8 The Border Gateway Protocol (BGP)

The Border Gateway Protocol (BGP) is an exterior routing protocol that guarantees the loop-free exchange of routing information between autonomous systems. There are three versions of the BGP protocol—versions 2, 3, and 4. The Alcatel implementation supports BGP version 4 as defined in RFC 1771. It is configured through command statements in the **gated.conf** file. Description of BGP command statements begins on page 8-16.

The Alcatel implementation of BGP is designed for enterprise WAN networks, specifically for satellite or branch offices within the same organization. Up to 50,000 route table entries and 250,000 next hop routes can be supported by BGP (as long as 64MB of DRAM is present on the MPM or MPX).

An OmniSwitch or Omni Switch/Router using BGP could be placed at the edge of an enterprise network to handle downstream Internet traffic. However, a switch using BGP should not be placed on the public Internet to handle upstream traffic. The BGP implementation in an Omni Switch/Router can handle up to 9 peers, but ideally should be configured with 2 peers. An example of such a configuration would be two (2) paths to the Internet.

BGP is intended for use in networks with multiple autonomous systems. It is not intended to be used as a LAN routing protocol, such as RIP or OSPF. In addition, when BGP is used as an internal routing protocol, it is best used in autonomous systems with multiple exit points as it includes features that help switch/routers decide among multiple exit paths.

BGP is a link-state protocol, like OSPF, but is has less overhead than OSPF. It does not require periodic refresh of its entire routing table, but messages are sent between BGP peers to ensure a connection is active. A BGP speaker must retain the current routing table of its peers during the life of a connection.

BGP uses TCP as its transport protocol, eliminating the need for it to implement mechanisms for updating fragmentation, retransmission, acknowledgment, and sequencing information. BGP uses TCP port 179 to establish its connections.

Several User Interface (UI) commands have been implemented to view BGP configuration parameters and BGP paths. Description of these UI commands can be found in page 8-10.

BGP Speakers

BGP supports two types of neighbors: internal and external. Internal sessions are run between BGP speakers in the same autonomous system (AS). External sessions are run between BGP speakers in different autonomous systems. Internal neighbors may be located anywhere within the same autonomous system while external neighbors are adjacent to each other and share a subnet. Internal neighbors usually share a subnet. However, it is possible to configure a group of internal neighbors that do not share a subnet but still reside in the same autonomous system; these groups are referred to as *internal routing groups*.

BGP speakers can be organized into groups that share similar parameters, such as metrics, timers, and route preferences. It is also possible to configure individual speakers with unique parameters. Group and individual speaker configuration is handled through the BGP group and peer statements, which are described in page 8-19 and page 8-25, respectively.

Routing Decisions

Several metrics are used to make BGP routing decisions. These metrics include the main GateD preference assigned to a route in an import/export statement, the BGP local preference, and the Multi-Exit Discriminator (MED). These metrics are organized into a hierarchy such that if a tie results, the next important criteria is used to break the tie until a decision is made for the route path. BGP routing decisions are discussed in page 8-3.

BGP Features

Alcatel's BGP supports the following features:

- Route Reflection. Assigns route exchange responsibility to one BGP speaker to allow for a partially meshed autonomous system while maintaining loop-free routing information exchange. Described further in page 8-7.
- Route dampening. Decreases the affect of route flapping. Described further in page 8-6.
- Communities. Creates groups of autonomous systems sharing similar attributes. Described further in page 8-6.

Classless Interdomain Routing (CIDR)

BGP supports Classless Interdomain Routing (CIDR), which reduces the size of routing table information by aggregating routes into *supernets*. CIDR eliminates the concept of class among IP addresses and uses the advertising of IP prefixes to identify IP networks. The use of IP prefixes significantly cuts down the number of bits required to store an address. However, IP masks are not checked when CIDR is enabled.

CIDR can be used with the BGP, OSPF and RIP protocols. It is mainly intended for use with BGP in Internet applications.

By default CIDR is disabled in the switch. Use the **ipclass** command to enable CIDR. This command enables and disables IP class address checking. The syntax for the **ipclass** command is as follows:

```
ipclass on | off
```

To enable CIDR, you need to turn *off* IP class checking by issuing the following command:

```
ipclass off
```

To disable CIDR (the default) you need to turn on IP class checking as follows:

```
ipclass on
```

BGP Path Selection

BGP selects the best path to an autonomous system from all known paths and propagates the selected path to its neighbors. GateD uses the following criteria, *in order*, to select the best path. If routes are equal at a given point in the selection process, then the next criterion is applied to break the tie.

1. The route with smallest preference, as determined by the routing policy defined in gated.conf file through an import or export statement.
2. The route with the highest BGP local preference as indicated by the local_pref variable. Local preference is discussed in page 8-3.
3. The route with the fewest autonomous systems listed in its AS Path.
4. The AS path origin. A route with an AS path origin of IGP (internal to the AS) is preferred. Next in preference is a route with AS path origin of EGP (external to the AS). Least preferred is an AS path that is incomplete. In summary, the path origin preference is as follows: IGP < EGP < Incomplete.
5. The route with the best Multi-Exit Discriminator (MED). MEDs are only compared between routes that are received within the same AS. This test is only applied if the local AS has two or more connections, or exits, to a neighbor AS.
6. The route with a closer next hop (with respect to the internal routing distance).
7. The source of the route. A strictly interior route is preferred, next in preference is a strictly exterior route, and third in preference is an exterior route learned from an interior session. In summary, the route source preference is as follows: IGP < EBGP < IBGP.
8. Lowest Router ID. The route whose next hop IP address is numerically lowest.

♦ Note ♦

If an interface goes down, policy information will be lost and GateD will have to be restarted to recover the information.

Local Preference

Local preference, as represented by the local_pref variable, is a key determinant of BGP routing policy. The local_pref variable is used to select between BGP routes that have the same GateD preference (as specified in an import or export statement). BGP routes with a larger local_pref are preferred.

Internal BGP routes require a local preference value. A BGP speaker uses local preference to inform other speakers in the same autonomous system of its degree of preference for an advertised route.

The local_pref variable is derived from the setpref variable, which is specified in a BGP group statement and used with internal BGP groups. If the setpref variable is not specified, then BGP uses a local_pref value of 100.

♦ Note ♦

You can view the local_pref value for a route by entering the **bgp path** UI command.

How local_pref is Derived

The local_pref variable is never set directly; it is derived from the GateD preference (from import/export statements) and setpref metrics. The setpref variable allows GateD to compute the local_pref to reflect GateD's own internal preference for a route; it is used only with internal BGP group types. The local_pref value, as exported to BGP peers, is calculated as:

local_pref = 254 – (global protocol preference for this route) + setpref

The global protocol preference for a route is the preference defined in an import or export statement. GateD will only send local_pref values between 0 and 254. A value greater than 254 will be reset to 254.

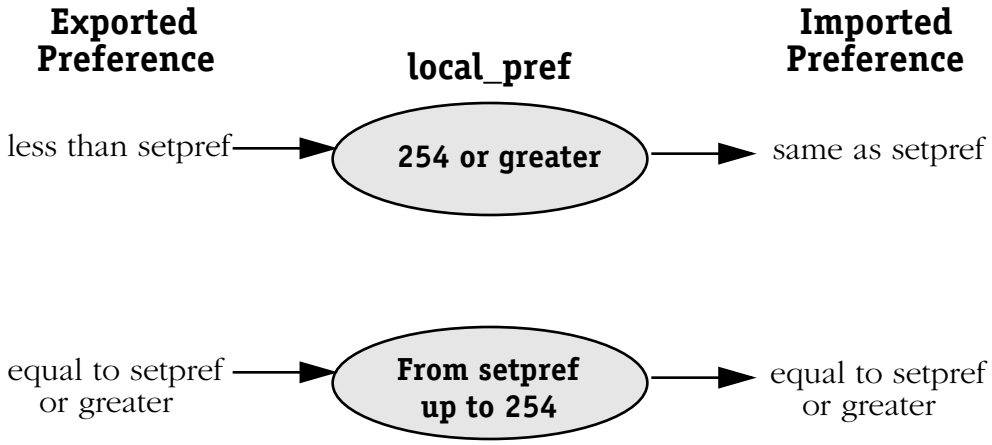
♦ Note ♦

Non-GateD internal BGP groups may send local_pref values that are greater than 254. When operating a mixed network of this type, it is recommended that all routers restrict themselves to sending local_pref values in the range of setpref to 254.

As an example, if the global preference for a route is 170 and the setpref metric is set to 100. then the local_pref value would be 184 (254–170+100=184).

How Route Preferences are Translated by local_pref

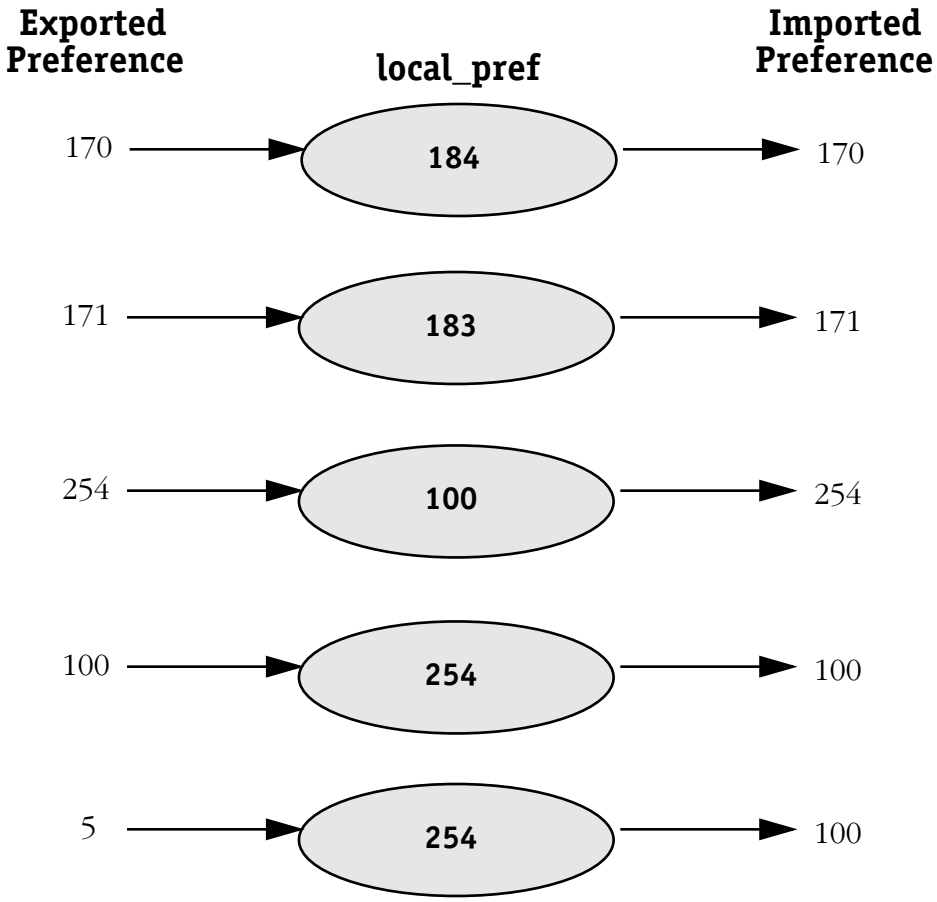
The translation of GateD's internal preference to and from local_pref is done as follows. The "exported preference" is the GateD preference (as specified in an export statement) of the exported route. The "imported preference" is the GateD preference (as specified in an import statement) assigned to the imported route.



How local_pref Affects Route Preference

Any GateD preference of less than setpref is exported with a preference *equal* to setpref. Any preference equal to or greater than setpref will be exported with the *same preference it had originally.*

The examples below show how a GateD preference will change depending on its value and the value for local_pref. All of these examples assume that GateD is sending routes to an internal BGP group using a setpref value of 100, and the routes are subsequently received by another router in the same BGP group also using a setpref value of 100.



Route Preference before and After local_pref

All routers in the same network that are running GateD and participating in IBGP should use setpref uniformly. That is, if one router has setpref set, all should set it, and all should use the same value for that metric. The value for this metric should be selected to be consistent with the import policy in use in the network. For example, if an import policy sets GateD preferences ranging from 170 to 200, a setpref metric of 170 would make sense. It is advisable to set the metric high enough to avoid conflicts between BGP routes and IGP or static routes.

Multi-Exit Discriminator (MED)

The Multi Exit Discriminator, or MED, is used by border routers (i.e., BGP speakers with links to neighboring autonomous systems) to help choose between multiple entry and exit points for an autonomous system. It is only relevant when an AS has more than one connection to a neighboring AS. If all other factors are equal, the path with the lower MED value takes preference over other paths to the neighbor AS.

The MED attribute (which was referred to as INTER-AS_METRIC in earlier versions of BGP) is a four-octet unsigned integer. If received on external links, the MED may be propagated over internal links to other BGP speakers in the same AS. However, the MED is never propagated to speakers in a neighboring AS.

MED is enabled through the main BGP statement and the value for this variable is set using the **metricout** statement. The metricout statement is configured in an export statement or in a BGP group or peer statement. See page 8-16 for more information on MED statements.

Communities

Distribution of routing information in BGP is typically based on IP address and AS number. To simplify the control of routing information, autonomous systems can be grouped into *communities* and routing decisions can be applied based on these communities. In this sense a group of ASs, when grouped into a community, can appear to be a single AS to BGP.

Communities are configured through the aspath-opt statement, which can be specified in a BGP group statement or in an import or export statement. Community statements are described in page 8-31.

Route Dampening

Route dampening minimizes the affect of flapping routes in a BGP network. Route flapping occurs when route information is updated erratically, such as when a route is announced and withdrawn at a rapid rate. Route flapping can cause problems in networks connected to the Internet, where route flapping will involve the propagation of many routes. Route dampening suppresses flapping routes and designates them as unreachable until they flap at a lower rate.

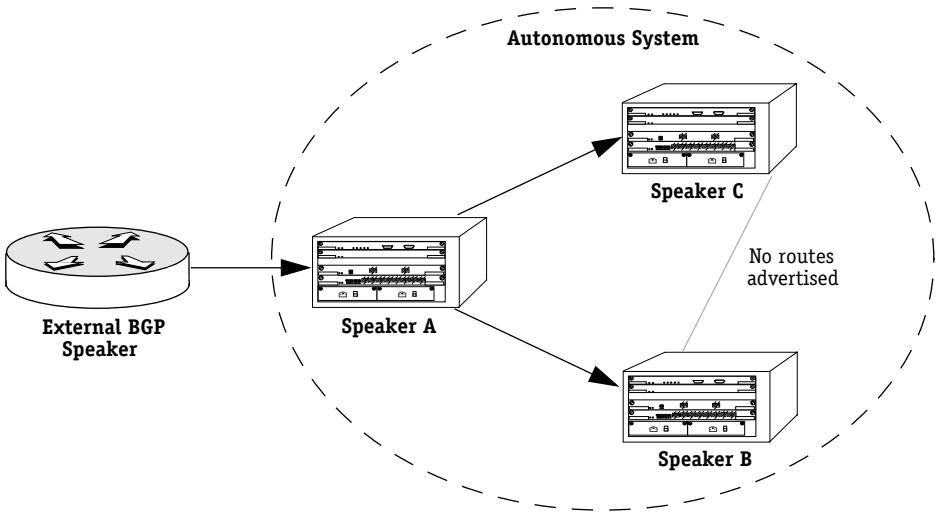
You can configure route dampening to adapt to the frequency and duration of a particular route that is flapping. The more a route flaps during a period of time, the longer it will be suppressed.

The statements for controlling BGP route flapping are not contained within the main BGP statement. Instead they are specified outside the BGP statement in a way similar to interface and kernel statements. See page 8-30 for route flapping configuration statements.

Route Reflection

BGP requires that all speakers in an autonomous system be fully meshed (i.e., each speaker must have a peer connection to every other speaker in the AS) so that external routing information can be distributed to all BGP speakers in an AS. However, fully meshed configurations are difficult to scale in large networks. For this reason, BGP supports *route reflection*, a configuration in which one or more speakers—route reflectors—handle intra-AS communication among all BGP speakers.

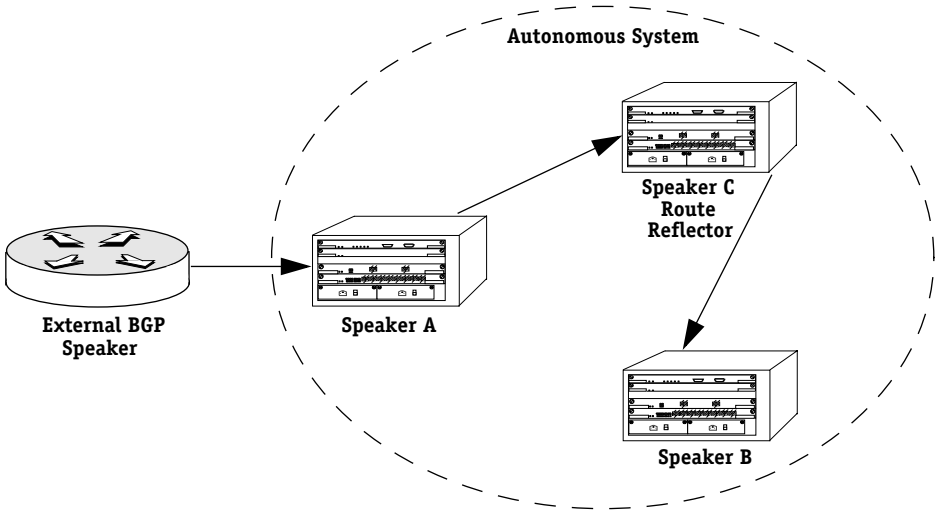
In a fully meshed BGP configuration, a BGP speaker that receives an external route must re-advertise the route to all internal peers. In the illustration below, BGP speaker A receives a route from an external BGP speaker and advertises it to both Speakers B and C in its autonomous system. Speakers B and C do not re-advertise the route to each other so as to prevent a routing information loop.



Fully Meshed BGP Speakers

In the above example, Speakers B and C do not re-advertise the external route they each received from Speaker A. However, this fundamental routing rule is relaxed for BGP speakers that are route reflectors.

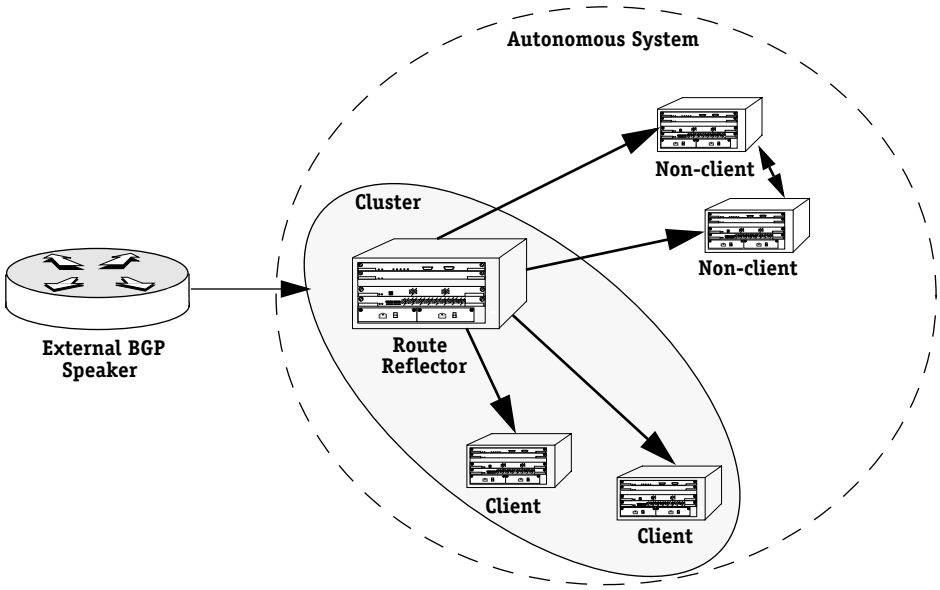
This same configuration using a route reflector would not require that all BGP speakers be fully meshed. One of the speakers is configured to be a route reflector for the group. In this case, the route reflector is Speaker C. When the route reflector (Speaker C) receives route information from Speaker A it advertises the information to Speaker B. This set up eliminates the peer connection between Speakers A and B.



Route Reflector Configuration

The internal peers of a route reflector are divided into two groups: client peers and non-client peers. The route reflector sits between these two groups and reflects routes between them. The route reflector, its *clients*, and *non-clients* are all in the same autonomous system.

The route reflector and its clients form a *cluster*. The client peers do not need to be fully meshed (and therefore take full advantage of route reflection), but the non-client peers must be fully meshed. The following illustration shows a route reflector, its clients within a cluster, and its non-client speakers outside the cluster.



Route Reflector, Clients and Non-Clients

Note that the non-client BGP speakers are fully meshed with each other and that the client speakers in the cluster do not communicate with the non-client speakers.

When a route reflector receives a route it selects the best path based on its policy decision criteria. The internal peers to which the route reflector advertises depends on the source of the route. The table below shows the rules the reflector follows when advertising path information:

Route Received From...	Route Advertised To..
External BGP Speaker →	All Clients and Non-Clients
Non-Client Peer →	All Clients
Client Peer →	All Clients and Non-Clients

Configuring Route Reflection

To configure a BGP speaker to be a route reflector you simply include the **reflector-client** statement in the group statement. With this statement, the route reflector will follow the standard rules for client route advertisement (i.e., routes from a client are sent to all clients and non-clients, except the source client).

It is possible to configure the route reflector to advertise only to non-clients when a route is received from a client. In such a configuration all clients in a cluster must be fully meshed. To configure a route reflector to advertise client routes only to non-clients use the **no-client-reflect** statement rather than the **reflector-client** statement.

All routes received from a non-client internal peer will be sent to all route reflection clients regardless of whether **reflector-client** or **no-client-reflect** is used to configure the route reflector.

No special configuration is required on BGP speakers that are not route reflectors (i.e., clients and non-clients). From a client's perspective, a route reflector is simply a normal internal BGP peer.

Redundant Route Reflectors

A single BGP speaker will usually act as the reflector for a cluster of clients. In such a case, the cluster is identified by the router-id of the reflector. It is possible to add redundancy to a cluster by configuring more than one route reflector, eliminating the single point of failure. Redundant route reflectors must be identified by a 4-byte cluster id, which is specified in the BGP statement. All route reflectors in the same cluster must be fully meshed and should have the exact same client and non-client peers.

♦ Note ♦

Using many redundant reflectors is not recommended as it places demands on the memory required to store routes for all redundant reflectors' peers.

BGP UI Commands

The switch's User Interface (UI) contains several commands for viewing BGP configuration parameters and path values. These UI commands operate slightly different than other commands in the UI in that you specify **bgp** before each command name. When you enter **bgp** alone you receive a display of commands as follows:

```
parameters for bgp root command are:
    stat                        BGP status
    peers                       List of BGP peers
    peer <Peer ID>              Detailed stats on a single peer
    paths                       BGP path list
    path <Path IP Address>      Detailed info on a BGP path
```

Each of these command options is described in the following sections. BGP is configured like other GateD routing protocols—through the gated.conf file command statements. These command statements are described in the section, page 8-16.

Viewing BGP Statistics

You can obtain general BGP statistics by entering the following command:

```
bgp stat
```

A screen similar to the following displays:

```
                    ***** BGP Status *****

BGP is running

BGP Identifier                          : 198.206.184.97
Local Autonomous System                 : 2
Global Preference for Incoming Routes   : 170
Default Outbound Metric                 : none
Route Reflection Cluster ID             : 198.206.184.97
```

The first line indicates whether BGP is currently operational. In the sample screen above, BGP is in operation and the message **BGP is running** displays. If BGP were not operational, the message **BGP is not running** would display.

BGP Identifier. The IP address indicated in the **routerid** statement of the gated.conf file. This is the first interface seen by GateD.

Local Autonomous System. The autonomous system where this BGP speaker resides.

Global Preference for Incoming Routes. The global preference for BGP incoming routes. The default preference is 170. This preference may be overridden by a preference specified on the group or peer statement or by an import policy. However, the local_pref variable is always derived from preference (see page 8-3 for a discussion of the local_pref variable).

Default Outbound Metric. Defines the metric used when advertising routes via BGP. If not specified, no metric is propagated. This metric may be overridden by a metric specified on the peer or group statements, or in an export policy.

Route Reflection Cluster ID. The cluster ID for the BGP route reflector group. The cluster ID defaults to the router ID of the route reflector in a cluster. It is only required when multiple route reflectors are used in an autonomous system. A single cluster ID should be selected and configured on all route reflectors in the AS. The only constraints on the choice of cluster ID are the following:

- IDs of clusters within an AS must be unique within that AS, **and**
- the cluster ID must not be 0.0.0.0.

Choosing the cluster ID to be the router ID of one router in the AS fulfills these criteria. If there is only one route reflector in the cluster, the cluster ID setting may be omitted.

Viewing a List of BGP Peers

You can obtain a list of BGP peers by entering the following command:

bgp peers

A screen similar to the following displays:

```
***** BGP Peer List *****

1 BGP peer(s)

Peer ID          State    Local IP Addr    Remote IP Addr    Remote AS
================================================================================
198.206.184.112  active   0.0.0.0          21.0.0.1          10
```

Peer ID. The IP address indicated in the **routerid** statement of the gated.conf file. This is the first interface seen by GateD.

State. The current BGP peer connection state. Possible states are as follows:

idle	BGP refuses all incoming connections. No resources are allocated to the peer. If a BGP speaker detects an error, it will shut down a connection and change the state to **idle**.
connect	BGP waits for the transport protocol connection to complete. If the transport protocol succeeds, the local system clears the connect retry timer, initializes, sends an OPEN message to the peer and changes its state to **opensent**. If the protocol fails, then the local system continues to wait and changes its state to **active**.
active	BGP is actively trying to acquire a peer through the transport protocol.
open sent	BGP waits for an OPEN message to be sent from the peer. If an OPEN message is received and it does not contain errors, then BGP sends a KEEPALIVE message and starts the KEEPALIVE timer. If there is an error in the OPEN message, then the system sends a NOTIFICATION message and changes the state of the connection to **idle**.
open confirm	BGP waits for a KEEPALIVE or NOTIFICATION message from the remote peer. If a KEEPALIVE message is received, the state of the connection is changed to **established**. If a NOTIFICATION message is received or a timer expires, then the connection state reverts back to **idle**.
established	BGP can exchange UPDATE, NOTIFICATION, and KEEPALIVE messages with the remote peer.

Local IP Addr. The local IP address of this BGP entry's BGP connection.

Remote IP Addr. The remote IP address of this entry's BGP connection.

Remote AS. The autonomous system in which the remote BGP peer is a member.

Viewing Detailed Peer Information

You can obtain detailed information on a single BGP peer by entering the following command:

bgp peer <peer ID>

where **<peer ID>** is the IP address for the peer. You can find this ID through the **bgp peers** command, which lists the IDs for all BGP peers.

A screen similar to the following displays:

******* BGP Statistics for Peer ID 198.206.184.112 *******

Peer Name is	**: 21.0.0.1 (External AS 10)**		
Local IP Address	**: 21.0.0.2**	**In Updates**	**: 2**
Local Port	**: 1331**	**Out Updates**	**: 2**
Remote IP Address	**: 21.0.0.1**	**In Total Messages**	**: 71**
Remote Port	**: 179**	**Out Total Messages**	**: 83**
Negotiated Version	**: 4**	**Established Transitions**	**: 4**
Remote AS	**: 10**	**Established Time**	**: 3813**
Connect Retry Interval	**: 32**	**In Update Elapsed Time**	**: 286354**
Hold Time Configured	**: 180**	**Peer Hold Time**	**: 180**
KeepAlive Configured	**: 60**	**Peer KeepAlive**	**: 60**
Current TTL	**: 1**	**Connects Failed**	**: 0**

Peer Name is. The IP address of this BGP peer. Information is also included on whether the peer is internal or external to the local BGP speaker's autonomous system and the autonomous system to which the BGP peer belongs.

Local IP Address. The IP address for the local BGP speaker to which this peer has a connection.

Local Port. The local port to which this peer is connected.

Remote IP Address. The IP address of the remote BGP peer.

Remote Port. The port to which the remote BGP connects on its remote system.

Negotiated Version. The version of BGP currently operational in this autonomous system. There are versions 2, 3, and 4 of BGP. The switch supports only BGP version 4.

Remote AS. The autonomous system number for the remote BGP peer.

Connect Retry Interval. The interval, in seconds, between BGP retries to set up a connection via the transport protocol. In the connect state, BGP tries to set up a connection with a remote peer. If the connection fails, then the connect retry interval is started. Once the interval is reached, BGP retries setting up the connection.

Hold Time Configured. The hold timer, shown in seconds, is used during the connection setup process and in on-going connection maintenance with BGP peers. This value is placed in an OPEN message. The actual hold time will be the lower of the configured time and the hold time received in an OPEN connection message. The default value for this timer is 180 seconds. The minimum value is 3.

KeepAlive Configured. The keepalive timer, shown in seconds, is used during the connection setup process and in on-going connection maintenance with BGP peers. The suggested value for this timer is 30 seconds.

Current TTL. Current time to live (TTL) . By default, GateD sets the IP TTL for local peers to one (1) and the TTL for non-local peers to 255.

In Updates. The number of BGP UPDATE messages received on this connection. UPDATE messages are used to transfer routing information between BGP peers.

Out Updates. The number of BGP UPDATE messages sent on this connection. UPDATE messages are used to transfer routing information between BGP peers.

In Total Messages. The total number of messages received from the remote peer on this connection. The message types included in this count are UPDATE (routing information), OPEN (used during connection setup), KEEPALIVE (indicates an open connection), and NOTIFICATION (updates peers on errors and state changes).

Out Total Messages. The total number of messages sent to the remote peer on this connection. The message types included in this count are UPDATE (routing information), OPEN (used during connection setup), KEEPALIVE (indicates an open connection), and NOTIFICATION (updates peers on errors and state changes).

Established Transitions. The total number of times this BGP peer transitioned into the established state from the openconfirm state.

Established Time. The number of seconds this BGP peer has been in the established state or how long *since* this BGP has been in the established state.

In Update Elapsed Time. The elapsed time, in seconds, since the last BGP UPDATE message was received from this BGP peer.

Peer Hold Time. The actual hold time, in seconds, negotiated for this BGP peer. The actual hold time will be the lower of the configured time and the hold time received in an OPEN connection message.

Peer KeepAlive. The actual keepalive time, in seconds, established with this BGP peer. This BGP speaker calculates this time by comparing it to the actual hold time. The ratio between the actual keepalive time and the actual hold time will be equal to the configured keepalive time and the configured hold time. For example, if the actual hold time is 180 and the configured hold time and keepalive time are 90 and 30, respectively, then the actual keepalive time will be 60 (90/30=180/60).

Connects Failed. The total number of BGP connections that have been initiated, but have failed before reaching the established state.

Viewing BGP Paths

You can view a list of BGP paths to destination networks received from all BGP peers by entering the following command:

bgp paths

A screen similar to the following displays:

```
                          ***** BGP Path List *****

     IP Address
(Prefix/Prefix Len)          Next Hop          AS Path
================================================================================
21.0.0.0          / 8        21.0.0.1
33.0.0.0          / 8        21.0.0.1
45.0.0.0          / 8        21.0.0.1
198.206.184.0     / 24       21.0.0.1
```

IP Address. The IP address of the destination network along this path. This column also includes the length of the address prefix in bit.

Next Hop. The IP address for the border router that should be used for the destination network.

AS Path. The autonomous system (AS) path for this route path. If the full path cannot fit in this table, you can view it through the **bgp path** command, which is described in page 8-15.

Viewing Detailed BGP Path Information

You can view more detailed information on an individual BGP path by entering the following command:

bgp path <path IP address>

where **<path IP address>** is the IP address for this path. You can find this address through the **bgp path** command, which lists the path IP addresses for all found paths.

The following is sample output from the **bgp paths** command.

```
***** BGP Path Information for 198.206.184.0 *****

Peer IP Address          : 21.0.0.1      Next Hop              : 21.0.0.1
Multi Exit Discrim.      : none          Local Preference      : none
Aggregator AS            : none          Aggregator Address    : 0.0.0.0
Calculated Local Pref    : 84            Atomic Aggregate      : no
Best Route               : False         Origin                : IGP

AS Path
-------------------------------------------------------------------------------------------------------------
```

Peer IP Address. The IP address for the BGP peer on this network path.

Multi Exit Discrim. The value of the Multi-Exit Discriminator (MED) metric, which is used to decide the correct exit point among multiple exit points within the adjacent autonomous system. A value of **none** indicates the MED has not been configured. See page 8-6 for further information on this metric.

Aggregator AS. The autonomous system number of the last BGP speaker that performed route aggregation.

Calculated Local Pref. The degree of preference calculated by the receiving BGP speaker for this advertised route. This is determined by subtracting the absolute value of the route's global preference from 254.

The global protocol preference for a route is the preference defined in an import or export statement.

Best Route. Indicates whether this route was chosen by BGP as the best route to the destination.

Next Hop. The IP address of the border router that should be used to reach this destination network.

Local Preference. The originating BGP speaker's degree of preference for this advertised route. Sent to other BGP speakers in the same autonomous system. See page 8-3 for further information on local preference.

Aggregator Address. The IP address of the last BGP speaker that performed route aggregation.

Atomic Aggregate. Indicates whether or not the local system has selected a less specific route without selecting a more specific route. A value of **yes** means the system has selected a less specific route, while a value of **no** means the system has not selected a less specific route.

Origin. Indicates whether this path information was learned within this BGP peer's autonomous system (IGP) or external to this autonomous system (EGP).

AS Path. The autonomous system (AS) path for this route path.

The BGP Statement

The following two pages show the full BGP command statement. These statements are entered in the gated.conf file. Each part of the statement (i.e., global statements, group statements, peer statements) is described in sections following this full statement.

bgp yes | no | on | off
[{
preference *preference* **;**
defaultmetric *metric* **;**
traceoptions *trace_options* **;**
[clusterid *host* **;]**
group type
(external peeras *autonomous_system*
[ignorefirstashop]
[med])
[localas *autonomous_system***])**
| (internal peeras *autonomous_system*
[ignorefirstashop]
[lcladdr *local_address* **]**
[outdelay *time* **]**
[metricout *metric* **]**
[reflector-client [no-client-reflect]]
[setpref *metric* **])**
| (routing peeras *autonomous_system* **proto** *proto*
interface *interface_list*
[ignorefirstashop]
[lcladdr *local_address* **]**
[outdelay *time* **]**
[metricout *metric* **]**
[reflector-client [no-client-reflect]]
[setpref *metric* **])**
| (test peeras *autonomous_system* **)**

[aspath-opt]
{
allow {
network
network **mask** *mask*
network **masklen** *number*
all
host *host*
} ;

```
                peer host
                        [ metricout metric ]
                        [ ignorefirstashop ]
                        [ nogendefault ]
                        [ gateway gateway ]
                        [ nexthopself ]
                        [ preference preference ]
                        [ preference2 preference ]
                        [ lcladdr local_address ]
                        [ holdtime time ]
                        [ version number ]
                        [ passive ]
                        [ sendbuffer number ]
                        [ recvbuffer number ]
                        [ outdelay time ]
                        [ keep [ all | none ] ]
                        [ show-warnings ]
                        [ noaggregatorid ]
                        [ keepalivesalways ]
                        [ v3asloopokay ]
                        [ nov4asloop ]
                        [ ascount count ]
                        [ logupdown ]
                        [ ttl ttl ]
                        [ traceoptions trace_options ]
                        ;
        };
};
```

The BGP statement can be divided into the following three parts: global statements, group statements, and peer statements. Each of these statement types will be covered in the following sections.

The syntax conventions used for showing BGP commands are the same as those used in other parts of the *Advanced Routing User's Guide*. See that manual for a full description of syntax conventions.

The autonomoussystem Statement

The **autonomoussystem** statement is required before the main BGP statement and any supporting group or peer statements. Without this statement, the switch will not know the AS to which its BGP peer belongs and BGP will not be able to communicate with other peers. The syntax for this statement is as follows:

autonomoussystem *autonomous_system*

where *autonomous_system* indicates the AS number supported by this switch. A switch supporting AS 2 would specify this statement as:

autonomoussystem 2

Global BGP Statements

The first set of BGP commands defines global parameters for the protocol, such as enabling the protocol and setting global route preference. The statements included as global statements are as follows:

bgp yes | no | on | off

[{

preference *preference* **;**

defaultmetric *metric* **;**

traceoptions *trace_options* **;**

[clusterid *host* **;]**

Each command statement is described below:

bgp yes | no | on | off

The bgp statement enables or disables BGP. By default, BGP is disabled. The default metric for announcing routes via BGP is no metric. Example:

bgp yes

preference *preference*

Sets the global preference for BGP incoming routes. The default preference is 170. This preference may be overridden by a preference specified on the group or peer statement or by an import policy. However, the local_pref variable is always derived from this preference value. Example:

preference 170

defaultmetric *metric*

Defines the metric used when advertising routes via BGP. If not specified, no metric is propagated. This metric may be overridden by a metric specified on the peer or group statements, or in an export policy.

traceoptions *trace_options*

Specifies the tracing options for BGP. By default, these options are inherited from the global trace options. These values may be overridden by a group or peer statement.

clusterid *host*

The cluster ID for the BGP route reflector group. The cluster ID defaults to the router ID of the route reflector in an autonomous system. The cluster ID is only required when multiple route reflectors are used in an AS. A single cluster ID should be selected and configured on all route reflectors in the AS. The only constraints on the choice of cluster ID are the following:

- IDs of clusters within an AS must be unique within that AS, **and**
- The cluster ID must not be 0.0.0.0.

Choosing the cluster ID to be the router ID of one router in the AS fulfills these criteria. If there is only one route reflector in the AS, the cluster ID setting may be omitted. Example:

clusterid 21.0.0.1

Group Statements

The BGP statement has group clauses and peer subclauses. Any number of peer subclauses may be specified within a group clause. A group clause usually defines default parameters for a group of peers. These group parameters apply to all peer subclauses. Any parameters from the peer subclause may be specified on the group clause to provide defaults for the whole group. When the same statement appears in the group and peer subclause, the peer statement will take precedence.

BGP peers are grouped by type and autonomous system. Any number of groups may be specified, but each must have a unique combination of type, peer autonomous system, and aspath-opt options. There are four possible group types:

- Classic external BGP groups
- Internal BGP groups using IP-level routing
- Internal BGP groups with no internal routing
- External BGP groups with test peers

Each of these four group types and their associated command statements are described in the following sections.

The group statements are as follows:

group type
(external peeras *autonomous_system*
[ignorefirstashop]
[med])
[localas *autonomous_system*] **)**
| (internal peeras *autonomous_system*
[ignorefirstashop]
[lcladdr *local_address* **]**
[outdelay *time* **]**
[metricout *metric* **]**
[reflector-client [no-client-reflect]]
[setpref *metric* **])**
| (routing peeras *autonomous_system* **proto** *proto*
interface *interface_list*
[ignorefirstashop]
[lcladdr *local_address* **]**
[outdelay *time* **]**
[metricout *metric* **]**
[reflector-client [no-client-reflect]]
[setpref *metric* **])**
| (test peeras *autonomous_system* **)**

[aspath-opt]

Classic External BGP Groups

In the classic external BGP group, full policy checking is applied to all incoming and outgoing advertisements. The external BGP speakers must be directly reachable through one of the local connections (i.e., the neighbor BGP speakers must share a subnet). The next hop transmitted is computed with respect to the shared interface. The following are statements for the classic external BGP group:

group type external peeras *autonomous_system*

The **group type external** clause indicates this is an external BGP group. The **peeras** *autonomous_system* clause indicates the autonomous system where this group resides. Example:

```
group type external peeras 3
```

med

By default, any Multi-Exit Discriminator (MED) metric received on an external BGP connection is ignored. To use MEDs in routing computations, the **med** option must be specified for a group. By default, MEDs are not sent on external connections. To configure MEDs, use the **metric** option of the export statement or the **metricout** peer/group parameter

ignorefirstashop

Some routers, known as route servers, are capable of propagating routes without appending their own AS to the AS Path. By default, GateD will drop such routes. Specifying **ignorefirstashop** on either the group clause or peer clause disables this feature. This option should only be used if you know the peer is a route server and not a normal router.

localas *autonomous_system*

Identifies the autonomous system that GateD is representing to this group of peers. The default is the autonomous system number that has been set globally in the **autonomoussystem** statement (which should appear before the BGP statement). Example:

```
localas 3
```

aspath-opt

Within a BGP group statement, the **aspath-opt** statement is used to generate community attributes. See page 8-31 for more detailed information on aspath-opt community statements. Example:

```
aspath-opt {
        comm-split 3 2
}
```

Internal BGP Groups Using Internal Protocol Routing

The second BGP group type is an internal group that uses the routes of an interior protocol to resolve forwarding addresses. This group type propagates external routes between routers that are not directly connected to the external network. This kind of group also computes immediate next hops for external routes by using the BGP next hop that arrives with the route as a forwarding address to be resolved via an internal protocol's routing information. In this group, internal BGP is used to carry AS external routes while the internal protocol carries AS internal routes and finds immediate next hops for BGP.

group type routing peeras *autonomous_system* **proto** *proto*

The **group type routing** clause indicates this is an internal routing BGP group. The **peeras** *autonomous_system* clause indicates the autonomous system where this group resides. The **proto** statement names the interior protocol to be used to resolve BGP route next hops, and may be the name of any internal routing protocol in the configuration, including **static**. By default, the next hop advertised to BGP routing peers will be set to the local address on the BGP connection to those peers, as it is assumed a route to this address will be propagated via the internal routing protocol. Example

group type routing peeras 3 proto rip

interface *interface_list*

The *interface_list* can optionally provide a list of interfaces with routes that are carried via the internal routing protocol. Third-party next hops may use this list instead of the local address of the BGP connection.

ignorefirstashop

Some routers, known as route servers, are capable of propagating routes without appending their own AS to the AS Path. By default, GateD will drop such routes. Specifying **ignorefirstashop** on either the group clause or peer clause disables this feature. This option should only be used if you know the peer is a route server and not a normal router.

lcladdr *local_address*

Specifies the address to be used on the local end of the TCP connection with the peer. For external peers the local address must be on an interface that is shared with the peer (i.e., a shared subnet). A session with an external peer will only be opened when an interface with the appropriate local address—through which the peer or gateway address is directly reachable—is operating. For non-external peers, a peer session will be maintained when any interface with the specified local address is operating. In either case incoming connections will only be recognized as matching a configured peer if they are addressed to the configured local address. For internal-routing group types, set the **lcladdr** in the group clause rather than in a peer clause. If this parameter is set on a peer clause, it must equal the value in the group clause. For internal-routing groups, it is recommended to set the **lcladdr** to a non-physical interface, such as a loopback interface.

outdelay *time*

Used to dampen route fluctuations. Outdelay is the amount of time, in seconds, a route must be present in the GateD routing database before it is exported to BGP. The default value is zero (0), meaning that route dampening is disabled. For an internal-routing group, set the **outdelay** in the group clause rather than in a peer statement. If this parameter is set on a peer statement, it must equal the value set up in the group statement. For more information on route dampening, see page 8-6; for information on route dampening command statements, see page 8-30.

metricout *metric*

The metric that may be used on all routes sent to the peers in this BGP group. This value is also known as the Multi-Exit Discriminator (MED). The metric hierarchy is as follows, starting from the most preferred:

1. The metric specified by the export policy.
2. The metricout specified on the peer clause.
3. The metricout specified on the group clause (this parameter).
4. The default metric.

For internal BGP groups, this parameter should be set on the group clause rather than the peer clause. If this parameter is set on a peer clause, it must equal the value set up in the group clause.

aspath-opt

Within a BGP group statement, the **aspath-opt** statement is used to generate community attributes. See page 8-31 for more detailed information on aspath-opt community statements. Example:

```
aspath-opt {
        comm-split 3 2
}
```

reflector-client [no-client-reflect]

The **reflector-client** option specifies that this switch will act as a route reflector. All routes received from any reflector cluster member will be sent to all other internal neighbors, and all routes received from any other internal neighbors will be sent to the reflector clients. Since the route reflector forwards routes in this way, the reflector-client cluster need not be fully meshed.

If the **no-client-reflect** option is specified, routes received from reflector clients will only be sent to internal neighbors that are not in the same cluster as the sending reflector client. In this case the reflector-client group should be fully meshed. In all cases, routes received from normal internal peers will be sent to all reflector clients. See page 8-7 for a more detailed discussion of route reflection.

Note that it is necessary to export routes from the local AS into the local AS when a switch is acting as a route reflector. For example, suppose that the local AS number is 2. The following export statement would ensure reflection worked correctly:

```
export proto bgp as 2 {
        proto bgp as 2 {all;};
};
```

If the cluster ID is changed and GateD is reconfigured, all BGP sessions with reflector clients will be dropped and restarted.

setpref *metric*

Allows the BGP local_pref variable to be used in setting the GateD preference. The setpref metric works as a lower limit, below which the imported local_pref may not set the GateD preference. See page 8-3 for more discussion on the local_pref and setpref variables.

Internal Group with No IP-Level Routing

The third BGP group type is an internal group operating with no IP-level internal routing protocol (e.g., an SMDS network or MILNET). All neighbors in this group are required to be directly reachable via a single interface, and all next-hop information is computed with respect to this interface. Import and export policy may be applied to group advertisements. Routes received from external BGP or external neighbors are by default readvertised with the received metric.

group type internal peeras *autonomous_system*

The **group type internal** clause indicates this is an internal BGP group with no IP-level routing. The **peeras** *autonomous_system* clause indicates the autonomous system where this group resides. Example:

```
group type internal peeras 3
```

ignorefirstashop

Some routers, known as route servers, are capable of propagating routes without appending their own AS to the AS Path. By default, GateD will drop such routes. Specifying **ignorefirstashop** on either the group clause or peer clause disables this feature. This option should only be used if you know the peer is a route server and not a normal router.

aspath-opt

Within a BGP group statement, the **aspath-opt** statement is used to generate community attributes. See page 8-31 for more detailed information on aspath-opt community statements. Example:

```
aspath-opt {
        comm-split 3 2
}
```

reflector-client [no-client-reflect]

The **reflector-client** option specifies that this switch will act as a route reflector. All routes received from any reflector cluster member will be sent to all other internal neighbors, and all routes received from any other internal neighbors will be sent to the reflector clients. Since the route reflector forwards routes in this way, the reflector-client cluster need not be fully meshed.

If the **no-client-reflect** option is specified, routes received from reflector clients will only be sent to internal neighbors that are not in the same cluster as the sending reflector client. In this case the reflector client group should be fully meshed. In all cases, routes received from normal internal peers will be sent to all reflector clients. See page 8-7 for a more detailed discussion of route reflection.

Note that it is necessary to export routes from the local AS into the local AS when a switch is acting as a route reflector. For example, suppose that the local AS number is 2. The following export statement would ensure reflection worked correctly:

```
export proto bgp as 2 {
        proto bgp as 2 {all;};
};
```

If the cluster ID is changed and GateD is reconfigured, all BGP sessions with reflector clients will be dropped and restarted.

lcladdr *local_address*

Specifies the address to be used on the local end of the TCP connection with the peer. For external peers the local address must be on an interface that is shared with the peer (i.e., a shared subnet). A session with an external peer will only be opened when an interface with the appropriate local address—through which the peer or gateway address is directly reachable—is operating. For non-external peers, a peer session will be maintained when any interface with the specified local address is operating. In either case incoming connections will only be recognized as matching a configured peer if they are addressed to the configured local address. For internal group types, set the **lcladdr** in the group clause rather than in a peer clause. If this parameter is set on a peer clause, it must equal the value in the group clause.

outdelay *time*

Used to dampen route fluctuations. Outdelay is the amount of time, in seconds, a route must be present in the GateD routing database before it is exported to BGP. The default value is zero (0), meaning that route dampening is disabled. For an internal group, set the **outdelay** in the group clause rather than in a peer clause. If this parameter is set on a peer clause, it must equal the value set up in the group statement. For more information on route dampening, see page 8-6; for information on route dampening command statements, see page 8-30.

metricout *metric*

The metric that may be used on all routes sent to the peers in this BGP group. This value is also known as the Multi-Exit Discriminator (MED). The metric hierarchy is as follows, starting from the most preferred:

1. The metric specified by the export policy.
2. The metricout specified on the peer clause.
3. The metricout specified on the group clause (this parameter).
4. The default metric.

For internal BGP groups, this parameter should be set on the group clause rather than the peer clause. If this parameter is set on a peer clause, it must equal the value set up in the group clause.

setpref *metric*

Allows the BGP local_pref variable to be used for GateD preference. The setpref metric works as a lower limit, below which the imported local_pref may not set the GateD preference. See page 8-3 for more discussion of the local_pref and setpref variables.

External Group with Test Peers

The fourth BGP group type is an extension to external BGP that implements a fixed policy using test peers. Fixed policy make test peers inexpensive to maintain. Test peers do not need to be on a directly attached network. If GateD and the peer are on the same (directly attached) subnet, the advertised next hop is computed with respect to that network; otherwise, the next hop is the local switch's current next hop. All routing information advertised by and received from a test peer is discarded, and all BGP-advertisable routes are sent back to the test peer. Metrics from external and BGP-derived routes are forwarded in the advertisement; otherwise, no metric is included.

group type test peeras *autonomous_system*

The **group type test** clause indicates this is an external BGP test group. The **peeras** *autonomous_system* clause indicates the autonomous system where this group resides.

Peer Statements

Within each group clause, individual peers can be specified using a peer clause. The peer clause configures an individual peer. In addition, a group of potential peers can be specified using the allow clause. The allow clause specifies a set of address masks. Both methods and clauses are described in the following sections.

The Allow Clause

The allow clause specifies peer connections from any address in a specified range of network and mask pairs. If GateD receives a BGP connection request from any address in the set specified, it will accept it and set up a peer relationship.

All parameters for the peers in an allow clause must be configured on the group clause. An internal peer is created when an incoming open request is received, and closed when the connection is broken. The following are the commands used in the allow clause:

allow {

network
network **mask** *mask*
network **masklen** *number*
all
host *host*

network

The IP address of the network to be included in this peer grouping. Usually specified in dotted decimal format with one to four values in the range of 0 to 255. For example, specifying 139.206.132 would allow peer connections from any address in the 139.206.132 network. The network mask in this statement is implied.

network **mask** *mask*

The IP address for the network plus a mask for that network to be included in this peer grouping. The network mask in this format is explicitly defined.

network **masklen** *number*

The IP address of the network plus the mask for the network as defined by the number of contiguous one bits to be included in this peer grouping.

all

All networks should be included in this peer grouping. This parameter is the same as specifying:

0.0.0.0 mask 0.0.0.0

host *host*

The networks specified by this host should be included in this peer grouping.

The Peer Clause

The peer clause configures an individual peer. Each peer inherits all parameters specified on its associated group statement as defaults. However, many group defaults may be overridden by parameters specified in the peer subclause. The following is the structure and commands used in the allow clause.

peer *host*
[metricout *metric* **]**
[ignorefirstashop]
[nogendefault]
[gateway *gateway* **]**
[nexthopself]
[preference *preference* **]**
[preference2 *preference* **]**
[lcladdr *local_address* **]**
[holdtime *time* **]**
[version *number* **]**
[passive]
[sendbuffer *number* **]**
[recvbuffer *number* **]**
[outdelay *time* **]**
[keep [all | none]]
[show-warnings]
[noaggregatorid]
[keepalivesalways]
[v3asloopokay]
[nov4asloop]
[ascount *count* **]**
[logupdown]
[ttl *ttl* **]**
[traceoptions *trace_options* **]**

The BGP peer subclause allows the following parameters, which can also be specified on the group clause. All of these parameters are optional.

metricout *metric*

The metric that may be used on all routes. This value is also known as the Multi-Exit Discriminator (MED). The metric hierarchy is as follows, starting from the most preferred:

1. The metric specified by the export policy.
2. The metricout specified on the peer clause (this parameter).
3. The metricout specified on the group clause.
4. The default metric.

For internal BGP groups, this parameter should be set on the group clause rather than the peer clause. If this parameter is set on a peer clause, it must equal the value set up in the group clause.

nogendefault

Prevents GateD from generating a default route when BGP receives a valid update from its neighbor. The default route is only generated when the **gendefault** option is enabled in the **options** statement of the GateD configuration file.

ignorefirstashop

Some routers, known as route servers, are capable of propagating routes without appending their own AS to the AS Path. By default, GateD will drop such routes. Specifying **ignorefirstashop** on either the group clause or peer clause disables this feature. This option should only be used if you know the peer is a route server and not a normal router.

gateway *gateway*

If a network is not shared with a peer, the **gateway** parameter specifies a router on an attached network to be used as the next hop router for routes received from the neighbor. The gateway parameter may also be used to specify a next hop for peers that are on shared networks. This parameter can be used to ensure that third-party next hops are never accepted from a given peer by specifying that peer's address as its own gateway. This parameter is not needed in most cases.

nexthopself

Sets the next hop in route advertisements for this peer to this BGP speaker's address, even if it would normally be possible to send a third-party next hop. Use of this option may cause inefficient routes to be followed, but it may be necessary in cases where the routers on the shared medium do not have full connectivity to each other.

preference *preference*

Specifies the preference used for routes learned from these peers. This value can differ from the default BGP preference set in the BGP statement, so that GateD can prefer routes from one peer group over others. This preference may be explicitly overridden by an import policy.

preference2 *preference*

In the case of a preference tie, this second preference—preference2—may be used to break the tie. The default value is 0.

lcladdr *local_address*

Specifies the address to be used on the local end of the TCP connection with the peer. For external peers the local address must be on an interface that is shared with the peer (i.e., a shared subnet). A session with an external peer will only be opened when an interface with the appropriate local address—through which the peer or gateway address is directly reachable—is operating. For other types of peers, a peer session will be maintained when any interface with the specified local address is operating. In either case incoming connections will only be recognized as matching a configured peer if they are addressed to the configured local address. For internal and internal-routing group types, set the **lcladdr** in the group clause rather than in a peer clause. If this parameter is set on a peer clause, it must equal the value set up in the group clause. For internal-routing groups, it is recommended to set the **lcladdr** to a non-physical interface, such as a loopback interface.

holdtime *time*

Specifies the BGP holdtime value, in seconds, to use when negotiating the connection with this peer. If GateD does not receive a KEEPALIVE, UPDATE, or NOTIFICATION message within the period specified in the BGP open message, then the BGP connection will be closed. This value must be at least 3.

version *version*

Specifies the version of the BGP protocol to use with this peer. If not specified, the highest supported version is used first and version negotiation is attempted. If it is specified, only the specified version will be offered during negotiation. Alcatel supports only BGP version 4.

passive

Not needed with BGP version 4. If this option is used, GateD will never try to open a BGP connection with this peer. Instead, it will wait for the peer to initiate a connection. This option was introduced to handle a problem in BGP version 3 and earlier in which two peers both attempt to initiate a connection at the same time. This problem is fixed in BGP version 4 so passive is not needed with BGP sessions.

sendbuffer *buffer_size*

recvbuffer *buffer_size*

Controls the amount of send and receive buffering asked of the kernel in bytes. The maximum buffer size supported is 65535 bytes although many kernels have a lower limit. By default, GateD configures the maximum supported. These parameters are not needed in normally functioning systems.

outdelay *time*

Used to dampen route fluctuations. Outdelay is the amount of time a route must be present in the GateD routing database before it is exported to BGP. The default value is zero (0), meaning that route dampening is disabled. For an internal group, set the **outdelay** in the group clause rather than in a peer clause. If this parameter is set on a peer clause, it must equal the value set up in the group clause.

keep [all | none]

Used to retain routes learned from a peer even if the routes' AS paths contain one of this switch's exported AS numbers.

show-warnings

When specified, GateD will issue warning messages when receiving questionable BGP updates such as duplicate routes and/or deletions of non-existing routes. Normally these events are not reported.

noaggregatorid

When enabled, GateD will specify the router-id in the aggregator attribute as zero (instead of as its router-id) to prevent different routers in an AS from creating aggregate routes with different AS paths.

keepalivesalways

When specified, GateD will always send KEEPALIVE messages, even when an UPDATE message could be used. This feature allows interoperability with routers that do not completely adhere to BGP message handling specifications.

v3asloopokay

By default GateD will not advertise routes with a looped AS path (i.e., with an AS appearing more than once in the path) to version 3 external peers. Setting this flag removes this constraint. This parameter is ignored when set on internal groups or peers.

nov4asloop

Prevents routes with looped AS paths from being advertised to version 4 external peers. This can be useful to avoid advertising such routes to peers that would incorrectly forward the routes to version 3 neighbors.

ascount *count*

The number of times GateD will insert the AS number when it sends the AS path to an external neighbor. The default is 1. Higher values are typically used to bias upstream neighbors' route selection. Most routers will prefer to use routes with shorter AS Paths. Using **ascount**, the AS Path can be artificially lengthened.

Note that **ascount** supersedes the **nov4asloop** option. Regardless of whether nov4asloop is set, GateD will still send multiple copies of this switch's AS if the ascount option is greater than one. Also, note that if the value of ascount is changed and GateD is reconfigured, routes will not be sent to reflect the new setting. If this is desired, it will be necessary to restart the peer session.

logupdown

When specified, GateD logs a message via the syslog mechanism whenever a BGP peer enters or leaves the established state.

ttl *ttl*

Time to live. By default, GateD sets the IP TTL for local peers to one (1) and the TTL for non-local peers to 255. This option can be used when communicating with improperly functioning routers that ignore packets sent with a TTL of one (1). Not all kernels allow the TTL to be specified for TCP connections.

traceoptions *trace_options*

Specifies the tracing options for this BGP peer. By default, trace options are inherited from group or BGP global trace options. Note that the state option works with BGP, but does not provide true state transition information. The following are BGP packet tracing options. They may be modified with **detail**, **send**, and **recv** parameters.

packets	All BGP packets
open	BGP OPEN packets that are used to establish a peer relationship.
update	BGP UPDATE packets that are used to pass network reachability information.
keepalive	BGP KEEPALIVE packets that are used to verify peer reachability.
all	Additional useful information, including additions/changes/deletions to the GateD routing table.

Route Dampening Statements

Route flapping in BGP is controlled by a set of statements outside the main BGP statement. The route dampening statements are as follows:

dampen-flap {
[suppress-above *metric* **;**
reuse-below *metric* **;**
max-flap *metric***;**
unreach-decay *time* **;**
reach-decay *time* **;**
keep-history *time* **;]**
};

GateD keeps track of an "instability metric," that is incremented by one (1) every time a route is unreachable, or every time a route "flaps." The route dampening parameters are configured in relation to this instability metric.

suppress-above *metric*

The value of the instability metric at which route suppression will take place. Once this route suppression level is reached, a route will not be announced even if it is reachable. The default value is 3.

reuse-below *metric*

The value of the instability metric at which a suppressed route will become unsuppressed, if it is reachable but currently suppressed. The value assigned to the **reuse-below** parameter must be less than the value for the **suppress-above** parameter. The default value is 2.

max-flap *metric*

The upper limit of the instability metric. This value must be greater than the **suppress-above** value. If suppress-above is less than one (1), then **max_flap** must be greater than one (1). The default value is 16

reach-decay *time*

Specifies the time, in seconds, desired for the instability metric value to reach one half of its current value when the route is reachable. This half-life value determines the rate at which the metric value is decayed. A smaller half-life value will make a suppressed route reusable sooner than a larger value. The default value is 300.

unreach-decay *time*

The same as reach-decay except that it specifies the rate at which the instability metric is decayed when a route is unreachable. It should have a value greater than or equal to **reach-decay**. The default value is 900.

keep-history *time*

Specifies the period, in seconds, over which the route flapping history is to be maintained for a given route. The size of the configuration information is directly affected by this value. The default value is 1800.

Community Statements

BGP communities are configured in the **aspath-opt** clause within a BGP group statement. The full aspath-opt statement for BGP is shown below:

[aspath-opt {
[comm-split *autonomous_system community_id* **]**
[community no-export | no-advertise | no-export-subconfed | none]
}]

comm-split *autonomous_system community_id*

This statement indicates the community to which a particular autonomous system should be assigned. The *autonomous_system* should be the local AS and the *community_id* is the community to which the AS will be assigned. Example:

comm-split 3 1

The communities that are actually sent are the combination of the communities received with a route, those communities specified in a BGP group policy, and those specified in an export policy. When communities are received, all communities specified in aspath-opt statements must be present in the UPDATE message; additional communities could be included in the UPDATE message.

community no-export | no-advertise | no-export-subconfed | none

These statements configure special community types. Each possible statement is described below:

community no-export

Routes for this community will not be advertised outside the AS.

community no-advertise

Routes for this community will not be advertised to other BGP peers within this AS.

community no-export-subconfed

Routes for this community will not be advertised to external BGP peers.

community none

Indicates that a received BGP update will only be matched if no communities are present in the UPDATE message. It does not affect the creation of communities.