVanta 5305 with 512 MB of random access memory (RAM).

As of AOS firmware release 18.3, the commands, organization, and configuration of BGP changed. If you are using firmware version 18.3 or later refer to the configuration guide [Configuring BGP in AOS](#) (for AOS data products running firmware 18.3 or later, and voice products running firmware R10.1.0 or later) for the correct configuration information. This guide is available online at <https://supportforums.adtran.com>.

Basic BGP Configuration Using the CLI

There are several commands that must be issued for BGP to operate at the basic level. The following steps outline the minimum configuration required to enable BGP on an AOS device.

Step 1: Enable BGP and Specify the Local AS

When enabling BGP, the local AS number must be specified from the Global Configuration mode:

```
(config)#router bgp <AS number>
```

<AS number> Specifies the AS number of the local system of which this BGP router is a member. Range is **1** to **4294967295**.



Upon entering this command, the AOS device is now in BGP Configuration mode. All subsequent commands in this section are entered from the BGP Configuration mode unless otherwise noted. If you are using AOS firmware 18.3 or later, please note that the command syntax and organization of BGP has changed. Refer to the configuration guide [Multi-VRF in AOS](https://supportforums.adtran.com), available online at <https://supportforums.adtran.com> for more information.

Step 2: Advertise Local Networks

Specify the local networks that remote sites should be able to access. Only networks that originate within the local AS should be advertised. The following command is used from the BGP Configuration mode to allow BGP to advertise a network:

```
(config-bgp)#network <ipv4 address> mask <subnet mask>
```

<ipv4 address> Specifies the IPv4 network address for the neighbor that AOS will advertise over BGP. IPv4 addresses should be expressed in dotted decimal notation (for example, **10.10.10.0**).

<subnet mask> Specifies the subnet mask that corresponds to a range of IPv4 addresses (network) or a specific host. Subnet masks can be expressed in dotted decimal notation (for example, **255.255.255.0**).

For example, to advertise the private network 10.1.10.0 255.255.255.0, enter:

```
(config-bgp)#network 10.1.10.0 mask 255.255.255.0
```

BGP is a classless protocol. Therefore, networks with variable length subnet masks can be specified. BGP can send out a route summary for the entire range of local subnets. For example, a customer's site includes 16 /24 networks from 10.1.0.0 /24 to 10.1.15.0 /24, which together make up network 10.1.0.0 /20. You can specify the entire range of subnets by entering:

```
(config-bgp)#network 10.1.0.0 mask 255.255.240.0
```

Since the BGP router is advertising a route, it searches its routing table for a route to the specified networks. It then sends this route to all authorized neighbors.



The subnet mask is an integral part of the IPv4 network address. If the BGP interface is specified to advertise routes to network 10.1.0.0 /20, it will not advertise routes to network 10.1.0.0 /16 or 10.1.0.0 /24. Therefore, when advertising a network or range of networks, it must be verified that the routing table includes the exact route that has been specified (including the same subnet mask or corresponding prefix length).

If the routing table does not include a route that has been specified in BGP, a null route must be configured. For example, a routing table only includes routes to the 24-bit networks, but not to the 20-bit network that contains them all. A route to network 10.1.0.0 /20 must be manually added so that the BGP interface can advertise it. The route is added from the Global Configuration mode with the **null 0** keyword indicating the next-hop address:

```
(config)#ip route 10.1.0.0 255.255.240.0 null 0
```



If a more specific route (for example, from connected interfaces, static routes, routing protocols, etc.) other than the one to null 0 does not exist in the routing table, then all traffic for the specified subnet will be dropped. Null 0 routes should only be used when a more specific route is available to which to direct traffic.

Step 3: Configure at Least One BGP Neighbor

BGP is different from many routing protocols because it does not allow a router to automatically search for peers from which to obtain routes. A separate BGP neighbor must be configured for each router with which the local router will communicate. Optional policies for each neighbor can be configured to dictate which routes the BGP interface sends to and accepts from the neighbor (refer to [Additional BGP Configuration on page 12](#)).

Set the BGP Neighbor ID (IPv4 address)

BGP identifies a peer router by its IPv4 address. A neighbor's ID is set when a policy is created for it. Enter the BGP Neighbor Configuration mode from the BGP Configuration mode as follows:

```
(config-bgp)#neighbor <ipv4 address>
```

<ipv4 address> Specifies the IPv4 address for the neighbor. IPv4 addresses should be expressed in dotted decimal notation (for example, **10.10.10.1**) or as a prefix length (for example, **/24**).



The IPv4 address entered in this command must match the address for the interface that the remote router is using as its update source.



The local router must be able to reach the IPv4 address configured as the neighbor ID. View the routing table and verify that it includes a route to this address.

Specify the Remote AS Number

The AS to which this neighbor belongs must also be specified. The **remote-as** command is issued from the BGP Neighbor Configuration mode:

```
(config-bgp-neighbor)#remote-as <value>
```

<value> Specifies the AS number. Range is **1** to **4294967295**.



*When configuring eBGP, the remote AS number must be different from that of the local router (which is defined using the **router bgp** command).*

The following example configures a remote AS number of **200** for neighbor **172.16.1.2**:

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

```
(config-bgp)#router bgp-neighbor 172.16.1.2
```

```
(config-bgp-neighbor)#remote-as 200
```

Assuming the neighbor is properly configured, the AOS device should now be able to connect to the neighbor and exchange routes with it.

Step 4: Save the Configuration

The configuration can be saved directly from the BGP Configuration mode or BGP Neighbor Configuration mode:

```
(config-bgp-neighbor)#do write
```



*When the command to save the configuration is issued from the Enable mode, the command is: **write**.*

Additional BGP Configuration

Depending on the network, additional BGP configuration might be needed. This section contains detailed explanations on additional BGP-related options that are available in AOS devices. The following topics are listed in this section:

- [Advertisement Interval on page 12](#)
- [BGP Communities on page 13](#)
- [Description on page 16](#)
- [Distance on page 16](#)
- [Distribute List on page 17](#)
- [eBGP Multihop on page 17](#)
- [Fast External Failover on page 17](#)
- [Hold Timer on page 18](#)
- [Local AS on page 18](#)
- [Log Neighbor Changes on page 19](#)
- [Maximum Paths on page 19](#)
- [Multi-Exit Discriminators \(MEDs\) on page 19](#)
- [Next-Hop Self on page 19](#)
- [No Default Originate on page 20](#)
- [Password on page 21](#)
- [Prefix List on page 21](#)
- [Route Map on page 23](#)
- [Applying a Route Map Entry to a BGP Neighbor on page 30](#)
- [Router ID on page 31](#)
- [Soft Reconfiguration Inbound on page 31](#)
- [Update Source on page 32](#)
- [Establishing Routing Preference on page 33](#)

Advertisement Interval

Use the **advertisement-interval** command to configure AOS to specify how long the BGP process waits before sending updates to the neighbor. This command sets the minimum interval between sending updates to the specified neighbor.

(config-bgp-neighbor)#**advertisement-interval** <value>

<value> Specifies the advertisement interval in seconds. Range is **0** to **600** seconds.



*By default, the advertisement interval is **30** seconds for external neighbors and **5** seconds for internal neighbors.*

BGP Communities

Within BGP, the community is an optional attribute that can be used for identification, security, or to signal a BGP peer that it should take a particular action. When used for identification and security, the attribute adds another layer of complexity that requires special configuration to bring the BGP connection up. When used to signal a peer, the attribute is commonly used when that peer is ignoring other attributes of the BGP advertisement (which is often the case in MPLS networks).

A route can be a member of one or more BGP communities. A community is simply a way of grouping routes together and applying a consistent policy to the group. A route can be placed into a community according to any attribute in that route. One of the most common ways of grouping routes is by IPv4 network address and prefix length, which is defined in a prefix list and ultimately referenced in a route map (refer to [Prefix List on page 21](#) and [Filtering Routes According to Network IPv4 Address on page 25](#)). In order for a route's membership in a community to have significance, administrators must define policies that apply to the community.

Several commands must be issued when configuring BGP communities on an AOS device. A route map must be configured, followed by the appropriate commands for sending and/or receiving a community string. Also, the **send-community standard** command must be enabled for any neighbor that will be sending or receiving community attributes (refer to [Enabling an AOS Device to Send or Receive BGP Communities on page 15](#)).

Configuring a Community List

A community list is used when an AOS device receives a community string using BGP. A community list can be used to:

- Select the communities to which the router will apply a specific policy, such as filtering advertised routes to those communities or applying policies to inbound routes from those communities.
- Define the communities that the BGP interface will delete from routes.

Use the **ip community-list** command to create a community list for BGP route map use. The communities defined in this command should match an existing community string that the AOS device receives. This command is issued from the Global Configuration mode:

```
(config)#ip community-list <name>
```

<name> Specifies the community list name. This is an arbitrary name for the list that is referenced in a BGP policy using community strings.

This command places the user in the Community List Configuration mode where one or more well-defined communities can be specified. A value for a privately defined community might also be specified.

The following command adds an entry to the community list that either **permits** or **denies** BGP routes containing the specified community string in the community attribute:

```
(config-comm-list)#[permit | deny] [<value> | internet | local-as | no-advertise | no-export]
```

<value> Specifies a privately defined community for routes that contain this value in their community attribute. This is a numeric value that can be an integer from **1** to **4294967295** or string in the form **aa:nn**, where **aa** is the AS number and **nn** is the community number. Multiple community number parameters can be present in the command.

internet Specifies routes that contain the reserved community number for the Internet community.

local-as Specifies routes that contain the reserved community number for NO_EXPORT_SUBCONFED. Routes containing this attribute should not be advertised to external BGP peers.

no-advertise Specifies routes that contain the reserved community number for NO_ADVERTISE. Routes containing this attribute should not be advertised to any BGP peer.

no-export Specifies routes that contain the reserved community number for NO_EXPORT. Routes containing this attribute should not be advertised to BGP peers outside a confederation boundary.

[Table 1](#) below summarizes the well-known communities and the policy expected for these communities.

Table 1. Well-Known Communities

Community	Advertise To
internet	All peers
local-as	Peers in the local AS
no-advertise	No peers
no-export	Internal peers only



*Multiple communities can be specified by stringing several keywords in the same command (for example, (config-comm-list)#**permit local-as no-export**).*

After the community list has been defined, it must be referenced in a route map entry. Refer to [Route Map on page 23](#) for information on creating route maps. Refer to [Filtering Routes According to Community on page 27](#) for information on how to reference a community list within the route map.

Defining a Community Policy

The community policy defines the action that will be taken on routes depending on the community string attached to those routes. This can be used to set certain attributes, such as local preference or metric, for routes in the community. The **set** commands are used to define the community policy on a community string. These commands are issued within Route Map Configuration mode (refer to [Route Map on page 23](#)) and the route map is applied outbound to a BGP neighbor (refer to [Applying a Route Map Entry to a BGP Neighbor on page 30](#)).



Before placing a route in a community, you should contact your service provider and discuss what options it supports for various communities. You should also consult your organization's policies.

Use the **set community** command to select the peers to which a neighbor advertises routes in the community:

```
(config-route-map)#set community <value> [add | internet | local-as | no-advertise | no-export]
```

<value> Specifies a privately defined community for routes serviced by this route map. This is a numeric value that can be an integer from **1** to **4294967295** or string in the form *aa:nn*, where *aa* is the AS number and *nn* is the community number. Multiple community number parameters can be present in the command.

add Appends the listed community number to the end of the community attribute for routes serviced by this route map.

internet Sets the community attribute to the reserved INTERNET community number for routes serviced by this route map.

local-as Sets the community attribute to the reserved NO_EXPORT_SUBCONFED community number for routes serviced by this route map. Routes containing this attribute should not be advertised to eBGP peers.

no-advertise Sets the community attribute to the reserved NO_ADVERTISE community number for routes serviced by this route map. Routes containing this attribute should not be advertised to any BGP peer.

no-export Sets the community attribute to the reserved NO_EXPORT community number for routes serviced by this route map. Routes containing this attribute should not be advertised to BGP peers outside a confederation boundary.

none Removes all communities from BGP routes serviced by this route map.



See [Table 1 on page 14](#) for a summary of the well-known communities (for example, INTERNET, NO_EXPORT_SUBCONFED, etc.) and the policy expected for these communities.

Attributes, such as local preference and metric, can also be defined for the community string. A discussion of the **set** commands used to define these community attributes begin with [Prepending Private AS Numbers for Load Balancing on page 28](#).

Enabling an AOS Device to Send or Receive BGP Communities

A BGP neighbor must be configured for BGP communities before it is able to send or receive a community attribute. Use the **send-community standard** command to enable this peer to accept a community attribute and add the community attribute to any advertisement sent by this peer. This command is issued from the BGP Neighbor Configuration mode:

```
(config-bgp-neighbor)#send-community standard
```

Deleting Communities from a Route

BGP communities are not completely standardized. External neighbors can place routes in a community for which the local network does not define a policy or for which it defines a substantially different policy. A network administrator might not want to apply the policies that the external neighbor is requesting with the community attribute. As a result, the administrator might need to remove certain communities from inbound routes in order to enforce the organization's policies.

Deleting communities from a route can be accomplished by first creating a community list that *permits* the communities that are to be deleted (refer to [Configuring a Community List on page 13](#)). Next, create a route map (refer to [Route Map on page 23](#)) and use the **set comm-list delete** command to specify a list of communities to delete:

```
(config-route-map)#set comm-list <name> delete
```

<name> Specifies the name of the community list that contains the communities to delete.

Apply the route map to the neighbor as an inbound policy (refer to [Applying a Route Map Entry to a BGP Neighbor on page 30](#)).

If a network defines local communities, the administrator might need to remove these from the routes before the local router advertises the routes to an external neighbor. This scenario requires the administrator to configure a community list that permits the local communities. The community list is then matched to a route map entry and the route map is applied to the BGP neighbor as an outbound policy.



*If a route map is already in place to set policies or filter routes, this route map should also be used to delete the specified communities. Each route map sequence number entry that could potentially let in a route should be evaluated to determine if deleting communities also applies. If so, the **set comm-list <name> delete** command should be added for each applicable route map sequence number. This is because the router stops processing a route map as soon as it finds a match. If a separate, earlier entry permits a route (as described in the preceding paragraphs), the router will immediately add the route to the BGP database without applying any policies that are set in later entries.*

Description

Use the **description** command to identify the specified interface connected to a BGP neighbor.

```
(config-bgp-neighbor)#description <text>
```

<text> Identifies the specified interface using up to **80** alphanumeric characters.

Distance

The administrative distance is a local variable that allows a router to choose the best route when there are multiple paths to the same network. Use the **distance bgp** command to set the administrative distance for BGP routes.

```
(config-bgp)#distance bgp <external> <internal> <local>
```

<external> Sets the administrative distance for BGP routes learned via eBGP sessions. Range is **1** to **255**.

<internal> Sets the administrative distance for BGP routes learned via iBGP sessions. Range is **1** to **255**.

<local> Sets the administrative distance for BGP routes learned via the network command and redistribution. Range is **1** to **255**.



*By default, external is set to **20**, internal to **200**, and local to **200**. Normally, these default settings should not be changed.*



Routes with lower administrative distances are favored.

Distribute List

Use the **distribute-list** command to add route filtering functionality by assigning inbound and outbound IPv4 access control lists (ACLs) to a BGP neighbor. Only one inbound/outbound pair of ACLs can be configured for a particular neighbor.

(config-bgp-neighbor)#**distribute-list** <ipv4 acl name> [**in** | **out**]

<ipv4 acl name> Specifies an IPv4 ACL name. This is a standard or extended IPv4 ACL against which the contents of the incoming/outgoing routing updates are matched.

in Applies route filtering to inbound data.

out Applies route filtering to outbound data.



Refer to the configuration guide [IP ACLs in AOS](https://supportforums.adtran.com) (available online at <https://supportforums.adtran.com>) for information on how to create a standard or extended IPv4 ACL.

eBGP Multihop

Use the **ebgp-multihop** command to configure the maximum hop count for BGP messages to a neighbor.

(config-bgp-neighbor)#**ebgp-multihop** <value>

<value> Specifies the maximum hop count of BGP messages to a neighbor. Range is **1** to **254** hops.

This command allows an eBGP neighbor to be on a network that is not directly connected. The default time to live (TTL) for BGP messages is **1** since eBGP peers are normally directly connected. However, in certain applications, a non-BGP device, such as a firewall or router, might reside between eBGP peers. The **ebgp multihop** command is required in this case to allow updates to have a TTL greater than 1 and to allow received BGP updates to be added to the BGP table when the next-hop address is not directly connected.

It is also good practice to create a static route to reach the eBGP neighbor when the neighbor is more than one hop away, as well as creating a backup route pointing to null 0 interface with a higher administrative distance. These routes prevent unnecessary BGP traffic from traversing the wrong connection and prevent the BGP neighbor relationship from incorrectly attempting to establish itself on that connection. It is also possible that sending invalid BGP traffic to some providers might result in the connection being automatically disabled by the provider as a security measure.

Fast External Failover

Use the **bgp fast-external-failover** command to enable the fast external failover feature.

(config-bgp)#**bgp fast-external-failover**

When failover is enabled, if the link interface goes down between this router and a BGP neighbor, the BGP session with the neighbor is immediately cleared. When failover is disabled and the link goes down, the session is maintained until the BGP hold timer expires (refer to [Hold Timer on page 18](#)).

Hold Timer

Use the **hold-timer** command to set the default hold time for BGP neighbors. The command can be issued in BGP configuration mode to set the default hold time for all neighbors in that BGP process, or the command can be issued in BGP Neighbor Configuration mode to set the hold time for only that neighbor.

```
(config-bgp)#hold-timer <value>
```

or

```
(config-bgp-neighbor)#hold-timer <value>
```

<value> Specifies a time interval (in seconds) within which a keepalive must be received from a peer before it is declared a dead peer. Range is **0** to **65535** seconds.

The peers will negotiate and use the lowest configured setting. The keepalive interval will be set to one-third of the negotiated hold time.



NOTE

The default hold time is 180 seconds.

Local AS

Some multihoming network designs require a customer to appear as a different AS number to individual service providers. Also, service providers sometimes assign the same AS to multiple sites, which can cause problems due to BGP's loop avoidance check mechanism (refer to [Example 2 on page 37](#)). The **local-as** command rectifies both situations by substituting an AS number that is different from the one specified in the command **router bgp <AS number>**.

Use the **local-as** command to specify an AS number for the unit to use when communicating with this BGP neighbor.

```
(config-bgp-neighbor)#local-as <value>
```

<value> Specifies the AS number to use when communicating with this neighbor. The value must be different than the AS number for this router and the peer router. It is only valid for eBGP connections. Range is **0** to **4294967295**. When **0** is used, it indicates that the BGP process local AS is used, because **0** is not a valid AS number.



NOTE

*Network advertisements from routers using the **local-as** command contain only the AS specified within that command. When the **local-as** command is used, the AS specified using the **router bgp** command is not advertised.*



NOTE

*By default, the **local-as** value is set to match the configured router BGP value.*

Log Neighbor Changes

Use the **bgp log-neighbor-changes** command to control the logging of neighbor state changes. This command controls logging of BGP neighbor state changes (up/down) and resets. This information is useful for troubleshooting and determining network stability.

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

Maximum Paths

Use the **maximum-paths** command to specify the number of equal cost parallel routes (shared paths) learned by BGP that can be exported to the route table. When IP load sharing is enabled, traffic is balanced to a specific destination across up to six equal paths.

```
(config-bgp)#maximum-paths <value>
```

<value> Specifies the number of equal cost parallel routes learned by BGP that can be exported to the route table. Valid range is 1 to 6.

Multi-Exit Discriminators (MEDs)

Use the **bgp** command to instruct AOS on how to handle multi-exit discriminators (MEDs) for all routes from the same AS.

```
(config-bgp)#bgp [always-compare-med | compare-med | deterministic-med | ignore-med]
```

always-compare-med Always compares MEDs for all paths for a route, regardless of the AS through which the route passes.

compare-med Compares MEDs for all received routes.

deterministic-med Compares the MEDs for all routes received from different neighbors within the same AS.

ignore-med Disregards MEDs for all received routes.



Refer to [Setting a MED Metric on page 28](#) for information on how to configure the value of the MED metric advertised outbound to BGP neighbors.

Next-Hop Self

Use the **next-hop-self** command to force the NEXT_HOP attribute to be changed to this unit's IPv4 address for each network it advertises to the neighbor address.

```
(config-bgp-neighbor)#next-hop-self
```

IGPs, such as RIP and OSPF, always use the source IPv4 address of a routing update as the next-hop address for each network that is placed in the routing table. Conversely, since BGP routes AS-to-AS, the default next hop that is advertised is the next AS. This behavior can present a problem in situations where an iBGP router learns about networks outside of its AS through one of its iBGP peers. By default, the next-hop address for the external networks advertised to the iBGP router is the entry point for the next AS. When the iBGP router receives packets destined for one of the external networks, it performs a recursive lookup of the entries in its own IGP routing table to determine how to reach the BGP next-hop address. Unless the iBGP router has a static route or an entry in its IGP routing table indicating how to reach the edge router in

the external AS, packets destined for those networks will be dropped. A remedy for this scenario is for the iBGP peer to advertise its own IPv4 address as the next-hop address to the external networks. Consider the following example:

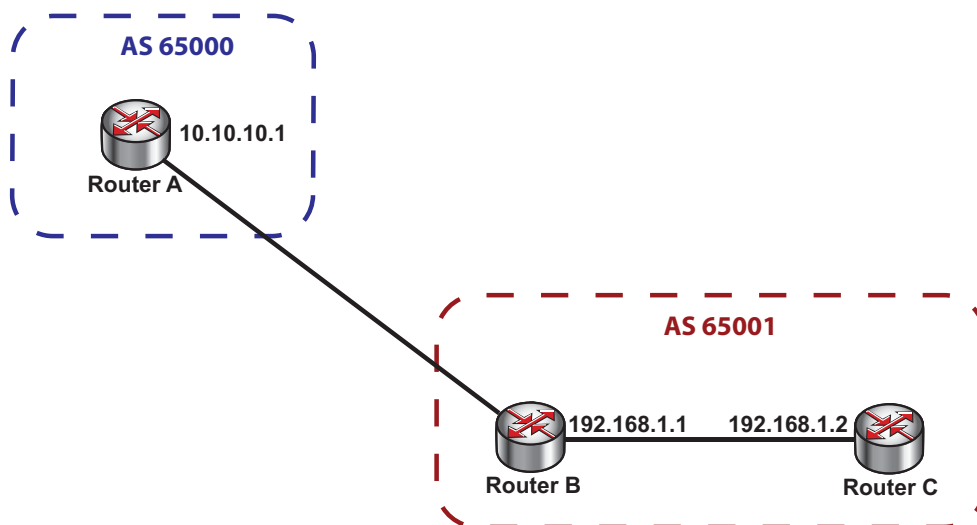


Figure 2. Using the Next-Hop-Self Command between iBGP Peers

Router B in AS 65001 has an eBGP neighbor relationship with Router A in AS 65000. All networks in AS 65000 are advertised from Router A to Router B with a next-hop IPv4 address of 10.10.10.1. Subsequently, when Router B announces these networks to its iBGP neighbor (Router C), the BGP default setting is to announce that the next hop to reach these networks is the entrance to AS 65000 (10.10.10.1). Router C **MUST** have either a static route or an entry in its IGP routing table, indicating a route to reach the edge router at 10.10.10.1. Otherwise, any information destined for networks in AS 65000 will be dropped by Router C. The **next-hop-self** command can be issued on Router B so that Router B's IP address (192.168.1.1) is advertised to Router C as the next-hop address for networks in AS 65000. Since Router B and C are directly connected, Router C's routing table contains a route to Router B.



*Example 4 on page 41 further demonstrates the use of the **next-hop-self** command.*

No Default Originate

The **no default-originate** command prevents the unit from sending the default route to a BGP neighbor. This is the default setting for AOS devices.

```
(config-bgp-neighbor)#no default-originate
```



*This command cannot be enabled. In other words, the command **default-originate** cannot be entered to enable the default route to be sent to a BGP neighbor.*

The transmission of default routes to BGP neighbors is accomplished by manually entering a default route in the BGP Neighbor Configuration. This is done by entering the following command:

```
(config-bgp-neighbor)#network 0.0.0.0 mask 0.0.0.0
```



An entry for the default route must appear in the IGP routing table in order for the previous command to work in BGP. Refer to [Example 7 on page 49](#) for an illustration using the **network** command to enable transmission of the default route to the eBGP neighbor.

Password

Use the **password** command to enable message digest 5 (MD5) password authentication on Transmission Control Protocol (TCP) segments exchanged with the BGP peer.

```
(config-bgp-neighbor)#password <password>
```

<password> Specifies the password string to be used for authentication. The password is case sensitive and must not exceed **80** characters.



Authentication must be configured on both peers using the same password.

Every BGP TCP segment sent is authenticated. Configuring authentication causes an existing session to be torn down and re-established using the currently specified authentication.

Prefix List

Prefix lists are used in BGP configurations to define the routes that a router can advertise to or receive from a neighbor. Common uses for prefix lists include:

- Preventing a network from becoming a transit for external traffic when multihoming
- Receiving only routes from remote virtual private network (VPN) sites
- Prohibiting the advertisement of a network
- Load balancing outbound traffic

IPv4 address, prefix length, or other attributes must be defined in a prefix list before it can be assigned to a BGP neighbor.



Refer to [Example 3 on page 39](#) for an example using prefix lists to filter routes.

First, use the **ip prefix-list** command to specify a prefix to be matched or a range of mask lengths:

```
(config)#ip prefix-list <name> seq <number> [deny | permit] <network ip/length>
```

```
(config)#ip prefix-list <name> seq <number> [deny | permit] <network ip/length> [ge | le] <value>
```

<name> Specifies the name of the list. Up to **80** characters are allowed in a name.


<number> Specifies the entry's unique sequence number that determines the processing order. Lower numbered entries are processed first. Range is **1** to **4294967294**.

permit <network ipv4 address/length> Permits access to entries matching the specified network IPv4 address and the corresponding network prefix length (for example, **10.10.10.0/24**).

deny <network ipv4 address/length> Denies access to entries matching the specified network IPv4 address and the corresponding network prefix length (for example, **10.10.10.0/24**).


le <value> Specifies the upper end of the range. Range is **0** to **32**.

ge <value> Specifies the lower end of the range. Range is **0** to **32**.

 **NOTE** *If the IPv4 network address is entered without specifying a range for prefix lengths, the router assumes that the route must be an exact match. For example, if the command **ip prefix-list TEST seq 5 permit 10.1.0.0/16** is entered, the BGP interface will only accept routes to the entire 10.1.0.0 /16 subnet. It will not accept routes to a network, such as 10.1.1.0/24, which was subdivided from the /16 network.*

Routes to subnets within the larger network can be permitted or denied by specifying the permitted range of prefix lengths. For example, the filter could allow all routes to subnets in the 10.1.0.0 /16 network with a prefix length up to and including 24:

```
(config)#ip prefix-list TEST seq 5 permit 10.1.0.0/16 ge 24 le 24
```

 **NOTE** *The **ge** keyword indicates that the length must be **greater than or equal to** that specified in order to match. The **le** keyword indicates that the length must be **less than or equal to** that specified in order to match. If **ge** is only specified, the router assumes 32 as the upper limit. If **le** is only specified, the router assumes the IPv4 network address's length as the lower limit.*

A filter that exactly matches a prefix length can be created by entering the length for both the **ge** and **le** values. For example, the filter could allow any routes to a /24 subnet in the 10.1.0.0 /16 range, but not accept a route to the entire 10.1.0.0 /16 network:

```
(config)#ip prefix-list TEST seq 5 permit 10.1.0.0/16 ge 24 le 24
```

Table 2. Common Prefix List Examples

Action	Example Prefix List Command
Deny all routes	ip prefix-list ALL seq 10 deny 0.0.0.0/0 le 32
Allow default route	ip prefix-list DEFAULT seq 10 permit 0.0.0.0/0
Deny default route, but allow everything else	ip prefix-list DEFAULT seq 10 deny 0.0.0.0/0 ip prefix-list DEFAULT seq 20 permit 0.0.0.0/le 32
Only allow RFC 1918 private addresses	ip prefix-list PRIVATE seq 10 permit 10.0.0.0/8 le 32 ip prefix-list PRIVATE seq 20 permit 172.16.0.0/12 le 32 ip prefix-list PRIVATE seq 30 permit 192.168.0.0/16 le 32 ip prefix-list PRIVATE seq 40 deny 0.0.0.0/0 le 32
Only allow 192.168.x.0/24 summarized route	ip prefix-list REMOTES seq 10 permit 192.168.0.0/16 ge 24 le 24

After a prefix list has been defined, use the **prefix-list** <name> command to assign the prefix list to a BGP neighbor and specify whether the list will be used to filter inbound or outbound routes.

```
(config-bgp-neighbor)#prefix-list <name> [in | out]
```

<name> Assigns the specified prefix list to this BGP neighbor.

in Specifies that all inbound BGP route updates received from the neighbor are filtered.

out Specifies that all outbound BGP route updates being sent to the neighbor are filtered.



A prefix list can be used to create even more complicated policies when it is applied to a route map entry rather than a BGP neighbor (as shown above). Refer to [Route Map on page 23](#) for more information on this option.

Route Map

Route maps allow configuration of more complex policies than prefix lists. In addition to filtering routes according to IPv4 network address and prefix length, routes can be filtered according to their AS path, metric value, or BGP community. A BGP community is a group of routes to which a BGP router applies the same policies. Refer to [BGP Communities on page 13](#) for more information on BGP communities.

Route maps can also be configured to apply various attributes to the routes it filters. A route map applied to outbound data determines how the router advertises routes to a neighbor. The **outbound** route map can be configured to perform such tasks as:

- Define the routes that the router can advertise according to specified attributes or prefixes
- Prepend private AS numbers to specific routes to help balance inbound traffic
- Set an MED on specific routes to help balance inbound traffic
- Request that the neighbor advertise the route to certain communities only

When a route map is applied to inbound data, it determines which of the service provider advertised routes the local router accepts. The **inbound** route map can be configured to perform such tasks as:

- Filter external routes according to specified attributes or prefixes
- Apply attributes to filtered routes, including:
 - Local preference
 - Community
 - MED value
 - Prepend AS path
- Delete communities defined for the routes

The route map itself is created first. Matching criteria and attributes are defined within the route map configuration menu. Once a route map has been established, it can be assigned to a BGP neighbor.

Use the **route-map** command to create a route map and enter the Route Map Configuration mode.

```
(config)#route-map <name> <number>
```

```
(config)#route-map <name> [deny | permit] <number>
```

<name> Specifies a name for the route map.

deny Specifies not to process routes matching the specified route map attributes.

permit Processes routes matching the specified route map attributes.

<number> Specifies a sequence number for this route entry. Range is **1** to **4294967295**.

 NOTE

After creating a route map, route map attributes can be defined from the Route Map Configuration mode. Enter ? at the **(config-route-map)#** prompt to explore the available options.

Defining Routes and Attributes to Advertise Outbound or Filter Inbound

The advertisements sent from a BGP interface to a neighbor or received by a BGP interface from a neighbor can be controlled according to the route's:

- IPv4 Network address and prefix length
- AS path
- Community
- Metric

Routes that the BGP interface will advertise outbound or filter inbound are selected by entering a **match** command in a route map entry. The difference between an inbound filter and an outbound filter is seen when the route map is applied to a BGP neighbor as an **inbound** policy rather than an **outbound** policy. Refer to [Applying a Route Map Entry to a BGP Neighbor on page 30](#). For a list of available filtering options for BGP, see [Table 3 on page 24](#).

Table 3. Defining Routes to Be Filtered

Filtering According To	Command Syntax
IPv4 Network address and/or prefix length	match ip address prefix-list <name>
IPv4 ACL	match ip address <ipv4 acl name>
AS_PATH	match as-path <name>
Community	match community <name> exact-match
Metric	match metric <value>

Detailed explanations of the above **match** commands begin with [Filtering Routes According to Network IPv4 Address on page 25](#).

 NOTE

If a BGP route does not contain a qualifying prefix or attribute that matches any of the filters specified in the route map or if a BGP route matches a deny route map entry, then the route will not be allowed in or out.

 NOTE

If the only action required is filtering of routes, then the **match** command is entered and the route map is applied to the BGP neighbor as either an outbound or inbound policy (refer to [Applying a Route Map Entry to a BGP Neighbor on page 30](#)).

If an attribute is to be applied to the route, then a **set** command must be entered in addition to the **match** command. Attributes are applied to the routes selected by the **match** command. The following attributes can be applied to inbound filtered or outbound advertised routes:

- Community
- Prepend AS path
- MED metric
- Local preference
- Delete a community list

Detailed explanations of the **set** command attributes begin with [Prepending Private AS Numbers for Load Balancing on page 28](#).

Filtering Routes According to Network IPv4 Address

One way to use route maps to filter routes is according to the IPv4 network address and/or prefix length. A prefix list is first created to define the routes that are to be filtered by the BGP interface (refer to [Prefix List on page 21](#)). The prefix list delineates either routes that the BGP interface will advertise outbound or inbound routes that should be filtered. An exact route can be specified or a range of prefix lengths for routes to variable length subnets. After the prefix list has been configured, it is referenced in a route map entry. The route map entry is then applied to a BGP neighbor (refer to [Applying a Route Map Entry to a BGP Neighbor on page 30](#)).

Use the **match ip address prefix-list** command to configure the route map to route traffic based on a prefix list route filter:

```
(config-route-map)#match ip address prefix-list <name>
```

<name> Specifies the name of the prefix list.

Another way to use route maps to filter routes according to network address is by using standard or extended IPv4 ACLs. As with prefix lists, an ACL is first created to define the routes that are to be filtered by the BGP interface. Refer to [IP ACLs in AOS](#) (available online at <https://supportforums.adtran.com>) for information on how to create a standard or extended IPv4 ACL. After the ACL has been configured, it is referenced in a route map entry (see below) or applied to a BGP neighbor using the **distribute-list** command (refer to [Distribute List on page 17](#)). The route map entry is then applied to a BGP neighbor (refer to [Applying a Route Map Entry to a BGP Neighbor on page 30](#)).

Use the **match ip address** command to configure the route map to process traffic based on the ACL name defined with the **ip access-list** command:

```
(config-route-map)#match ip address <ipv4 acl name>
```

<ipv4 acl name> Specifies the name of the IPv4 ACL to match.



Use **set** commands to configure any attributes (prepend `AS_PATH`, `MULTI_EXIT_DISC`, `LOCAL_PREF`, etc.) to be applied to the routes prior to associating the route map with the BGP neighbor. Refer to [Prepending Private AS Numbers for Load Balancing on page 28](#) for detailed explanations of the **set** command attributes.



Refer to [Example 6 on page 46](#) for a detailed BGP configuration example featuring the use of route maps to filter routes according to network address.

Filtering Routes According to AS Path

Routes can also be filtered according to the hops listed in the AS_PATH field. For advertised routes, this type of filtering allows a degree of influence over which autonomous systems external neighbors can access. For example, service provider routers can filter routes with paths that include customer AS numbers to prevent themselves from advertising private customer routes to unauthorized peers. Private networks do not typically transmit traffic from AS to AS. Therefore, filtering advertised routes according to AS path is not usually necessary when configuring eBGP in a private network.

A list of the AS paths to be filtered is created first. The AS path list is then referenced in a route map entry to define the paths to be filtered. Lastly, the route map is applied to a BGP neighbor (refer to [Applying a Route Map Entry to a BGP Neighbor on page 30](#)).

Use the **ip as-path-list** command to create AS path lists for route map use:

```
(config)#ip as-path-list <name>
```

<name> Specifies the name of the AS path list.

Next, specify the AS numbers to be filtered and specify whether the routes containing these AS numbers should be permitted or denied:

```
(config-as-path-list)#[deny | permit] <value>
```

<value> Specifies permitting or denying routes that contain this value in their AS_PATH attribute. This is a numeric value that can be an integer from **1** to **4294967295** or a string that follows the format of AS regular expressions to filter an AS path. Refer to [AS Regular Expressions on page 83](#) for a detailed list of valid AS regular expressions.

The AS path list is only compared against the AS_PATH attribute on a BGP prefix, which is also displayed in the output of the **show ip bgp** command.

For example, a router can be permitted to advertise only routes that use both AS 200 and AS 400:

```
(config-as-path-list)#permit (\b200\b.*400\b)|(\b400\b.*200\b)
```

However, the statement above only permits routes that use both AS 200 and AS 400. Permit any routes that use either AS by entering separate statements:

```
(config-as-path-list)#permit \b200\b
```

```
(config-as-path-list)#permit \b400\b
```

Permitting AS number 200 selects any routes that include that value, even if the AS field also includes other values. In other words, entering **permit 200** permits routes containing AS 200, as well as AS 200 and AS 400, while entering **permit 200 400** only permits routes containing both AS 200 and AS 400. Therefore, it might be necessary to explicitly deny any values that should not be included in the field.

Another example might be where the router is allowed to advertise routes that use AS 200 or AS 400, but not routes that force traffic to travel through both AS 200 and AS 400:

```
(config-as-path-list)#deny (\b200\b.*400\b)|(\b400\b.*200\b)
```

```
(config-as-path-list)#\b200\b
```

```
(config-as-path-list)#permit lb400lb
```



*It is important to enter any **deny** commands before the **permit** commands since the router processes statements in the AS path list in the order that they are entered.*

After configuring the AS path list, use the **match** command to reference the list in a route map entry.

```
(config-route-map)#match as-path <name>
```

<name> Specifies the name of the AS path list.



*Use **set** commands to configure any attributes (prepend AS_PATH, MULTI_EXIT_DISC, LOCAL_PREF, etc.) to be applied to the routes prior to applying the route map to the BGP neighbor. Refer to [Prepending Private AS Numbers for Load Balancing on page 28](#) for detailed explanations of the **set** command attributes.*

The route map is then applied to a BGP neighbor (refer to [Applying a Route Map Entry to a BGP Neighbor on page 30](#) for more information).

Filtering Routes According to Community

If a network places routes in communities, the routes that the local router advertises can be filtered according to these communities. The first step to filtering routes according to community is to create a community list (refer to [Configuring a Community List on page 13](#)) that either permits or denies BGP routes based on well-known or privately defined BGP communities. Next, unless previously configured, a route map must be created (refer to [Route Map on page 23](#)). Use the **match** command to reference the community list in the route map entry:

```
(config-route-map)#match community <name> exact-match
```

<name> Specifies the name of the community list.

exact-match Optional. Specifies that the route map must match the community name exactly. When the **exact-match** keyword is used, the entire community string must be defined as advertised for it to qualify as an exact match. Without this keyword, matches will result if the defined string appears *anywhere* in the community field.



This command does not define a community for routes. It selects routes according to their predefined community or communities. Other BGP neighbors, either internal or external, should have placed the route in a community.



*Use **set** commands to configure any policies (prepend AS_PATH, MULTI_EXIT_DISC, LOCAL_PREF, etc.) to be applied to the community. Refer to [Prepending Private AS Numbers for Load Balancing on page 28](#) for detailed explanations of the **set** command attributes.*

Lastly, the route map should be applied to a BGP neighbor as an outbound policy (refer to [Applying a Route Map Entry to a BGP Neighbor on page 30](#)).

Prepending Private AS Numbers for Load Balancing

A router sends identical routes to all neighbors unless policies are configured to filter and add attributes to the routes. When service provider routers receive multiple identical routes from an organization, it is up to the service provider to select the connection over which inbound traffic is sent to the organization. The customer can attempt to load balance inbound traffic over multiple Internet connections by influencing the service provider routers' selection process. One way to accomplish this is to prepend extra hops in the AS path of certain routes. For example, a router has two connections to the Internet: one to Service Provider A and one to Service Provider B. Inbound traffic always arrives over the connection from Service Provider A. Several fabricated AS hops can be prepended to the routes for half of the private networks sent from the router to Service Provider A. The advertisements containing extra AS hops would make the service provider routers more likely to route traffic destined to these networks through Service Provider B.

To prepend AS hops to a route, a route map is created and the **match** command is used to select the routes to which the router should prepend the AS hops. Generally, routes are selected according to their network address and prefix length. However, routes can be selected according to other attributes as well.



Refer to [Route Map on page 23](#) for information on how to create a route map. Refer to [Filtering Routes According to Network IPv4 Address on page 25](#) for information on the **match** command used to select routes according to network address and prefix length.

Use the **set as-path prepend** command to prepend the hops to the selected routes. The router can be configured to prepend one or more fabricated AS hops to the selected routes.

```
(config-route-map)#set as-path prepend <number>
```

<number> Specifies a number to be prepended to the AS_PATH value as an AS number. Valid range is **1** to **4294967295**.

Alternatively, the router can simply repeat the last AS in a route up to ten times.

```
(config-route-map)#set as-path prepend last-as <number>
```

<number> Specifies the number of times to repeat the last AS. Valid range is **1** to **10**.



Be sure to consult with the service provider before prepending any fabricated AS numbers to a path. It is important to ensure that the fabricated AS path does not conflict with route policies that the service provider router implements. It is also important to discuss with the ISP what BGP attributes they consider when making routing decisions. This information will verify whether AS_PATH is a valid way to influence traffic to your network.

Setting a MED Metric

Another way to influence neighbors to select a certain connection for inbound traffic is to set different metrics on the routes that are sent to separate neighbors. Since BGP prefers routes with a lower metric, the connection to the neighbor that receives the route with the lowest metric is more likely to be selected.



The algorithm BGP uses to select routes relies on many factors, some of which are dependent upon configurations on the remote router. It is impossible to ensure that the route with the lower metric will actually be selected.

This metric is sometimes called the multi-exit discriminator or MED because it is used to differentiate routes sent over various external connections to the same neighboring AS.

When MEDs are used, routes to a specific part of the network are typically classified according to their destination address. This classification is accomplished using one prefix list or several; depending on the network setup and the goal. Refer to [Prefix List on page 21](#) for information on how to create prefix lists. Separate route maps are then configured for each neighbor to which the router connects. Refer to [Route Map on page 23](#) for information on how to create a route map. A prefix list is associated with each route map entry. Again, depending on the network setup and the goal, the same prefix list can be associated with each route map entry or there might be a different prefix list associated with every route map entry.

Within each route map entry, use the **set metric** command to specify a metric value for the routes that have been selected:

```
(config-route-map)#set metric <value>
```

<value> Sets the metric value. Valid range is **0** to **4294967295**.



The route with the lowest MULTI_EXIT_DISC value is preferred in BGP. Refer to [Multi-Exit Discriminators \(MEDs\) on page 19](#) for information on options available in AOS for handling MEDs received.



Refer to [Example 6 on page 46](#) for a detailed BGP configuration example featuring the use of the MED metric to influence which path is selected for inbound traffic to a local network.



*When specifying a MULTI_EXIT_DISC value, the attribute should be applied **outbound** to a BGP neighbor. Refer to [Applying a Route Map Entry to a BGP Neighbor on page 30](#) for additional information.*

Setting Local Preference for Inbound Routes

The local preference attribute can be used to influence the best path used to transmit information from a local network to a private remote network. Adjusting the local preference value on inbound routes from a remote network can affect the local router's decision when transmitting traffic outbound to the remote network.



The local preference attribute can be set for outbound routes, but it is only relevant in iBGP scenarios because the local preference attribute is not retained across different autonomous systems.

Use the **set local-preference** command to change the LOCAL_PREF value for selected inbound routes:

(config-route-map)#**set local-preference** <value>

<value> Sets the local preference value. Valid range is **0** to **4294967295**.



*The default local preference for all BGP routes in AOS is **100**. The default value can be changed using the **bgp default local-preference** <value> command. The valid range for the default value is **0** to **4294967295**.*



The route with the largest local preference value is preferred in BGP.



Refer to [Example 6 on page 46](#) for a detailed BGP configuration example featuring the use of local preference on inbound routes to influence which path is selected for outbound traffic to a private remote network.

Deleting a Community List

Use the **set comm-list delete** command to specify a list of communities to delete from the route:

(config-route-map)#**set comm-list** <name> **delete**

<name> Specifies the name of the community list that contains the list of community strings to delete.



*A community list must be defined using the **ip community-list** command before the **set comm-list delete** command can be used. Refer to [Configuring a Community List on page 13](#) for detailed information on configuring community lists.*

Applying a Route Map Entry to a BGP Neighbor

After a route map entry has been configured, it must be applied to a BGP neighbor. Use the **route-map** command in BGP Neighbor Configuration mode to assign the route map to a specific neighbor:

(config-bgp-neighbor)#**route-map** <name> [**in** | **out**]

<name> Assigns the specified route map to this BGP neighbor.

in Specifies the filtering/modification of all inbound BGP route updates (for filtering external routes and setting inbound policies).

out Specifies the filtering/modification of all outbound BGP route updates (for advertising routes to the external neighbor and setting outbound policies).



*Before a route map can be assigned to a BGP neighbor, it must first be defined using the **route-map** command in the Global Configuration mode (refer to [Route Map on page 23](#)).*



Refer to [Example 6 on page 46](#) for a detailed BGP configuration example where route maps are applied inbound and outbound to different BGP neighbors.

Router ID

The BGP interface identifies itself to neighbors with its router ID. Often this ID is the IPv4 address of the logical interface that connects to each neighbor. However, the ID can also be the address of a loopback interface used as the update source. A loopback interface that is the update source for BGP ensures that a BGP session stays open even if one connection goes down. The following command specifies the router ID:

```
(config-bgp)#bgp router-id <ipv4 address>
```

<ipv4 address> Designates the IPv4 address this router should use as its BGP router ID. IPv4 addresses should be expressed in dotted decimal notation (for example, **10.10.10.1**) or as a prefix length (for example, **/24**).



If no IPv4 address is configured at BGP startup, it uses the highest IPv4 address configured on a loopback interface. If no loopback interfaces are configured, it uses the highest IPv4 address configured on any interface that is active. If the specified router ID is changed, existing sessions with BGP neighbors are reset.

Soft Reconfiguration Inbound

Soft reconfiguration enables the router to store all updates from a neighbor in case the inbound policy is changed. The command is issued in BGP Neighbor Configuration mode and allows a network administrator to reconfigure BGP policies without clearing active BGP sessions. Administrators can then institute new policies at any time without forcing the neighbors to reestablish their connection and possibly disrupt traffic.

BGP updates are stored prior to filtering; thus, allowing the **clear ip bgp soft** command to be used in the absence of route refresh (RFC 2918) capability. The unfiltered table is used when an inbound policy is changed; allowing the router to implement policy changes immediately based on the stored table instead of having to wait on a new table to be built after a hard reset. A soft reset is beneficial over a hard reset because it allows policy updates without disrupting network traffic flow. A hard reset terminates the existing BGP session, effectively removing all routes learned from a neighbor. A new session is then created and all of the routes must be relearned. Because this process takes place with a hard reset, a network outage can potentially occur until the BGP database and route table have been rebuilt.



Refer to [Clear IP BGP on page 74](#) for more information on this command.

Use the **soft-reconfiguration inbound** command to enable the AOS device to store BGP updates for the specified neighbor:

```
(config-bgp-neighbor)#soft-reconfiguration inbound
```

Update Source

Use the **update-source** command to specify which interface's IPv4 address will be used as the source IPv4 address for the BGP TCP connection.

```
(config-bgp-neighbor)#update-source <interface>
```

<interface> Specifies the interface to be used as the source IPv4 address. Specify an interface in the format <interface type [slot/port | slot/port.subinterface id | interface id | interface id.subinterface id | ap / ap/radio | ap/radio.vap]>, for example, for an Ethernet interface, use **eth 0/1**; for a PPP interface, use **ppp 1**; and for an ATM subinterface, use **atm 1.1**. Enter **update-source ?** for a complete list of valid interfaces.



This command is most often configured as a loopback interface that is reachable by the peer router. The peer will specify this address in its neighbor commands for this router.

iBGP

When an AOS device sends a BGP packet, the routing table is first consulted to determine how to reach the intended peer. By default, the IPv4 address assigned to the egress interface of the local router the packet uses to reach the peer is the source address for the BGP packet. This address can be overwritten by using the **update source** command.

Loopback interfaces should be used when there are multiple paths to reach a single iBGP neighbor. Using a loopback address as the update source forces the BGP messages to use only the IP address of the loopback interface instead of the IP addresses associated with the egress interface of any of the multiple links from a single neighbor. The loopback interface is first created and then advertised as the update source toward the intended iBGP neighbor.



Using a loopback interface as the update source for an iBGP neighbor is often done in conjunction with an interior routing protocol, such as OSPF, that can dynamically advertise routes to reach a particular destination when failures and topology changes occur.

eBGP

Loopback interfaces can also be used in eBGP scenarios where multiple links to the same neighbor exist and a multilink protocol is not being used. Similar to the iBGP scenario above, the loopback address would need to be configured and then specified with the **update-source** command on a per-neighbor basis. However, some additional configuration is required for eBGP applications. Unlike iBGP, eBGP assumes that the neighbor is one hop away on a directly connected interface. When loopback addresses are used, an extra hop is created between the neighbors, thus requiring the use of the **ebgp-multihop** command to increase the TTL value for BGP messages. If the value for this command is not changed in accordance to the new number of hops, the TTL for BGP messages intended for eBGP neighbors defaults to 1 and will expire before the packet reaches the destined loopback interface on the peer router. Refer to [eBGP Multihop on page 17](#) for more information on this command.



*When a loopback interface is used as the update source address, one extra hop is created between eBGP neighbors. It is important to account for this extra hop when calculating the TTL value set in the **ebgp-multihop** command.*

The peering router would also use a loopback interface in this scenario. The neighbor address configured on each router would be the IPv4 address of the loopback interface on the peering router. Since the loopback address enables the BGP neighbors to use an IPv4 address that is not reachable by a directly connected interface, separate static routes would need to be specified for each individual link that can reach the neighbor's loopback IPv4 address. This ensures that the loopback address on the BGP peer remains reachable if one of the links goes down.

Establishing Routing Preference

When a route is redistributed into BGP from a static route, another routing protocol, or when a prefix is specifically advertised using a network statement in BGP Configuration mode, BGP must consider it in its route calculations.

Assuming the LOCAL_PREF value of the two routes is equal, the BGP decision logic prefers routes learned by the local router from statically configured routes or routing protocols over routes learned from a BGP peer.



Assuming both routes have an equal LOCAL_PREF value, it doesn't matter if the administrative distance of a locally learned route is configured higher than that of BGP. The local route will still be used instead of the route learned from the BGP peer. This scenario occurs only when the same route is learned locally and by BGP.

After the best valid route has been selected by the BGP decision logic, the administrative distance for BGP is applied to the route. The routing table then chooses the best route to install based on the lowest administrative distance. If the BGP algorithm chooses a locally learned route with a higher administrative distance that is already installed in the routing table, it will not replace that existing route since it points to the same next-hop IPv4 address. As this specific scenario is manifested, it can be observed in the output of the **show ip route** command. The routing source will show the locally learned route from its original source instead of a BGP sourced route (denoted with a B).



Despite administrative distance and assuming equal LOCAL_PREF value, BGP learned routes are always less preferable than routes learned by the local router from statically configured routes or routing protocols.

The best way to make a static route or a prefix learned from another routing protocol more or less preferable than any route learned using BGP is to redistribute the protocol into BGP (using the **redistribute [connected | ospf | rip | static] [metric <value> | route-map <map>]** command) and use a route map to modify the local preference of the intended routes. If this approach is used, an additional prefix list might need to be configured and applied outbound to all neighbors to prevent unwanted redistributed subnets from being advertised.

The default LOCAL_PREF value for BGP routes is **100**. To prefer the redistributed route over a route learned through BGP, the LOCAL_PREF of the redistributed route must be set to a value higher than 100. To prefer a route learned through BGP over a redistributed route, an inbound route map should be applied to the appropriate BGP neighbor to set the LOCAL_PREF of the intended route higher than 100.

The following example configuration demonstrates the basic commands used to prefer a static route or a route learned through another routing protocol (local prefix) over a BGP learned route:

```
!
route-map REDISTRIBUTE permit 10
  match ip address prefix-list <name>
```

```

    set local preference 110
!
router bgp <AS number>
    redistribute <source protocol> route-map REDISTRIBUTE
!

```



The **redistribute** command supports connected, OSPF, RIP, and static routes. These are all considered locally originated routes to BGP. However, locally learned routes are only considered in the BGP path selection process if the locally learned routes are active in the route table. For example, if a duplicate route is learned using OSPF and BGP, and the OSPF route is not active in the route table (which can occur if the OSPF route is learned after the duplicate route has first been learned by BGP), the OSPF route is not selected for the BGP path. This behavior can lead to different routes being in the route table depending on dynamic events such as the timing of routing protocol convergence and the losing and relearning of routes.

The following example configuration demonstrates the basic commands used to prefer the prefix learned through BGP over a static route or a route learned through another routing protocol (local prefix):

```

!
route-map PREFER permit 10
    match ip address prefix-list <name>
    set local-preference 110
!
router bgp <AS number>
    neighbor <ipv4 address>
    route-map PREFER in

```



[Example 8 on page 52](#) further demonstrates the use of the above example configurations to establish preferences for certain routes.

Example Configurations

The example scenarios contained within this section are designed to enhance understanding of BGP configurations on AOS products. The examples describe some of the common real-world applications of BGP. All configurations provided in this section use the command line interface (CLI).

NOTE

The configuration parameters entered in these examples are sample configurations only. These applications should be configured in a manner consistent with the needs of your particular network. CLI prompts have been removed from the configuration examples to provide a method of copying and pasting configurations directly from this configuration guide into the CLI. These configurations should not be copied without first making the necessary adjustments to ensure they will function properly in your network.

Some commands shown in the example configurations in this guide are already enabled as the default setting in the unit. These commands will not appear in the output when the **show running-config** command is issued. Issue the **show run verbose** command to see all commands (including those that do not appear when the **show running-config** command is issued).

Example 1: MPLS Basic Setup for Private Internet Protocol (PIP)

This example illustrates a typical PIP setup where several remote sites are connected by an MPLS provider. The local AOS router is acting as the customer edge (CE) router and will form a neighbor relationship with the provider edge (PE) router to exchange BGP routes over a Point-to-Point Protocol (PPP) connection. The PE router will learn all of the other remote customer subnets (192.168.2.0 /24 and 192.168.3.0 /24) using BGP and advertise them to the local AOS router. The local AOS router will have a static default route to a firewall on the local area network (LAN) for Internet access.

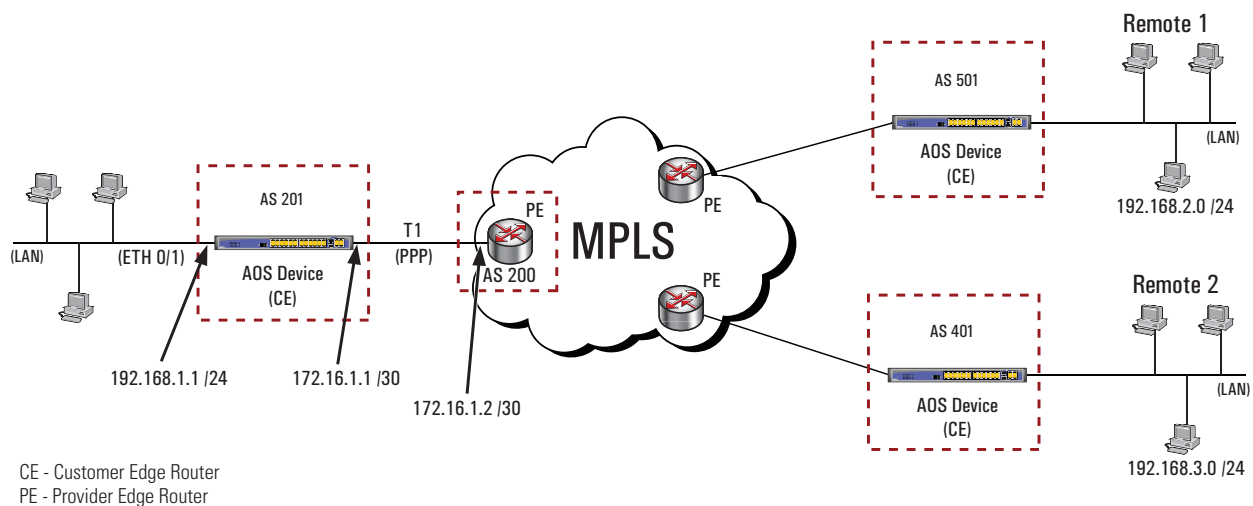


Figure 3. Typical PIP Application

The following configuration applies to Example 1:

```
!  
interface eth 0/1  
    ip address 192.168.1.1 255.255.255.0  
    no shutdown  
!  
interface t1 1/1  
    clock source line  
    tdm-group 1 timeslots 1-24 speed 64  
    no shutdown  
!  
interface ppp 1  
    ip address 172.16.1.1 255.255.255.252  
    no shutdown  
    cross-connect 1 t1 1/1 1 ppp 1  
!  
router bgp 201  
    no auto-summary  
    no synchronization  
    network 192.168.1.0 mask 255.255.255.0  
    neighbor 172.16.1.2  
        no default-originate  
        soft-reconfiguration inbound  
        remote-as 200  
!  
ip route 0.0.0.0 0.0.0.0 192.168.1.254  
!
```



Depending on the provisioning from the MPLS provider, a default route for Internet access can be advertised through BGP.



*When the **eth 0/1** interface is up, there will be a directly connected route for the 192.168.1.0 /24 subnet in the routing table. The route will allow this network to be advertised in BGP using the network command. When the interface is not up, the route will not be advertised in BGP because it is not in the routing table of the router.*



*The command **clock source line** is enabled by default. Therefore, this command will not appear in the output when the **show running-config** command is issued.*

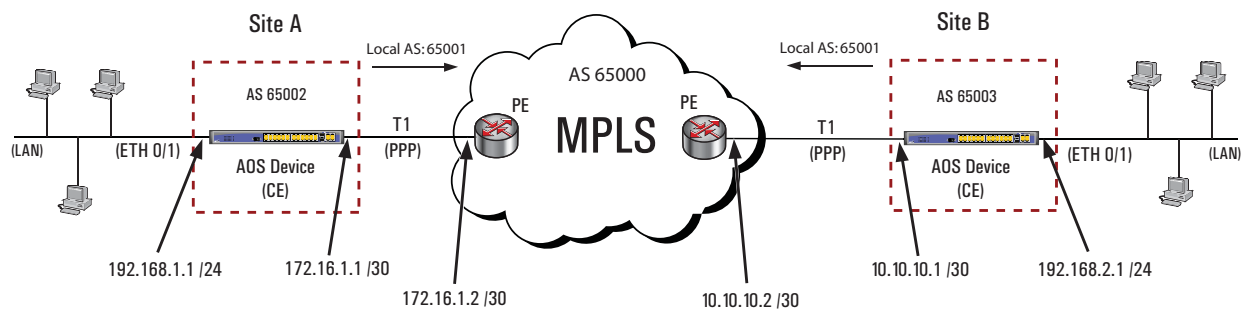
Example 2: Exchanging Routes between Peers with the Same AS Number

Some MPLS providers assign the same AS to every remote location in a customer's network. This type of assignment is problematic because eBGP has a built-in loop avoidance mechanism that prevents the protocol from adding any routes that include its own configured AS from the **router bgp** command in the AS path list. The issue can be avoided by using the **local-as** command from BGP Neighbor Configuration mode in AOS. This command alters the AS path attribute to replace the AS number specified in the **router bgp** command with the AS number specified in the **local-as** command. In this case, the specified AS number is the one the MPLS provider has assigned to multiple remote locations (65001). As a result, each remote site can have a true AS configured globally using the **router bgp** command, but will transmit routing advertisements to the MPLS provider that appear to be sourced from the AS number specified in the **local-as** command. In other words, the **local-as** command replaces the true AS number (the number configured globally with the **router bgp** command), making it appear that the path to the network is through the local AS. Configuring a router in this manner allows routing updates from the far-end customer site to overcome the AS path check because the true AS configured on the router will not match any of the AS numbers listed in the AS path of the routing update.

The following example illustrates the use of the **local-as** command such that routing updates can be exchanged between peers that have been assigned the same AS number by the MPLS provider. The provider has allocated AS 65001 to both customer sites (see [Figure 4 on page 38](#)). Routes from the customer must be sent from AS 65001 to be able to peer with the provider routers. However, the remote locations will not be able to exchange routing information with each other if they are both configured in AS 65001 using the global **router bgp** command. Therefore, instead of using the provider assigned AS of 65001, an arbitrary AS is chosen. The router at Site A is configured with an AS of 65002, and the router at Site B is configured with an AS of 65003 using the **router bgp** command. The AS from which the provider expects to see routing advertisements (65001) is configured into the routers at both Site A and Site B using the **local-as** command in BGP Neighbor Configuration mode. This command alters BGP routing advertisements sent from each router to be sourced from AS 65001, which satisfies the provider's requirements.

In addition to altering the AS path of transmitted BGP routing advertisements to a given neighbor, the messages received from a neighbor are altered as well. The AS configured with the **local-as** command is inserted as the more recent path for prefixes that it learns from that neighbor in addition to routing advertisements to that neighbor. This extra prepended AS on incoming routing advertisements is considered to be iBGP, which prevents the loop avoidance check from failing even though the same AS might be seen twice in a **show ip bgp** command.

For illustrative purposes, it is beneficial to investigate what the AS path will look like in the advertisements for this example. Both sites will essentially see the same AS path on incoming routes: **local-as** command of local router, service provider AS, **local-as** command of remote router. BGP routing advertisements sent to Site A from Site B contain an AS path of 65001, 65000, 65001. This path will not cause an AS path loop because Site A is configured with an AS of 65002, which does not exist in the AS path of the received message. Similarly, advertisements sent to Site B from Site A will contain an AS path of 65001, 65000, 65001. This will not cause an AS path loop as Site B is configured with an AS of 65003. In each case, an extra 65001 is prepended to incoming routes, but is not factored in the loop avoidance check.



CE - Customer Edge Router
PE - Provider Edge Router

Figure 4. Remote Locations with the Same AS Assigned by the MPLS Provider

The following configuration applies to Example 2:

Site A:

```
!
interface eth 0/1
  ip address 192.168.1.1 255.255.255.0
  255.255.255.0
  no shutdown
!
interface t1 1/1
  clock source line
  tdm-group 1 timeslots 1-24 speed 64
  no shutdown
!
interface ppp 1
  ip address 172.16.1.1 255.255.255.252
  255.255.255.252
  no shutdown
  cross-connect 1 t1 1/1 1 ppp 1
!
router bgp 65002
  no auto-summary
  no synchronization
  network 192.168.1.0 mask 255.255.255.0
  255.255.255.0
  neighbor 172.16.1.2
    no default-originate
    local-as 65001
    soft-reconfiguration inbound
    remote-as 65000
!
ip route 0.0.0.0 0.0.0.0 192.168.1.254
!
```

Site B:

```
!
interface eth 0/1
  ip address 192.168.2.1
  no shutdown
!
interface t1 1/1
  clock source line
  tdm-group 1 timeslots 1-24 speed 64
  no shutdown
!
interface ppp 1
  ip address 10.10.10.1
  no shutdown
  cross-connect 1 t1 1/1 1 ppp 1
!
router bgp 65003
  no auto-summary
  no synchronization
  network 192.168.2.0 mask
  neighbor 10.10.10.2
    no default-originate
    local-as 65001
    soft-reconfiguration inbound
    remote-as 65000
!
ip route 0.0.0.0 0.0.0.0 10.10.10.2
!
```



*The command **clock source line** is enabled by default. Therefore, this command will not appear in the output when the **show running-config** command is issued.*

Example 3: Filtering Routes with Prefix Lists

BGP routes advertised and received on an interface can be filtered using prefix lists. The following example illustrates the use of prefix lists to discard incoming routing information and to limit the routes advertised to certain peers. The AOS device in *Figure 5* is expecting a specific route from its eBGP neighbor (208.61.209.253). All other advertised routes from this neighbor are to be discarded. A prefix list (EXPECTED-ROUTE) is used to define the specific subnet (208.61.209.0 /29) the AOS device is expecting from the eBGP neighbor. The implicit *deny* at the end of a prefix list denies all other routes. This prefix list is applied inbound from the eBGP neighbor.

All BGP routes learned from one neighbor are advertised to all other BGP neighbors by default. The customer wants to advertise a specific route from the AOS device to the eBGP neighbor (208.61.209.253) and at the same time prevent the eBGP neighbor from learning about other BGP routes advertised to the AOS device from its iBGP neighbor (65.162.109.202). A prefix list (ADVERTISE) is used to define the specific route (65.162.109.201 /29) that is to be advertised from the AOS device to the eBGP neighbor. The implicit *deny* at the end of the prefix list will prevent any other BGP routes from being advertised on the link. The prefix list is applied outbound toward the eBGP neighbor.

Lastly, it is desired that the AOS device in Figure 5 learn routes from its iBGP neighbor (65.162.109.202), but not advertise any routes to this neighbor. A prefix list (FILTER) is used to create a *deny all* statement. The prefix list is applied outbound toward the iBGP neighbor.

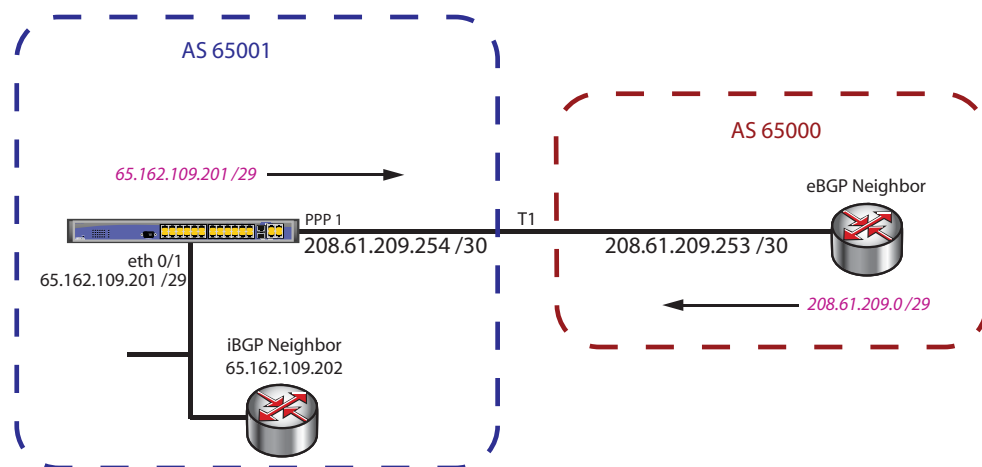


Figure 5. Using Prefix Lists to Filter Routes Sent to and Received from BGP Neighbors

The following configuration applies to Example 3:

```
!
interface eth 0/1
  ip address 65.162.109.201 255.255.255.248
  no shutdown
!
interface t1 1/1
  clock source line
  tdm-group 1 timeslots 1-24 speed 64
  no shutdown
!
```

```
interface ppp 1
  ip address 208.61.209.254 255.255.255.252
  no shutdown
  cross-connect 1 t1 1/1 1 ppp 1
!
ip prefix-list ADVERTISE seq 10 permit 65.162.109.201/29
ip prefix-list EXPECTED-ROUTE seq 10 permit 208.61.209.0/29
ip prefix-list FILTER seq 10 deny 0.0.0.0/0 le 32
!
router bgp 65001
  no auto-summary
  no synchronization
  neighbor 208.61.209.253
    no default-originate
    prefix-list EXPECTED-ROUTE in
    prefix-list ADVERTISE out
    soft-reconfiguration inbound
    remote-as 65000
  neighbor 65.162.109.202
    no default-originate
    prefix-list FILTER out
    soft-reconfiguration inbound
    remote-as 65001
!
```



*The command **clock source line** is enabled by default. Therefore, this command will not appear in the output when the **show running-config** command is issued.*

Example 4: Multihoming and Influencing Traffic over a Preferred Path

Multihoming is when a router has more than one connection to the Internet. The following example illustrates a customer multihoming to two different Internet service providers (ISPs). The customer owns a public block 208.61.209.0 /29 that will be advertised to both ISPs. The preferred path for incoming traffic is the high speed Metro-Ethernet connection to ISP 1. The secondary path is the T1 connection to ISP 2. AS path prepend is used to influence ISP 2 to direct inbound traffic destined to the public block over the Metro-Ethernet connection versus the T1 connection, except when the Metro-Ethernet connection is unavailable. This is accomplished with the route map (NOT-PREFERRED). This route map also automatically filters the routes that are advertised by matching only prefixes defined in the prefix list (PUBLIC-SUBNET) and dropping the rest due to the implicit *discard* all at the end of the route map.

Since the wide area network (WAN) connections are not of equal bandwidth, the customer also prefers to send outbound traffic over the Metro-Ethernet connection. A route map (PREFERRED) is used to create a preference for the default route learned from the neighbor across the Metro-Ethernet connection rather than the neighbor across the T1 connection. The route map is assigned to the Metro-Ethernet eBGP neighbor (65.162.109.202) and matches a prefix list (DEFAULT) specifying the default route. The route map also applies a LOCAL_PREF value of 110 to the specified default route; making it more desirable than the default route learned from the T1 connection to ISP 2, which is assigned a default LOCAL_PREF value of 100.

When multihoming to two different ISPs, it is good practice to advertise only intended networks to prevent becoming a transit AS. The customer's network in Figure 6 could become a transit AS, if ISP 1 sent traffic destined for ISP 2 through the customer's AS (AS 500) or vice versa. This example uses an outbound prefix list (PUBLIC-SUBNET) to advertise only the customer public block to both ISPs. This prefix list will prevent any routes learned by the AOS device using BGP from one ISP from being advertised to the other ISP. The prefix list is applied explicitly to the Metro-Ethernet neighbor with the **prefix-list PUBLIC-SUBNET out** command, and implicitly to the T1 neighbor through the NOT-PREFERRED route map applied outbound. If only default routes are learned from the ISPs, the potential of becoming a transit AS is not an issue. However, it is good practice to use outbound prefixes as a preventative measure for multihoming setups.

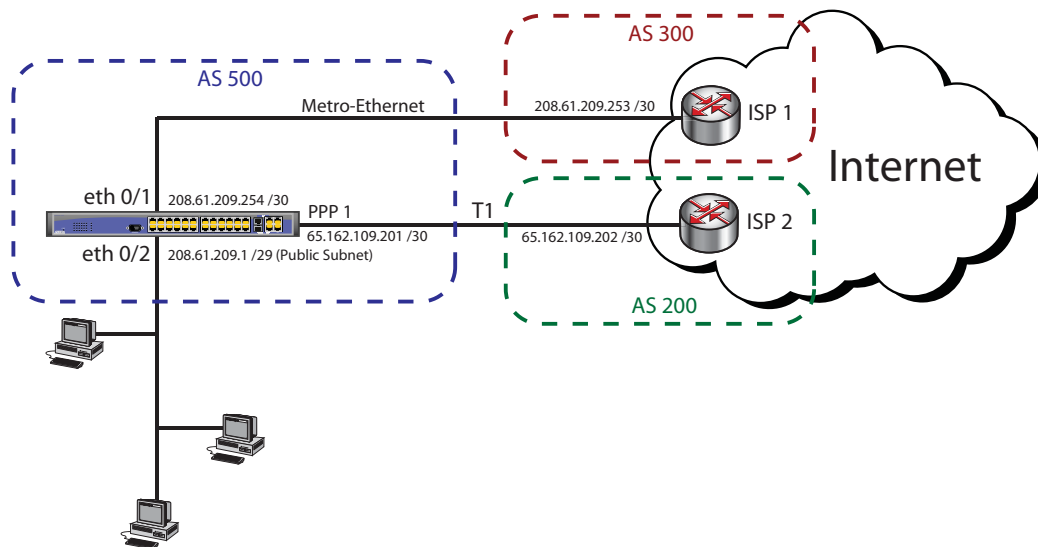


Figure 6. Multihoming to Two Different ISPs

The following configuration applies to Example 4:

```
!  
interface eth 0/1  
  description 10 Mbps Metro-Ethernet connection to ISP 1  
  ip address 208.61.209.254 255.255.255.252  
  traffic-shape rate 10000000  
  no shutdown  
!  
interface eth 0/2  
  description Public Block of IPs being advertised to both ISPs  
  ip address 208.61.209.1 255.255.255.248  
  no shutdown  
!  
interface t1 1/1  
  clock source line  
  tdm-group 1 timeslots 1-24 speed 64  
  no shutdown  
!  
interface ppp 1  
  description T1 connection to ISP 2  
  ip address 65.162.109.201 255.255.255.252  
  no shutdown  
  cross-connect 1 t1 1/1 1 ppp 1  
!  
ip prefix-list PUBLIC-SUBNET seq 10 permit 208.61.209.0/29  
ip prefix-list DEFAULT seq 10 permit 0.0.0.0/0  
!  
route-map NOT-PREFERRED permit 10  
  match ip address prefix-list PUBLIC-SUBNET  
  set as-path prepend 500 500 500 500 500  
route-map PREFERRED permit 10  
  match ip address prefix-list DEFAULT  
  set local-preference 110  
!  
router bgp 500  
  no auto-summary  
  no synchronization  
  network 208.61.209.0 mask 255.255.255.248  
  neighbor 65.162.109.202  
    no default-originate  
    route-map NOT-PREFERRED out  
    soft-reconfiguration inbound  
    remote-as 200  
  neighbor 208.61.209.253  
    no default-originate  
    prefix-list PUBLIC-SUBNET out  
    route-map PREFERRED in  
    soft-reconfiguration inbound  
    remote-as 300  
!
```



Consult with the ISP to determine which BGP attributes they will honor when making a decision on routing traffic back to your advertised AS.



The command **clock source line** is enabled by default. Therefore, this command will not appear in the output when the **show running-config** command is issued.

Example 5: Load Sharing When Multihomed to Multiple ISPs

AOS allows multiple equal cost routes to be used for the purposes of load sharing outbound traffic.



The maximum number of equal cost routes supported in AOS is 6.

The need for load sharing is typically found in BGP applications where an AOS device is multihoming with multiple connections to different ISPs. The BGP protocol does not provide support for load sharing. Therefore, BGP will always export the single best path for a given prefix to the IP route table. However, there are methods that can be implemented that will allow multiple BGP-derived routes to be imported into the IPv4 route table. Aside from the BGP-specific configuration, load sharing must be globally enabled on the AOS device to allow the presence of multiple equal cost routes in the IP route table.



At the global level, load sharing has two different implementation options: per packet and per destination. Refer to [Configuring IP Load Sharing in AOS](https://supportforums.adtran.com) (available online at <https://supportforums.adtran.com>) for more information on these load sharing options.

The following example illustrates load sharing across multiple links where the customer's router is multihomed to two different ISPs. Each ISP is advertising a default route to the AOS device. The default routes contain equal BGP attributes, therefore one route is no more desirable than the other according to the BGP selection process. The objective is to ensure that outbound traffic from the customer's network is load balanced (load shared) between the two Internet connections.

Several configuration steps are needed to allow BGP load sharing to take place. The **ip load-sharing per-destination** command must be enabled in Global Configuration mode. This command allows duplicate routes to exist in the routing table. The command **maximum-paths 2** is issued in BGP Configuration mode to allow up to two equal cost routes from BGP to be exported to the routing table. In order for multiple BGP routes to the same destination to be candidates for load sharing, they must be equal cost and share the same AS number in the AS path attribute. Since each ISP in this example owns their own unique AS number that is added on to the BGP path attribute, an arbitrary AS number must be prepended to each eBGP neighbor's BGP advertisement. This AS number enables the BGP routes to become candidates for load sharing since the routes now appear to originate from the same AS. This is accomplished with the route map (LOAD-SHARE) applied inbound to each eBGP neighbor. It matches the prefix list (DEFAULT) that only allows the default route, which automatically filters any other advertised routes. The route map then prepends the same arbitrary AS path (65009) to the default route prefix learned by each neighbor. The prefix list (PUBLIC-BLOCK) is used to advertise only the customer public block

outbound to both ISPs. This prefix will prevent any routes learned by the AOS device using BGP from one ISP from being advertised to the other ISP. If default routes are only learned from the ISPs, then the potential of becoming a transit AS is not an issue. However, it is good practice to use outbound prefixes as a preventative measure for multihoming setups.

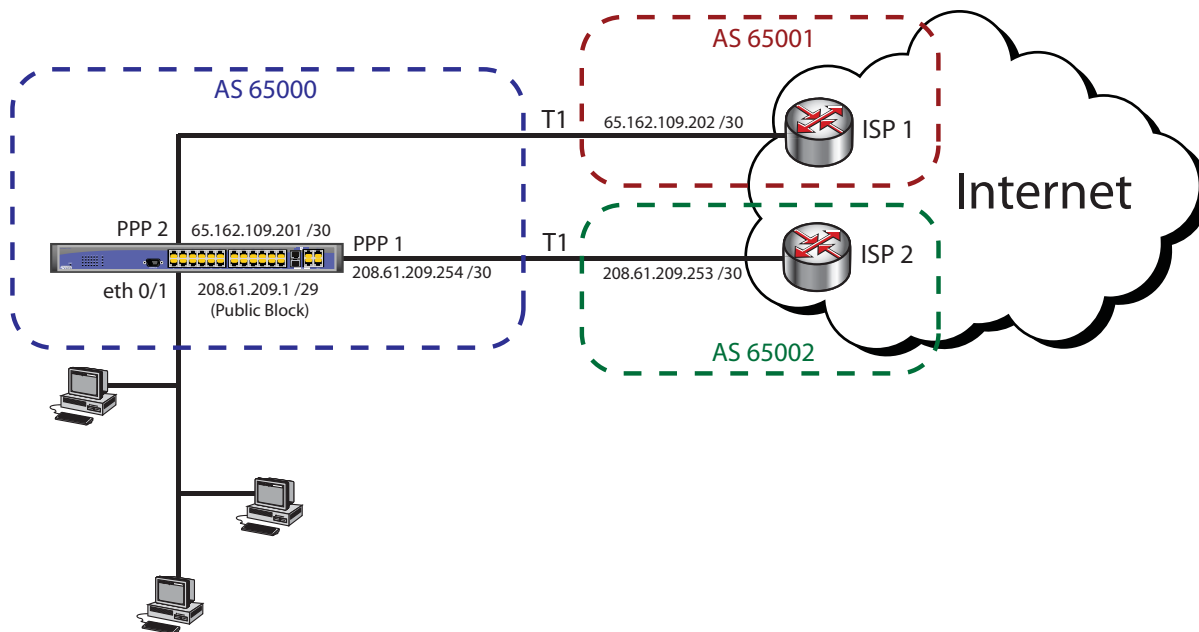


Figure 7. Load Balancing Across Multihomed Connections to Two ISPs

The following configuration applies to Example 5:

```

!
ip load-sharing per-destination
!
interface eth 0/1
 ip address 208.61.209.1 255.255.255.248
 no shutdown
!
interface t1 1/1
 clock source line
 tdm-group 1 timeslots 1-24 speed 64
 no shutdown
!
interface t1 2/1
 clock source line
 tdm-group 1 timeslots 1-24 speed 64
 no shutdown
!
interface ppp 1
 ip address 208.61.209.254 255.255.255.252
 no shutdown
 cross-connect 1 t1 1/1 1 ppp 1
!

```

```

interface ppp 2
  ip address 65.162.109.201 255.255.255.252
  no shutdown
  cross-connect 2 t1 2/1 1 ppp 2
!
ip prefix-list PUBLIC-BLOCK seq 10 permit 208.61.209.0/29
ip prefix-list DEFAULT seq 10 permit 0.0.0.0/0
!
route-map LOAD-SHARE permit 10
  match ip address prefix-list DEFAULT
  set as-path prepend 65009
!
router bgp 65000
  no auto-summary
  no synchronization
  maximum-paths 2
  network 208.61.209.0 mask 255.255.255.248
  neighbor 65.162.109.202
    no default-originate
    prefix-list PUBLIC-BLOCK out
    route-map LOAD-SHARE in
    soft-reconfiguration inbound
    remote-as 65002
  neighbor 208.61.209.253
    no default-originate
    prefix-list PUBLIC-BLOCK out
    route-map LOAD-SHARE in
    soft-reconfiguration inbound
    remote-as 65001
!

```



*The command **clock source line** is enabled by default. Therefore, this command will not appear in the output when the **show running-config** command is issued.*



This configuration uses two separate T1 network interface modules (NIMs), allowing independent T1 clocking. The NetVanta dual T1 NIM (Part Number 1200872L1) does not allow independent clocking and cannot be used in applications where the T1 connections are terminated from two separate service providers.

Example 6: Configuring Local Preference, MED, and Next-Hop-Self on an AOS Router with Both iBGP and eBGP Neighbors

The following example illustrates a scenario where an AOS router has both eBGP and iBGP neighbors. The AOS device in AS 65000 has two eBGP neighbors: Remote Router 1 and Remote Router 2. This means that there are multiple exit points from the local AS (65000). One exit is through the AOS router over the Ethernet WAN interface (eth 0/1) to Remote Router 1. Another exit is through the AOS device over the PPP interface (PPP 1) to Remote Router 2.

The preferred path for traffic originating from the 192.168.1.0 /24 network and destined for the remote private 172.16.5.0 /24 network is the Metro-Ethernet connection. Since the remote network 172.16.5.0 /24 is advertised by both eBGP neighbors (Remote Routers 1 and 2), the local preference attribute is modified to ensure that the Metro-Ethernet connection is selected as the best path to the remote network. This modification is accomplished by creating a route map (SETLOCALPREF) that matches a prefix list (MATCHPREFIX), which specifies the 172.16.5.0 /24 network. Within this route map, the LOCAL_PREF attribute is modified to 110, a higher value than the default LOCAL_PREF value of 100. The route map is applied inbound from the Ethernet WAN eBGP neighbor (192.168.2.2).



The route with the highest local preference value is preferred in BGP.

The Metro-Ethernet connection is also the preferred path for traffic originating from the remote 172.16.5.0 /24 network inbound to the 192.168.1.0 /24 network. The MED is modified to ensure that the Metro-Ethernet connection is selected as the best path inbound to the local network. This modification is accomplished by creating two separate route maps (SETMULTIEXIT1 and SETMULTIEXIT2); each matching a prefix list (NETWORK) that specifies the 192.168.1.0 /24 network. The MULTI_EXIT_DISC value in the route map SETMULTIEXIT1 is set to 100, whereas the MULTI_EXIT_DISC value in the route map SETMULTIEXIT2 is set to 200. SETMULTIEXIT1 is applied outbound to the Ethernet WAN eBGP neighbor (192.168.2.2). SETMULTIEXIT2 is applied outbound to the T1 eBGP neighbor (10.10.10.2).



The route with the lowest MED metric value is preferred in BGP.

The AOS device has two iBGP neighbors: Local Router 1 and Local Router 2. The **next-hop-self** command is used to advertise the IPv4 address of the AOS router (192.168.1.1) to its iBGP neighbors as the path to reach any of the networks advertised by the eBGP peers.



All iBGP routers are fully meshed for both AS 65000 and AS 65001.

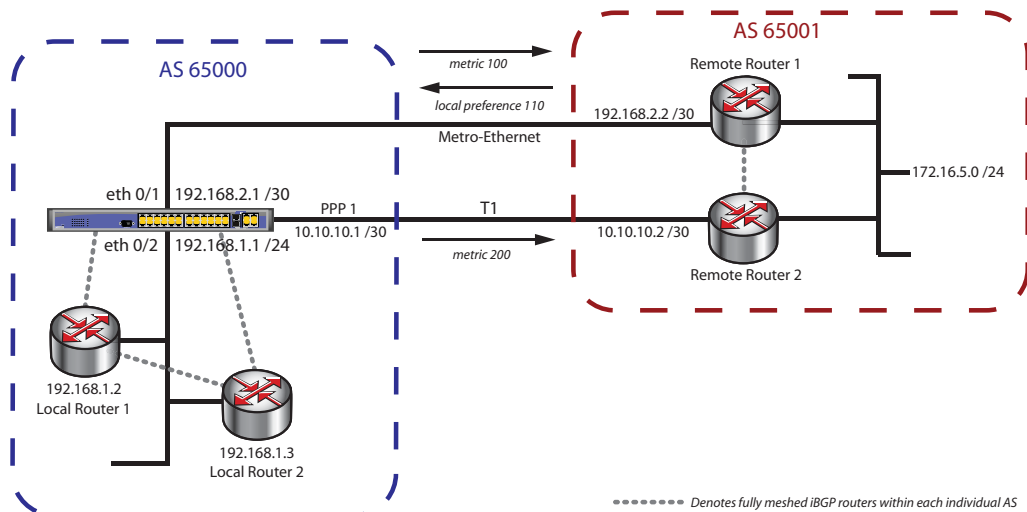


Figure 8. An AOS Device with Both iBGP and eBGP Neighbors

The following configuration applies to Example 6:

```

!
interface eth 0/1
  ip address 192.168.2.1 255.255.255.252
  no shutdown
!
interface eth 0/2
  ip address 192.168.1.1 255.255.255.0
  no shutdown
!
interface t1 1/1
  clock source line
  tdm-group 1 timeslots 1-24 speed 64
  no shutdown
!
interface ppp 1
  ip address 10.10.10.1 255.255.255.252
  no shutdown
  cross-connect 1 t1 1/1 1 ppp 1
!
ip prefix-list MATCHPREFIX seq 10 permit 172.16.5.0/24
ip prefix-list NETWORK seq 10 permit 192.168.1.0/24
!
route-map SETLOCALPREF permit 10
  match ip address prefix-list MATCHPREFIX
  set local-preference 110
route-map SETMULTIEXIT1 permit 10
  match ip address prefix-list NETWORK
  set metric 100

```

```
route-map SETMULTIEXIT2 permit 10
  match ip address prefix-list NETWORK
  set metric 200
!
router bgp 65000
  no auto-summary
  no synchronization
  network 192.168.1.0 mask 255.255.255.0
  neighbor 192.168.2.2
    no default-originate
    route-map SETLOCALPREF in
    route-map SETMULTIEXIT1 out
    soft-reconfiguration inbound
    remote-as 65001
  neighbor 10.10.10.2
    no default-originate
    route-map SETMULTIEXIT2 out
    soft-reconfiguration inbound
    remote-as 65001
  neighbor 192.168.1.2
    no default-originate
    next-hop-self
    soft-reconfiguration inbound
    remote-as 65000
  neighbor 192.168.1.3
    no default-originate
    next-hop-self
    soft-reconfiguration inbound
    remote-as 65000
!
```



*The command **clock source line** is enabled by default. Therefore, this command will not appear in the output when the **show running-config** command is issued.*

Example 7: Using Local Preference to Promote a BGP Route as the Primary Internet Connection over a Backup Static Route

The following example illustrates how to configure an AOS device to prefer a default route learned from an ISP using eBGP as superior to a manually configured static route for the on-site backup Internet connection. In addition, the AOS device will advertise the default route to a local iBGP neighbor.



*AOS only supports the **no default-originate** command, which prevents a unit from sending the default route to a BGP neighbor. However, transmission of default routes to BGP neighbors can be accomplished by issuing the command **network 0.0.0.0 mask 0.0.0.0**.*

An inbound prefix list (DEFAULT) is used in conjunction with a route map to ensure only the default route is learned from the ISP, and an outbound prefix list (FILTER) ensures no routes are advertised to the ISP from the AOS device. The DEFAULT prefix list is also used outbound to only allow the default route to be advertised to the iBGP neighbor.

In this example, a default route is advertised from the ISP to the AOS device using BGP. A static default backup route with an administrative distance of 30 is also specified in the AOS device (using the command **ip route 0.0.0.0 0.0.0.0 192.168.1.254 30**) for the on-site backup Internet connection. In addition, the **network 0.0.0.0 mask 0.0.0.0** command is issued to ensure that the AOS device transmits the default route learned using BGP to its iBGP neighbor. Anytime a route is specified with the above network command or redistributed into BGP, it is subject to the BGP rules for determining the best path. Since the prefix of the static default backup route is specifically advertised by the network statement under BGP, the backup route is injected into the BGP process. This means that the default route advertised from the ISP and the static default backup route will be compared by BGP to determine the best path.

Both the advertised default route and the injected static route will be assigned the default LOCAL_PREF value of 100 in BGP. The learned default route advertised from the ISP is the preferred primary route. However, since both routes have the same value for LOCAL_PREF, the static default backup route will be selected as the best path by the BGP algorithm. The reason for this selection is because the static default backup route is a locally originated route, which is preferable to the BGP selection process than routes learned from a BGP neighbor. The eBGP learned route can be promoted as the preferred route by increasing its LOCAL_PREF value to 110. This is accomplished by applying an inbound route map (DEFAULT-ROUTE-IN) to the eBGP neighbor (ISP).

The route map match criteria specifies a prefix list (DEFAULT) that defines the default route advertised by the eBGP neighbor to the AOS device. When the match is made, the LOCAL_PREF value for that route is set to 110. Thus, the advertised default route is now more desirable with a LOCAL_PREF value of 110 than the static backup route with the default LOCAL_PREF value of 100. As a result, the eBGP learned default route is exported to the route table of the AOS device anytime the route is available. When the eBGP learned default route is not available, the AOS device will send traffic to the internal backup Internet router (192.168.1.254) from the floating static default route.

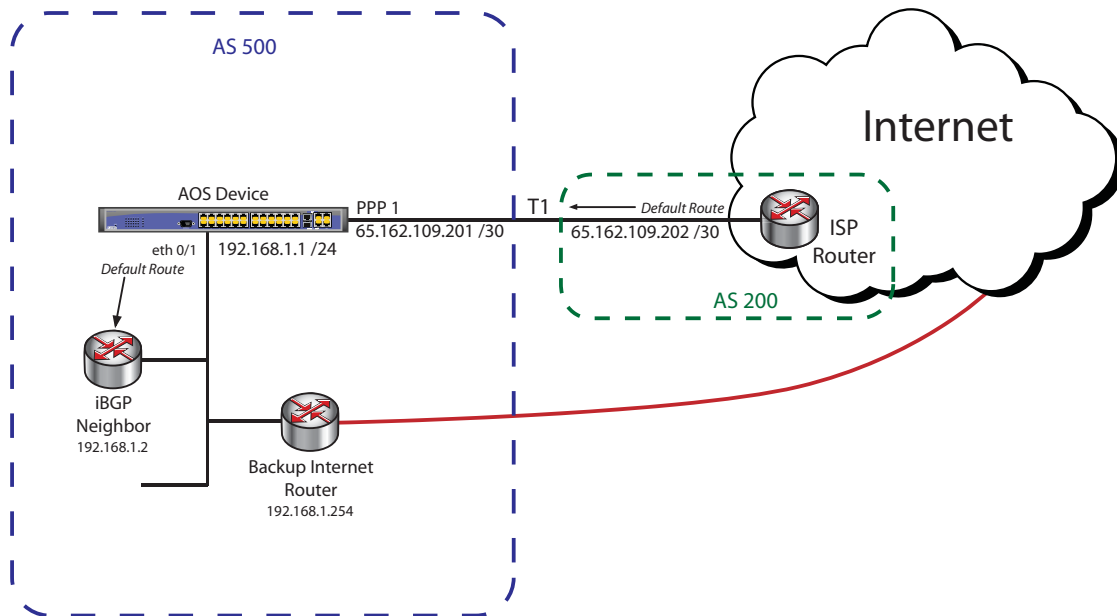


Figure 9. The BGP Advertised Default Route Is the Preferred Primary Internet Connection

The following configuration applies to Example 7:

```

!
interface eth 0/1
  ip address 192.168.1.1 255.255.255.0
  no shutdown
!
interface t1 1/1
  clock source line
  tdm-group 1 timeslots 1-24 speed 64
  no shutdown
!
interface ppp 1
  ip address 65.162.109.201 255.255.255.252
  no shutdown
  cross-connect 1 t1 1/1 1 ppp 1
!
!
ip prefix-list DEFAULT seq 10 permit 0.0.0.0/0
ip prefix-list FILTER seq 10 deny 0.0.0.0/0 le 32
!
route-map DEFAULT-ROUTE-IN permit 10
  match ip address prefix-list DEFAULT
  set local-preference 110
!

```

```
router bgp 500
  no auto-summary
  no synchronization
  network 0.0.0.0 mask 0.0.0.0
  neighbor 65.162.109.202
    no default-originate
    prefix-list FILTER out
    route-map DEFAULT-ROUTE-IN in
    soft-reconfiguration inbound
    remote-as 200
  neighbor 192.168.1.2
    no default-originate
    next-hop-self
    prefix-list DEFAULT out
    soft-reconfiguration inbound
    remote-as 500
!
ip route 0.0.0.0 0.0.0.0 192.168.1.254 30
!
```



The command **clock source line** is enabled by default. Therefore, this command will not appear in the output when the **show running-config** command is issued.



If the AOS router is used to provide Internet access to privately addressed hosts, many to one network address translation (NAT) needs to be configured by running the IPv4 Firewall Wizard. Refer to [Configuring Internet Access \(Many to One NAT\) with the Firewall Wizard in AOS](#) (ADTRAN's Knowledge Base article 2185) for more information.

Example 8: Using BGP Communities in an MPLS Network to Change Local Preference

The most common application for BGP community strings occurs in MPLS networks. Since MPLS providers tend to ignore the AS path attribute and the MED, a community string is often sent to signal the provider that the local preference should be changed. The change in local preference is necessary so that one prefix is seen as less preferable than another identical prefix. Consider the network in [Figure 10 on page 53](#). There are three Internet-provisioned sites on an MPLS network (Primary, Secondary, and Tertiary) that will provide Internet access to remote sites on the network that do not have their own Internet circuits. All PE routers within the MPLS cloud are fully meshed iBGP neighbors. Routing information for the MPLS AS, including local preference information for exiting the AS, is synchronized among each of these PE routers.

The Primary site does not advertise a community string, and its prefixes are adopted with the default LOCAL_PREF value of 100. This local preference setting is the highest advertised in the MPLS network, making the Primary site the preferred connection for Internet traffic for the remote sites. A community string is used to manipulate the local preference in the MPLS cloud for the Secondary site, which will provide backup Internet access for the remote locations, if the Primary site's connection fails. A different community string is used to manipulate the local preference for the Tertiary site to make it the third backup Internet connection. In this example, the MPLS provider has configured the community string 65000:90 for the Secondary site with a LOCAL_PREF value of 90. The community policy is defined using the command **set community 65000:90** under a route map (BGP-OUT) on the AOS device at the Secondary site. Similarly, the MPLS provider has configured the community string 65000:70 for the Tertiary site with a LOCAL_PREF value of 70. The community policy is defined using the command **set community 65000:70** under a route map (BGP-OUT) on the AOS device at the Tertiary site. The local preference attribute is shared and synchronized among all iBGP neighbors in the AS and serves to select the exit point out of the AS when multiple exit points exist for a particular route. In this example, the default route is chosen by local preference and triggers all of the PE routers to send customer Internet traffic to the designated Internet-provisioned customer site. Since the route with the largest local preference is preferred in BGP, the MPLS cloud will prefer the prefix with a LOCAL_PREF value of 100, 90, and finally 70, respectively.



The local Internet connection at the Primary, Secondary, and Tertiary sites must be monitored by some device to determine when a failure occurs. Although this task can be performed by most AOS devices, this topic will not be covered in this configuration guide. For purposes of this example, the external firewall (see [Figure 10 on page 53](#)) will be monitoring the connection and advertising the default route to the MPLS router using RIP.

The final configuration consideration is whether the Secondary and Tertiary sites prefer their own Internet connections to the Primary connection. Some organizations perform monitoring at the Primary site to keep track of their employees browsing habits, or to filter out certain sites from being available for browsing. In either case, the LOCAL_PREF value of the default route (learned from RIP in this example) must be set to either above or below the default LOCAL_PREF value in BGP (100) and redistributed into BGP. These settings allow the failover and failback Internet operations to function correctly. The Secondary site in this example prefers to use its own Internet connection rather than the Primary connection. So, the Secondary site's default prefix is redistributed into BGP and a route map (REDISTRIBUTE) is used to modify the LOCAL_PREF to the value 110 (a value greater than 100). The Tertiary site prefers to use the Primary site for Internet connectivity. Like the Secondary site, the Tertiary site's default prefix is also redistributed into BGP. However, the LOCAL_PREF value is set to 90 (a value less than 100).

NOTE

This example represents a typical MPLS BGP configuration where there is only one PE router that is an eBGP neighbor to the customer's router. Much of the BGP community configuration is done by the MPLS provider, leaving the customer with a much simpler configuration for the desired failover application. A sample configuration that shows what the configuration would look like if the MPLS cloud was a single AOS unit is provided at the end of this example.

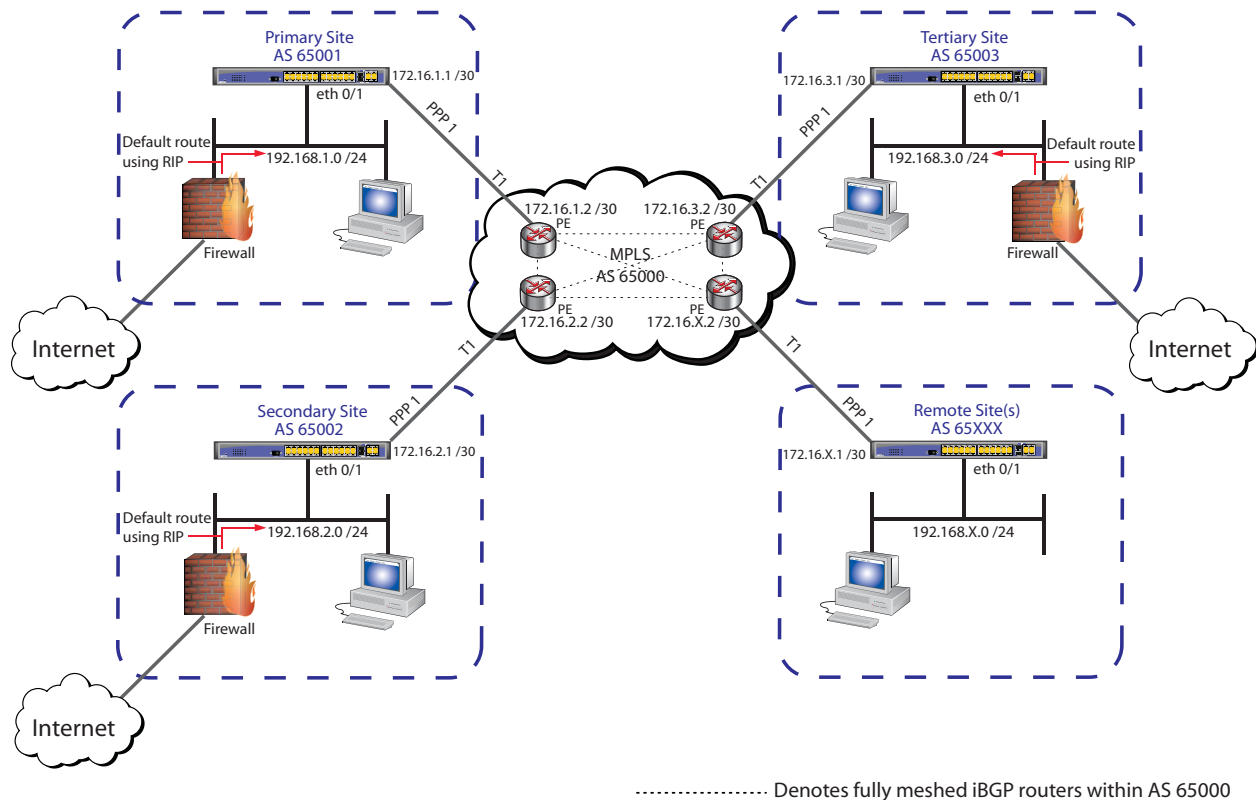


Figure 10. Three Internet-Provisioned Sites on an MPLS Network Provide Internet Access for Remote Sites

The following configuration applies to Example 8:

Primary Site

In this example, the default route advertised with RIP by the external firewall (local prefix) is to be preferred.

```

!
interface eth 0/1
  ip address 192.168.1.1 255.255.255.0
  no shutdown
!
    
```

```
interface t1 1/1
  clock source line
  tdm-group 1 timeslots 1-24 speed 64
  no shutdown
!
interface ppp 1
  ip address 172.16.1.1 255.255.255.252
  no shutdown
  cross-connect 1 t1 1/1 1 ppp 1
!
ip prefix-list DEFAULT seq 10 permit 0.0.0.0/0
!
route-map REDISTRIBUTE permit 10
  match ip address prefix-list DEFAULT
  set local-preference 110
!
router bgp 65001
  redistribute rip route-map REDISTRIBUTE
  network 192.168.1.0 mask 255.255.255.0
  neighbor 172.16.1.2
  remote-as 65000
!
```

Secondary Site

In this example, the default route advertised with RIP by the external firewall (local prefix) is to be preferred.

```
!
interface eth 0/1
  ip address 192.168.2.1 255.255.255.0
  no shutdown
!
interface t1 1/1
  clock source line
  tdm-group 1 timeslots 1-24 speed 64
  no shutdown
!
interface ppp 1
  ip address 172.16.2.1 255.255.255.252
  no shutdown
  cross-connect 1 t1 1/1 1 ppp 1
!
ip prefix-list DEFAULT seq 10 permit 0.0.0.0/0
!
route-map REDISTRIBUTE permit 10
  match ip address prefix-list DEFAULT
  set local preference 110
!
route-map BGP-OUT permit 10
  set community 65000:90
!
```

```

router bgp 65002
  redistribute rip route-map REDISTRIBUTE
  network 192.168.2.0 mask 255.255.255.0
  neighbor 172.16.2.2
    route-map BGP-OUT out
    send-community standard
  remote-as 65000
!
```

Tertiary Site

In this example, the BGP prefix is to be preferred.

```

!
interface eth 0/1
  ip address 192.168.3.1 255.255.255.0
  no shutdown
!
interface t1 1/1
  clock source line
  tdm-group 1 timeslots 1-24 speed 64
  no shutdown
!
interface ppp 1
  ip address 172.16.3.1 255.255.255.252
  no shutdown
  cross-connect 1 t1 1/1 1 ppp 1
!
ip prefix-list DEFAULT seq 10 permit 0.0.0.0/0
!
route-map REDISTRIBUTE permit 10
  match ip address prefix-list DEFAULT
  set local-preference 90
!
route-map BGP-OUT permit 10
  set community 65000:70
!
router bgp 65003
  redistribute rip route-map REDISTRIBUTE
  network 192.168.3.0 mask 255.255.255.0
  neighbor 172.16.3.2
    route-map BGP-OUT out
    send-community standard
  remote-as 65000
!
```

Remote Site(s)

In this example, there is no local Internet connection, so it requires no special configuration. The Internet access for this site would come from the MPLS cloud using the site currently chosen by local preference.

```
!  
interface eth 0/1  
  ip address 192.168.X.1 255.255.255.0  
  no shutdown  
!  
interface t1 1/1  
  clock source line  
  tdm-group 1 timeslots 1-24 speed 64  
  no shutdown  
!  
interface ppp 1  
  ip address 172.16.X.1 255.255.255.252  
  no shutdown  
  cross-connect 1 t1 1/1 1 ppp 1  
!  
router bgp 65XXX  
  network 192.168.X.0 mask 255.255.255.0  
  neighbor 172.16.X.2  
    remote-as 65000  
!
```

MPLS Cloud Sample Configuration



The following configuration represents the BGP community setup and full meshing in the MPLS cloud if it were represented on a single AOS device. The provided configuration is only intended to be an example for administrators whose network is performing the full meshing and BGP definitions that is normally realized by the MPLS network.

```
!  
ip community-list C100  
  permit 65000:100  
!  
ip community-list C90  
  permit 65000:90  
!  
ip community-list C70  
  permit 65000:70  
!  
route-map BGP-IN permit 10  
  match community C100  
  set local-preference 100  
!  
route-map BGP-IN permit 20  
  match community C90  
  set local-preference 90  
!  
route-map BGP-IN permit 30  
  match community C70  
  set local-preference 70  
!  
route-map BGP-IN permit 40
```



```
!  
router bgp 65000  
  neighbor 172.16.1.1  
    route-map BGP-IN in  
    send-community standard  
    remote-as 65001  
  neighbor 172.16.2.1  
    route-map BGP-IN in  
    send-community standard  
    remote-as 65002  
  neighbor 172.16.3.1  
    route-map BGP-IN in  
    send-community standard  
    remote-as 65003  
  neighbor 172.16.X.1  
    route-map BGP-IN in  
    send-community standard  
    remote-as 65XXX
```

Configuration Command Summary



*It is important to note that BGP sessions must be cleared for BGP policy changes, such as alterations to the prefix list filters, to take effect. Use the **clear ip bgp** command to clear BGP neighbors. Typically, soft resets should be used because hard resets can disrupt the network. Refer to [Clear IP BGP on page 74](#) for detailed information on how to properly use this command.*

Table 4. Basic BGP Configuration Steps

	Command	Description
Step 1	(config)# router bgp <AS number>	Enables BGP and specifies the local AS.
Step 2	(config-bgp)# network <ipv4 address> mask <subnet mask>	Specifies the local IPv4 networks that remote sites should be able to access.
Step 3	(config-bgp)# neighbor <ipv4 address>	Configures at least one BGP neighbor.
	(config-bgp-neighbor)# remote-as <value>	Specifies the AS to which this neighbor belongs. Range is 1 to 4294967295 .
Step 4	(config-bgp-neighbor)# do write	Saves the configuration



Refer to [Basic BGP Configuration Using the CLI on page 9](#) for detailed information on the commands used in the table above.

Table 5. Additional BGP Configuration Options

Command	Description
(config-bgp)# distance bgp <external> <internal> <local>	Specifies the administrative distance for BGP routes learned via eBGP (external), iBGP (internal), the network command and redistributed routes (local). By default, external is set to 20 , internal to 200 , and local to 200 .
(config-bgp)# bgp fast-external-failover	Enables the fast-external-failover feature.
(config-bgp)# hold-timer <value>	Specifies a default hold time for all neighbors in this BGP process. This is the time interval (in seconds) within which a keepalive must be received from a peer before it is declared a dead peer. Range is 0 to 65535 seconds.
(config-bgp)# bgp log-neighbor-changes	Turns on the logging of BGP neighbor state changes. State change messages will appear on the screen when connected to the console.
(config-bgp)# maximum-paths <value>	Specifies the maximum number of equal cost parallel routes (shared paths) BGP can export to the route table. Range is 1 to 6 .
(config-bgp)# bgp [always-compare-med compare-med deterministic-med ignore-med]	Instructs AOS on how to handle MEDs for all routes from the same AS.

Table 6. Additional BGP Neighbor Configuration Options

Command	Description
(config-bgp-neighbor)# advertisement-interval <value>	Specifies the minimum interval between sending updates to the neighbor. The default advertisement interval is 30 seconds for external neighbors and 5 seconds for internal neighbors. Range is 0 to 600 seconds.
(config-bgp-neighbor)# description <text>	Identifies the specified interface connected to the BGP neighbor using up to 80 alphanumeric characters.
(config-bgp-neighbor)# distribute-list <ipv4 acl name> [in out]	Assigns an inbound or outbound IPv4 ACL to this BGP neighbor for filtering.
(config-bgp-neighbor)# ebgp-multihop <value>	Configures the maximum hop count of BGP messages to a neighbor. The default TTL for BGP messages is 1 . Range is 1 to 254 hops.
(config-bgp-neighbor)# hold-timer <value>	Specifies a default hold time for this BGP neighbor. This is the time interval (in seconds) within which a keepalive must be received from the peer before it is declared a dead peer. Range is 0 to 65535 seconds.
(config-bgp-neighbor)# local-as <value>	Specifies an AS number that is different from the one specified in the router bgp command. The AS number is used when communicating with this BGP neighbor.
(config-bgp-neighbor)# password <password>	Enables MD5 password authentication to a BGP peer and specifies the password string to be used for authentication.
(config-bgp-neighbor)# next-hop-self	Forces the next-hop attribute to be changed to this unit's IPv4 address for each network it advertises to the iBGP neighbor.
(config-bgp-neighbor)# no default-originate	Prevents the unit from sending the default route to a BGP neighbor. This is a default setting and cannot be modified. A default route can be transmitted to a BGP neighbor by manually entering a default route in BGP Configuration mode with the network 0.0.0.0 mask 0.0.0.0 command.
(config-bgp-neighbor)# prefix-list <name> [in out]	Assigns a prefix list to a BGP neighbor and specifies whether the list will be used to filter inbound or outbound routes.
(config-bgp-neighbor)# route-map <name> [in out]	Applies a previously configured route map entry to a BGP neighbor and specifies whether this route map will filter/modify inbound or outbound BGP route updates.
(config-bgp-neighbor)# soft-reconfiguration inbound	Enables the AOS device to store BGP updates for the specified neighbor.
(config-bgp-neighbor)# update-source <interface>	Specifies which interface's IPv4 address will be used as the source IPv4 address for the BGP TCP connection.



Refer to [Additional BGP Configuration on page 12](#) for more detailed information on the commands referenced in the table above.

Table 7. Filtering Routes and Applying Attributes with Route Maps

	Command	Description
Step 1	(config)# route-map <name> [deny permit] <number>	Creates a route map and enters the Route Map Configuration mode. Range is 1 to 4294967295 .
Step 2 (see note)	Select routes for the BGP interface to advertise outbound or filter inbound using the match commands.	
	(config-route-map)# match ip address prefix-list <name>	Filters according to a prefix list.
	(config-route-map)# match ip address <ipv4 acl name>	Filters according to an IPv4 ACL.
	(config-route-map)# match as-path <name>	Filters according to AS path.
	(config-route-map)# match community <name> [exact-match]	Filters according to a community list.
	(config-route-map)# match metric <value>	Filters according to metric.
Step 3 (Optional)	Apply attributes to the route using the set commands.	
	(config-route-map)# set as-path prepend <number> [last-as] <number>	Prepends AS hops to the selected route(s). The last-as keyword is optional and specifies to repeat the last AS in a route up to 10 times.
	(config-route-map)# set metric <value>	Specifies a MED value. Range is 0 to 4294967295 .
	(config-route-map)# set local-preference <value>	Specifies a local-preference value. Range is 0 to 4294967295 .
Step 4	(config-route-map)# exit	Exits Route Map Configuration mode.
Step 5	(config)# router bgp <AS number>	Enters BGP Configuration mode. Range is 1 to 4294967295 .
Step 6	(config-bgp)# neighbor <ipv4 address>	Enters BGP Neighbor Configuration mode for the neighbor to which the route map is to be applied.
Step 7	(config-bgp-neighbor)# route-map <name> [in out]	Applies the route map to a BGP neighbor and specifies whether it is to be applied to inbound or outbound routes.
Step 8	(config-bgp-neighbor)# do write	Saves the configuration.



Lists referenced in Step 2 (prefix, IPv4 ACL, AS path, and community) first must be created separately before they can be used within the route map for filtering. Refer to [Route Map on page 23](#) for additional information on route maps and related commands used to filter routes and apply attributes.

Table 8. Defining a Community Policy

	Command	Description
Step 1	(config)# ip community-list <name>	Creates a community list for a BGP route map to use. The communities defined in this command should match an existing community string that the AOS device is receiving.
Step 2	(config-comm-list)# [permit deny] [<value> internet local-as no-advertise no-export]	Adds an entry to the community list that either permits or denies BGP routes containing the specified community string in the community attribute.
	(config-comm-list)# exit	Exits the Community List Configuration mode.
Step 3	(config)# route-map <name> [deny permit] <number>	Creates a route map, if one does not already exist, to reference the community list.
Step 4	(config-route-map)# match community <name> [exact-match]	References the <name> of the community list created in Step 1 to add it to the route map entry. The exact-match keyword is optional and specifies that the route map must match all configured community strings in the specified list exactly.
Step 5	Use set commands to configure any policies to be applied to the community.	
	(config-route-map)# set as-path prepend <number> [last-as] <number>	Prepends AS hops to the selected route(s). The last-as keyword is optional and specifies to repeat the last AS in a route up to 10 times.
	(config-route-map)# set metric <value>	Specifies a MULTI_EXIT_DISC value for the selected route(s). Range is 0 to 4294967295 .
	(config-route-map)# set local-preference <value>	Changes the LOCAL_PREF value for selected route(s). Range is 0 to 4294967295 .
Step 6	(config-route-map)# exit	Exits Route Map Configuration mode.
Step 7	(config)# router bgp <AS number>	Enters BGP Configuration mode. Range is 1 to 4294967295 .

Table 8. Defining a Community Policy (Continued)

	Command	Description
Step 8	(config-bgp)# neighbor <ipv4 address>	Enters BGP Neighbor Configuration mode for the neighbor to which the route map is to be applied.
Step 9	(config-bgp-neighbor)# send-community standard	Enables this AOS device to accept a community attribute or adds a community attribute to any advertisement sent by this peer.
Step 10	(config-bgp-neighbor)# route-map <name> in	Applies the route map to the BGP neighbor as an inbound policy.
Step 11	(config-bgp-neighbor)# do write	Saves the configuration.



Refer to *BGP Communities on page 13* for additional configuration options and more detailed information on the BGP community commands listed in *Table on page 62*.

Table 9. Advertising a BGP Community

	Command	Description
Step 1	(config)# ip prefix-list <name> seq <number> [deny permit] <network ipv4 address/length> [ge le] <value>	Creates a prefix list to define the routes that are to be tagged with a community string. Range is 1 to 4294967294 .
Step 2	(config)# route-map <name> [deny permit] <number>	Creates a route map (if one does not already exist) to reference the prefix list. Range is 1 to 4294967294 .
Step 3	(config-route-map)# match ip address prefix-list <name>	References the <name> of the prefix list created in Step 1 to add it to the route map entry.
Step 4	(config-route-map)# set community <value> [add internet local-as no-advertise no-export]	Sets the community attribute to either a privately defined community or one of the well-known community numbers for routes serviced by this route map.
Step 5	(config-route-map)# exit	Exits Route Map Configuration mode.
Step 6	(config)# router bgp <AS number>	Enters BGP Configuration mode. Range is 1 to 4294967295 .
Step 7	(config-bgp)# neighbor <ipv4 address>	Enters BGP Neighbor Configuration mode for the neighbor to which the route map is to be applied.
Step 8	(config-bgp-neighbor)# send-community standard	Enables this AOS device to accept a community attribute or add a community attribute to any advertisement sent by this peer.
Step 9	(config-bgp-neighbor)# route-map <name> out	Applies the route map to the BGP neighbor as an outbound policy.
Step 10	(config-bgp-neighbor)# do write	Saves the configuration.

**NOTE**

Refer to [Prefix List on page 21](#) for more information on creating prefix lists. Refer to [Defining a Community Policy on page 14](#) for more information on the **set community** command.

Table 10. Deleting a Community

	Command	Description
Step 1	(config)# ip prefix-list <name> seq <number> [deny permit] <network ipv4 address/length> [ge le] <value>	Creates a prefix list to define the routes from which a community is to be deleted. Range is 1 to 4294967294 .
Step 2	(config)# ip community-list <name>	Creates a community list for a BGP route map use. The communities defined in this list will be deleted.
Step 3	(config-comm-list)# permit [<value> internet local-as no-advertise no-export]	Adds an entry to the community list that defines the community string in the community attribute that should be deleted.
Step 4	(config-comm-list)# exit	Exits the Community List Configuration mode.
Step 5	(config)# route-map <name> [deny permit] <number>	If one does not already exist, creates a route map. Range is 1 to 4294967295 .
Step 6	(config-route-map)# match ip address prefix-list <name>	Specifies the routes from which a community is to be deleted. Reference the <name> of the prefix list created in Step 1 to add it to the route map entry.
Step 7	(config-route-map)# set comm-list <name> delete	Specifies a list of communities to delete. Reference the <name> of the community list created in Step 2.
Step 8	(config-route-map)# exit	Exits Route Map Configuration mode.
Step 9	(config)# router bgp <AS number>	Enters BGP Configuration mode. Range is 1 to 4294967295 .
Step 10	(config-bgp)# neighbor <ipv4 address>	Enters BGP Neighbor Configuration mode for the neighbor to which the route map is to be applied.
Step 11	(config-bgp-neighbor)# send-community standard	Enables this AOS device to accept a community attribute or add a community attribute to any advertisement sent by this peer.
Step 12	(config-bgp-neighbor)# route-map <name> in	Applies the route map to the BGP neighbor as an inbound policy.
Step 13	(config-bgp-neighbor)# do write	Saves the configuration.



Refer to [Prefix List on page 21](#) for more information on creating prefix lists. Refer to [Deleting Communities from a Route on page 15](#) for more information on the **set comm-list delete** command.

Troubleshooting

After configuring BGP, several different commands can be issued from Enable mode in the CLI to assist in troubleshooting. The following tables contain the **show** and **debug** commands that are implemented specifically for BGP.



Please note that as of AOS firmware release 18.3, the command syntax, organization, and configuration of BGP changed. If you are using firmware 18.3 or later, please refer to the configuration guide *Configuring BGP in AOS* (for AOS data products using firmware 18.3 or later and voice products using firmware R10.1.0 or later). The guide is available online at <https://supportforums.adtran.com>.

Table 11. Viewing BGP Information

Command	Description
#show ip as-path-list <name>	Displays any AS path lists that have been configured in the router, along with any permit and deny clauses in each list. Optionally, enter the <name> of a specific AS path list to display only the list matching the specified AS path list name.
#show ip bgp	Displays the local router ID and AS, plus detailed information about all BGP routes. Information provided for each route includes: origin, destination, next hop, AS path, and whether selected as best.
#show ip bgp <ipv4 address> <subnet mask>	Displays BGP information about a specific IPv4 route, including: advertising router IPv4 address, advertising router ID, and neighbors to which the route is advertised.
#show ip bgp summary	Displays summarized BGP information. This includes the local router ID and AS, the number of paths received and the number of BGP attribute entries. All BGP neighbors are summarized in a table that displays the remote ID, remote AS, and the number of messages sent and received.
#show ip bgp community [<number> internet local-as no-advertise no-export]	Displays routes known by the router that contain the specified <number> or well-known community string: internet , local-as , no-advertise , or no-export in their community attribute.
#show ip bgp community-list <name> [exact]	Displays the BGP routes that are permitted by the specified community list. The optional exact keyword restricts the routes displayed to only those whose community lists exactly match those specified in the named community list. If this parameter is omitted, all routes matching any part of the specified community list will be displayed.
#show ip bgp neighbors <ipv4 address>	Displays information for all BGP neighbors. Information for each neighbor includes: neighbor IPv4 address, neighbor ID, remote AS, settings for BGP intervals, connection status, number of messages, and the local BGP interface IPv4 address. Optionally, limit the information displayed by entering the <ipv4 address> of a specific neighbor.
#show ip bgp neighbors <ipv4 address> advertised-routes	Displays all IPv4 BGP routes being advertised to the specified neighbor.

Table 11. Viewing BGP Information (*Continued*)

Command	Description
#show ip bgp neighbors <ipv4 address> received-routes	Displays all IPv4 routes (accepted and rejected) advertised by the specified neighbor.
#show ip bgp neighbors <ipv4 address> routes	Displays all accepted received IPv4 routes advertised by the specified neighbor. Routes displayed have passed inbound filtering.
#show ip bgp regexp <expression>	Displays a summary of the BGP route table that includes routes whose AS path matches the specified expression.
#show running-config router bgp	Displays only the BGP configuration of the router.



The output of all **show** commands can be limited by appending the following modifiers to the end of the command: | **begin** <text>, | **exclude** <text>, and | **include** <text>. The **include** modifier limits output to lines that contain the specified text, the **exclude** modifier excludes any lines with the specified text, and the **begin** modifier displays the first line of output with the specified text and all lines thereafter.

Table 12. Viewing BGP Messages

Command	Description
#debug ip bgp	Displays general BGP events, such as sent/received message summaries, route processing actions, and results. Keepalive packets are not debugged with this command.
#debug ip bgp events	Displays significant BGP events, such as neighbor state changes.
#debug ip bgp [in out]	Displays the same information as debug ip bgp , but limits messages to the specified direction.
#debug ip bgp keepalives	Displays BGP keepalive packets.
#debug ip bgp updates [quiet]	Displays detailed information on BGP updates for all neighbors. The optional quiet keyword displays a one-line summary with information about BGP neighbor updates.



The **bgp log-neighbor-changes** command can be issued to log state changes of a BGP neighbor. Refer to [Log Neighbor Changes on page 19](#) for more information.

Show Commands

Show IP AS-Path List

Use the **show ip as-path-list** command to display any AS path lists that have been configured in the router, along with any permit and deny clauses in each list.

#show ip as-path-list <name>

<name> Optional. Specifies that the command display only the list matching the specified AS path list name. If not specified, all AS path lists are displayed.

In the following example, all AS path lists defined in the router are displayed:

```
#show ip as-path-list
ip as-path-list AsPathList1:
  permit 100
  permit 200
  permit 300
  deny 6500
ip as-path-list ASPathList2:
  permit 400
  permit 500
```

In the following example, only the AS path list with the name AsPathList2 is displayed:

```
#show ip as-path-list AsPathList2
ip as-path-list AsPathList2:
  permit 400
  permit 500
```

Show IP BGP

Use the **show ip bgp** command to display details about the specified route, including the advertising router IPv4 address, router ID, and the list of neighbors to which this router is being advertised.

#show ip bgp <ipv4 address> <subnet mask>

<ipv4v address> Optional. Specifies a valid IPv4 address. IPv4 addresses should be expressed in dotted decimal notation (for example, **10.10.10.1**).

<subnet mask> Optional. Specifies the subnet mask that corresponds to a range of IPv4 addresses (network) or a specific host. Subnet masks can be expressed in dotted decimal notation (for example, **255.255.255.0**) or as a prefix length (for example, **/24**).

The following example shows detailed output of the **show ip bgp** command:

```
#show ip bgp
BGP local router ID is 172.16.70.1, local AS is 501.
Status codes: * valid, > best, i - internal, o - local
Origin codes: i - IGP, e - EGP, ? - incomplete
Network NextHop Metric LocPrf Path
*> 172.16.55.0/30 10.100.13.151 500 i
*>o172.16.70.0/30 0.0.0.0 i
Total RIB entries = 2
```



The exact prefixes that are being transmitted and received are shown in this output. The *o* in front of the 172.16.70.0/30 route indicates it is a received route. For received routes to be exported to the route table, they must be the best valid route indicated with the **>* for the received route.

Use the **show ip bgp summary** command to display a summary of the BGP route table.

#show ip bgp summary

The following sample output of the **show ip bgp summary** command shows a summarized list of the configured BGP neighbors, as well as their status and statistics.

#show ip bgp summary

BGP router identifier 192.168.3.1, local AS number 304

8 network entries, 5 paths, and 23 BGP path attribute entries

Neighbor	V	AS	MsgRcvd	MsgSent	InQ	OutQ	Up/Down	State/PfxRcd
10.22.131.1	4	302	95	104	0	0	01:30:06	9
10.22.131.9	4	302	97	105	0	0	01:30:07	21
10.22.132.9	4	303	200	179	0	0	02:43:09	21
10.22.134.1	4	304	166	178	0	0	02:43:15	3
10.22.134.10	4	304	174	179	0	0	02:43:24	7
10.22.134.26	4	304	172	174	0	0	02:41:43	10
10.22.134.34	4	304	164	174	0	0	02:41:40	4

Show IP BGP Community

Use the **show ip bgp community** command to display only those routes learned via BGP that match the community numbers specified in the command. If no communities are specified, all BGP routes containing a community attribute are shown.

#show ip bgp community [*<number>* | **internet** | **local-as** | **no-advertise** | **no-export**]

<number> Optional. Displays routes that contain this value in their community attribute. This is a numeric value that can be an integer from **1** to **4294967295** or string in the form aa:nn, where the value of aa is the AS number and the value of nn is the desired local preference to be used in the service provider network. Multiple community-number parameters can be present in the command.

internet Optional. Displays routes that contain this value in their community attribute. This represents the well-known reserved community string INTERNET.

local-as Optional. Displays routes that contain this value in their community attribute. This represents the well-known reserved community string NO_EXPORT_SUBCONFED. Routes containing this attribute should not be advertised to external BGP peers.

no-advertise Optional. Displays routes containing this value in the community attribute. This represents the well-known reserved community string NO_ADVERTISE. Routes containing this attribute should not be advertised to any BGP peer.

no-export Optional. Displays routes containing this value in the community attribute. This represents the well-known reserved community number for NO_EXPORT. Routes containing this attribute should not be advertised to BGP peers outside a confederation boundary.

In the following example, all BGP routes are displayed whose community attributes match those listed in the **show ip bgp community** command.

#show ip bgp community local-as 10:405

BGP local router ID is 10.22.131.241, local AS is 302.

Status codes: * valid, > best, i - internal, o - local

Origin codes: i - IGP, e - EGP, ? - incomplete

Network	Next Hop	Metric	LocPrf	Path
10.22.152.20/30	10.22.131.10	304		302 300 1 3 4 i
10.22.152.24/29	10.22.131.10	304		302 300 1 3 4 5 i
10.22.152.36/30	10.22.131.10	304		302 300 1 3 4 i
10.22.152.52/30	10.22.131.10	304		302 300 1 3 4 i
11.0.0.0/30	10.22.131.10	304		302 300 1 3 4 6 i
12.0.0.0/30	10.22.131.10	304		302 300 1 3 4 6 i
13.0.0.0/30	10.22.131.10	304		302 300 1 3 4 6 i
14.0.0.0/30	10.22.131.10	304		302 300 1 3 4 6 i

Total RIB entries = 8

Information displayed includes: the ID of this router and its AS number; the destination Network address of the route learned; the Next-Hop address to that network; the Metric; the Local Preference (LocPrf) value; and the AS Path to the destination network.

Show IP BGP Community-List

Use the **show ip bgp community-list** command to display BGP routes that are permitted by the specified community list.

#show ip bgp community-list <name> [exact]

<name> Specifies the name of the community list whose routes are to be displayed.

exact Optional. Restricts the routes displayed to only those whose community lists exactly match those specified in the named community list. If this parameter is omitted, all routes matching any part of the specified community list will be displayed.

In the following example, all BGP routes are displayed whose community number match those defined in the community list named CLIST1.

#show ip bgp community-list CLIST1

BGP local router ID is 10.22.131.241, local AS is 302.

Status codes: * valid, > best, i - internal, o - local

Origin codes: i - IGP, e - EGP, ? - incomplete

Network	Next Hop	Metric	LocPrf	Path
10.22.152.20/30	10.22.131.10	304		302 300 1 3 4 i
10.22.152.24/29	10.22.131.10	304		302 300 1 3 4 5 i
10.22.152.36/30	10.22.131.10	304		302 300 1 3 4 i
10.22.152.52/30	10.22.131.10	304		302 300 1 3 4 i
11.0.0.0/30	10.22.131.10	304		302 300 1 3 4 6 i
12.0.0.0/30	10.22.131.10	304		302 300 1 3 4 6 i
13.0.0.0/30	10.22.131.10	304		302 300 1 3 4 6 i
14.0.0.0/30	10.22.131.10	304		302 300 1 3 4 6 i
20.0.0.0/30	10.22.131.10	304		302 300 1 3 4 5 i
21.0.0.0/30	10.22.131.10	304		302 300 1 3 4 5 i

Total RIB entries = 10

Show IP BGP Neighbors

Use the **show ip bgp neighbors** command to display information for the specified BGP neighbor.

#show ip bgp neighbors [*<ipv4 address>*] [**advertised-routes** | **received-routes** | **routes**]

<ipv4 address> Optional. Displays information for the specified neighbor. IPv4 addresses should be expressed in dotted decimal notation (for example, 10.10.10.1). If no IPv4 address is entered, information for all neighbors is displayed.

advertised-routes Optional. Displays all routes being advertised to the specified neighbor. Command output is the same as for **show ip bgp** except filtered to only the BGP routes being advertised to the specified neighbor.

received-routes Optional. Displays all routes (accepted and rejected) advertised by the specified neighbor. Routes can be rejected by inbound filters, such as prefix list filters.

routes Optional. Displays all accepted received routes advertised by the specified neighbor. Routes displayed have passed inbound filtering. This command output is the same as **show ip bgp** except the output is filtered to those learned from the specified neighbor.


NOTE

Entries that are not filtered by a local BGP policy are marked with an asterisk () to show they are valid. Entries that are deemed the best path to an advertised route are marked with a caret (>).*

The following examples show a couple of output variations of the **show ip bgp neighbors** command:

#show ip bgp neighbors

```
BGP neighbor is 10.15.43.17, remote AS 100, external link
Configured hold time is 180, keepalive interval is 60 seconds
Default minimum time between advertisement runs is 30 seconds
Connections established 6; dropped 5
Last reset: Interface went down
  Connection ID: 15
    BGP version 4, remote router ID 8.1.1.1
    BGP state is Established, for 01:55:05
    Negotiated hold time is 180, keepalive interval is 60 seconds
    Message statistics:
      InQ depth is 0, OutQ depth is 0
Local host: 10.15.43.18, Local port: 179
  Sent Rcvd
  Opens:1 1
  Notifications: 0 0
  Updates: 0 8
  Keepalives: 116 116
  Unknown: 0 0
  Total: 117 125
Foreign host: 10.15.43.17, foreign port: 1048
  Flags: passive open
```

#show ip bgp neighbors 10.15.43.34 advertised-routes

BGP local router ID is 10.0.0.1, local AS is 101.

Status codes: * valid, > best, i - internal

Origin codes: i - IGP, e - EGP, ? - incomplete


	Network	NextHop	Metric Path
*>	1.0.0.0/8	10.15.43.17	1 100 i
*>	2.0.0.0/9	10.15.43.17	1 100 i

Show IP BGP Regexp


Use the **show ip bgp regexp** command to display a summary of the BGP route table that includes routes whose AS path matches the specified expression.

#show ip bgp regexp <expression>

<expression> Specifies the regular expression to match when displaying BGP routes. Only those routes whose AS path matches this expression will be displayed in the output.



Regular expressions are strings of characters used in BGP to identify routes by their AS path. Refer to [AS Regular Expressions on page 83](#) for a detailed list of valid AS regular expressions.



Entries that are not filtered by a local BGP policy are marked with an asterisk () to show they are valid. Entries that are deemed the best path to an advertised route are marked with a caret (>).*

The following sample output of the **show ip bgp regexp_303_** command shows all of the entries in the BGP database that contain **303** in the AS path.

#show ip bgp regexp _303_

BGP local router ID is 192.168.3.1, local AS is 304.

Status codes: * valid, > best, i - internal, o - local

Origin codes: i - IGP, e - EGP, ? - incomplete

Network	NextHop	Metric	LocPrf Path
10.22.130.8/29	10.22.132.9		303 304 302 i
* i10.22.130.240/28	10.22.132.1		100 303 300 i
* 10.22.130.240/28	10.22.132.9		303 300 i
10.22.131.0/29	10.22.132.9		303 304 302 i
10.22.131.8/29	10.22.132.9		303 304 302 i
* i10.22.131.16/29	10.22.132.1		0 100 303 i
* 10.22.131.16/29	10.22.132.9		0 303 i
* i10.22.131.240/28	10.22.132.1		100 303 300 i
* 10.22.131.240/28	10.22.132.9		303 300 i
* 10.22.132.0/29	10.22.131.1		0 302 303 i
* 10.22.132.0/29	10.22.131.9		0 302 303 i
* i10.22.132.0/29	10.22.132.1		0 100 303 i
*> 10.22.132.0/29	10.22.132.9		0 303 i
* 10.22.132.8/29	10.22.131.1		0 302 303 i
* 10.22.132.8/29	10.22.131.9		0 302 303 i


```

* 10.22.132.8/29      10.22.132.9      0 303 i
* i10.22.132.240/28 10.22.132.1      0 100 303 i
*> 10.22.132.240/28 10.22.132.9      0 303 i
10.22.134.0/29      10.22.132.9      303 304 i
10.22.134.8/29      10.22.132.9      303 304 i
10.22.134.16/29     10.22.132.9      303 304 i
10.22.134.24/29     10.22.132.9      303 304 i
10.22.134.32/29     10.22.132.9      303 304 i
10.22.134.40/29     10.22.132.9      303 304 i
10.22.134.48/29     10.22.132.9      303 304 i
10.22.134.56/29     10.22.132.9      303 304 i
10.22.134.64/29     10.22.132.9      303 304 i
10.22.134.80/29     10.22.132.9      303 304 i
10.22.135.0/29      10.22.132.9      303 304 305 i
10.22.135.8/29      10.22.132.9      303 304 305 i
Total RIB entries = 30

```

Show Running-Config Router BGP

Use the **show running-config router bgp** command to display only the BGP portion of the running configuration of the router.

#show running-config router bgp

Whereas the **show running-config** command displays all the nondefault parameters contained in the current running configuration file, the **show running-config router bgp** is helpful when a user would like to view only the BGP portion of the running configuration. This command is particularly useful when the running configuration is long and scrolls through many screens.



*Route maps, prefix-lists, and other lists or filters that are being used with BGP are not displayed in the output of the **show running-config router bgp** command.*

The following sample output of the **show running-config router bgp** command shows the BGP portion of a running configuration:

#show run router bgp

```

Building configuration...
!
!
router bgp 6500
no auto-summary
no synchronization
network 192.168.5.0 mask 255.255.255.0
neighbor 192.168.100.1
no default-originate
route-map FILTER out
soft-reconfiguration inbound
remote-as 6510
!
end

```

Debug Commands

Debug IP BGP

Use the **debug ip bgp** command to activate debug messages associated with BGP. Debug messages are displayed in real time. Use the no form of this command to disable the debug messages.



Turning on a large amount of debug information can adversely affect the performance of your unit.

#debug ip bgp [events | in | out | keepalives | updates | updates quiet]

events Optional. Displays significant BGP events, such as a neighbor state change.

in/out Optional. Displays the same information as the **debug ip bgp** command, but limits messages to the specified direction (in or out).

keepalives Optional. Displays BGP keepalive packets.

updates Optional. Displays detailed information on BGP updates for all neighbors.

updates quiet Optional. Displays summary information about BGP neighbor updates. (Note: **updates quiet** displays a one-line summary of the output that is displayed when **debug ip bgp updates** is issued.)



*If no arguments are given, the **debug ip bgp** command displays general BGP events, such as sent/received message summaries, route processing actions, and results. Keepalive packets are not debugged with this command.*

The following example enables debug messages on general outbound BGP messages and events:

#debug ip bgp out

#07:42:39: BGP OUT 10.15.240.1[2]: Transmitting msg, type=UPDATE (2), len=142

Clear Commands

Clear IP BGP

BGP sessions must be cleared for BGP policy changes, such as alterations to prefix list filters or actions taken by a route map, to take effect. Use the **clear ip bgp** command to clear BGP neighbors as specified:

#clear ip bgp [* | <number> | <ipv4 address>] [in | out | soft]

* Clears all BGP neighbors.

<number> Clears all BGP neighbors with the specified AS number. Range is **1** to **4294967295**.

<ipv4 address> Clears the BGP neighbor with the specified IPv4 address. IPv4 addresses should be expressed in dotted decimal notation (for example, **10.10.10.1**).

in Causes a soft reset inbound with a neighbor, reprocessing routes advertised by that neighbor.

out Causes a soft reset outbound with a neighbor, resending advertised routes to that neighbor.

soft Causes a soft reset both inbound and outbound.



Typically, soft resets should be used because hard resets can disrupt the network. A hard reset clears the TCP connection with the specified peers, which results in clearing the table. This method of clearing is disruptive and causes peer routers to record a route flap for each route. Refer to [Soft Reconfiguration Inbound on page 31](#) for additional information.

Strategies for Troubleshooting Specific BGP Problems

An AOS device running BGP might not send or receive the routes that it should for several reasons:

- It cannot communicate with a neighbor
- It is not authorized to transmit, or to accept, the routes in question

View BGP neighbors to make certain the neighbor exists by entering the command **show ip bgp neighbor** at the Enable mode prompt.

If the AOS device is able to communicate with the neighbor, but it is not receiving the routes that it should, then BGP filters need to be examined.

Removing Filters

iBGP allows a user to configure policies to filter routes accepted from and advertised to neighbors. These policies are configured in prefix lists, route maps, and IPv4 ACLs. Since the policies can be quite complicated, they open room for errors.

One of the first steps in troubleshooting BGP is to remove any inbound or outbound filters from the neighbor. If the AOS device begins receiving or advertising the expected routes, then the conclusion is that the filters are causing the problem. The following commands can be issued from BGP Neighbor Configuration mode to remove the appropriate filter:

Remove a prefix list:

```
(config-bgp-neighbor)#no prefix-list <name> [in | out]
```

<name> Specifies the name of the prefix list to be removed.

in Specifies to remove the inbound filter.

out Specifies to remove the outbound filter.

Remove a route map:

```
(config-bgp-neighbor)#no route-map <name> [in | out]
```

<name> Specifies the name of the route map to be removed.

in Specifies to remove the inbound filter.

out Specifies to remove the outbound filter.

Remove an IPv4 ACL:

```
(config-bgp-neighbor)#no distribute-list <ipv4 acl name> [in | out]
```

<ipv4 acl name> Specifies the name of the IPv4 ACL to be removed.

in Specifies to remove the inbound filter.

out Specifies to remove the outbound filter.

Clear the BGP neighbor with a soft reset and observe if the AOS device begins to receive routes. If routes are received, then it is confirmed that the filter is the problem. Reconfigure the prefix list, route map, or IPv4 ACL keeping in mind that the AOS device processes entries in order by sequence number and stops as soon as it finds a match.

A prefix list or route map can also be monitored to see how it is affecting traffic by viewing the list or map and checking the number of packets the router has matched to it. Refer to [Troubleshooting a Prefix List on page 79](#) and [Troubleshooting a Route Map on page 80](#).

If a prefix list is being used with a route map, then it is important to determine whether it is the prefix list or the route map configuration that has the error. An entry in the prefix list that permits all routes can be configured:

```
(config)#ip prefix-list <name> seq <number> permit 0.0.0.0/0 le 32
```

<name> Specifies the name of the prefix list.

<number> Specifies the entry's unique sequence number that determines the processing order. Lower numbered entries are processed first. Range is **1 to 4294967294**.

Clear the BGP neighbor with a soft reset and observe if the AOS device begins to receive routes. If routes are received, then it can be concluded that the prefix list must be reconfigured. Otherwise, the route map is the problem.



When adding an entry to a prefix list to permit all routes, the sequence number specified should be the lower than all other entries contained within the prefix list.

BGP Will Not Accept Routes

If it is suspected that filters are keeping the AOS device from receiving routes, compare the routes that BGP receives from a neighbor to those it actually accepts. Enter:

```
#show ip bgp neighbor <ipv4 address> received-routes
```

```
#show ip bgp neighbor <ipv4 address> routes
```

Note any routes that are displayed when the first command is entered, but not displayed when the second command is entered. These routes are being filtered out. Also, determine if the filter is rejecting a route by locating the asterisk (*) in front of the network address. The absence of an asterisk means that the AOS device considers the displayed route invalid. See [Figure 11 on page 77](#).

```

Router# show ip bgp neighbors 192.168.0.25 routes
BGP local router ID is 192.168.88.1, local AS is 501.
Status codes: * valid, > best, i - internal, o - local
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          NextHop          Metric LocPrf Path
*> 192.168.5.0/24   192.168.0.25           500 i
Total RIB entries = 2

```

The AOS device is accepting one route from the neighbor.

```

Router# show ip bgp neighbors 192.168.0.25 received-routes
BGP local router ID is 192.168.88.1, local AS is 501.
Status codes: * valid, > best, i - internal, o - local
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          NextHop          Metric LocPrf Path
*> 192.168.5.0/24   192.168.0.25           500 i
   192.168.77.0/24  192.168.0.25           500 502 i
Total RIB entries = 2

```

No * indicates that the AOS device considers the route invalid.

The router is filtering out the route to 192.168.77.0 /24.

Figure 11. Comparing Accepted Routes to All Routes Received

The next step in troubleshooting an interface that will not accept routes is to remove filters from the neighbor (refer to [Removing Filters on page 75](#)). If the filter is the problem, then troubleshoot it as described in [Troubleshooting a Prefix List on page 79](#) and [Troubleshooting a Route Map on page 80](#).

BGP Cannot Communicate with a Neighbor

Unlike other routing protocols, BGP does not automatically search for and exchange routes with connected routers. Each BGP neighbor must be manually added and configured on the AOS device.


First, view the BGP neighbor and double-check its IPv4 address:

#show ip bgp neighbors

Ping the neighbor and check connectivity.

If the ping is successful, but the AOS device does not seem to be exchanging BGP messages, the maximum hop count for BGP messages might need to be adjusted using the **ebgp-multihop** command. Typically, external neighbors can be reached from a directly connected interface making them one hop away. If they are not, then the number of hops between the interface and the neighbor must be specified. For example:

#ebgp-multihop 4



NOTE A loopback interface adds a hop to the route. Even if the external neighbor is directly connected, **ebgp-multihop** must be enabled if a loopback interface is used as the source BGP interface. Refer to [eBGP Multihop on page 17](#) for additional information on this command.

Next, record the current settings in the AOS device and verify that they match those that have been agreed upon with the entity that controls the external AS. [Table 13 on page 78](#) displays the key information that should be verified and how to view the settings on the AOS device.

Table 13. Checking BGP Configuration

Key Information	How to View	Record the AOS Device Setting
local AS	show ip bgp [summary]	
local router ID	show ip bgp [summary]	
local router IPv4 address	show ip bgp neighbor	
neighbor router ID	show ip bgp neighbor	
neighbor IPv4 address	show ip bgp neighbor	
remote AS	show ip bgp neighbor	

[Figure 12](#) below shows sample output and where to locate some of the information for [Table 13 on page 78](#).

```

Router#show ip bgp summary
BGP router identifier 192.168.88.1, local AS number 501 ← Local AS
0 network entries, 0 paths, and 1 BGP path attribute entries

Neighbor      V      AS  MsgRcvd  MsgSent  InQ  OutQ  Up/Down    State/PfxRcd
192.168.0.25  4      500     58       57     0    0  00:55:07  State/PfxRcd 2
    ↑
  Remote address

    ↑
  Local Router ID
    
```

Figure 12. Viewing Local ID and Local AS

When the BGP interface cannot reach the configured neighbor, the following debug messages are received on the console:

```

BGP EVT 1.1.1.1[1]: IDLE->CONNECT
BGP EVT 1.1.1.1[1]: CONNET->IDLE
BGP OUT 1.1.1.1[1]: TCP error 0 connecting to peer (events:connect)
    
```

In this example, the interface is attempting to connect to a peer through the peer’s loopback IPv4 address (1.1.1.1), which the router does not consider to be directly connected.

When configuring the BGP neighbor, it is important to always identify it by the IPv4 address for the connecting interface, even if the remote router uses a different router ID. For example, [Figure 13](#) displays information about a local router’s BGP neighbor. The neighbor uses the IPv4 address 192.168.5.1 for its router ID. However, the remote IPv4 address is 192.168.0.25, and this is the IPv4 address that should be entered when configuring the neighbor.

```

Router#show ip bgp neighbors
BGP neighbor is 192.168.0.25, remote AS 500, external link
Configured hold time is 180, keepalive interval is 60 seconds
Default minimum time between advertisement runs is 30 seconds
Connections established 2; dropped 1
Last reset: Peer closed connection
Connection ID: 5
  BGP version 4, remote router ID 192.168.5.1
  BGP state is Established, for 00:58:15
  Negotiated hold time is 180, keepalive interval is 60 seconds
  Message statistics:
    InQ depth is 0, OutQ depth is 0
      Sent      Rcvd
  Opens:           1         1
  Notifications:   0         0
  Updates:         0         2
  Keepalives:     59        58
  Unknown:         0         0
  Total:          60        61
  Local host: 192.168.0.34, Local port: 179
  Foreign host: 192.168.0.25, foreign port: 1042
  Flags: passive open

```

Figure 13. Viewing a BGP Neighbor

BGP Will Not Send Routes to a Neighbor

If a remote host cannot reach the local network, it is possible that BGP might not be sending the remote unit the correct routes. View the routes the AOS device is advertising to the neighbor by entering the following command:

```
#show ip bgp neighbor <ipv4 address> advertised-routes
```

Verify that BGP has been configured to advertise the network by viewing the running configuration. Also, check outbound filters (both prefix lists and route maps) as you would inbound filters.

If the AOS device still cannot send or receive routes, then it is probably having trouble connecting to the neighbor (refer to [BGP Cannot Communicate with a Neighbor on page 77](#)).

Troubleshooting a Prefix List

Use the following Enable mode command to view a prefix list:

```
#show ip prefix-list [detail | summary] <name>
```

The **detail** and **summary** keywords are optional and mutually exclusive. If only the name of the prefix list is entered, then the permit and deny statements are displayed and listed by sequence number. The **summary** keyword produces output that lists the number of statements, their sequence numbers, and the number of these statements that include a range of valid prefixes. The **detail** keyword produces all the information shown by the other two commands, as well as the number of times a BGP prefix has matched each statement.

Check all of the statements for hits. Statements that have been misconfigured often have no hits. If the entire list has no hits, then the list might not have been applied to the BGP neighbor (if the list is applied to a route map, ensure that the map has been applied to the BGP neighbor). Also, verify that the list is correctly applied to either inbound or outbound data.

Figure 14 on page 80 shows sample output of the detailed command.

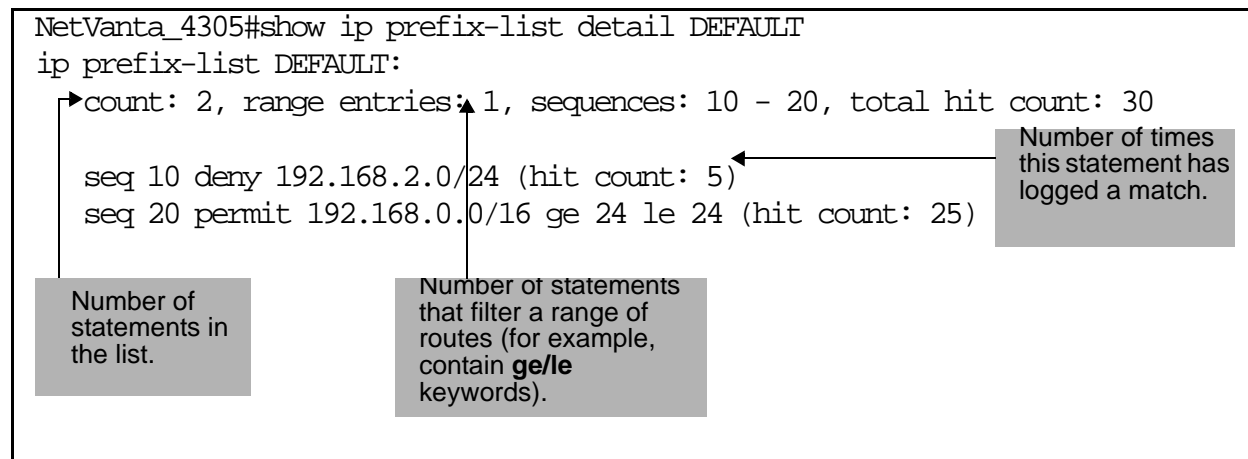


Figure 14. Viewing a Prefix List

The following are useful tips to keep in mind when searching for misconfigurations in a prefix list:

- If a statement does not include a range of prefixes, then a route must match the statement exactly in order to be selected. Make sure that the prefix length is correct.
- Sequence numbers are important. The router stops processing the list after it finds a match. In *Figure 14 on page 80*, the deny statement must have a lower sequence number than the permit statement because the route specified in the deny statement also matches the permit statement.
- The **ge** and **le** keywords match prefixes equal to length specified, as well as those greater or lesser than the specified length. That is, the statement **permit 0.0.0.0 le 17** will allow /17 routes.

Troubleshooting a Route Map

Use the following Enable mode command to view a route map:

#show route-map [*<name>*]

Include the name of a route map to display only the route map matching the specified name. Issuing the command without entering a name will display all route maps configured on the router.

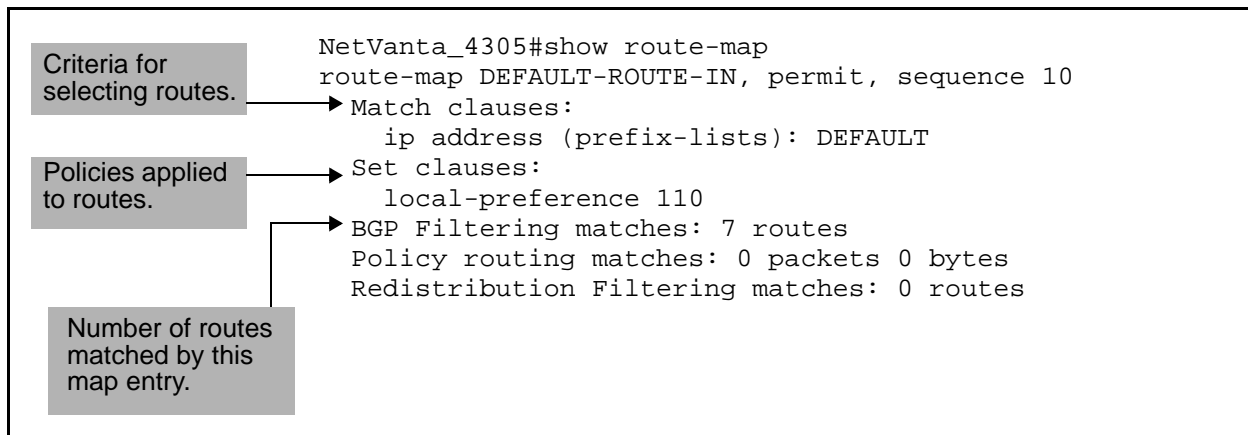


Figure 15. Viewing BGP Policies in a Route Map

The output shows how many routes have been matched to the route map. If the AOS device does not seem to be filtering any routes, verify that the route map has been applied to the correct neighbor and as the correct policy (inbound or outbound).

The following are useful tips to keep in mind when searching for misconfigurations in a route map:

- If attributes are to be applied to routes filtered by an inbound route map, the **set** commands must be entered for the attributes in the same route map entry in which the **match** command is entered to select permitted routes.
- If an entry does not include a match statement, then the policy in that entry will be applied to all routes.
- If an entry is being used to place a route in one or more communities with the **set community** command, then the **send-community standard** command must be issued from BGP Neighbor Configuration mode.

Other Common BGP Problems

After a BGP session has been opened and routes have been exchanged with neighbors, several problems might arise:

- An ISP might refuse to accept the local router's routes
- The local network is flooded with external traffic
- Routers are not defined in the correct communities

An ISP Router Refuses Local Routes.

Verify that the ISP allows the advertisement of private routes.

Network Flooded with External Traffic

One of the most common uses for BGP is BGP multihoming. Multihoming allows connections to two different ISPs. An unintended consequence of multihoming is that the ISPs can advertise routes to each other through the multihomed router. This results in the local router becoming a transit network for external traffic. This should be prevented by utilizing prefix lists that only allow specific subnets to be advertised to each provider. This will keep the ISPs from being able to advertise other routes for the Internet to each other through the local router. (For more information on proper configuration for multihoming applications, refer to [Example 4 on page 41](#)).

Routes in Incorrect Communities

There are several things that could prevent a remote neighbor from applying the correct policies to routes that have been defined as members of particular communities.

- The AOS device has not been enabled to send community attributes to this neighbor. Enter the **send-community standard** command from the BGP Neighbor Configuration mode context.
- The BGP neighbor defines different policies for the community or the BGP neighbor does not accept community attributes in customer routes. Consult with the ISP about what communities it supports.

There might also be problems with the local policy that has been configured for communities on the AOS device. Look at the configured route maps and examine entries that include a match clause for a community list. Then verify that the **set** clauses implement the correct policies for communities in this list.

Use the following Enable mode command to view a community list:

```
#show ip community-list <name>
```

Use the following command to view the routes that match the community list:

```
#show ip bgp community-list <name>
```

Monitor the communities of routes that the router receives by entering this command:

```
#show ip bgp community [<number> | internet | local-as | no-advertise | no-export]
```

The CLI displays all routes in the specified community. Enter the command without keywords (**#show ip bgp community**) for the community to see all routes known by the router that have a community attribute.

Appendix

AS Regular Expressions

Regular expressions, also known as **regexp**, are used in BGP to identify routes that are to be included in outbound routing advertisements or filtered from received inbound routing updates. Regular expressions identify an expected pattern to match against the AS path associated with a BGP route. AS paths identified by the regular expressions are subject to the actions specified in AOS.

Regular expressions are defined in the Global Configuration mode using the command **ip as-path-list**. Refer to *Filtering Routes According to AS Path on page 26* for more information on this command.

There are four types of regular expressions: atom, range, piece, and branch.

Atom

An atom is a single character.

Table 14. Atom

Character	Definition
.	Matches any single character.
^	Matches the beginning of the input string.
\$	Matches the end of the input string.
\	Matches the character.
-	Matches a comma (,), left brace ({), right brace (}), the beginning of the input string, the end of the input string, or a space

Range

A range is a sequence of characters contained within square brackets (for example, [0-9]).

Piece

A piece is an atom followed by one of the following symbols:

Table 15. Piece

Character	Definition
*	Matches 0 or more sequences of the atom.
+	Matches 1 or more sequences of the atom.
?	Matches the atom or the null string.

Branch

A branch is 0 or more integrated pieces.

Examples of Regular Expressions

.*	Matches any character or sequence of characters.
3+	Matches any BGP route entry with at least one occurrence of the number 3 in the AS path.
\b300\b	Matches any BGP route entry containing 300 in the AS path.
\b300\$	Matches any BGP route entry with an AS path ending in 300.
^300\b.*	Matches any BGP route entry with an AS path that begins with 300.
^\$	Matches any BGP route entry with an AS path containing only the local AS.
^300\b.*301	Matches any BGP route entry with an AS path that starts with 300 and ends with 301.
^300\$	Matches any BGP route entry with an AS path that contains only 300.