Chapter 18
Multi-Protocol - Border Gateway Protocol (MPBGP)

allow

Name

allow - permits peer connections from any addresses in the specified range (including single host or "**all**") of network and mask pairs

Syntax

```
[inet6] allow {
all ;
    | host ipaddress ;
    | classful_network ;
    | network mask mask ;
    | network masklen masklennumber ;
    | network / masklennumber ;
} ;
```

Parameters

classful_network - the IPv4 address of the network to be peered, in dotted-quad format (xxx.xxx.xxx.xxx)

network - the IPv4 or IPv6 address of the network to be peered

mask - the address mask (modification) in dotted-quad format (xxx.xxx.xxx.xxx)

masklennumber - the number of contiguous one bits at the beginning of the mask

ip_address - the IPv4 or IPv6 address of the host network

Description

allow permits peer connections from any addresses in the specified range of networks. Multiple networks may be specified in the **allow** clause, but must be separated with semicolons.

All parameters for these peers must be configured on the **group** clause. The internal peer structures are created when an incoming open request is received, and destroyed when the connection is broken.

For more details on specifying the network/mask pairs, see “Route Filtering” on page 129 in *Configuring GateD*.

Default

The default is an empty **allow** clause, allowing no non-explicitly configured peers.

Context

mpbgp group type external statement

mpbgp group type internal statement

mpbgp group type routing statement

Examples

Example 1

This allows anyone to peer with us and is not recommended.

```
allow {
    all;
} ;
```

Example 2

This allows a specific host to peer with us. Note that this is equivalent to specifying a **peer** statement without any options.

```
allow {
    host 192.0.2.1 ;
} ;
```

Example 3

This allows all hosts within a classful subnet.

```
allow {
    10 ; # Allow all 10/8
} ;
```

Example 4

This allows all hosts within a network:

```
allow {
    192.0.2.0 mask 255.255.255.0 ;
};
```

which is the same as:

```
allow {
    192.0.2.0 masklen 24 ;
```

```
} ;
```

which is the same as:

```
allow {
    192.0.2.0/24 ;
} ;
```

Example 5

You can specify more than one network in an **allow** clause.

```
allow {
    192.0.2.0/24 ;
    10.0.0.0/8 ;
    host 172.16.0.1 ;
} ;
```

Example 6

IPv6 allow lists can also be specified.

```
inet6 allow {
    all;
};
```

Example 7

IPv6 and IPv4 lists can be used together to allow multiple address families.

```
inet6 allow {
    fec0:0:0:70a::/64;
};
allow {
        #IPv4 allow list
    all;
};
```

See Also

mpbgp on page 482

“Route Filtering” on page 129 in *Configuring GateD*

ascount

Name

ascount - configures the number of times that this router will prepend its autonomous system (AS) number to a route's AS Path when it sends the route to an external peer

Syntax

ascount *count*

Parameters

count - an integer between 1 and 25, inclusive

Description

ascount configures the number of times that this router will prepend its AS number to a route's AS Path when it sends the route to an external peer. Larger values are typically used to bias upstream peers' route selection. All things being equal, most routers will prefer routes with shorter AS Paths. Using **ascount**, the AS Path this router sends can be artificially lengthened.

The AS number that is prepended to the AS Path is the **localas** value, if specified, or the global **autonomoussystem** value.

ascount supersedes the deprecated **nov4asloop** option. Regardless of whether **nov4asloop** is set, this router will still send multiple copies of its own AS if the **ascount** option is set to something greater than 1.

Default

ascount 1

Context

mpbgp group type external statement

mpbgp peer statement

Examples

Example 1

The **ascount** is specified for a peer group.

[**autonomoussystem** and **routerid** omitted]

```
mpbgp on {
    group type external peeras 64512 ascount 5 {
        peer 192.0.2.2 ; # peer inherits ascount of 5 from group
        peer 192.0.2.3 ; # peer inherits ascount of 5 from group
    } ;
} ;
```

[**import** and **export** statements omitted]

Example 2

The **ascount** is specified for a given peer.

[**autonomoussystem** and **routerid** omitted]

```
mpbgp on {
    group type external peeras 64512 {
        peer 192.0.2.2 ascount 5 ; # This peer has an ascount of 5
        peer 192.0.2.3 ;           # This peer defaults to ascount 1
        } ;
    } ;
} ;
```

[**import** and **export** statements omitted]

See Also

localas on page 469

mpbgp on page 482

“Route Filtering” on page 129 in *Configuring GateD*

caps

Name

caps - sets multiprotocol capabilities to be advertised to a group of peers

Syntax

caps { *caps_list* **}**

caps no-caps

Parameters

caps_list - specify one or more of the following: **v4u**, **v4m**, **v4um**, **v6u**, **v6m**, **v6um** (separated by spaces)

Description

caps sets a list of multiprotocol capabilities to send to a group of peers (using the BGP Capability Advertisement mechanism defined in RFC 2842).

v4u, **v4m**, **v4um**, **v6u**, **v6m**, and **v6um** represent Address Family Identified/Subsequent Address Family Identifier (AFI/SAFI) pairs (see RFC 2858 - Multiprotocol Extensions for BGP-4). "**v4**" represents the IPv4 AFI. "**v6**" represents the IPv6 AFI. "**u**" represents unicast (SAFI 1), "**m**" represents multicast (SAFI 2), "**um**" represents unicast and multicast (SAFI 3). **no-caps** specifies that capabilities will not be understood. "**v6**" represents the IPv6 AFI.

All peers in this group must include all of the capabilities that are listed in the **caps** clause. If the remote peer is missing any of the capabilities specified in this clause, the peering session will not be established.

Default

none (capability announcement not performed)

Context

mpbgp group type external statement

mpbgp group type internal statement

Example

[**autonomoussystem** and **routerid** omitted]

```
mpbgp on {
    group type internal peeras 65534 caps { v4u v4m v4um } {
        # announce all ipv4 capabilities to this peer
        peer 192.0.2.10;
    } ;
    group type external peeras 65533 caps { v4u } {
        # announce only capability v4u to this peer
        peer 192.0.2.11;
```

```
    } ;
    group type external peeras 65532 {
        # no capabilities sent to this peer
        peer 192.0.2.12;
    } ;
    group type external peeras 65531 caps { v6u } localv4addr 192.0.2.9 {
        # announce the ipv6 unicast capability to this peer
        peer fec0:0:0:ff::d;
    } ;
} ;
```

clusterid

Name

clusterid - specifies the route reflection cluster ID for MPBGP

Syntax

clusterid *host-id* **;**

Parameters

host-id - a router ID, in dotted-quad format (xxxx.xxxx.xxxx.xxxx), used by route reflectors to prevent route propagation loops within the cluster

Description

clusterid specifies the route reflection cluster ID for MPBGP. (See "Route Reflection" on page 71 in *Configuring GateD*.) The cluster ID defaults to be the same as the router ID. (See "Router ID Syntax" on page 30 in *Configuring GateD* for more information about router IDs.) If a router is to be a route reflector, then a single cluster ID should be selected and configured on all route reflectors in the cluster. The only constraints on the choice of cluster ID are that:

- IDs of clusters within an autonomous system (AS) must be unique within that AS.
- The cluster ID must not be 0.0.0.0.

Choosing the cluster ID to be the router ID of one router in the cluster will always fulfill these criteria. If there is only one route reflector in the cluster, the **clusterid** setting may be omitted because the default will suffice.

Default

the globally configured **routerid**

Context

mpbgp statement

Examples

Example 1

The **clusterid** is specified.

[**autonomoussystem** omitted]

```
routerid 192.0.2.1 ;
mpbgp on {
    clusterid 192.0.2.254 ;
    group type internal peeras 65534 {
        peer 192.0.2.2 ;
        peer 192.0.2.3 ;
```

```
        } ;
# Routes received from clients will have the clusterid of
# 92.0.2.254 added to their BGP attributes.
    group type internal peeras 65534 reflector-client {
        peer 192.0.2.10 ;
        peer 192.0.2.11 ;
        peer 192.0.2.12 ;
        peer 192.0.2.13 ;
    } ;
} ;
```

[**import** and **export** statements omitted]

Example 2

The **clusterid** is omitted.

[**autonomoussystem** omitted]

```
routerid 192.0.2.1 ;
mpbgp on {
    group type internal peeras 65534 {
        peer 192.0.2.2 ;
        peer 192.0.2.3 ;
    } ;
    # Routes received from clients will have the clusterid of 192.0.2.1
    # added to their BGP attributes.
    group type internal peeras 65534 reflector-client {
        peer 192.0.2.10 ;
        peer 192.0.2.11 ;
        peer 192.0.2.12 ;
        peer 192.0.2.13 ;
    } ;
} ;
```

[**import** and **export** statements omitted]

See Also

mpbgp on page 482

“Route Reflection” on page 71 in *Configuring GateD*

comm

Name

comm - specifies the community attributes added to routes sent to routers in a peer group

Syntax

comm {*community_values***}**

Parameters

community_values include:

comm-hex *hex-number hex-number* - This parameter specifies any arbitrary community that is the concatenation of the two 16-bit numbers specified.

comm-split *autonomous_system community-id* - This parameter specifies a community that is the concatenation of the AS number ASN and the arbitrary 16-bit number.

community no-export - Specifies the well-known community NO_EXPORT as defined in RFC 1997. Routes tagged with this community are not to be exported outside of a confederation boundary.

community no-advertise - Specifies the well-known community NO_ADVERTISE as defined in RFC 1997. Routes tagged with this community are not to be advertised to any other peers.

community no-export-subconfed - Specifies the well-known community NO_EXPORT_SUBCONFED as defined in RFC 1997. Routes tagged with this community are not to be advertised to external peers, even if they are within the same confederation.

Description

comm specifies the community attributes added to routes sent to routers in a peer group. Communities may also be manipulated on import and export of routes from peers.

See “BGP Communities” on page 77 in *Configuring GateD* for more information.

Default

none

Context

mpbgp group type external statement

mpbgp group type internal statement

mpbgp group type routing statement

Examples

Example 1

Add the no-export community to a group.

[**autonomoussystem** and **routerid** omitted]

```
mpbgp on {
    group type external peeras 65534
        comm { community no-export ; } {
        peer 192.0.2.1;
        } ;
} ;
```

[**import** and **export** statements omitted]

Example 2

Add the community 64512:100 to a group.

[**autonomoussystem** and **routerid** omitted]

```
mpbgp on {
    group type external peeras 65534
        comm { comm-split 64512 100 ; } {
        peer 192.0.2.1;
        } ;
} ;
```

[**import** and **export** statements omitted]

Example 3

Add the community 64512:100 and community no-export-subconfed to a group.

[**autonomoussystem** and **routerid** omitted]

```
mpbgp on {
    group type external peeras 65534
        comm { comm-split 64512 100 ; community no-export-subconfed ; } {
        peer 192.0.2.1;
    } ;
} ;
```

[**import** and **export** statements omitted]

See Also

"BGP Communities" on page 77 in *Configuring GateD*

import on page 605

export on page 623

confed

Name

confed - marks a BGP group as being part of a BGP confederation

Syntax

confed

Parameters

none

Description

confed configures a BGP group to be part of a BGP confederation. The **confed-id** keyword must have been previously specified. When the **confed** keyword is present on a BGP group statement, GateD will treat all members of that group as confederation peers. Additionally, BGP will use the configured **autonomoussystem** number in its peering session for its AS number instead of **confed-id**.

When sending UPDATEs to confederation peers, the AS_PATH is modified using AS_CONFED_SET and AS_CONFED_SEQUENCEs instead of the normal CONFED_SET and CONFED_SEQUENCE. Additionally, the restriction against propagating MED and LOCAL_PREF values is relaxed and these values are propagated to confederation external peers.

Default

off

Context

bgp group statement

Examples

The following gated.conf shows a confederation border router. It has two peers outside of the confederation, one inside the confederation, and some confederation internal peers.

```
autonomoussystem 64512;
confed-id 100;
mpbgp on {
    group type routing peeras 64512 confed proto ospf {
        peer 192.168.1.1 ;
        peer 192.168.1.4 ;
    } ;
    group type external peeras 65000 confed {
        peer 10.132.10.1 ;
    } ;
    group type external peeras 200 {
        peer 172.16.50.1 ;
   } ;
} ;
```

```
# Import everything from our internal confederation peers
    import proto mpbgp as 64512 {
        all ;
   } ;

# Import everything from our external confederation peer
    import proto bgp as 65000 {
        all ;
    } ;

# Import everything from our external non-confederation peer
    import proto bgp as 200 {
        all ;
    } ;

# Redistribute everything from our external non-confederation and our
   # external confederation peer to our internal peers. Note that we are
   # not operating as a route reflector, so we do not redistribute routes
   # from our internal peers to our other internal peers.
export proto bgp as 64512 {
    proto bgp as 200 {
        all ;
    } ;
    proto bgp as 65000 {
        all ;
    } ;
} ;

# Redistribute our routes from our external confederation peer to
   # our internal confederation peers and our external
   # non-confederation peer.
export proto bgp as 65000 {
    proto bgp as 200 {
        all ;
    } ;
    proto bgp as 64512 {
        all ;
    } ;
} ;

# We want to receive traffic for this AS on our external links, so
   # propogate everything from our confederation.
export bgp as 200 {
    proto bgp as 64512 {
        all ;
    } ;
    proto bgp as 65000 {
        all ;
    } ;
} ;
```

See also

RFC 3065 at http://www.ietf.org/rfc/rfc3065.txt

defaultmetric

Name

defaultmetric - defines the metric (MED) used when advertising routes via MPBGP

Syntax

defaultmetric *metric* **;**

Parameters

metric

Description

defaultmetric defines the metric (MED) used when advertising routes via MPBGP. If not specified, no metric is propagated. This metric may be overridden by a metric specified on the **peer** or **group** statements or in export policy.

Default

none

Context

mpbgp statement

Examples

[**autonomoussystem** and **routerid** omitted]

```
mpbgp on {
    defaultmetric 100 ;
    group type external peeras 64512 {
# Metric of 100 is sent to this peer by default.
        peer 192.0.2.2 ;
# Metric of 150 is sent to this peer.
        peer 192.0.2.3 metricout 150 ;
    } ;
} ;
```

[**import** and **export** statements omitted]

See Also

metricout on page 480

discard-nonprefixed-confederations

Name

discard-nonprefixed-confederations - discards malformed AS_PATHs containing non-prefixed confederation segments

Syntax

discard-nonprefixed-confederations

Parameters

none

Description

discard-nonprefixed-confederations causes MPBGP to discard AS_PATHs that are received from BGP peers where there are BGP Confederation AS_PATH segments (AS_CONFED_SEQUENCE, AS_CONFED_SET) occurring anywhere other than at the left hand side of the AS_PATH. This feature is useful because of “buggy” routers on the Internet that will illegally advertise AS_PATHs containing confederation segments outside of a confederation boundary. If these routes propagate beyond the confederation boundary edge, they will cause the peering session of any router that does not accept confederation segments from non-confederation peers to drop the peering session and thus disrupt service.

This option will cause routes containing non-prefixed confederation segments to be logged and discarded.

Default

not enabled

Context

mpbgp statement

Examples

```
routerid 192.0.0.1;
autonomoussystem 64512;

mpbgp on {
    discard-nonprefixed-confederations;

    group type external peeras 65534 {
        allow {
            all;
        };
    };
```

```
};
```

[**static**, **import**, and **export** clauses omitted]

See Also:

“Chapter 22 Multi-Protocol - Border Gateway Protocol (MPBGP)” on page 111 of *Configuring GateD*

gateway

Name

gateway - instructs GateD to use a form of multihop External Border Gateway Protocol (EBGP)

Syntax

gateway *gateway_ip_address* **;**

Parameters

gateway_ip_address - A gateway is an intermediate destination by which packets are delivered to their ultimate destination. A gateway is the IP address of the gateway host.

Description

gateway instructs GateD to use a form of multihop EBGP. If a network is not shared with this group, **gateway** specifies a router on an attached network to be used as the next hop router for routes received from this peer. The **gateway** parameter may also be used to specify a next hop for groups that are on shared networks. For example, you might use **gateway** to ensure that third-party next hops are never accepted from a given group by specifying that group's address as its own gateway. The **gateway** specified must have consistent routing information to prevent routing loops. **gateway** is not needed in most cases.

Default

none

Context

mpbgp group type external statement

mpbgp group type internal statement

mpbgp group type routing statement

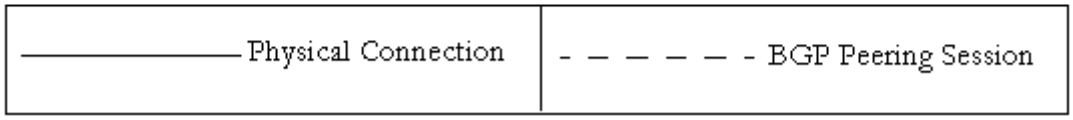
mpbgp peer statement

Examples

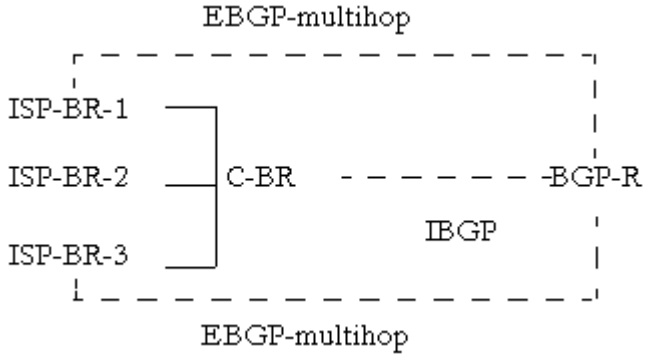
Consider the following example:

THE ROUTE SIEVE

Key:



A company has three Internet feeds. The customer border router has direct connections to the ISP border routers. However, the company's router does not have enough resources to hold three full views of the Internet. The company uses a GateD workstation running EBGP multihop to establish peering sessions with the ISP border routers and uses IBGP to send the selected routes to the EBGP-multihop company border router.



.

Router	AS	Router ID/Interface
ISP-BR-1	65501	192.168.1.1 -- DS3 to C-BR
ISP-BR-2	65502	10.0.0.1 -- DS3 to C-BR
ISP-BR-3	65503	172.16.0.1 -- DS3 to C-BR
C-BR	64512	192.0.2.1 -- Gig Ethernet to BGP-R
BGP-R	64512	192.0.2.2 -- Gig Ethernet to C-BR

Example 1

```
autonomoussystem 64512 ;
routerid 192.0.2.1 ;
mpbgp on {
    group type external peeras 65501 gateway 192.168.1.1 {
        peer 192.168.1.1;
    } ;
    group type external peeras 65502 gateway 10.0.0.1 {
        peer 10.0.0.1 ;
    } ;
    group type external peeras 65503 gateway 172.16.0.1 {
        peer 172.16.0.1 ;
    } ;
```

```
        group type internal peeras 64512 {
            peer 192.0.2.1
        } ;
    } ;
```

[**import** and **export** statements omitted]

Example 2

```
    autonomoussystem 64512 ;
    routerid 192.0.2.1 ;
    mpbgp on {
        group type external peeras 65501 {
            peer 192.168.1.1 gateway 192.168.1.1 ;
        } ;
        group type external peeras 65502 {
            peer 10.0.0.1 gateway 10.0.0.1 ;
        } ;
        group type external peeras 65503 {
            peer 172.16.0.1 gateway 172.16.0.1 ;
        } ;
        group type internal peeras 64512 {
            peer 192.0.2.1
        } ;
    } ;
```

[**import** and **export** statements omitted]

See Also

“Third Party Routing” on page 70 in *Configuring GateD*

group type

Name

group type - specifies an MPBGP group as internal, external or routing

Syntax

group type external peeras *autonomous_system*

group type internal peeras *autonomous_system*

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

Parameters

autonomous_system - a number between 1 and 65535 inclusive, specifying a set of routers under a single technical administration and assigned by the Internet Assigned Numbers Authority

protocol - name of the protocol to be used to resolve MPBGP route next hops, including **static**, **rip**, **ospf**, and **isis**

Groups also have the following parameters, which are described throughout the BGP section of this manual:

ascount *count*
caps *cap_list*
comm *community_list*
confed
gateway *host*
holdtime *time*
ignorefirstashop
keep (all | none)
keepalivesalways
localtcp *local_address*
localas *autonomous_system*
localv4addr *ipv4_address*
localv6addr *ipv6_address*
logupdown
med
metricout *metric*
nexthopself
noaggregatorid
nogendefault
nov4asloop
outdelay *time*
passive
preference *grouppreference*
preference2 *grouppreference2*
recvbuffer *buffer_size*
reflector-client [no-client-reflect]
remotev4addr *ipv4_address*
remotev6addr *ipv6_address*
routetopeer
sendbuffer *buffer_size*

setpref *metric*
showwarnings
traceoptions *trace_options*
ttl *ttl*

Description

Within a group, MPBGP peers may be configured in one of two ways. They may be implicitly configured with the **allow** statement or explicitly configured with a **peer** statement.

In **group type external**, full policy checking is applied to all incoming and outgoing advertisements. The external peers must be directly reachable through one of the machine's local interfaces. If they are not, BGP may be "multi-hopped" through the use of the **gateway** statement. The next hop transmitted is computed with respect to the shared interface.

The **group type internal** specifies an internal group operating where there is no IP-level IGP, for example, an SMDS network or MILNET. If the peer is not directly connected, the **route-to-peer** and **ttl** statements may be used to "multihop" the IBGP session. Import and export policy may be applied to group advertisements. Routes received from external MPBGP peers are by default readvertised with the received metric.

The **group type routing** propagates external routes between routers that are not directly connected. **group type routing** also computes immediate next hops for those external routes by using the MPBGP next hop that arrived with the route as a forwarding address to be resolved via an internal protocol's routing information. In essence, internal MPBGP is used to carry AS external routes, and the IGP is expected to carry only AS internal routes. The latter is used to find immediate next hops for the former. **proto** *protocol* names the interior protocol to be used to resolve MPBGP route next hops, and may be the name of any IGP in the configuration, including static. By default, the next hop in MPBGP routes advertised to **group type routing** peers will be set to the local address on the BGP connection to those peers because it is assumed that a route to this address will be propagated via the IGP. The **interface list** can optionally provide a list of interfaces whose routes are carried via the IGP for which third-party next hops may be used instead.

localtcp, **outdelay**, and **metricout** must be set in the **group** clause, not on a per-peer basis, for the group types **internal** and **routing**. If these options are set on the **peer** subclause, they must equal the values set on the corresponding **group** clause.

Default

none

Context

mpbgp statement

Examples

```
autonomoussystem 65531
mpbgp on {
    group type external peeras 65534 {
        peer 192.0.2.2 ;
```

```
        } ;
        group type internal peeras 65531 {
            peer 192.0.2.3 ;
        } ;
        group type routing peeras 65531 proto ospf {
            peer 192.0.2.4 ;
        } ;
    } ;
```

See Also

mpbgp on page 482

holdtime

Name

holdtime - specifies the MPBGP hold time value, in seconds, to use when negotiating a peering session

Syntax

holdtime *time*

Parameters

time - time in seconds, an integer that is at least 3

Description

holdtime specifies the MPBGP hold time value, in seconds, to use when negotiating a peering session with this group. If GateD does not receive a keepalive, update, or notification message within the period specified in the hold time field of the MPBGP Open message, then the MPBGP connection will be closed. The value must be at least 3.

The negotiated holdtime value is the lesser of the values sent in the exchanged MPBGP Open messages.

Default

holdtime 180 ;

Context

mpbgp group type external statement

mpbgp group type internal statement

mpbgp group type routing statement

mpbgp peer statement

Examples

Example 1

holdtime set in the **group** statement

[**autonomoussystem** and **routerid** omitted]

```
mpbgp on {
    group type external peeras 64512 holdtime 30 {
        peer 192.0.2.1 ; # Holdtime is 30 seconds
        peer 192.0.2.2 ; # Holdtime is 30 seconds
    } ;
} ;
```

[**import** and **export** statements omitted]

Example 2

holdtime set in the **group** and **peer** statements

[**autonomoussystem** and **routerid** omitted]

```
mpbgp on {
    group type external peeras 64512 holdtime 90 {
        peer 192.0.2.1 ; # Holdtime is 90 seconds
        peer 192.0.2.2 holdtime 30 ; # Holdtime is 30 seconds
    } ;
} ;
```

[**import** and **export** statements omitted]

See Also

keepalivesalways on page 467

ignorefirstashop

Name

ignorefirstashop - directs GateD to keep route servers' routes

Syntax

```
ignorefirstashop ;
```

Parameters

none

Description

Some routers, known as "route servers," are capable of propagating routes without appending their own autonomous system (AS) number to the AS Path. By default, GateD will drop such routes. Specifying **ignorefirstashop** on the **group** clause allows GateD to keep these routes. **ignorefirstashop** should be used only if there is no doubt that these peers are route servers and not normal routers.

Default

disabled

Context

mpbgp group type external statement

mpbgp peer statement for external groups

Examples

Example 1

ignorefirstashop set in **group** statement

[**autonomoussystem** and **routerid** omitted]

```
mpbgp on {
    group type external peeras 64512 ignorefirstashop {
        peer 192.0.2.1 ;
        peer 192.0.2.2 ;
    } ;
} ;
```

[**import** and **export** statements omitted]

Example 2

ignorefirstashop set in **peer** statement

[**autonomoussystem** and **routerid** omitted]

```
mpbgp on {
    group type external peeras 64512 {
        peer 192.0.2.1 ignorefirstashop ;
        peer 192.0.2.2 ;
    } ;
} ;
```

[**import** and **export** statements omitted]

ignore-nonprefixed-confederations

Name

ignore-nonprefixed-confederations - ignores malformed AS_PATHs containing non-prefixed confederation segments

Syntax

ignore-nonprefixed-confederations ;

Parameters

none

Description

ignore-nonprefixed-confederations causes BGP to ignore AS_PATHs that are received from BGP peers where there are BGP Confederation AS_PATH segments (AS_CONFED_SEQUENCE, AS_CONFED_SET) occurring anywhere other than at the left-hand side of the AS_PATH. This feature is useful due to “buggy” routers on the Internet that will illegally advertise AS_PATHs containing confederation segments outside of a confederation boundary. If these routes propagate beyond the confederation boundary edge, they will cause the peering session of any router that does not accept confederation segments from non-confederation peers to drop the peering session and thus disrupt service. This option will cause routes containing non-prefixed confederation segments to be logged and stored in the RIB with a preference of -1. The GII command "**show mpbgp bad-as-path**" will then show all of these routes.

Default

not enabled

Context

mpbgp statement

Examples

```
routerid 192.0.0.1;
autonomoussystem 64512;

bgp on {
    ignore-nonprefixed-confederations;

    group type external peeras 65534 {
        allow {
            all;
        };
```

```
    };
 };
```

[**static**, **import**, and **export** clauses omitted]

See Also

“Chapter 22 Multi-Protocol - Border Gateway Protocol (MPBGP)” on page 111 of *Configuring GateD*

discard-nonprefixed-confederations on page 450

RFC 3065 - Autonomous System Confederations for BGP

interface

See **interface** on page 28

keep

Name

keep - specifies whether to keep routes containing a router's own autonomous system (AS) number

Syntax

```
keep ( all | none );
```

Parameters

all or **none**

Description

keep all retains routes learned from a peer even if the routes' AS paths contain the router's own AS number.

keep none causes GateD to disregard routes containing the router's own AS number.

Default

```
keep none ;
```

Context

mpbgp group type external statement

mpbgp group type internal statement

mpbgp group type routing statement

mpbgp peer statement

Examples

Example 1

keep all set in **group** statement

[**autonomoussystem** and **routerid** omitted]

```
mpbgp on {
    group type external peeras 64512 keep all {
        peer 192.0.2.1 ; # keeps all
        peer 192.0.2.2 ; # keeps all
    } ;
} ;
```

[**import** and **export** statements omitted]

Example 2

keep all set in **peer** statement

[**autonomoussystem** and **routerid** omitted]

```
mpbgp on {
    group type external peeras 64512 {
        peer 192.0.2.1 keep all ;
        peer 192.0.2.2 ; # keeps none
    } ;
} ;
```

[**import** and **export** statements omitted]

keepalivesalways

Name

keepalivesalways - causes GateD to always send keepalives

Syntax

```
keepalivesalways ;
```

Parameters

none

Description

keepalivesalways causes GateD to always send keepalives, even when an **update** could have correctly substituted for one. **keepalivesalways** allows interoperability with routers that do not completely obey the protocol specifications on this point.

Default

disabled

Context

mpbgp group type external statement

mpbgp group type internal statement

mpbgp group type routing statement

mpbgp peer statement

Examples

Example 1

keepalivesalways set in **group** statement

[**autonomoussystem** and **routerid** omitted]

```
mpbgp on {
    group type external peeras 64512 keepalivesalways {
        peer 192.0.2.1 ; # keepalivesalways
        peer 192.0.2.2 ; # keepalivesalways
    } ;
} ;
```

[**import** and **export** statements omitted]

Example 2

keepalivesalways set in **peer** statement

[**autonomoussystem** and **routerid** omitted]

```
mpbgp on {
    group type external peeras 64512 {
        peer 192.0.2.1 keepalivesalways ;
        peer 192.0.2.2 ; # Normal behavior
    } ;
} ;
```

[**import** and **export** statements omitted]

localas

Name

localas - identifies the autonomous system that GateD is representing to this group of peers

Syntax

localas *autonomous_system* **;**

Parameters

autonomous_system - a number between 1 and 65535 inclusive, specifying a set of routers under a single technical administration and assigned by the Internet Assigned Numbers Authority

Description

localas identifies the autonomous system that GateD is representing to this group of peers. The default is that which has been set globally in the **autonomoussystem** statement.

Default

inherited from the global **autonomoussystem**

Context

mpbgp group type external statement

Examples

[**routerid** omitted]

```
autonomoussystem 64512
mpbgp on {
    group type external peeras 65000 { # use as 64512 (inherited)
        peer 192.0.2.1 ; # keepalivealways
    } ;
    group type external peeras 65001 localas 65534 {
       peer 192.0.2.2 ; # keepalivealways
    } ;
} ;
```

[**import** and **export** statements omitted]

localtcp

Name

localtcp - specifies the address to be used on the local end of the TCP connection with the group

Syntax

localtcp *local_address* **;**

Parameters

local_address - the host address of an attached interface. This is the address of a broadcast, NBMA or loopback interface and the local address of a point-to-point interface. As with any host address, it may be specified symbolically or in an IPv4 dotted-quad format (a.b.c.d) or IPv6 format (a : b : c : d :: e).

Description

localtcp specifies the IP 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 or with the peer's gateway when **gateway** is used. A session with an external peer will be opened only when an interface with the appropriate local address (through which the peer or gateway address is directly reachable) is operating. For **internal** and **routing** peers, a peer session will be maintained when any interface with the specified local address is operating. In any case, an incoming connection will be recognized as a match for a configured peer only if it is addressed to the configured local address.

For group types **internal** and **routing**, set this **localtcp** on the **group** clause.

Default

the IP address of a shared interface

Context

mpbgp group type external statement

mpbgp group type internal statement

mpbgp group type routing statement

mpbgp peer statement

Examples

Example 1

[**autonomoussystem** and **routerid** omitted]

```
mpbgp on {
    group type external peeras 64512 localtcp 192.168.1.1 {
        peer 192.0.2.1 ;
    } ;
} ;
```

[**import** and **export** statements omitted]

Example 2

```
mpbgp on {
    group type external peeras 64513 caps {v6u} localv4addr 192.0.2.2
       localtcp fec0:0:0:ff::d {
        peer fec0:0:0:ff::e ;
    } ;
} ;
```

localv4addr

Name

localv4addr - specifies an IPv4 address to send as a next hop when advertising IPv4 NLRI to an IPv6 peer.

Syntax

localv4addr - *ipv4 address* ;

Parameters

ipv4 address

Description

BGP peering sessions can be established over IPv4 or IPv6, but NLRI of both address families may be advertised to a given peer. When peering over IPv6, the configured address of the peer is used to determine acceptable next hops to advertise to that peer. However, GateD has no knowledge of the peer's IPv4 configuration. **localv4addr** specifies an IPv4 address for GateD to advertise as a next hop for IPv4 NLRI advertised to IPv6 peers. **localv4addr** must be configured for all IPv6 peers unless an appropriate address can be determined from the configuration of **remotev4addr**.

Default

none (A parse error will result if not configured and it is required.)

Context

mpbgp group type external statement

mpbgp group type internal statement

mpbgp group type routing statement

mpbgp peer statement

Examples

Example 1

```
mpbgp on {
        group type external peeras 64512 caps {v4u v6u} localv4addr
            192.0.2.1 {
                peer fec0:0:0:ff::d;
        };
};
```

Example 2

```
mpbgp on {
```

```
            group type external peeras 64512 caps {v4u v6u} localv4addr
                192.0.2.1 {
                    peer fec0:0:0:ff::d;
                    # override group definition of localv4addr
                    peer fec0:0:0:ff::e localv4addr 192.0.2.10;
            };
    };
```

See Also

localv6addr on page 474

remotev4addr on page 500

remotev6addr on page 502

localv6addr

Name

localv6addr - specifies an IPv6 address to send as a next hop when advertising IPv6 NLRI to an IPv4 peer

Syntax

localv6addr - *ipv6_address* ;

Parameters

ipv6_address

Description

BGP peering sessions can be established over IPv4 or IPv6, but NLRI of both address families may be advertised to a given peer. When peering over IPv4, the configured address of the peer is used to determine acceptable next hops to advertise to that peer. However, GateD has no knowledge of the peer's IPv6 configuration. **localv6addr** specifies an IPv6 address for GateD to advertise as a next hop for IPv6 NLRI advertised to IPv4 peers. **localv6addr** must be configured for all IPv6 peers unless an appropriate address can be determined from the configuration of **remotev6addr**.

Default

none (A parse error will result if not configured and it is required.)

Context

mpbgp group type external statement

mpbgp group type internal statement

mpbgp group type routing statement

mpbgp peer statement

Examples

Example 1

```
mpbgp on {
        group type external peeras 64512 caps {v4u v6u} localv6addr
            fec0:0:0:ff::d {
                peer 192.0.2.1;
        };
};
```

Example 2

```
mpbgp on {
```

```
            group type external peeras 64512 caps {v4u v6u} localv6addr
                fec0:0:0:ff::d {
                    peer 192.0.2.1;
                    # override group definition of localv6addr
                    peer 192.0.2.2 localv6addr fec0:0:0:ff::e;
            };
    };
```

See Also

localv4addr on page 472

remotev4addr on page 500

remotev6addr on page 502

logupdown

Name

logupdown - causes a message to be logged via the syslog mechanism whenever a BGP group enters or leaves the Established state

Syntax

```
logupdown ;
```

Parameters

none

Description

logupdown causes a message to be logged via the syslog mechanism whenever a BGP group enters or leaves the Established state.

Default

disabled

Context

mpbgp group type external statement

mpbgp group type internal statement

mpbgp group type routing statement

mpbgp peer statement

Examples

Example 1

Enable **logupdown** in the group.

[**autonomoussystem** and **routerid** omitted]

```
mpbgp on {
    group type external peeras 64512 logupdown {
        peer 192.0.2.1 ; # logupdown enabled
        peer 192.0.2.2 ; # logupdown enabled
    } ;
} ;
```

[**import** and **export** statements omitted]

Example 2

Enable **logupdown** in the peer.

[**autonomoussystem** and **routerid** omitted]

```
mpbgp on {
    group type external peeras 64512 {
        peer 192.0.2.1 logupdown ; # logupdown enabled
        peer 192.0.2.2 ;           # logupdown disabled
    } ;
} ;
```

[**import** and **export** statements omitted]

See Also

traceoptions on page 508

med

Name

med - allows MEDs to be used in routing computations

Syntax

```
med ;
```

Parameters

none

Description

By default, any MED (*Multi_Exit_Disc*) received on a BGP connection is ignored. If MEDs are to be used in routing computations, the **med** option must be specified on the **group** or **peer** clauses. By default, MEDs are not sent on external connections. To send MEDs, use the **metric** option of the **export** statement or the **metricout** peer/group parameter.

When two routes to the same destination are received from different peers within the same **peer-as**, they could have different MEDs. When choosing between these routes, assuming that nothing else makes one preferable to the other (such as configured policy), the values of the differing MEDs are used to choose which route to use. In this comparison, the route with the lowest MED is preferred. Routes without MEDs are treated as having the best possible MED. MEDs are not propagated to internal peers unless **med** is enabled.

Default

disabled

Context

mpbgp group type external statement

mpbgp group type internal statement

mpbgp group type routing statement

mpbgp peer statement

Examples

Example 1

Enable **med** processing on the **group** statement.

[**autonomoussystem** and **routerid** omitted]

```
mpbgp on {
    group type external peeras 64512 med {
        peer 192.0.2.1 ; # med comparison is enabled
        peer 192.0.2.2 ; # med comparison is enabled
```

```
    } ;
} ;
```

[**import** and **export** statements omitted]

Example 2

Enable **med** processing on the **peer** statement.

[**autonomoussystem** and **routerid** omitted]

```
mpbgp on {
    group type external peeras 64512 {
        peer 192.0.2.1 med ; # med comparison is enabled
        peer 192.0.2.2 ;     # med comparison is disabled
    } ;
} ;
```

[**import** and **export** statements omitted]

See Also

“Preferences and Route Selection” on page 11 in *Configuring GateD*

metricout

Name

metricout - causes BGP to send a Multi-Exit Discriminator (MED) when routes are advertised to peers

Syntax

metricout *metric* **;**

Parameters

metric

Description

The **metricout** statement causes a BGP MED to be set on routes when they are advertised to peers. The metric hierarchy is as follows, starting from the most preferred:

1. the metric specified by export policy
2. peer-level **metricout**
3. group-level **metricout**
4. **defaultmetric**

For group types **internal** and **routing**, set **metricout** on the **group** clause instead of on the **peer** subclause (MED needs to be common among all peers in an internal group or looping may occur).

Default

no metric, or, if set, **defaultmetric**

Context

mpbgp group type external statement

mpbgp group type internal statement

mpbgp group type routing statement

mpbgp peer statement

Examples

Example 1

Set a **metricout** of 50 on the **group**.

[**autonomoussystem** and **routerid** omitted]

```
mpbgp on {
    group type external peeras 64512 metricout 50 {
        peer 192.0.2.1 ; # Send MED of 50
        peer 192.0.2.2 ; # Send MED of 50
```

```
        peer 192.0.2.3 ; # Send MED of 50
    } ;
} ;
```

[**import** and **export** statements omitted]

Example 2

Set a **metricout** of 50 on a specific **peer**.

[**autonomoussystem** and **routerid** omitted]

```
mpbgp on {
    group type external peeras 64512 {
        peer 192.0.2.1 metricout 50 ; # Send MED of 50
        peer 192.0.2.2 ;              # Send no MED
        peer 192.0.2.3 ;              # Send no MED
    } ;
} ;
```

[**import** and **export** statements omitted]

Example 3:

Set a **metricout** of 50 on a specific **peer**.

[**autonomoussystem** and **routerid** omitted]

```
defaultmetric 100;
mpbgp on {
    group type external peeras 64512 {
        peer 192.0.2.1 metricout 50 ; # Send MED of 50
        peer 192.0.2.2 ;              # Send MED of 100
        peer 192.0.2.3 ;              # Send MED of 100
    } ;
} ;
```

[**import** and **export** statements omitted]

See Also

defaultmetric on page 449

“Preferences and Route Selection” on page 11 in *Configuring GateD*

mpbgp

Name

mpbgp - enables or disables MPBGP

Syntax

mpbgp [on | off] {*mpbgp_parameters***}**

Parameters

mpbgp_parameters - includes all the parameters in this section

Description

mpbgp enables or disables MPBGP. *mpbgp_parameters* includes all the parameters in this section.

Default

mpbgp off ;

Context

global

Examples

```
mpbgp on {
    group type external peeras 64512 ignorefirstashop {
        peer 192.0.2.1 ;
        peer 192.0.2.2 ;
    } ;
} ;
```

See Also

“Multi-Protocol - Border Gateway Protocol” on page 111 of *Configuring GateD*

nexthopself

Name

nexthopself - sets this group or peer’s next hop to the router's own address on advertisement

Syntax

nexthopself

Parameters

none

Description

nexthopself sets this group's next hop to the router's own address even if it would normally be possible to send a third-party next hop.

nexthopself can cause inefficient routes to be followed. It might be needed in some cases to deal with improperly bridged interconnect media (in cases where the routers on the “shared” medium do not really have full connectivity to each other) or when political situations cause broken links.

nexthopself can be used only for external groups.

Default

disabled

Context

mpbgp group type external statement

mpbgp peer statement (in external groups)

Examples

Consider the following topology:

```
R1----R2
        |
       R3
```

Three routers that have interfaces all in the same subnet are in a partially meshed ATM network.

- R1 and R2 have a Permanent Virtual Circuit (PVC.)
- R2 and R3 have a PVC.
- R1 and R3 do not have any direct connectivity.
- R1, R2, and R3 share a common subnet.

Because all routers are in the same subnet, normally, routes exchanged between R1, R2, and R3 would all use third-party routing. This would result in routes exchanged between R1 and R3 to result in a blackhole because R1 and R3 do not have a valid physical connection.

When R2 exchanges routes with R1 and R3, it should use **nexthopself** to disable the third-party routing.

Configuration for R2:

```
mpbgp on {
    group type external peeras 64512 nexthopself {
        peer 192.0.2.1 ; # R1
    } ;
    group type external peeras 64513 nexthopself {
        peer 192.0.2.2 ; # R2
    } ;
} ;
```

[**import** and **export** statements omitted]

This could also be written as:

```
mpbgp on {
    group type external peeras 64512 {
        peer 192.0.2.1 nexthopself ; # R1
    } ;
    group type external peeras 64513 {
        peer 192.0.2.2 nexthopself ; # R2
    } ;
} ;
```

See Also

“Third Party Route Advertisement” on page 70 in *Configuring GateD*

noaggregatorid

Name

noaggregatorid - causes GateD to specify the **routerid** in the aggregator attribute as 0

Syntax

```
noaggregatorid ;
```

Parameters

none

Description

noaggregatorid causes GateD to specify the **routerid** in the aggregator attribute as 0 (instead of the **routerid** of the router) in order to prevent different routers in an autonomous system (AS) from creating aggregate routes with different AS Paths.

Default

disabled

Context

mpbgp group type external statement
mpbgp group type internal statement
mpbgp group type routing statement
mpbgp peer statement

Examples

[**autonomoussystem** and **routerid** omitted]

```
mpbgp on {
    group type external peeras 64512 noaggregatorid {
        peer 192.0.2.1 ;
    } ;
# Any aggregates will have a routerid of 0.0.0.0.
    aggregate 10.0.0.0 masklen 9 {
        proto bgp {
        10.0.2.0 masklen 24 ;
        10.0.4.0 masklen 24 ;
        } ;
    } ;
} ;
```

[**import** and **export** statements omitted]

See Also

"Route Aggregation" on page 155 in *Configuring GateD*

nogendefault

Name

nogendefault - prevents GateD from generating a default route when BGP receives a valid update from its peer

Syntax

```
nogendefault ;
```

Parameters

none

Description

nogendefault prevents GateD from generating a default route when MPBGP receives a valid update from its peer. The default route is generated only when the **gendefault** option is enabled.

Default

disabled

Context

mpbgp group type external statement

mpbgp group type internal statement

mpbgp group type routing statement

mpbgp peer statement

Examples

[**autonomoussystem** and **routerid** omitted]

```
mpbgp on {
    group type external peeras 64512 nogendefault {
        peer 192.0.2.1 ;
    } ;
} ;
```

[**import** and **export** statements omitted]

See Also

"Options Statements" on page 21 in *Configuring GateD*

nov4asloop

Name

nov4asloop - prevents routes with looped autonomous system (AS) Paths from being advertised to version 4 external peers

Note: **nov4asloop** is deprecated.

Syntax

```
nov4asloop ;
```

Parameters

none

Description

nov4asloop prevents routes with looped AS paths from being advertised to version 4 external peers. Use **nov4asloop** to avoid advertising routes to peers that would incorrectly forward the routes on to version 3 peers.

In this context, "looped" refers to "AS Path stuffing" where a given AS is inserted multiple times in an AS Path.

Default

disabled

Context

mpbgp group type external statement

mpbgp group type internal statement

mpbgp group type routing statement

mpbgp peer statement

Examples

[**autonomoussystem** and routerid omitted]

```
mpbgp on {
    group type external peeras 64512 nov4asloop {
        peer 192.0.2.1 ;
    } ;
} ;
```

[**import** and **export** statements omitted]

See Also

"BGP Communities" on page 133 of *Configuring GateD*

open-on-accept

Name

open-on-accept - causes GateD to send a BGP Open message immediately after the transport (TCP) connection has completed

Syntax

```
open-on-accept ;
```

Parameters

None

Description

When GateD receives an incoming BGP peering session, it will delay sending the BGP Open message for a short time after the transport connection (TCP) has completed. This a violation of the BGP Finite State Machine, but is used to allow the calling peer to send its Open message first. This is used to determine if a peering session matches a given **allow** clause. When the **open-on-accept** keyword is present, GateD will immediately send the Open message when the TCP connection has completed for configured peers. If the peer is unconfigured (is matched by an **allow** clause, but not a **peer** keyword), or is **passive**, GateD will continue to wait for the Open message from the remote peer before sending its own BGP Open message.

Default

Off

Context

mpbgp statement

Examples

```
autonomoussystem 64512;
mpbgp yes {
    open-on-accept ;
    group type external peeras 65534  {
        peer 192.168.1.1;
    };
};
```

See Also

"Chapter 22 Multi-Protocol - Border Gateway Protocol (MPBGP)" on page 111 of *Configuring GateD*

outdelay

Name

outdelay - damps route fluctuations

Syntax

outdelay *time* **;**

Parameters

time - time in seconds

Description

outdelay damps route fluctuations. The **outdelay** *time* is the amount of time a route must be present in the GateD routing database before it is exported to MPBGP. **outdelay** is intended only for use with external peers.

Note: Weighted Route Damping is better suited for improving overall network stability. The use of this option may delay route convergence for well-behaved routers.

Default

outdelay 0 ; (This feature is disabled.)

Context

mpbgp group type external statement

mpbgp peer statement for external groups

Examples

[**autonomoussystem** and **routerid** omitted]

```
mpbgp on {
    group type external peeras 64512 outdelay 10 {
         peer 192.0.2.1 ;
    };
    group type external peeras 64513 {
         peer 192.0.2.2 outdelay 15 ;
         peer 192.0.2.3 ; # no outdelay
    };
} ;
```

[**import** and **export** statements omitted]

See Also

"Route Flap Damping" on page 163 in *Configuring GateD*

passive

Name

passive - prevents GateD from ever trying to open a BGP connection with peers in this group

Syntax

```
passive ;
```

Parameters

none

Description

passive prevents GateD from ever trying to open a BGP connection with peers in this group. Instead, the router will wait for the peer to initiate a connection. **passive** was introduced to handle a problem in BGP3 and earlier in which two peers might both attempt to initiate a connection at the same time. This problem is fixed in the BGP4 protocol, so the **passive** option is not needed with BGP4 sessions.

Note: If it is applied to both sides of a peering session, **passive** will prevent the session from ever being established. For this reason, and because it is generally not needed, the use of **passive** is discouraged.

Default

disabled

Context

mpbgp group type external statement

mpbgp group type internal statement

mpbgp group type routing statement

mpbgp peer statement

Examples

[**autonomoussystem** and **routerid** omitted]

```
mpbgp on {
    group type external peeras 64512 passive {
        peer 192.0.2.1 ;
    } ;
} ;
```

[**import** and **export** statements omitted]

peer

Name

peer - configures an individual peer

Syntax

peer *host* **[** *peer_options* **] ;**

Parameters

host - the IPv6 or IPv4 address of the host machine

Peer options are described throughout the BGP section and include:

ascount *count*
gateway *gateway*
holdtime *time*
ignorefirstashop
keep (all | none)
keepalivesalways
localas *autonomous_system*
localtcp *local_address*
localv4addr *ipv4_address*
localv6addr *ipv6_address*
logupdown
med
metricout *metric*
nexthopself
noaggregatorid
nogendefault
nov4asloop
outdelay *time*
passive
preference *peerpreference*
preference2 *peerpreference2*
recvbuffer *buffer_size*
remotev4addr *ipv4_address*
remotev6addr *ipv6_address*
routetopeer
sendbuffer *buffer_size*
showwarnings
traceoptions *trace_options*
ttl *ttl*

Description

peer configures an individual peer. Each peer inherits all parameters specified on a **group** clause as defaults. Many defaults may be overridden by parameters explicitly specified on the **peer** subclause. Within each **group** clause, individual peers can be specified, or a group of potential peers can be specified using **allow**. Use **allow** to specify a set of address

masks. If GateD receives an MPBGP connection request from any address in the set specified, it will accept it and set up a peer relationship.

Default

not applicable

Context

mpbgp statement

Examples

Example 1

[**autonomoussystem** and **routerid** omitted]

```
mpbgp on {
    group type external peeras 64512 {
        peer 192.0.2.1 ;
        peer 10.0.0.1 passive ;
        peer 192.168.1.1 nexthopself ;
    } ;
} ;
```

[**import** and **export** statements omitted]

Example 2

```
mpbgp on {
    group type external peeras 64513 caps {v6u} localv4addr 192.168.1.2 {
        peer } ffe:0:0:ff::e ;
    } ;
} ;
```

preference

Name

preference - specifies how active routes that are learned from MPBGP (compared to other protocols) will be selected

Syntax

preference *mpbgppreference*

Parameters

mpbgppreference - an assigned integer between 0 (directly connected) and 255 (for EGP)

Description

preference specifies how active routes that are learned from MPBGP (compared to other protocols) will be selected. When a route has been learned from more than one protocol, the active route will be selected from the protocol with the lowest preference. Each protocol has a default **preference** in this selection. This **preference** may be overridden by a **preference** specified on the **group** or **peer** statements or by import policy.

Default

```
preference 170 ;
```

Context

mpbgp statement

mpbgp group type external statement

mpbgp group type internal statement

mpbgp group type routing statement

mpbgp peer statement

Examples

Example 1

The following example sets the global MPBGP preference to 140, which makes MPBGP routes better than OSPF AS-external routes.

[**autonomoussystem** and **routerid** omitted]

```
mpbgp on {
preference 140 ;
    group type external peeras 64512 {
        peer 192.0.2.1 ;
    } ;
} ;
```

[**import** and **export** statements omitted]

Example 2

The following example sets the preference for a specific MPBGP group to be better than routes from another MPBGP group. This ensures that, for a given prefix, these routes are always preferred.

[**autonomoussystem** and **routerid** omitted]

```
mpbgp on {
# These routes are always preferred compared with
   # other mpbgp routes
    group type external peeras 64512 preference 165 {
        peer 192.0.2.1 ;
    } ;
    group type external peeras 65000 { # default preference of 170
        peer 192.0.2.2 ;
   } ;
} ;
```

[**import** and **export** statements omitted]

Example 3

Policy for all

[**autonomoussystem** and **routerid** omitted]

```
mpbgp on {
    group type external peeras 64512 {
        peer 192.0.2.1 ;
    } ;
} ;
    import proto bgp as 64512 {
    # We will prefer this route over almost anything other than
       # a directly attached interface.
        10.0.0.0 masklen 24 preference 5 ;
        all ; # default 170
    } ;
```

[**export** statement omitted]

See Also

“Preferences and Route Selection” on page 11 in *Configuring GateD*

preference2

Name

preference2 - breaks a **preference** tie between groups

Syntax

preference2 *grouppreference2* **;**

Parameters

grouppreference2 - an integer between 0 and 255

Description

preference2 breaks a **preference** tie between groups. Preferences are the first criteria of comparison for route selection.

Default

preference2 0 ;

Context

mpbgp group type external statement

mpbgp group type internal statement

mpbgp group type routing statement

mpbgp peer statement

Examples

[**autonomoussystem** and **routerid** omitted]

```
mpbgp on {
    # Prefer these routes.
    group type external peeras 64512 preference 5 {
         peer 192.0.2.1 ;
    } ;
    group type external peeras 65000 { # default preference 170
    # Routes from this peer are preferred over 192.0.2.3.
        peer 192.0.2.2 preference2 10 ;
        peer 192.0.2.3 preference2 20 ;
    } ;
} ;
```

[**import** and **export** statements omitted]

See Also

"Preferences and Route Selection" on page 11 in *Configuring GateD*

recvbuffer

Name

recvbuffer - controls the amount of memory requested from the kernel for the receive buffer

Syntax

recvbuffer *buffer_size* **;**

Parameters

buffer_size - an integer between 1 and 65535

Description

recvbuffer controls the amount of memory requested from the kernel for the receive buffer. The maximum supported is 65535 bytes, although many kernels have a lower limit. By default, GateD configures the maximum supported. **recvbuffer** is not needed on normally functioning systems.

Default

recvbuffer 65535 ;

Context

mpbgp group type external statement

mpbgp group type internal statement

mpbgp group type routing statement

mpbgp peer statement

Examples

[**autonomoussystem** and **routerid** omitted]

```
mpbgp on {
    group type external peeras 64512 recvbuffer 32768 {
        peer 192.0.2.1 ;
    } ;
} ;
```

[**import** and **export** statements omitted]

See Also

sendbuffer on page 505

reflector-client [no-client-reflect]

Name

reflector-client - specifies that GateD will act as a route reflector for this group

no-client-reflect - specifies that GateD will not act as an intra-group reflector and thus will not reflect routes back to peers within the same group. If used, this keyword must follow **reflector-client**.

Syntax

```
reflector-client [ no-client-reflect ] ;
```

Parameters

none

Description

reflector-client specifies that GateD will act as a route reflector for this group.

no-client-reflect specifies that GateD will not act as an intra-group reflector and thus will not reflect routes back to peers within the same group. This is used when client peers within a route-reflection group are fully meshed.

Default

none

Context

mpbgp group type (only internal or routing group types)

Examples

[**autonomoussystem** omitted]

```
routerid 192.0.2.1 ;
mpbgp on {
    clusterid 192.0.2.254 ;
    group type internal peeras 65534 {
        peer 192.0.2.2 ;
        peer 192.0.2.3 ;
    } ;
    # Routes received from clients will have the clusterid of
    # 192.0.2.254 added to their BGP attributes.
    group type internal peeras 65534 reflector-client {
        peer 192.0.2.10 ;
        peer 192.0.2.11 ;
        peer 192.0.2.12 ;
```

```
        peer 192.0.2.13 ;
    } ;
} ;
```

[**import** and **export** statements omitted]

See Also

"Route Reflection" on page 71 in *Configuring GateD*

remotev4addr

Name

remotev4addr - specifies a peer's IPv4 address to use as a next hop for routes learned from an IPv6 peer sending IPv4 NLRI to GateD. Also used to determine a value to use for **localv4addr**.

Syntax

remotev4addr - *ipv4_address* ;

Parameters

ipv4_address

Description

BGP peering sessions can be established over IPv4 or IPv6, but NLRI of both address families may be learned from a given peer. IBGP peers may advertise next hops on networks to which we are not directly connected. When the advertised routes have the same address family as the IBGP peering session, BGP can simply install the route with the next hop of the peer's address. In the case of an IPv6 IBGP peer advertising IPv4 NLRI, **remotev4addr** is used to specify a next hop for the routes.

remotev4addr can also be specified instead of **localv4addr**. If a local interface is configured on the same network as **remotev4addr**, this interface address will be used for **localv4addr** if it is unspecified.

Default

none

Context

mpbgp group type external statement

mpbgp group type internal statement

mpbgp group type routing statement

mpbgp peer statement

Examples

```
mpbgp on {
     group type internal peeras 64512 {v4u v6u} remotev4addr
         192.0.1.1 {
              peer fec0:0:0:ff::d;
     };
     group type external peeras 64513 {v4u v6u} remotev4addr
         192.0.1.2 {
              peer fec0:0:0:ff::e;
```

```
        };
    };
```

See Also

localv4addr on page 472

localv6addr on page 474

remotev6addr on page 502

remotev6addr

Name

remotev6addr - specifies a peer's IPv6 address to use as a next hop for routes learned from an IPv4 peer sending IPv6 NLRI to GateD. Also used to determine a value to use for **localv6addr**.

Syntax

remotev6addr - *ipv6_address* ;

Parameters

ipv6_address

Description

BGP peering sessions can be established over IPv4 or IPv6, but NLRI of both address families may be learned from a given peer. IBGP peers may advertise next hops on networks to which we are not directly connected. When the advertised routes have the same address family as the IBGP peering session, BGP can simply install the route with the next hop of the peer's address. In the case of an IPv4 IBGP peer advertising IPv6 NLRI, **remotev6addr** is used to specify a next hop for the routes.

remotev6addr can also be specified instead of **localv6addr**. If a local interface is configured on the same network as **remotev6addr**, this interface address will be used for **localv6addr** if it is unspecified.

Default

none

Context

mpbgp group type external statement

mpbgp group type internal statement

mpbgp group type routing statement

mpbgp peer statement

Examples

```
mpbgp on {
     group type internal peeras 64512 {v4u v6u} remotev6addr
         fec0:0:0:ff::d {
             peer 192.0.2.1;
     };
     group type external peeras 64513 {v4u v6u} remotev6addr
         fec0:0:0:ff::e {
             peer 192.0.2.2;
```

```
        };
    };
```

See Also

localv4addr on page 472

localv6addr on page 474

remotev4addr on page 500

routetopeer

Name

routetopeer - specifies the actual time to live (TTL) used on a socket in all cases

Syntax

```
routetopeer ;
```

Parameters

none

Description

routetopeer specifies the actual TTL used on a socket in all cases. In particular, if GateD realizes that two BGP speakers are peering over a single network, GateD automatically sets the **dontroute** option on their peering session. This, in turn, causes the TTL of the packets to be set to 1. **routetopeer** prevents the **dontroute** option from being set. If you specify **routetopeer**, but don't specify a TTL, and you are directly connected, GateD will set the TTL of your peering session to 1. If you want a TTL greater than 1 for directly connected peers, you must specify both **routetopeer** and the **ttl** that you require.

Default

disabled

Context

mpbgp group type external statement

mpbgp group type internal statement

mpbgp group type routing statement

mpbgp peer statement

Examples

[**autonomoussystem** and **routerid** omitted]

```
mpbgp on {
    group type external peeras 64512 {
        peer 192.0.2.1 routetopeer ttl 5 ;
    } ;
} ;
```

[**import** and **export** statements omitted]

See Also

ttl on page 510

sendbuffer

Name

sendbuffer - controls the amount of send buffering asked of the kernel

Syntax

sendbuffer *buffer_size* **;**

Parameters

buffer_size - an integer from 1 to 65535 inclusive

Description

sendbuffer controls the amount of send buffering asked of the kernel. The maximum supported is 65535 bytes, although many kernels have a lower limit. By default, GateD configures the maximum supported. **sendbuffer** is not needed on normally functioning systems.

Default

sendbuffer 65535 ;

Context

mpbgp group type external statement

mpbgp group type internal statement

mpbgp group type routing statement

mpbgp peer statement

Examples

[**autonomoussystem** and **routerid** omitted]

```
mpbgp on {
    group type external sendbuffer 32768 {
        peer 192.0.2.1 ;
    } ;
} ;
```

[**import** and **export** statements omitted]

See Also

recvbuffer on page 497

setpref

Name

setpref - allows BGP's Local_Pref attribute to be used to set the GateD preference on reception, and allows GateD preference to set the **Local_Pref** on transmission

Syntax

setpref *metric* **;**

Parameters

metric

Description

setpref allows BGP's **Local_Pref** attribute to be used to set the GateD preference on reception, and allows GateD preference to set the **Local_Pref** on transmission. The **setpref** *metric* works as a lower limit, below which the imported **Local_Pref** may not set the GateD preference. (For full details, see “Preferences and Route Selection” on page 11 in *Configuring GateD*.)

Default

none

Context

mpbgp group type internal statement

mpbgp group type routing statement

Examples

See “Setpref/Local_Pref Overview” on page 75 in *Configuring GateD.*

See Also

“Preferences and Route Selection” on page 11 in *Configuring GateD*

showwarnings

Name

showwarnings - causes GateD to issue warning messages when receiving questionable BGP updates such as duplicate routes and/or deletions of non-existing routes

Syntax

showwarnings

Parameters

none

Description

showwarnings causes GateD to issue warning messages when receiving questionable MPBGP updates such as duplicate routes and/or deletions of non-existing routes. Normally, these events are silently ignored.

Default

disabled

Context

mpbgp group type external statement

mpbgp group type internal statement

mpbgp group type routing statement

mpbgp peer statement

Examples

[**autonomoussystem** and **routerid** omitted]

```
mpbgp on {
    group type external peeras 64512 showwarnings {
        peer 192.0.2.1 ;
    } ;
} ;
```

[**import** and **export** statements omitted]

See Also

traceoptions on page 508

traceoptions

Name

traceoptions - specifies the tracing options for this group

Syntax

traceoptions *trace_options* **;**

Parameters

Trace options include:

packets - Trace all BGP packets.

open - Trace BGP Open packets.

update - Trace BGP Update packets.

keepalive - Trace BGP Keepalive packets.

all - Trace changes to the GateD routing table.

Description

traceoptions specifies the tracing options for this group. By default, these are inherited from the MPBGP or global trace options. These values may be overridden on the peer statements.

Default

inherited from the global **traceoptions**

Context

mpbgp statement

mpbgp group type routing statement

mpbgp peer statement

Examples

[**autonomoussystem** and **routerid** omitted]

```
mpbgp on {
    traceoptions packets ;
    group type external peeras 64512 {
        peer 192.0.2.1 ;
    } ;
} ;
```

[**import** and **export** statements omitted]

See Also

showwarnings on page 507

ttl

Name

ttl - specifies time to live

Syntax

ttl *ttl* ;

Parameters

ttl - *ttl* has two units: seconds and number of hops. Either can be used.

Description

By default, GateD sets the IP TTL for local peers to 1, and the TTL for non-local peers to the default kernel value. The **ttl** option is provided mainly when attempting to communicate with improperly functioning routers that ignore packets sent with a **ttl** of **1**. Not all kernels allow the TTL to be specified for TCP connections.

Default

calculated

Context

mpbgp group type routing statement

mpbgp peer statement

Examples

[**autonomoussystem** and **routerid** omitted]

```
mpbgp on {
    group type external peeras 64512 {
        peer 192.0.2.1 ; # ttl of 1 (directly connected network)
        peer 192.168.1.1 routetopeer ttl 2 ;
    } ;
} ;
```

[**import** and **export** statements omitted]

See Also

routetopeer on page 504