Chapter 49

Border Gateway Protocol version 4 (BGP-4)

Introduction

This chapter describes the Border Gateway Protocol version 4 (BGP-4), how it is implemented on the router, and how to configure the router to use it.

BGP-4 is enabled with a special feature license that you can obtain by contacting an Allied Telesyn authorised distributor or reseller.

BGP-4 runs on hardware with 16 MB or more of RAM and is used with IPv4. See the Hardware Reference for your router for memory details.

Overview of BGP-4

The Border Gateway Protocol version 4 (BGP-4) is an external gateway protocol. It allows two routers in different routing domains, known as *Autonomous Systems*, to exchange routing information to facilitate the forwarding of data across the borders of the routing domains. The basic operation of BGP-4 is described in RFC 1771, "A Border Gateway Protocol 4 (BGP-4)".

An Autonomous System (AS) is a set of routers under a single technical administration that uses:

- one or more internal gateway protocols (IGP)
- one or more sets of common metrics to route packets within its own AS
- an external gateway protocol (EGP) to route packets to other ASs

Every public AS is identified by an Autonomous System Number (ASN) in the range 1 to 64511. An ASN in this range is a globally unique number that the IANA assigns to every AS on the internet.

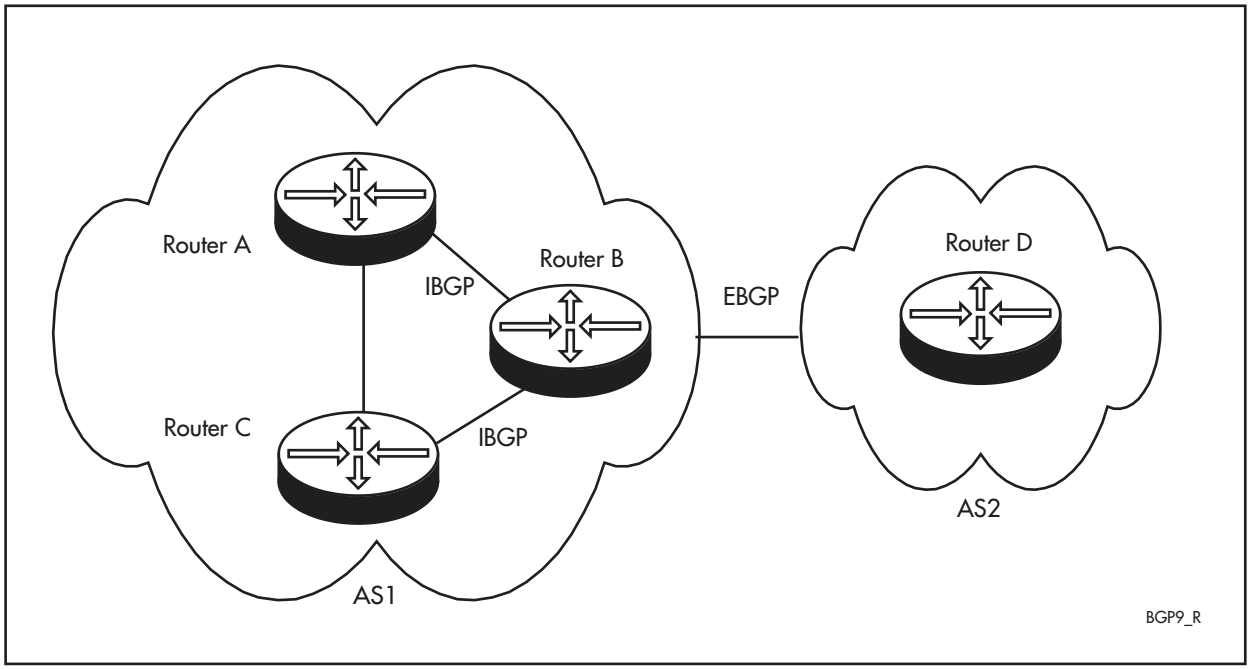
BGP lets routers learn multiple paths, choose the best path, and install it in the IP routing table. BGP-4 is based on distance vector (DV) protocol algorithms and uses TCP as its transport protocol on TCP port 179.

When BGP is used as an external gateway protocol to exchange routing information across AS borders, it is known as *External BGP (EBGP)*. EBGP connections are established between BGP *speakers* that have different AS numbers. A BGP speaker is any host that can use BGP to exchange routing information with another BGP-capable host. A BGP speaker does not necessarily have to be a router – it could be a host that passes routes learned by BGP to a router via another means, such as RIP.

A speaker may advertise any routes it knows to other speakers over an EBGP connection, as long as the speaker being advertised to has a different AS number from the speaker that is advertising.

The routes advertised over an EBGP connection can be learned by any means, for example IGP, EGP, or static assignment.

Figure 49-1: Example of the use of IBGP and EBGP



When BGP is used as an internal gateway protocol to transfer routing information within an AS, it is known as an *Internal BGP (IBGP)*. IBGP connections are established between BGP speakers that have the same AS number. Figure 49-1 on page 49-4 shows the use of IBGP and EBGP.

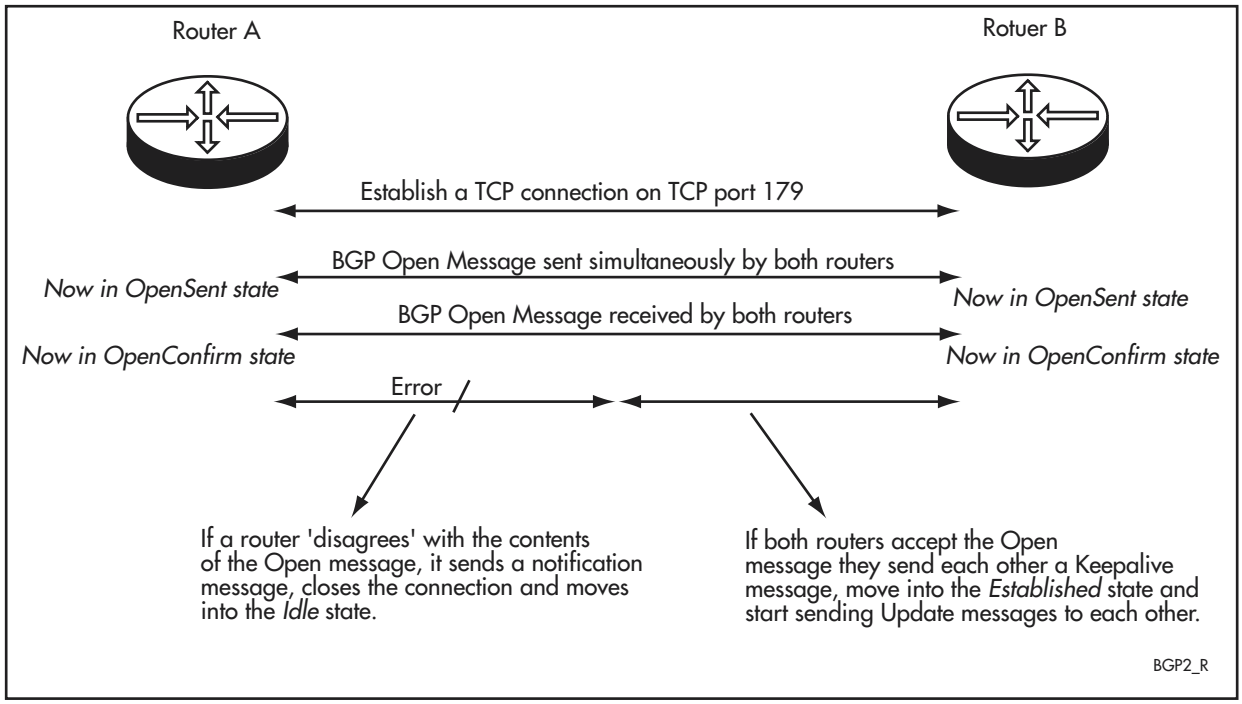
For speakers within an AS to learn the routes known by all the other speakers in the AS, they must have IBGP connections to all the other speakers in the AS. This is because a speaker cannot advertise routes learned over an IBGP connection to a speaker over another IBGP connection, unless confederations are used. However, a speaker can advertise routes learned from other means (for example, RIP and OSPF) to another speaker over an IBGP connection. A speaker can also advertise routes learned from an IBGP connection to another speaker over an EBGP connection.

An AS with one BGP speaker and a single external BGP connection is referred to as a *stub* AS.

BGP Operation

BGP is a protocol between two BGP speakers, which are called *peers*. Two routers become BGP peers when a TCP connection is established on Port 179 between them. The communication flow is illustrated in Figure 49-2.

Figure 49-2: Communication flow in a BGP session



Establishing a connection

BGP peer sessions start in the Idle state. In this state, BGP refuses all incoming BGP connections and does not allocate resources to peers. When you trigger a Start event—by enabling a peer—the router initiates a TCP connection to the peer and moves that peer session into the Connect state.

If the TCP connection attempt to a peer fails, the session moves into the Active state, waits until its ConnectRetry time expires, then tries to establish the connection again.

When the TCP connection is established, BGP peers immediately identify themselves to each other by simultaneously sending *open* messages, and move into the OpenSent state. The open messages let the peers agree on various protocol parameters, such as timers, and negotiate shared capabilities.

When each router receives an open message, it checks all the fields. If it "disagrees" with the contents of the open message, it sends a *notification* message, closes the connection and goes into the Idle state. If it finds no errors, it moves into the OpenConfirm state and sends back a *keepalive* message.

When both routers have received a keepalive message, they move into the Established state. The BGP session is now open. BGP sessions typically stay in the Established state most of the time. They only leave the Established state if an error occurs, or the hold time expires with no contact from the far end.

Exchanging routing information

Once a router has established a BGP connection with a peer, it starts to exchange routing information with that peer. Initially the peers exchange a complete copy of their routing tables. After this, they exchange updates to this routing information, in update messages.

The routing information contained within an update message consists of:

- a set of attribute values

  Attributes describe properties of the routes the update message contains. They are described in detail in "BGP Attributes" below.

- a list of one or more prefixes

  A prefix is the network address and CIDR mask. Each prefix contained within an update message represents a network that can be reached through the IP address given in the NextHop attribute contained in the same update message.

  The attribute values contained in the attributes section of the update message apply to *all* the prefixes that are advertised in that update message. Update messages can advertise multiple routes by listing multiple prefixes, as long as all their attributes are the same.

Update messages can also list routes that are withdrawn from service. These routes do not have to have the same attributes as the advertised routes.

Maintaining and closing a connection

Peers regularly exchange keepalive messages to prevent sessions from expiring. These messages are sent every 1 to 21845 seconds, every 30 seconds by default.

When an error occurs during a BGP session, the router that perceives the error sends a notification message to its peer identifying the error. It then closes the TCP connection and moves into the Idle state. Each router stops using routing information it heard from the other.

If a BGP speaker receives neither an update message or a keepalive message from a peer for a configurable period of time, called the hold time, it resets the session and withdraws any routes it learned from that peer.

BGP Attributes

An important part of the BGP protocol operation is the set of *attributes* associated with prefixes. Each BGP update message contains a set of attributes (Table 49-1 on page 49-7). These attributes describe some of the properties of the routes, and can be used in making decisions about which routes to accept and which to reject, in the following ways:

- If the router has multiple routes to a destination, it checks the attributes of each route to determine which one to use (see "BGP Route Selection" on page 49-8).
- You can create filters to reject or accept routes on the basis of their attributes (see "How to Filter Routes for BGP" on page 49-26).
- You can create route maps to change the attributes of particular update messages (see "How to Configure Route Maps" on page 49-29).

Table 49-1: BGP attributes

Attribute	Description
Origin	How the prefix came to be routed by BGP at the origin AS. The router can learn prefixes from various sources and then put them into BGP. Sources include directly connected interfaces, manually configured static routes, and dynamic internal or external routing protocols. Values are IGP (internal protocols such as RIP and OSPF), EGP (other EGPs) and INCOMPLETE (static routes or other means). Every update message has this attribute.
AS_path	A list of the autonomous systems through which the announcement for the prefix has passed. As prefixes pass between autonomous systems, each autonomous system adds its Autonomous System Number (ASN) to the beginning of the list. This means the AS_path can be used to make routing decisions. Every update message has this attribute, although it may be empty.
Next_hop	The address of the next node to which the router should send packets to get the packets closer to the destination. Every update message has this attribute.
Multi_Exit_Discriminator (MED)	A metric expressing the optimal path by which to reach a particular prefix in or behind a particular AS. One AS sets the value and a different AS uses that value when deciding which path to choose.
Local_preference	A metric used in IBGP so each host knows which path inside the AS it should use to reach the advertised prefix. EBGP peers do not send this value, and ignore it on receipt.
Atomic_aggregate	An attribute that allows BGP peers to inform each other about decisions they have made about overlapping routes. If Router A receives overlapping routes, and selects the less specific (more general route) only, then it attaches the atomic_aggregate attribute. When one of its neighbours receives a prefix with the atomic_aggregate attribute set, that neighbour must not take the prefix and de-aggregate it into any more specific entries in BGP.
Aggregator	An attribute that can be attached to an aggregated prefix to specify the AS and IP address of the router that performed the aggregation.
Community	Where the prefix is relevant to and should be advertised to. By default, all prefixes belong to the Internet community, which is the community of all BGP peers. Other communities have been globally defined that limit the scope of prefix advertisement or export, or you can identify a community by a community number.

BGP Route Selection

The route selection process involves selecting paths toward any prefix that should be put into a forwarding table. All routes that have been learned and accepted by the local system can be selected.

When there is only one route toward a particular prefix, it is the one used. When there are multiple routes for a particular prefix, then the system uses the following rules to decide which one to use. If a rule results in selection of a single route, the router uses that route. If multiple routes still match, the router goes to the next rule.

1. Selecting on local_preference

   The router chooses the route with the highest local preference. How the router determines the local preference depends on the source of the route:

   - for routes the router learned via an EBGP session, or for routes it learned from sources such as an IGP or static configuration, the router calculates the value of the preference itself.
   - for routes the router learned from an IBGP peer, the router uses the preference supplied by the peer—the update message for that route contains a local_preference attribute indicating the degree of preference.

2. Selecting on AS_path

   The router chooses the route with the shortest AS path.

3. Selecting on the Multi_Exit_Discriminator value

   If the local system is configured to take into account the value of the Multi_Exit_Discriminator (MED), and if the multiple routes are learned from the same neighbouring AS, the router chooses the route with the lowest MED value.

4. Selecting on the next_hop attribute

   The router chooses the route that has the minimum cost to the next hop specified in the next_hop attribute. Deciding the cost involves looking into the IP route table.

5. Selecting on routes learned via EBGP

   If the router learned only one of the possible routes via EBGP, it chooses that route. If the router learned more than one of the possible routes via EBGP, it chooses the route with the lowest router ID.

6. Selecting on routes learned via IBGP

   When the router learned all the possible routes via IBGP, it chooses the route that it learned from the IBGP neighbour with the lowest router ID.

Classless Inter-domain Routing (CIDR) and Aggregation

Interfaces, static routes, and routes learned via IGP (such as RIP and OSPF) can have a specific classful subnet mask (Class A, B, C or D). However, BGP routes are *classless*, (whereby IP addresses are not part of a class) so that shorter route tables can be exchanged and the number of advertised routes (prefixes) is less.

Figure 49-3 shows an example of inter-domain routing without CIDR. The service provider has many customers, all with Class C addresses that start with 204.71. The service provider announces each of the networks individually into the global Internet routing mesh.

Figure 49-3: Inter-domain routing without CIDR

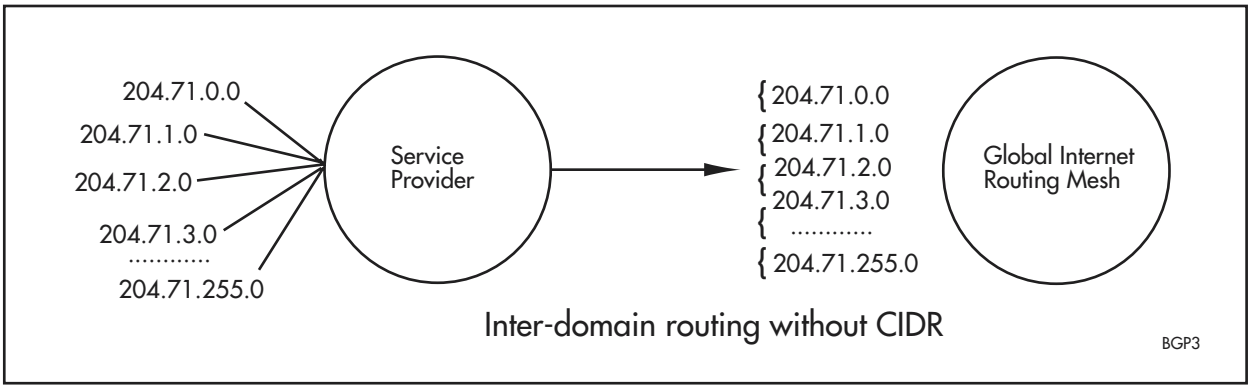
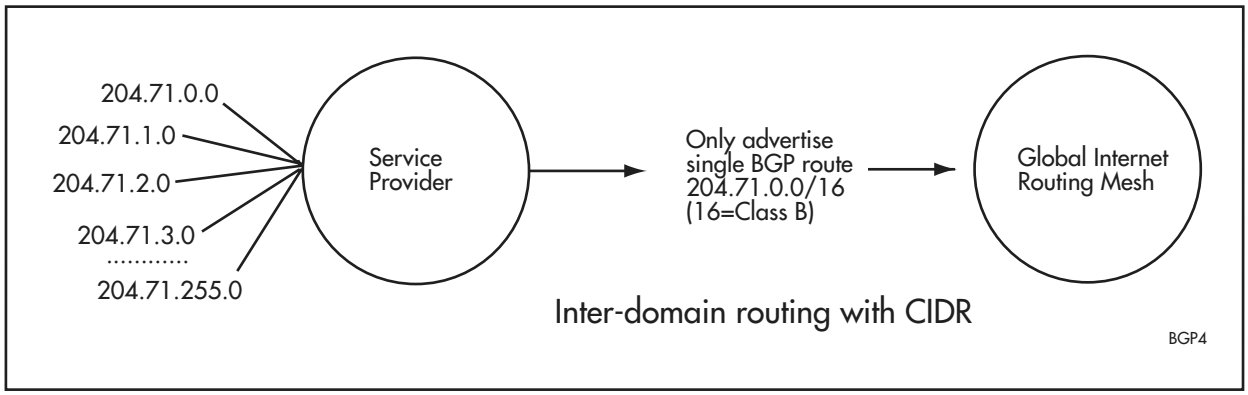


Figure 49-4 on page 49-9 shows an example of inter-domain routing with CIDR. By routing with CIDR, the service provider can aggregate the classful networks used by its customers into single classless advertisements. This provides routing for hundreds of customers by announcing only one advertisement into the global Internet routing mesh.

Figure 49-4: Inter-domain routing with CIDR



CIDR also copes with overlapping routes/prefixes because the route with the longest match (greatest number of bits/more specific netmask) is always chosen first. As the prefix traverses Autonomous Systems, each AS adds its AS number to the AS path bits, in both the BGP attribute and also the source IP and next-hop address.

BGP Multi-Homing

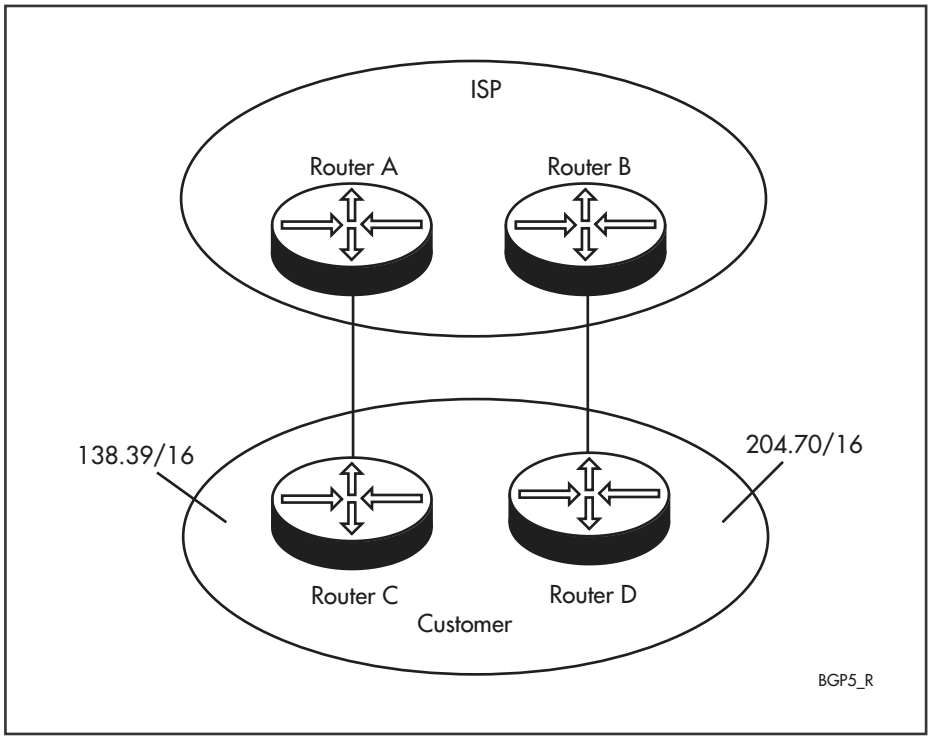
An AS may have multiple EBGP speakers connected to different ASs. This is known as BGP *multi-homing*. Multi-homing is used to improve reliability of the AS connection to the Internet, and to improve network performance owing to the fact that the network's bandwidth is the sum of all the circuits' bandwidth. Network performance increases when more than one connection is used at a time, otherwise, maximum performance is the bandwidth being used by one connection at a given time. An even split of traffic across multiple connections is called *load balancing*.

Sites can be multi-homed in the following ways:

- to a single Internet Service Provider (ISP) or Network Service Provider (NSP)
- to more than one ISP or NSP

The following figure illustrates the most reliable multi-homing topology to a single ISP involving different routers in the ISP and different routers in the customer network.

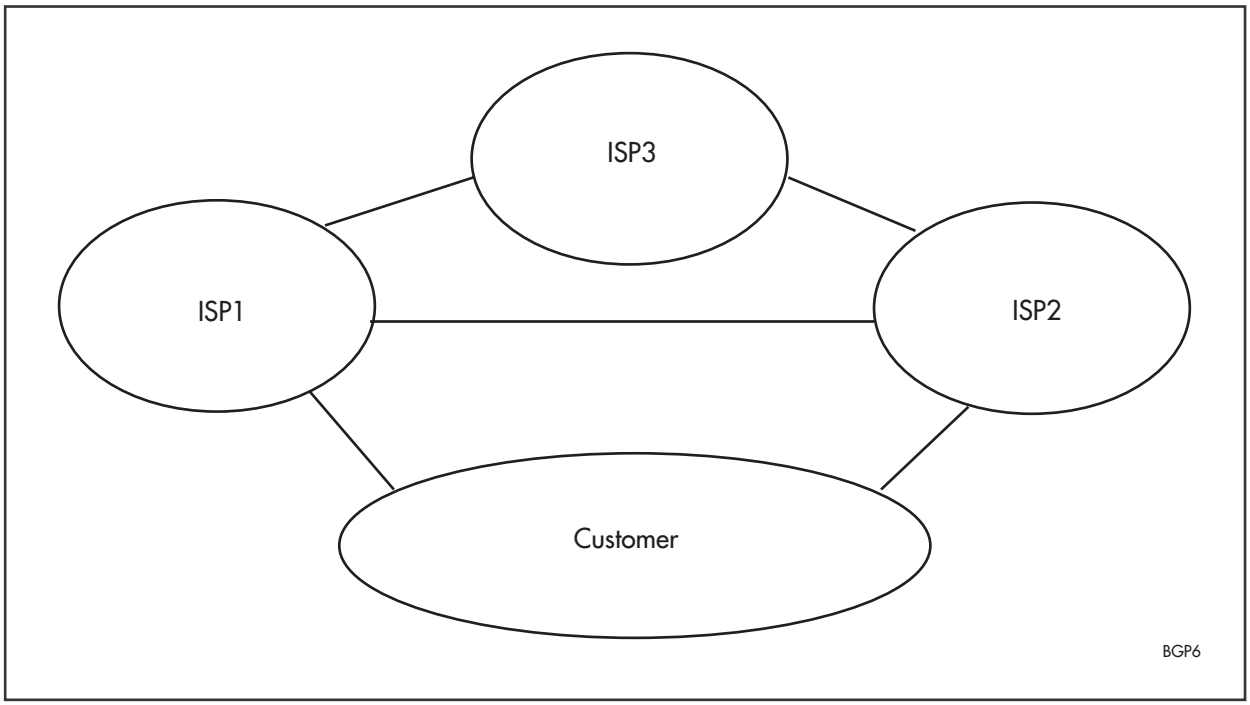
Figure 49-5: Multi-homing to a single ISP



This example is the most reliable because no equipment is shared between the two links. If the traffic between the two networks is equal, the approach to load balancing would be to use the link between Router A and Router C for traffic going to 138.39/16 and use the link between Router B and Router D for traffic going to 204.70/16.

Multi-homing to more than one provider is shown in Figure 49-6. The customer is multi-homed to ISP1 and ISP2; ISP1, ISP2, and ISP3 connect to each other. The customer has to decide how to use address space, as this is critical for load balancing from the ISPs to the customer, whether it delegates it by ISP1, ISP2, both, or independently.

Figure 49-6: Multi-homing to more than one provider



When the customer uses the address space delegated to it by ISP1, the customer uses a more specific prefix out of ISP1's aggregate and ISP1 can announce only the aggregate to ISP2. When the customer gets as much traffic from ISP1 as it gets from both ISP2 and ISP3, load balancing can be good. When ISP2 and ISP3 together send substantially more traffic than ISP1, load balancing can be poor. When the customer uses the address space delegated to it by ISP2, it does the same although ISP1 is the ISP to announce the more-specific route and attract traffic to it. Load balancing may be quite good if ISP1's address space is used, but not very good if ISP2's space is used.

When the customer uses the address space delegated to it by both ISP1 and ISP2, the degree of load balancing from ISP1 and ISP2 to the customer depends on the amount of traffic destined for the two ISPs. If the amount of traffic for the two is about the same, load balancing towards the customer can be quite good, if not, load balancing can be poor.

The option of the customer getting its own address space from a registry rather than from either ISP1 or ISP2 provides the most control but there is no aggregation. *Aggregation* is the combining of several different routes so that a single route can be advertised. This minimises the size of the routing table. The customer needs address space that makes it through the *route filters* (see "BGP Route Filtering"); otherwise, the customer may end up having no connectivity. If the customer gets address space that makes it through the route filters, there must be control over which provider uses which path to reach the customer.

When ISP1 is the largest, it may want to reach the customer via the ISP1-customer link, but have ISP2 and ISP3 go through the ISP2-customer link. When the path learned from ISP2 is shorter than the path learned from ISP1, ISP3 uses ISP2 to reach the customer.

BGP Route Filtering

BGP route filtering enables network providers to control their routing tables and meet the terms of business relationships they have with the networks they are connected to.

As a provider, you can filter the routing information that your routers receive from the networks they connect to, and that they advertise to those networks. This gives you control over the path of any traffic originating from or traversing your network.

Usually, one or more of your BGP routers form peer relationships with BGP routers at other ISPs with which you have entered into data transporting agreements. The process of BGP filtering is, in effect, the process of specifying the routes that your routers send or receive from each of their peers.

There are three filter types that you can apply to the BGP update messages that your router exchanges with a particular BGP peer:

- AS path filters

  Path filters look at the AS_path attribute in update messages. If the attribute in the update message matches the filter criteria then the whole update message is filtered out (or accepted, depending on what action the filter entry has been configured to carry out).

- prefix filters

  Prefix filters look at the individual prefixes within an update message, and compare them against an IP routing filter. If a prefix within the update message matches the filter criteria then that individual prefix is filtered out (or accepted).

- route maps

  Route maps have a complex combination of match criteria and actions. When applied to a BGP peer, they can:

  - accept or reject update messages on the basis of origin, community, AS path or MED
  - accept or reject particular routes, by comparing the update message's routes with a prefix list
  - alter the attribute values in matching update messages.

You can use filters in both the incoming and outgoing directions. In the incoming direction, they filter the update packets that the router receives from the peer. In the outgoing direction, they filter the update packets that the router sends to the peer.

If you create more than one type of filter, the router first applies the AS path filter, then the prefix filter, then the route map. Note that the router stops checking after the first filter entry that excludes the update or prefix, so an update or prefix is only included if all the applied filters result in it being included.

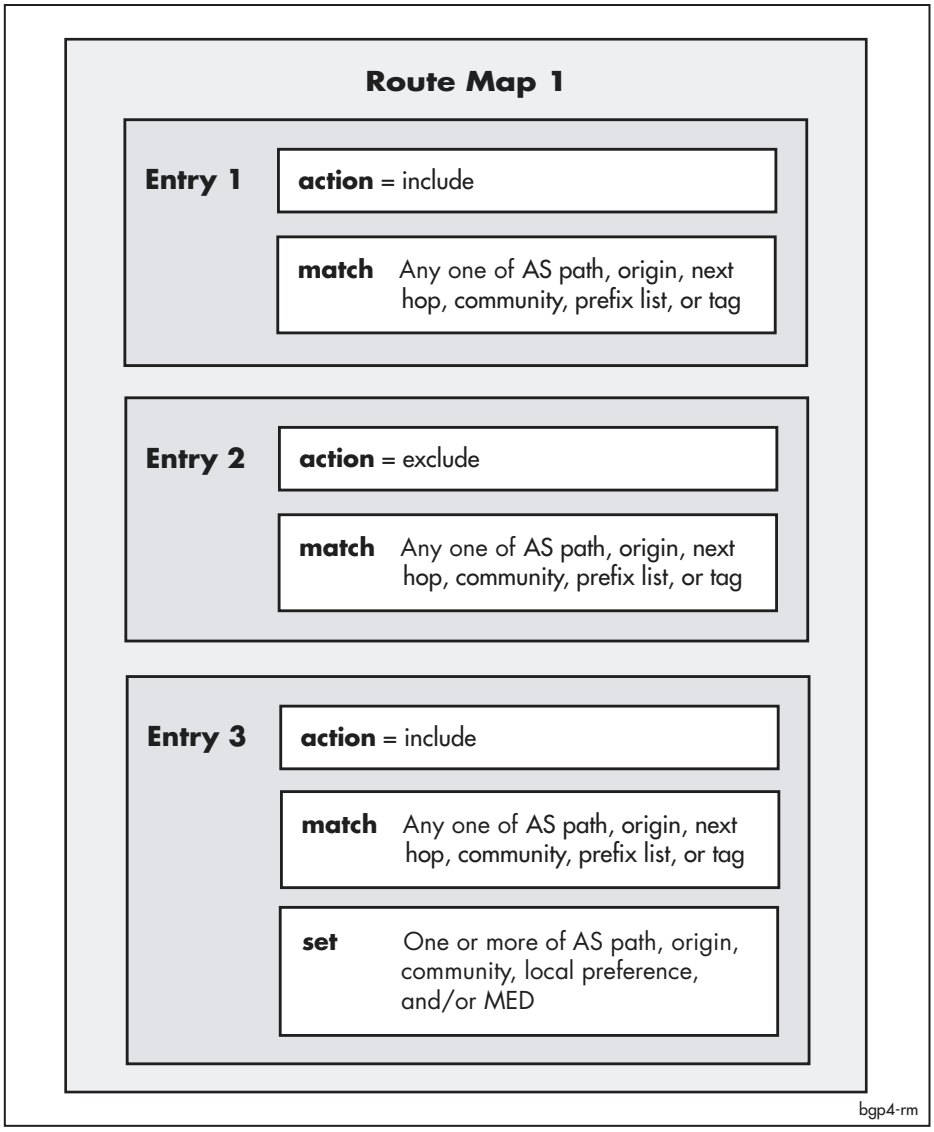
Route Maps

Route maps allow you to configure complex flexible filters. They achieve this by having several levels of structure:

- each route map consists of multiple entries
- each entry consists of an *action* (include or exclude) and at least one clause:
  - zero or one *match* clause, which determines which update messages or prefixes match the entry. If you do not specify a match clause, every update message or prefix matches.
  - zero or more *set* clauses, which change the BGP attributes of matching update messages or prefixes.

The following figure shows valid combinations of action and clause.

Figure 49-7: The structure of a route map



Through their set clauses, route maps can:

- add up to 10 AS numbers to the beginning of the AS_path attribute
- replace the community attribute with another of up to 10 entries or add up to 10 communities to the community attribute

- change the local_preference attribute to another value
- change the MED attribute to another value, or remove it
- change the origin attribute to another value

When a BGP process uses a route map:

1. It checks the entries in order, starting with the lowest numbered entry, until it finds a match.
2. Then it takes the action specified by that entry's action parameter. If the action is **exclude**, it filters that update or prefix out. If the action is **include**, it filters that update or prefix in.
3. Then if the action is **include**, it modifies attributes as specified by the entry's set clauses, if there are any.
4. Then it stops processing the route map; it does not check the remaining entries.

Every route map ends with an implicit entry that matches all routes with an action of **include**. This ensures that if no entries in a route map generate a match, the update message or route is included without modification.

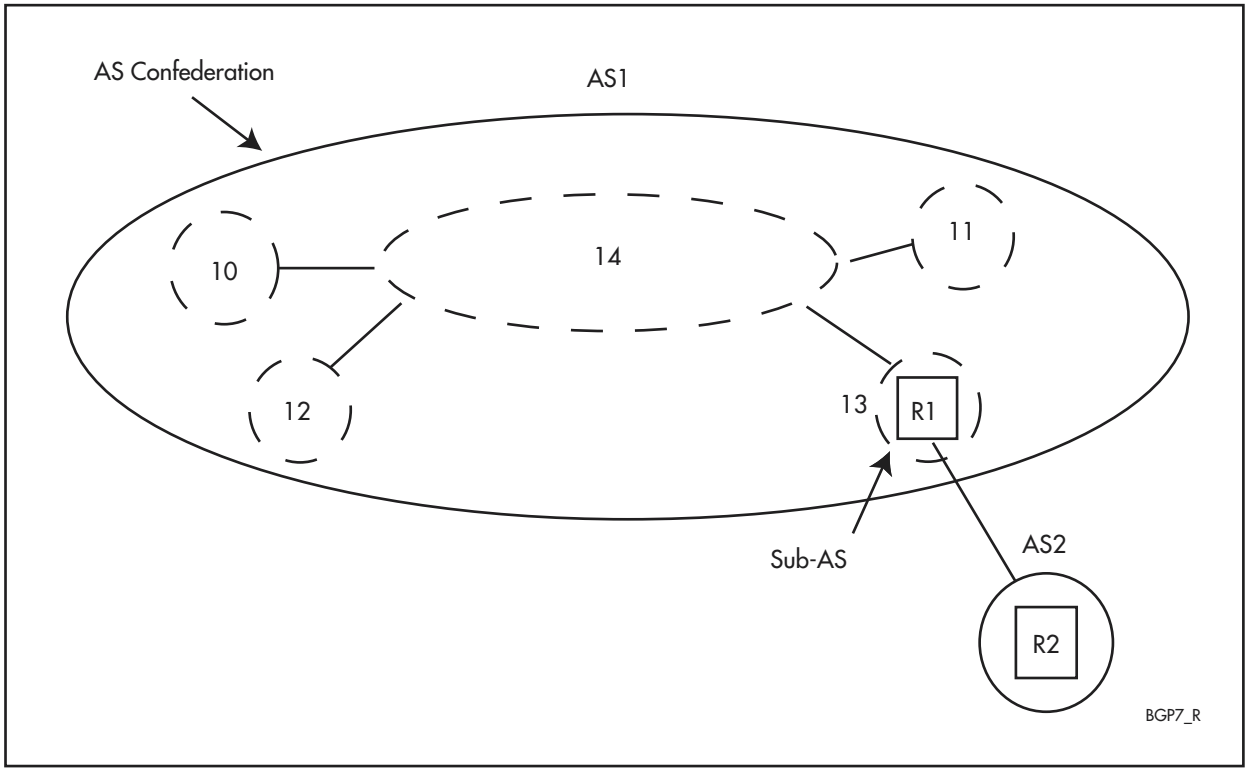
For information on creating and applying filters, see "How to Filter Routes for BGP" on page 49-26.

AS Confederations

An AS confederation is a collection of autonomous systems that are advertised as a single AS number to BGP speakers that are not members of the confederation. The autonomous systems in a confederation communicate between themselves using Confederation BGP (CBGP). AS confederations are used to subdivide autonomous systems with a very large number of BGP speakers to control routing policy.

Figure 49-8 is an example of an AS confederation. AS1 has been split into several sub-ASs (AS10-AS14). The original AS does not look any different to router 2, which is outside the confederation. Router 2 still sees AS1 rather than AS10-AS14. This implies that routers within the confederation are configured with an AS number for the confederation (for example, AS1), and an *AS member number*, an AS number visible only to those members within the confederation (for example, AS10).

Figure 49-8: Example of an AS confederation



Splitting a large AS into several smaller ones significantly reduces the number of intra-domain BGP connections. Unfortunately, splitting an AS may increase the complexity of routing policy based on AS path information for all members on the Internet. It also increases the maintenance overhead of coordinating external peering when the internal topology of this collection of ASs is modified.

Dividing an AS may unnecessarily increase the length of the sequence portions of the AS path attribute, and may adversely affect optimal routing of packets through the Internet.

Triggers

The Trigger Facility can be used to automatically run specified command scripts when particular triggers are activated. When a trigger is activated by an event, parameters specific to the event are passed to the script that is run. For a full description of the Trigger Facility, see Chapter 30, Trigger Facility.

Triggers can be created for two BGP events:

- when the router runs low on memory
- when a peer changes state.

Module MODULE=BGP

Event MEMORY
Description The router has run low enough on memory that BGP has had to start dropping routes.
Parameters There are no command parameters for this event.
Script Arguments There are no arguments to pass to the script.

Event PEERSTATE
Description The PEERSTATE trigger causes a trigger whenever a state change occurs that matches the peer, state and direction conditions. If the state is ANY and the direction is BOTH, two triggers are generated, one for leaving the old state and one for entering the new state.
Parameters The following command parameters can be specified in the **create** and **set trigger** commands:

Parameter	Description
peer=any|<*ipaddress*>	The IP address of the peer for which the state changes are interested in. This parameter is required in the **create trigger** command for BGP triggers, but is optional in the **set trigger** command unless the trigger event is changing from **memory** to **peerstate**.
bgpstate=idle|connect|active|opensent|openconfirm|established|any	The BGP state for which the trigger is required. This parameter is required in the **create trigger** command for BGP triggers, but is optional in the **set trigger** command, unless the trigger event is changing from **memory** to **peerstate**.
direction=enter|leave|both	Whether a match is made for the state the peer is leaving, entering, or both. This parameter is required in the **create trigger** command for BGP triggers, but is optional in the **set trigger** command unless the trigger event is changing from **memory** to **peerstate**.

Script Arguments The trigger passes the following arguments to the script:

Argument	Description
%1	The peer ID of the peer that has just undergone the state change.
%2	The state just left or entered.
%3	Whether the state was left or entered.

Example To create trigger 1, which activates whenever the router becomes low on memory, initiating the script MEMLOW.SCP, use the command:

```
create trigger=1 module=bgp event=memory script=memlow.scp
   repeat=yes
```

To create trigger 2, which activates whenever peer=172.30.1.2 leaves the ESTABLISHED state, initiating the script PEERDOWN.SCP, use the command:

```
create trigger=2 module=bgp event=peerstate peer=172.30.1.2
   bgpstate=established direction=leave script=peerdown.scp
   repeat=yes
```

To modify trigger 2, which activates when any peer leaves the ESTABLISHED state, use the command:

```
set trigger=2 peer=any
```

How to Configure BGP Peers

How to Create a Basic BGP AS

This section describes how to configure your network as an Autonomous System by using EBGP to send and receive routing information from an external peer (for example, an ISP), and by using IBGP to communicate routes within the AS.

For this basic BGP setup, you need to configure:

- the external BGP speaker. This is the router connected to the remote peer. In Figure 49-9, router B is the external speaker and router D is the remote peer. For the configuration procedure, see Table 49-2 on page 49-19. For checking and debugging, see Table 49-4 on page 49-21.
- the internal BGP speakers. These are all the other routers in the AS, connected to the external BGP speaker. In Figure 49-9, routers A and C are internal speakers. For the configuration procedure, see Table 49-3 on page 49-20. For checking and debugging, see Table 49-4 on page 49-21.

Figure 49-9: Example of the use of IBGP and EBGP

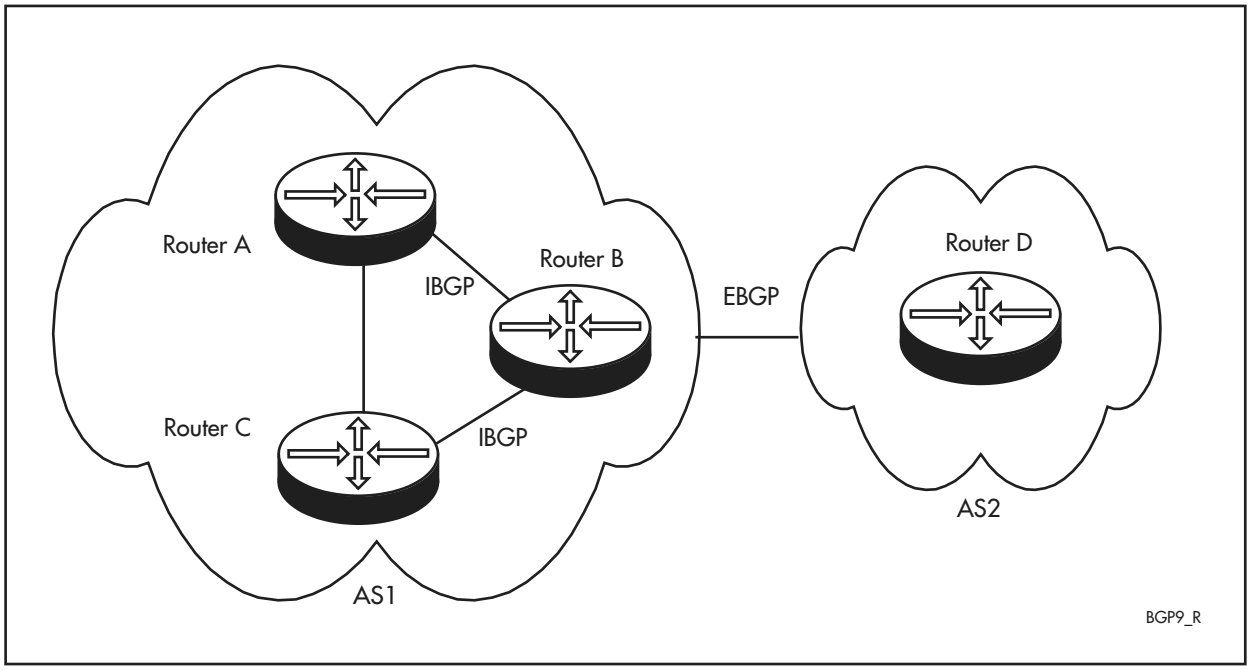


Table 49-2: Procedure for configuring the external BGP speaker

Step	Command	Action
1		Configure the lower-layer protocols that link the router to the remote peer, for example frame relay, PPP.
2	**set ip autonomous**=1..65534	Assign your AS number to the router.
3	**add ip interface**=*interface* ipaddress=*ipadd* [*other-options*] **add ip route**=0.0.0.0 interface=*interface* nexthop=*remote-peer-ipadd* **enable ip**	Configure IP on the interface that links the router to the remote peer: • assign an IP address • create a default route, if the IP addresses assigned to the interfaces connecting routers B and D are on different subnets • enable IP
4	**add ip interface**=*interface* ipaddress=*ipadd* [*other-options*]	Configure IP on the interfaces that link the router to each internal speaker.
5	**set bgp** routerid=*ipadd* [*other-options*]	Configure an interface on the router to be the source of IP packets generated by BGP. This step is not required but is good practice.
6	**add bgp peer**=*remote-peer-ipadd* remoteas=*remote-peer-asn* [authentication={md5|none}] [connectretry={default|0..4294967295}] [description=*description*] [ehops={default|1..255}] [holdtime={default|0|3..65535}] [infilter={none|300..399}] [inpathfilter={none|1..99}] [inroutemap=routemap] [keepalive={default|1..21845}] [local={none|1..15}] [maxprefix={off|1..4294967295}] [maxprefixaction={terminate|warning}] [minasoriginated={default|0..3600}] [minrouteadvert={default|0..3600}] [nexthopself={no|yes}] [outfilter={none|300..399}] [outpathfilter={none|1..99}] [outroutemap=*routemap*] [password=*password*] [sendcommunity={no|yes}]	Add the remote peer to the router. For the **peer** parameter, enter the IP address of the interface, on the remote peer, that the external speaker connects to. For the **remoteas** parameter, enter the remote peer's ASN.
7	**enable bgp peer**=*remote-peer-ipadd*	Enable the peer. The router establishes a BGP connection with the remote peer and exchanges routing information.

Table 49-3: Procedure for configuring an internal BGP speaker

Step	Command	Action
1		Connect the router directly to the external speaker and configure lower-layer protocols as required, for example VLANs.
2	**set ip autonomous**=1..65534	Assign your AS number to the router.
3	**add ip interface**=*interface* ipaddress=*ipadd* [*other-options*] **add ip route**=*ext-speaker-ipadd* interface=*interface* nexthop=*ipadd* **enable ip**	Configure IP on the interface that links the router to the external speaker: • assign an IP address • create a route to the external speaker if necessary • enable IP
4	**set bgp** routerid=*ipadd* [*other-options*]	Configure an interface on the router to be the source of IP packets generated by BGP. This step is not required but is good practice.
5	**add bgp peer**=*external-speaker-ipadd* remoteas=*external-speaker-asn* [authentication={md5|none}] [connectretry={default|0..4294967295}] [description=*description*] [ehops={default|1..255}] [holdtime={default|0|3..65535}] [infilter={none|300..399}] [inpathfilter={none|1..99}] [inroutemap=routemap] [keepalive={default|1..21845}] [local={none|1..15}] [maxprefix={off|1..4294967295}] [maxprefixaction={terminate|warning}] [minasoriginated={default|0..3600}] [minrouteadvert={default|0..3600}] [nexthopself={no|yes}] [outfilter={none|300..399}] [outpathfilter={none|1..99}] [outroutemap=*routemap*] [password=*password*] [sendcommunity={no|yes}]	Add the external speaker to the router as a BGP peer. For the **peer** parameter, enter the IP address of the external speaker's interface that this internal speaker connects to. For the **remoteas** parameter, enter the external speaker's ASN (which is the same as the internal speaker's ASN).
6	**enable bgp peer**=*ext-speaker-ipadd*	Enable the peer. The router establishes a BGP connection with the external speaker and exchanges routes.

Table 49-4: Procedure for checking and debugging BGP peers

Step	Command	Action
1	**show bgp peer** **show bgp peer**=*external-speaker-ipadd*	Check that the connections are established and that the internal and external speakers are exchanging messages. For example output and definitions, see Figure 49-19 on page 49-122 and Figure 49-20 on page 49-123.
2	**show bgp**	Check the number of routes learned, and other information. For example output and definitions, see Figure 49-11 on page 49-111.
3	**show bgp route**[=*prefix*] [regexp=*aspathregexp*] [community={internet|noadvertise| noexport|noexportsubconfed| 1..4294967295}]	List information about all or a subset of the learned routes. Note that BGP may learn many thousands of routes. For example output and definitions, see Figure 49-22 on page 49-130.
4	**enable bgp** **debug**={msg|state|update|all} [,...] [peer=*ipadd*]	Enable BGP debugging. Note that debugging may produce very large amounts of data.

How to Create BGP Peers Using Peer Templates

Peer templates make it easier to create BGP peers when many peers have identical inbound and outbound filtering policies, or timer values. They enable you to define a template set of these values, which you can subsequently apply to many different peers. You can assign a template to a BGP peer either when you create the peer, or afterwards.

Table 49-5: Procedure for using a template to create a BGP peer

Step	Command	Action
1	**add bgp peertemplate**=1..30 [connectretry={default|0..4294967295}] [description=*description*] [holdtime={default|0|3..65535}] [infilter={none|300..399}] [inpathfilter={none|1..99}] [inroutemap=*routemap*] [keepalive={default|1..21845}] [local={none|1..15}] [maxprefix={off|1..4294967295}] [maxprefixaction={terminate|warning}] [minasoriginated={default|0..3600}] [minrouteadvert={default|0..3600}] [nexthopself={no|yes}] [outfilter={none|300..399}] [outpathfilter={none|1..99}] [outroutemap=*routemap*] [sendcommunity={no|yes}]	Create the template. You can specify most of the peer settings in the template.
2	**show bgp peertemplate**[=1..30]	Check the template settings.
3	**add bgp peer**=*ipadd* remoteas=*asn* policytemplate=1..30 [authentication={md5|none}] [password=*password*] [description=*description*] [ehops={default|1..255}]	Create the peer entry and attach the template to it. You can also specify peer settings that are not available in the template.

How to Modify BGP Peers (Without Templates)

To modify a peer, unless the peer is using a template, use the command:

```
set bgp peer
```

You do not need to disable the peer first.

For information on changing peers that use templates, see "How to Modify BGP Peers that Use a Template" .

Once you have modified the peer, the router needs to update that peer. The router supports the following RFCs for updating modified peers:

- RFC 2918 Route Refresh Capability for BGP-4.
- RFC 2842 Capabilities Advertisement with BGP-4.

Automatic updates

You can configure the router to make updates automatically by using the command:

```
enable bgp autosoftupdate
```

This is disabled by default. Note that you must enable automatic updating before you modify the peer.

Manually-triggered updates

Alternatively, you can manually trigger the BGP peer to reset by using the command:

```
reset bgp peer={all|ipadd} soft={in|out|all}
```

The **soft** parameter determines the direction to update. There are two types of updates:

- Inbound updates, which reset routes that the router receives from the peer. To trigger these, the router sends a Route Refresh message to the peers it receives routes from. The Route Refresh message triggers the peers to resend a BGP Update message.
- Outbound updates, which reset routes the router sends. To reset these, the router simply sends a BGP Update message to the affected BGP peers.

If you do not manually or automatically trigger an immediate update, changes to the peer take effect when the router next receives an update message from that peer or sends an update message to it.

To see if automatic updating is enabled, use one of the commands:

```
show bgp
```

```
show bgp peer
```

In the command **show bgp peer**, you can see that the router and its peer have negotiated automatic updating when the "Capabilities" entry contains "Route Refresh". This command also displays the number of route refresh messages received from and sent to the peer.

How to Use a Template to Modify BGP Peers

You can apply a template to an existing peer, which overrides its current settings with the template settings.

Table 49-6: Procedure for using a template to modify a BGP peer

Step	Command	Action
1	**add bgp peertemplate**=1..30 [connectretry={default|0..4294967295}] [description=*description*] [holdtime={default|0|3..65535}] [infilter={none|300..399}] [inpathfilter={none|1..99}] [inroutemap=*routemap*] [keepalive={default|1..21845}] [local={none|1..15}] [maxprefix={off|1..4294967295}] [maxprefixaction={terminate|warning}] [minasoriginated={default|0..3600}] [minrouteadvert={default|0..3600}] [nexthopself={no|yes}] [outfilter={none|300..399}] [outpathfilter={none|1..99}] [outroutemap=*routemap*] [sendcommunity={no|yes}]	Create the template. You can specify most of the peer settings in the template.
2	**show bgp peertemplate**[=1..30]	Check the template settings.
3	**set bgp peer**=*ipadd* remoteas=*asn* policytemplate=1..30 [authentication={md5|none}] [password=*password*] [description=*description*] [ehops={default|1..255}]	Attach the template to the peer. You can also specify peer settings that are not available in the template.
4	**reset bgp peer soft** reset bgp peer=*ipadd* soft={in|out|all}	If automatic updating is not enabled, trigger the peer to update.

How to Modify BGP Peers that Use a Template

Once you have assigned a template to a peer, the method you use to modify the peer depends on the type and scope of the modification.

- To change a parameter on **all** peers that use the template, when the parameter **is** available in the template, change the template. Use the command:

```
set bgp peertemplate=1..30
    [connectretry={default|0..4294967295}]
    [description=description]
    [holdtime={default|0|3..65535}]
    [infilter={none|300..399}] [inpathfilter={none|1..99}]
    [inroutemap=routemap] [keepalive={default|1..21845}]
    [local={none|1..15}] [maxprefix={off|1..4294967295}]
    [maxprefixaction={terminate|warning}]
    [minasoriginated={default|0..3600}]
    [minrouteadvert={default|0..3600}]
    [nexthopself={no|yes}] [outfilter={none|300..399}]
    [outpathfilter={none|1..99}] [outroutemap=routemap]
    [sendcommunity={no|yes}]
```

- To change an **individual** peer when the parameter **is not** available in the template, specify the peer and parameter by using the command:

- `set bgp peer=ipadd remoteas=asn [authentication={md5|none}] [password=password] [description=description] [ehops={default|1..255}]` To change an **individual** peer when the parameter **is** available in the template, first remove the template from the peer by using the command:

```
set bgp peer=ipadd policytemplate=
```

  Specifying **policytemplate=** like this with no number removes the template. The peer retains the template's settings. Then change the settings you need to for that peer by using the command:

```
set bgp peer=ipadd [other-options]
```

For information on resetting peers after modification, see "How to Modify BGP Peers (Without Templates)" .

How to Delete BGP Peers

Table 49-7: Procedure for deleting a BGP peer

Step	Command	Action
1	**disable bgp peer**={*ipadd*|all}	Disable the peer. Enabled peers cannot be deleted.
2	**delete bgp peer**={*ipadd*|all}	Delete the peer.

As soon as the peer session goes down, the router removes any routes it learned from that peer. Once the route selection timer expires, the router withdraws the routes from any other peers it had advertised them to.

How to Filter Routes for BGP

Filters allow you to control the routes that BGP learns and advertises. There are three different types of filter: AS path filters, prefix filters and route maps. For a description of the filter types, see "BGP Route Filtering" on page 49-12.

In very general terms, configuring any of these filters involves two steps:

1. Create the filter.
2. Apply it.

   You can apply AS path filters, prefix filters and route maps to update messages received from a BGP peer, or sent to the BGP peer.

   You can also apply a route map to a particular prefix, independent of the peer. See "How to apply route maps to imported routes" on page 49-34.

How to Configure AS Path Filters

The AS path attribute lists the AS numbers of every Autonomous System that the routing information in an update message has passed through. It shows the path the update message has taken, and how "close" the routes are to the router. You can filter to accept or reject update messages on the basis of all or part of their AS path.

Creating AS path lists

The first step is to create an *AS path list* and add entries to it by using one of the commands:

```
add ip aspathlist=1..99 [entry=1..4294967295]
   include=aspath-reg-exp
```

```
add ip aspathlist=1..99 [entry=1..4294967295]
   exclude=aspath-reg-exp
```

Each entry uses a regular expression, *aspath-reg-exp*, to both specify the AS numbers that the entry matches, and to establish whether matching AS numbers are included or excluded. Table 49-8 shows regular expression syntax and examples.

Table 49-8: Syntax for AS path regular expressions

Token	Description	Examples	Meaning of example
<AS number>	Matches that identical AS number.	123	Matches any AS path attribute that contains AS 123 (but not 1234, 12345, or 5123).
^	Matches the start of the AS path attribute.	^123	Matches AS path attributes that have AS 123 as the first AS.
$	Matches the end of the AS path attribute.	^$	Matches an empty AS path attribute.
		^123$	Matches an AS path attribute with a single AS number, 123.
<space>	Separates AS numbers in a regular expression.	"123 456"	Matches AS path attributes that contain ASs 123 and 456, in that order, with no other AS numbers between them.
" "	Surrounds regular expressions that contain spaces.		
.	Matches any AS number.	.*	Matches all AS path attributes.
*	Matches zero or more repetitions of the preceding token in the AS path list being filtered	"123 .* 456"	Matches AS path attributes that contain ASs 123 and 456, in that order, with any number of other AS numbers between them.
+	Matches one or more repetitions of the preceding token in the AS path list being filtered.	"123 .+ 456"	Matches AS path attributes that contain ASs 123 and 456, in that order, with at least one other AS number between them.

You can apply AS path lists directly to BGP peers, or use them in route maps (see "How to Configure Route Maps" on page 49-29).

Applying path lists to peers

To apply the AS path list directly as a filter on a BGP peer, use one of the commands:

```
add bgp peer=ipadd remoteas=asn [inpathfilter=1..99]
    [outpathfilter=1..99] [other-options]

set bgp peer=ipadd [inpathfilter=1..99] [outpathfilter=1..99]
    [other-options]
```

The **inpathfilter** parameter applies the AS path list as a filter on update messages that the router receives from the peer. The router only accepts update messages if they match an AS path list entry that has the action **include**. If an update message matches an entry with the action **exclude**, the router rejects the update. If an update message does not match any entry in the AS path list, the router rejects the update. This is because each non-empty AS path list ends with an implicit entry that matches any AS path list and has the action **exclude**.

The **outpathfilter** parameter applies the AS path list as a filter on update messages that the router sends to the peer. The router only sends update messages if the update's AS path attribute matches an entry that has the action **include**. If a route matches an entry with the action **exclude**, the router does not advertise it to that peer. If an update message does not match any entry in the AS path list, the router does not advertise it to that peer.

An empty AS path list is equivalent to a path list that matches all AS numbers and has the action **include**.

How to Configure Prefix Filters

Prefix filters use IP routing filters. A routing filter is an IP filter with a number in the range 300-399. It matches on the source and mask of the prefix, and specifies whether matching prefixes are included or excluded.

You can use a prefix filter to reject some of the routes from an update message, without rejecting the whole update. This enables you to configure the router to accept only routes for particular networks from a particular peer, and to send only routes for particular networks to a particular peer. Therefore, the router can send or receive subsets of routes that have originated from or traversed a particular AS (or list of ASs), which is not possible with AS path filtering.

Creating IP routing filters

To create a routing filter, use the command:

```
add ip filter=300..399 action={include|exclude} source=ipadd
    [smask=ipadd] [entry=1..255]
```

The **source** parameter is the network IP address of the subnet to be filtered.

The **smask** parameter determines how many bits of the prefix are significant. When the router checks prefixes in an update message against the filter, it only checks the significant bits.

By default, new entries are added at the end of the filter. If you want the entry to be checked before some of the other entries, give it a lower entry number. This pushes existing entries with the same or higher number further down the list.

Applying routing filters to peers

To apply the routing filter as a prefix filter on a BGP peer, use one of the commands:

```
add bgp peer=ipadd remoteas=asn [infilter=300..399]
    [outfilter=300..399] [other-options]

set bgp peer=ipadd [infilter=300..399] [outfilter=300..399]
    [other-options]
```

The router checks every route in the update message against every entry in the filter, starting with the entry with the lowest entry number, until it finds a match or gets to the end of the filter.

The **infilter** parameter applies the filter to update messages that the router receives from the peer. If the router finds a match and that match has action **exclude**, the router rejects that route. If the match has action **include**, or there is no match, the router accepts the route.

The **outfilter** parameter applies the filter to update messages that the router sends to the peer. If the router finds a match and that match has action **exclude**, the router removes that route from the update message. If the match has action **include**, or there is no match, the router leaves the route in the update message and therefore advertises it to the peer.

How to Configure Route Maps

A route map consists of multiple entries, which are in effect individual filters. Each entry specifies both what it matches on, in a *match* clause, and what is done to matching traffic, in the entry's *action* and any *set* clauses it has.

Most set clauses modify the BGP attributes of matching update messages. If you want to change the attributes of all candidate routes, configure an entry with no match clause. Such an entry matches all update messages.

This section describes how to:

- create a route map
- configure match clauses
- configure set clauses
- apply the route map to a BGP peer
- apply the route map to a BGP prefix, independent of the peer

How to create a route map

You do not have to create a route map as a separate step—adding the first entry automatically creates it.

How to configure an entry with a match clause

The match clause for a route map entry determines which update messages or prefixes match the entry. Each entry can only match on one characteristic. Available characteristics are:

- AS path list
- community list
- origin attribute
- next_hop attribute
- prefix list
- tag

Matching on AS path list

The first step is to create an *AS path list* and add entries to it by using one of the commands:

```
add ip aspathlist=1..99 [entry=1..4294967295]
    include=aspath-reg-exp

add ip aspathlist=1..99 [entry=1..4294967295]
    exclude=aspath-reg-exp
```

Table 49-8 on page 49-27 shows the valid syntax for the regular expression *aspath-reg-exp* and gives syntax examples.

Then use the AS path list in the match clause of a route map by using the command:

```
add ip routemap=routemap entry=1..4294967295
    [action={include|exclude}] match aspath=1..99
```

When the router uses this route map to examine an update message, the router goes through the entries in the AS path list. The update matches if an entry in

the AS path list matches the AS_path in the update message, **and** that AS path list entry is an **include** entry.

If the update message matches, the router carries out the action of the route map. This is one of:

- exclude the update message
- include the update message without modification
- include the update message and modify its attributes

Note that the action (include/exclude) of the AS path list and the action of the route map entry are separate. Table 49-9 shows the effect of each combination.

Table 49-9: The effect of actions in AS path list and route map entries

AS path list entry	Route map entry	Action when route map applied
include	include	An update message with that AS_path matches, and is processed
include	exclude	An update message with that AS_path matches, and is discarded
exclude	include	An update message with that AS_path does not match. The router continues checking to see if the update message matches other entries in the route map.
exclude	exclude	An update message with that AS_path does not match. The router continues checking to see if the update message matches other entries in the route map.

In this context, the parameters **include** and **exclude** in the AS path list do not indicate whether the matching update message is allowed or dropped; they simply indicate whether the update matches or does not match the path list. This is different to the behaviour when you use the AS path list itself as a filter, as described in "How to Configure AS Path Filters" on page 49-26.

Example comparing AS path filter and route map

Compare this configuration, which uses an AS path filter:

```
add ip aspathlist=2 entry=1 exclude="^$"
add ip aspathlist=2 entry=2 include="15557"
set bgp peer=192.168.200.201 outpathfilter=2
```

with this configuration, which uses a route map and matches on AS path list:

```
add ip aspathlist=2 entry=1 include="^$"
add ip aspathlist=2 entry=2 exclude="15557"
add ip routemap=outdef3 entry=1 action=exclude match
   aspathlist=2
set bgp peer=192.168.200.201 outroutemap=outdef3
```

With both these configurations, the router drops update messages with empty AS paths, and advertises update messages with an AS path containing 15557. For the route map to achieve this (the second configuration):

- The AS path list has to **include** empty paths, so that the empty path matches the path list, and therefore is included into the route map's action of dropping packets that match the path list.
- The AS path list has to **exclude** updates whose AS path includes 15557. This excludes those updates from the route map's action of dropping packets that match the path list, so they are not dropped.

Matching on community list

The first step is to create a *community list* and add entries to it by using one of the commands:

```
add ip communitylist=1..99 [entry=1..4294967295]
    include={internet|noexport|noadvertise|
    noexportsubconfed|1..4294967295}[,...]

add ip communitylist=1..99 [entry=1..4294967295]
    exclude={internet|noexport|noadvertise|
    noexportsubconfed|1..4294967295}[,...]
```

Then use the community list in the match clause of a route map by using the command:

```
add ip routemap=routemap entry=1..4294967295
    [action={include|exclude}] match community=1..99
    [exact={no|yes}]
```

Note that the action (include/exclude) of the community list and of the route map entry are separate. This leads to the same behaviour as the distinction between the AS path list include/exclude parameters and the route map entry action. For a discussion of the distinction between these two include/exclude actions, see "Matching on AS path list" on page 49-29 and Table 49-9 on page 49-30.

If you specify **exact=yes**, an update message only matches the route map entry if its community attribute contains all the communities specified in the community list, and no other communities. If you specify **exact=no**, which is the default, then the set of communities in the attribute list of the update message must contain all the communities in the specified community list, but can also contain other communities.

Matching on next hop

Create a route map entry that specifies the IP address of the next hop by using the command:

```
add ip routemap=routemap entry=1..4294967295
    [action={include|exclude}] match nexthop=ipadd
```

This lets you select or discard routes that traverse a particular node.

Matching on origin

Create a route map entry that specifies the value of the origin attribute by using the command:

```
add ip routemap=routemap entry=1..4294967295
    [action={include|exclude}] match
    origin={egp|igp|incomplete}
```

This lets you select or discard routes depending on how BGP learned them: internally, externally, or from another means (such as statically-configured routes).

Matching on prefix list

A prefix list consists of a list of entries, each of which specifies:

- an IPv4 prefix, and a mask length or range of mask lengths. Together these specify the prefixes that the entry applies to.
- whether those prefixes explicitly match or explicitly do not match the prefix list.

All the match options described previously—AS path, community, next hop and origin—match on the **attributes** in an update message. Prefix list does not; it matches prefixes.

To use a prefix list, first create the prefix list and add entries to it by using the command:

```
add ip prefixlist=name entry=1..65535
    [action={match|nomatch}] [masklength=range] [prefix=ipadd]
```

The **masklength** parameter specifies the range of prefix mask lengths matched by this entry in the prefix list. The *range* is either a single CIDR mask from 0 to 32, or two masks separated by a hyphen. These options are valid for setting the mask length:

- As a mask length range (**masklength=*a*-*b*)**.

  For a route to match against this entry, its prefix mask length must be between *a* and *b* inclusive. *a* must be less than *b*.

- As a single mask length (**masklength=*a***).

  For a route to match against this entry, its prefix mask length must be exactly *a*.

- As an implicit mask length, by not specifying **masklength** (for example, **prefix=192.168.0.0**).

  For a route to match against this entry, its prefix mask length must correspond exactly to the mask for the class of the given address—in this example, 24.

Once you have created the prefix list, use it in the match clause of a route map by using the command:

```
add ip routemap=routemap entry=1..4294967295
    [action={include|exclude}] match prefixlist=name
```

Note that the action of the prefix list and of the route map entry are separate. Table 49-10 shows the effect of each combination.

Table 49-10: The effect of actions in prefix list and route map entries

Prefix list entry	Route map entry	Action when route map applied
match	include	An update message that contains the prefix matches the route map entry. The prefix is processed.
match	exclude	An update message that contains the prefix matches the route map entry. The prefix is removed from the update message. Other prefixes in the update are not removed.
nomatch	include	An update message that contains the prefix does not match the route map entry. The router continues checking to see if the update message matches other entries in the route map.
nomatch	exclude	An update message that contains the prefix does not match the route map entry. The router continues checking to see if the update message matches other entries in the route map.

In this context, the parameters **match** and **nomatch** in the prefix list do not indicate whether the prefix is allowed or dropped; they simply indicate whether the prefix matches or does not match the prefix list.

Matching on tag

See "How to Control Import of Static Routes" on page 49-37 for instructions on tagging routes and using the tags.

How to configure an entry with a set clause

Once you have determined what update messages or prefixes a route map entry matches, you can configure set clauses to change the attributes of matching items.

To create a set clause for an entry, use one of the commands shown in Table 49-11.

Table 49-11: The available set clauses for route maps

Command	Result
add ip routemap=*routemap* entry=1..4294967295 set aspath={1..65534[,...]}	Adds up to 10 AS numbers at the beginning of the AS path attribute.
add ip routemap=*routemap* entry=1..4294967295 set community={noexport|noadvertise| noexportsubconfed|1..4294967295}[,...]} [add={no|yes}]	Either: • replaces the community attribute with a list of up to 10 community values, if **add=no** (the default), or • adds up to 10 community values to the community attribute, if **add=yes**
add ip routemap=*routemap* entry=1..4294967295 set localpref=0..4294967295	Replaces the existing local_preference attribute, or sets it if it was not already set.
add ip routemap=*routemap* entry=1..4294967295 set med={0..4294967295|remove}	Replaces the existing MED attribute, or sets it if it was not already set, or if you specify **med=remove**, deletes the MED attribute.
add ip routemap=*routemap* entry=1..4294967295 set origin={igp|egp|incomplete}	Replaces the existing origin attribute, or sets it if it was not already set.

A prefix list can match a subset of prefixes in an update message. You can use this to change the attributes of some of the prefixes in an outgoing update, without having to change the attributes of all the prefixes. However, an update message contains just one set of attributes, which must apply to all the prefixes in the update. Therefore, the router splits the original update into two updates:

- one that contains the original attribute values and the prefixes that were not included by the route map entry, and
- one that contains the new attribute values and the prefixes that were included by the route map entry

How to apply route maps to BGP peers

To use the route map to filter or modify update messages that it receives from a peer, use one of the commands:

```
add bgp peer=ipadd remoteas=asn inroutemap=routemap
    [other-options]

set bgp peer=ipadd inroutemap=routemap [other-options]
```

To use the route map to filter or modify update messages that it sends to a peer, use one of the commands:

```
add bgp peer=ipadd remoteas=asn outroutemap=routemap
    [other-options]

set bgp peer=ipadd outroutemap=routemap [other-options]
```

The router checks every route in the update message against every entry in the filter, starting with the entry with the lowest entry number, until it finds a match or gets to the end of the filter.

If your route map is intended to modify the community attribute of outgoing update messages, you also need to enable the router to set the community attribute in messages to that peer. Use one of the commands:

```
add bgp peer=ipadd remoteas=asn outroutemap=routemap
    sendcommunity=yes [other-options]

set bgp peer=ipadd outroutemap=routemap sendcommunity=yes
    [other-options]
```

How to apply route maps to imported routes

The router is able to import routes into BGP that it learnt by non-BGP means—static routes, or routes learnt by OSPF or RIP.

You can apply a route map to this importation process so that the imported routes are given certain attributes, or so that certain routes are blocked from being imported. You can apply the route map by using either of the commands:

```
add bgp import={interface|ospf|rip|static} routemap=routemap

add bgp network=prefix[/0..32] [mask=mask] routemap=routemap
```

The router uses the route map to:

- filter routes or set attributes when it imports the routes into BGP
- set attributes on any update message in which it advertises the routes

Note that the entries in route maps that are applied to BGP importing cannot have match clauses that match on AS path or community. These attributes are not relevant to non-BGP routes. The route map entries can match on origin, next hop or tag.

How to Optimise BGP

How to Handle Spikes in Memory Use

The problem

While BGP is running, other software modules may cause a spike, or surge, of system memory utilisation for brief periods of time.

The solution: BGP backoff

BGP backoff enables BGP to elegantly handle low system memory situations. When memory is heavily utilised, BGP backs off and delays its processing until system memory is more abundant.

The backoff utility allows other processes access to the memory resources they need, without actually shutting BGP down unless it determines that BGP has backed off for a prolonged period of time. By default, BGP delays its processing for 10 seconds if system memory utilisation reaches 95%.

How to configure BGP backoff

To change BGP system memory backoff settings, use the command:

```
set bgp backoff[=0..100] [consecutive=1..20]
   [multiplier=0..1000] [basetime=0..100]
   [totallimit=0..1000] [step=0..1000]
```

This command provides the following configuration options:

- percentage limit of total system memory utilisation that causes BGP to back off—the **backoff** parameter
- time that BGP backs off for—a combination of the **step**, **basetime** and **multiplier** parameters (see "How long BGP backs off" below)
- total number of backoffs before all BGP peers are disabled—the **totallimit** parameter
- total consecutive number of backoffs before all BGP peers are disabled—the **consecutive** parameter.

How long BGP backs off

The backoff time is recalculated after a given number of backoffs. This is termed a *step*. The first backoff time is calculated as:

```
base time x multiplier/100
```

The backoff time is recalculated after each step based on the current backoff time:

```
current backoff time x multiplier/100
```

The value is rounded down to the nearest second (unless it is less than 1 second, in which case it is set to 1 second).

For example, a base time of 60 seconds with a multiplier of 110 increases the timeout by 10 percent every time the backoff time is recalculated. Thus, a step value of 2 and multiplier of 110 results in the following numbers:

Backoff Iteration	Time to Backoff (secs)
0	60
1	60
2	66
3	66
4	72
5	72
6	79
7	79

A multiplier of less than 100 percent gives the effect of a decay mechanism, and a multiplier of greater than 100 percent gives the effect of an accumulative mechanism.

Consecutive backoffs

If BGP gets to the end of the backoff period and system memory is still heavily utilised, BGP will immediately back off again without performing any processing. Such backoffs are called *consecutive backoffs*. By default, the number of consecutive backoffs is limited to 20. After BGP reaches this limit, the router considers that BGP is irrecoverable and disables all peers. You can change the limit.

The router counts the number of consecutive backoffs. It resets the count to zero whenever BGP is able to perform some processing after a backoff.

How to Stop BGP from Overloading System Memory

BGP memory accounting limits BGP's use of total system memory, to 95% of the total memory by default. The router disables BGP if it uses more than this percentage of system memory. The router shuts down BGP peers, and therefore drops all routes learnt from those peers.

To change the memory limit, use the command:

```
set bgp memlimit[=0..100]
```

To see the current limit and usage, use the command:

```
show bgp memlimit
```

To see detailed technical information about memory usage, use the command:

```
show bgp memlimit scan
```

How to Control Import of Static Routes

You can control which static routes you import into BGP by *tagging* routes with an identification number.

Table 49-12: Procedure for importing particular static routes

Step	Command	Action
1	**add ip route**=*ipadd* interface=*interface* nexthop=*ipadd* tag=1..65535 [*other-options*] or **set ip route**=*ipadd* interface=*interface* mask=*mask* nexthop=*ipadd* tag=1..65535 [*other-options*]	Specify a number to tag each static route that you want to import. You can also tag routes you specifically want to exclude.
2	**show ip route**	Check the number that the route is tagged with.
3	**add ip routemap**=*routemap* entry=1..4294967295 [action={include|exclude}] match tag=1..65535	Create a route map with entries that match the tagged routes.
4	**add bgp import**=static routemap=*routemap* **add bgp network**=*prefix*[/0..32] [mask=*mask*] routemap=*routemap*	Use the route map when importing routes into BGP.

Availability You can only use a route map that matches on **tag** when you use the **add bgp network** and **add bgp import** commands to import static routes from IP to BGP. Tagging does not filter routes that are sent to BGP peers, and does not match update messages that are received from BGP peers.

How to Set the IP Address By Which the Router Identifies Itself

When the router is acting as a BGP speaker, it uses an IP address to identify itself to its peers in these situations:

- when establishing the TCP session and sending TCP messages
- in the *open* message it sends at the beginning of the session
- when it considers itself to be the next hop for a route that it is advertising to its peers.

Address selection rules

The address the router uses in each of these situations depends on the situation and whether you have configured a router ID or a local interface address. The rules for each situation are:

1. TCP session source address

   If a local IP address has been set for the peer, use it. Otherwise allow TCP to select a source IP address, which it will do based on the outgoing interface.

2. BGP Identifier in *open* message

   If the router ID has been set, use it. Otherwise, if a local IP address has been set for the peer, use that. If neither has been set, use the highest IP address configured on any of the router's interfaces.

3. Next hop address

   If the router learned the route from an IBGP peer, use the learned next hop address—the next hop that the IBGP peer supplied for the route.

   If the router learned the route from an EBGP peer and the learned next hop is in the same subnet as the router, use the learned next hop.

   If the router learned the route from an EBGP peer and the learned next hop is in a different subnet to the router, then:

   - if a local IP address has been set for the peer to which the router is sending the update, use it
   - otherwise, if the router has an IP route to that network, use the IP address of the interface via which the route reaches that network
   - otherwise, use the IP address of the interface via which the router reaches the peer to which it is sending the update

How to configure router ID

To configure a router ID, use the command:

```
set bgp routerid=ipadd [other-options...]
```

How to configure local interface

To configure a local interface, first create the local interface and give it an IP address by using the command:

```
add ip local=1..15 ipaddress=ipadd [other-options...]
```

Then apply the local interface to the BGP peer by using one of the commands:

```
add bgp peer=ipadd remoteas=1..65534 local=1..15
   [other-options...]
```

```
set bgp peer=ipadd local=1..15 [other-options...]
```

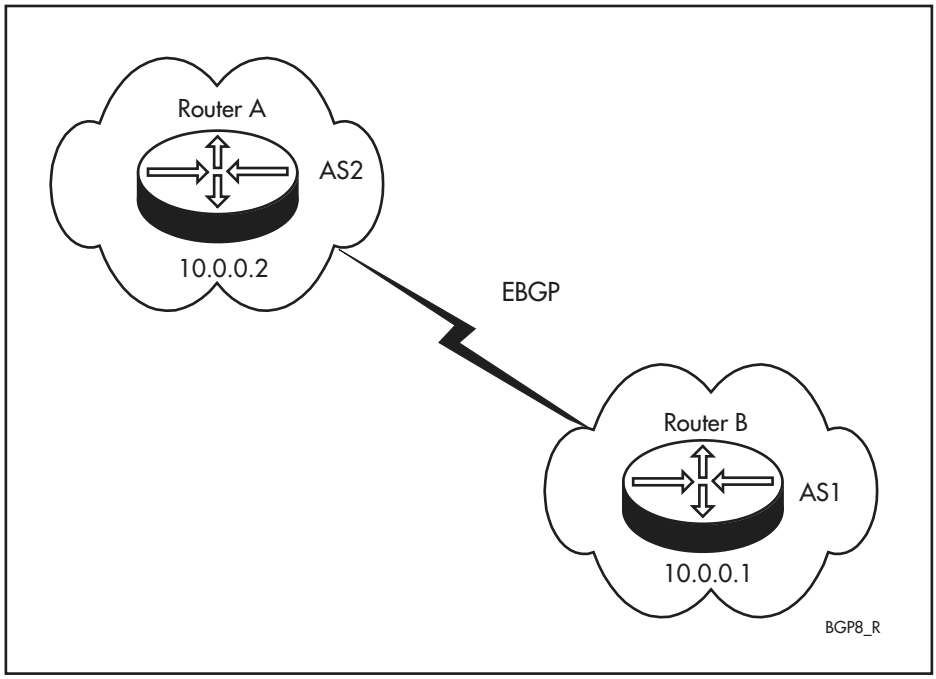
Configuration Examples

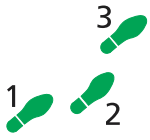
The following examples illustrate the steps required to configure BGP-4 on the router. The first example shows a simple configuration without filtering, the other examples illustrate configurations using filtering parameters.

Example One

Router A has been configured with IP address 10.0.0.2, AS number 2, and is to establish a BGP session with a peer, (Router B), configured with IP address 10.0.0.1, AS number 1. This example assumes that the IP address has already been configured and the two peers can ping each other. Figure 49-10 on page 49-39 illustrates a simple BGP-4 configuration.

Figure 49-10: Example of a simple BGP-4 configuration





To configure Router A and Router B as peers.

1. **Set the AS number**

   To set the autonomous system number for Router A, use the command:

   ```
   set ip autonomous=2
   ```

2. **Set the local IP address**

   To set the configuration of Router A's local IP interface, use the command:

   ```
   set ip local ip=10.0.0.2
   ```

   The local IP interface is a virtual interface that represents the IP routing module itself. The interface can be assigned an IP address, which can then be used as the source address of IP packets generated internally by IP protocols such as RIP, OSPF, PING, and NTP. (See the **set ip local** command on page 14-149 of Chapter 14, Internet Protocol (IP)).

3. Add Router B as a peer.

To add Router B as a peer to Router A, use the command:

```
add bgp peer=10.0.0.1 remoteas=1
```

There is a limit of 64 configurable peers.

The peer is now configured, but not yet able to establish a connection.

4. Establish a connection between Router A and Router B.

To enable Router B so that a connection can be established, use the command:

```
enable bgp peer=10.0.0.1
```

Router A can now attempt to establish a connection with Router B.

5. Check that the connection has been successfully established.

To verify that the connection is successfully established, use the command:

```
show bgp peer
```

This produces the following output.

```
BGP peer entries

Peer               State     AS      InMsg         OutMsg
-------------------------------------------------------------
-
10.0.0.1           Estab     1       23456         3245
```

6. Show the detailed output for a particular router.

To see detailed information about the status of Router B, use the command:

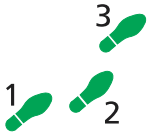
```
show bgp peer=10.0.0.1
```

This produces the following output.

```
Peer ................ 10.0.0.1
Description ......... -
State ............... Established
Remote AS ........... 1
Connect Retry ....... 120s
Hold time ........... 90s (actual 0s)
Keep alive .......... 30s (actual 0s - no KEEPALIVES)
Min AS originated ... 15
Min route advert .... 30
Filtering
  In filter ......... -
  In path filter .... -
  In route map ...... -
  Out filter ........ -
  Out path filter ... -
  Out route map ..... -
Max prefix .......... OFF
External hops ....... 1 (EBGP multihop disabled
Next hop self ....... No
Send community ...... No
Messages In/Out ..... 23456/3245
Debugging ........... -
Device .............. -
```

Example Two

Router A has been configured with IP address 10.0.0.2, AS number 2, Router B is configured as a peer with IP address 10.0.0.1, AS number 1. This example illustrates a configuration using the **inpathfilter** filtering parameter. The **inpathfilter** filters received BGP update messages based upon their AS path attributes. The filter is defined on a per peer basis.



To set up a peer with an IP address of 10.0.0.1 and an AS of 1 that has all received update messages that have originated from AS 300 filtered out (that is, the originating AS is at the end of the AS list)

1. **Add an AS path list entry.**

```
add ip aspathlist=1 entry=1 exclude="300$"
```

2. **Set up the BGP peer.**

```
add bgp peer=10.0.0.1 remoteas=1 inpathfilter=1
```

All update messages sent to this peer are included by default.

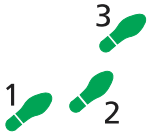
3. **An extra step to exclude routes that did not originate from AS 300 but went through AS 200 would be to use the command:**

```
add ip aspathlist=1 entry=2 exclude="200".
```

If an update message has AS 200 in it, and it originated from AS 300, then the first entry in the **aspathlist** is matched and the second entry match is not attempted.

Example Three

Router A has been configured with IP address 10.0.0.2, AS number 2, Router B is configured as a peer with IP address 10.0.0.1, AS number 1. This example illustrates a configuration using the **outpathfilter** filtering parameter. Using **outpathfilter** filters out BGP update messages being transmitted based upon their AS path attributes. The filter is defined on a per peer basis.



To set up a peer with an IP address of 10.0.0.1 and an AS of 1 that does not have any routes that have originated from AS550 advertised to it

1. **Add an AS path list entry.**

```
add ip aspathlist=2 entry=1 exclude="550$"
```

2. **Set up the BGP peer.**

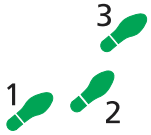
```
add bgp peer=10.0.0.1 remoteas=1 outpathfilter=1
```

All other update messages received from this peer are included by default.

Example Four

Router A has been configured with IP address 10.0.0.2, AS number 2, Router B is configured as a peer with IP address 10.0.0.1, AS number 1. This example illustrates a configuration using the **outroutemap** filtering parameter. The **outroutemap** filter is applied after all other filters have acted on an update message. The **outroutemap** filter is applied to an update message being transmitted, and filters it based on its AS path attribute if its match clause specifies an **aspathlist**.

The advantage of routemap filters over path filters is that they can be used to modify the attributes of a received BGP update message, whereas the path filters only includes or excludes messages.



To set up a peer with an IP address of 10.0.0.1 and an AS of 1 that does not have any routes that have passed through AS550 advertised to it

1. **Add an AS path list entry.**

```
add ip aspathlist=2 entry=1 include="550"
```

2. **Exclude routes that originated from AS550.**

```
add ip routemap=as550 entry=1 match aspathlist=2
   action=exclude
```

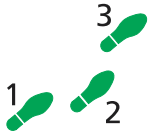
The **aspathfilter** has an **include** so that a match is made for the routemap's **entry**, and the routemap's **entry** action is taken. If the **aspathlist entry** was an **exclude**, then an AS Path that matched this line from the AS Path filter would stop processing the AS Path filter, would not match a Routemap Entry, and would move to the next routemap entry without any of the current routemap entry's **action/set** having been applied.

3. **Set up the BGP peer.**

```
add bgp peer=10.0.0.1 remoteas=1 outroutemap=as550
```

Example Five

Router A has been configured with IP address 10.0.0.2, AS number 2, Router B is configured as a peer with IP address 10.0.0.1, AS number 1. This example illustrates a configuration using the **inroutemap** filtering parameter. The **inroutemap** filter is applied after all other filters have acted on an update message. The **inroutemap** filter is applied to a received update message, and filters it based on its AS path attribute if its match clause specifies an **aspathlist**.



To set up a peer with an IP address of 10.0.0.1 and an AS of 1 that have all received update messages that have originated from AS 300 filtered out (that is, the originating AS is at the end of the AS list)

1. **Add an AS path list entry.**

```
add ip aspathlist=1 entry=1 include="300$"
```

2. **Filter out messages from AS300**

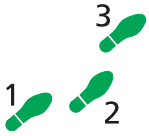
```
add ip routemap=as300 entry=1 match aspathlist=1
   action=exclude
```

3. **Set up the BGP peer.**

```
add bgp peer=10.0.0.1 remoteas=1 inroutemap=as300
```

Example Six

Router A has been configured with IP address 10.0.0.2, AS number 2, Router B is configured as a peer with IP address 10.0.0.1, AS number 1. This example illustrates a configuration using the **infilter** filtering parameter as Router B advertises a route to network 101.0.0.0, but there is a preferred route (because of an arrangement with the network) to use.



To set up a peer with an IP address of 10.0.0.1 and an AS of 1 that accepts all routes learned, except routes to network 101.0.0.0/8

1. **Add a filter entry to exclude a network.**

```
add ip filt=300 entry=1 action=exclude source=101.0.0.0
   smask=255.0.0.0
```

2. **Add a filter entry to include all routes.**

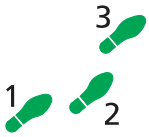
```
add ip filt=300 entry=2 action=include source=0.0.0.0
   smask=0.0.0.0
```

3. **Set up the INFILTER parameter.**

```
add bgp peer=10.0.0.1 remoteas=1 infilter=300
```

Example Seven

Router A has been configured with IP address 10.0.0.2, AS number 2, Router B is configured as a peer with IP address 10.0.0.1, AS number 1. This example illustrates a configuration where the peer is one that does not want to advertise a route to network 102.0.0.0 to use by using the **outfilter** filtering parameter.



To set up a peer with an IP address of 10.0.0.1 and an AS of 1 that sends all routes used, except routes to network 102.0.0.0/8

1. **Add a filter entry to exclude a network.**

```
add ip filt=301 entry=1 action=exclude source=102.0.0.0
   smask=255.0.0.0
```

2. **Add a filter entry to include all routes.**

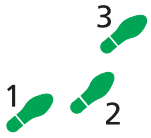
```
add ip filt=301 entry=2 action=include source=0.0.0.0
   smask=0.0.0.0
```

3. **Set up the INFILTER parameter.**

```
add bgp peer=10.0.0.1 remoteas=1 outfilter=301
```

Example Eight

Router A has been configured with IP address 10.0.0.2, AS number 2, Router B is configured as a peer with IP address 10.0.0.1, AS number 1. This example illustrates a configuration filtering on the basis of the community attribute, filtering on inbound and outbound routes.



To set the community attributes.

1. **Create routemaps that set the community attribute.**

   Calculate your community number using the following equation: (AS number x 65536)+1 = Community 1. For example, the AS number=2, so Community 1 is (2 x 65536)+1 = 131073.

   ```
   add ip routemap=map0 entry=1 set community=131073
   add ip routemap=map1 entry=1 set community=131074
   add ip routemap=map2 entry=1 set community=131075
   add ip routemap=map3 entry=1 set community=131076
   add ip routemap=map4 entry=1 set community=131077
   add ip routemap=map5 entry=1 set community=131078
   add ip routemap=map6 entry=1 set community=131079
   ```

2. **Associate the routemaps with subnets, which can then be applied to the routes as they are added to the BGP routing tables.**

   ```
   add bgp net=192.168.0.0/24 routemap=map0
   add bgp net=192.168.1.0/24 routemap=map1
   add bgp net=192.168.2.0/24 routemap=map2
   add bgp net=192.168.3.0/24 routemap=map3
   add bgp net=192.168.4.0/24 routemap=map4
   add bgp net=192.168.5.0/24 routemap=map5
   add bgp net=192.168.6.0/24 routemap=map6
   add bgp net=192.168.7.0/24 routemap=map6
   add bgp net=192.168.8.0/24 routemap=map6
   add bgp net=192.168.9.0/24 routemap=map6
   add bgp net=192.168.10.0/24 routemap=map6
   ```

   Note that the community attribute of the last five routes are set to the same value (16). Any routemaps not included may be used for other BGP peers.

3. **Filter the inbound routes on Router B.**

   ```
   add ip community=1 entry=1 include=16
   add ip community=1 entry=2 exclude=internet
   ```

   This builds a community list consisting of those routes with the community attribute value set to 16 and exclude all other routes.

4. **Associate the community filter with a routemap.**

   ```
   add ip routemap=mapin entry=1 match communitylist=1
   add ip routemap=mapin entry=2 action=exclude
   ```

5. **Apply the routemap to the BGP peer.**

   ```
   set bgp peer=10.0.0.2 sendcommunity=yes inr=mapin
   ```

Command Reference

This section describes the commands available on the router to enable, configure, control and monitor BGP. See Chapter 14, Internet Protocol (IP) for the commands required to enable and configure IP to use BGP.

The shortest valid command is denoted by capital letters in the Syntax section. See "Conventions" on page xcv of Preface in the front of this manual for details of the conventions used to describe command syntax. See Appendix A, Messages for a complete list of messages and their meanings.

add bgp aggregate

Syntax

```
ADD BGP AGGregate=prefix[/0..32] [MASK=mask]
   [SUMmary={NO|YES}] [ROUTEMap=routemap]
```

Description

This command adds an aggregate entry to BGP. When a peer advertises a route that is a subset of the entry's prefix, the router adds the aggregate entry to its database, as well as the entry for the more specific route. This can increase the efficiency of BGP, by allowing the router to process and advertise a single route, instead of a large number of more specific subnets.

Note that the router does not use the aggregate route for IP routing. The router only uses the aggregate to determine which routes to advertise.

The router does not add the aggregate entry to its database until it receives an advertisement of a more specific subnet.

The router advertises the aggregate route as coming from the router's autonomous system, and sets the aggregate's atomic_aggregate attribute.

Parameter	Description	
AGGregate	The network prefix to be used for this aggregate entry. This is expressed as the base IP address of the network, in dotted decimal notation, optionally followed by a "/" character and the number of bits in the network mask. If you do not specify the CIDR mask, the router uses the value from the **mask** parameter, if present, or otherwise the natural mask for the network, based on whether it is a class A, B, or C network. Default: no default	
MASK	The network mask for the aggregate entry. This parameter is provided for compatibility with other router commands that specify an IP address and mask; we recommend that you instead specify the mask in the **aggregate** parameter. If you specify a mask in this parameter and the **aggregate** parameter, an error results unless the two masks agree. Default: The natural mask for the network, based on whether it is a class A, B, or C network	
SUMmary	Whether the router advertises only the aggregate route, or also the more specific routes that make up the aggregate. Default: **no**	
	no	The router advertises the more specific routes that make up the aggregate.
	yes	The router only advertises the aggregate route. Note that unadvertised routes are still displayed in the output of the **show bgp route** command, but are marked with an "s".
ROUTEMap	The route map used to filter the more specific routes that make up the aggregate, or to set attributes for the aggregate route. The *routemap* is the name of the appropriate pre-existing map. Default: no route map (routes are not filtered and attributes are not set)	

Tip The shortest string you can enter is shown in capital letters.

Examples

To add an aggregate entry for the network 198.168.0.0, with a mask of 255.255.0.0, use the command:

```
add bgp agg=192.168.0.0/16
```

As soon as the router learns a more specific route, such as 192.168.1.0/24, BGP also adds an entry for 192.168.0.0/16 to the routing table.

To add an aggregate entry for the network 192.168.8.0/21 and use route map *agg_map*, use the command:

```
add bgp agg=192.168.8.0/21 routem=agg_map
```

Related Commands

delete bgp aggregate
set bgp aggregate
show bgp aggregate
show bgp route

add bgp confederationpeer

Syntax `ADD BGP CONFEDerationpeer=1..65534`

Description This command adds an Autonomous System to the AS confederation to which this router belongs. An AS confederation is a group of Autonomous Systems which communicate between themselves using confederation BGP, and communicate to Autonomous Systems outside the confederation as if they were a single Autonomous System. For more information about AS confederations, see "AS Confederations" .

The **confederationpeer** parameter specifies the number of an Autonomous System that is to be treated as one of the Autonomous Systems in the confederation. This number cannot be the same as this router's AS number, or this router's confederation ID. The specified AS number should not already have been added with this command.

A router need not be configured with all of the AS numbers in the AS confederation, but only those with which it is to have peer relationships. Similarly, the confederation ID for the router only has to be configured on routers that are to have peer relationships with BGP routers outside the confederation.

When you create the peer relationship by using the **add bgp peer** command on page 49-51, specify the peer's confederation ID in the **remoteas** parameter.

Examples To set up a confederation with AS numbers 65502, 65503 and 65504, whose external AS number is 1234, and with this router in AS 65501, use the commands:

```
set ip au=1234
set bgp conf=65501
add bgp confed=65502
add bgp confed=65503
add bgp confed=65504
```

Related Commands
delete bgp confederationpeer
set bgp
set ip autonomous
show bgp confederation

add bgp import

Syntax

```
ADD BGP IMPort={INTerface|OSPF|RIP|STAtic}
    [ROUTEMap=routemap]
```

Description

This command adds an import entry to BGP. This instructs BGP to import routes from a given route source into the BGP route table. Optionally, you can specify a route map to allow filtering of routes and setting of BGP attributes.

When BGP imports routes from a protocol, it only imports routes which are the best routes to their destination networks. BGP determines that a route is the best route if:

- the route goes over an active interface
- that interface does not have an infinite or unreachable route
- the route has a higher routing preference metric than other candidate routes.

Parameter	Description	
IMPort	The source of routing information for the routes that are to be imported into BGP. Default: no default	
	INTerface	Imports interface routes.
	OSPF	Imports OSPF routes.
	RIP	Imports RIP or RIP2 routes.
	STAtic	Imports statically configured routes.
ROUTEMap	The route map used to filter the routes imported into BGP and to set attributes for the routes as advertised by BGP. The *routemap* is the name of the appropriate pre-existing map. The route map can **match** on origin, next hop, prefix list or tag, and can use any of the **set** parameters. Default: no route map (routes are not filtered and attributes are not set)	

Tip The shortest string you can enter is shown in capital letters.

Examples

To import OSPF routes into BGP and use the route map *ospf_bgp_map* to filter and set attributes, use the command:

```
add bgp imp=ospf routem=ospf_bgp_map
```

Related Commands

add ip routemap
delete bgp import
set bgp import
show bgp import

add bgp network

Syntax

```
ADD BGP NETwork=prefix[/0..32] [MASK=mask]
    [ROUTEMap=routemap]
```

Description

This command adds a network to the list of networks that the router can advertise to remote BGP peers. You can also optionally specify a route map to filter the networks and/or to set attributes on the routes sent.

Statically defining a BGP network with this command does not cause the router to advertise the network immediately—the router does not know the next hop for the network. Defining a BGP network informs BGP that if the router learns a route to the network by a non-BGP means, for example statically or from OSPF, then BGP should advertise the network.

Parameter	Description
NETwork	The network to add to the list of networks that can be advertised. This is expressed as the base IP address of the network, in dotted decimal notation, optionally followed by a "/" character and the number of bits in the network mask. If you do not specify the CIDR mask, the router uses the value from the **mask** parameter, if present, or otherwise the natural mask for the network, based on whether it is a class A, B, or C network. Default: no default
MASK	The network mask for the network. This parameter is provided for compatibility with other router commands that specify an IP address and mask; we recommend that you instead specify the mask in the **network** parameter. If you specify a mask in this parameter and the **network** parameter, an error results unless the two masks agree. Default: The natural mask for the network, based on whether it is a class A, B, or C network
ROUTEMap	The route map used to filter this network and to set attributes on routes that are sent in the BGP update messages that advertise this route. The *routemap* is the name of the appropriate pre-existing map. The route map can **match** on origin, next hop, prefix list or tag, and can use any of the **set** parameters. Default: no route map (routes are not filtered and attributes are not set)
Tip The shortest string you can enter is shown in capital letters.	

Examples

To add the network 192.169.2.0 to the list of networks advertised by BGP and to use the route map "normal", use the command:

```
add bgp net=192.169.2.0/24 routem=normal
```

Related Commands

delete bgp network
show bgp network

add bgp peer

Syntax

```
ADD BGP PEer=ipadd REMoteas=1..65534
   [AUthentication={MD5|NONE}]
   [CONnectretry={DEFault|0..4294967295}]
   [DESCription=description] [EHOps={DEFault|1..255}]
   [HOLdtime={DEFault|0|3..65535}]
   [INFilter={NONE|300..399}] [INPathfilter={NONE|1..99}]
   [INRoutemap=routemap] [KEEpalive={DEFault|1..21845}]
   [LOCal={NONE|1..15}] [MAXPREFIX={OFF|1..4294967295}]
   [MAXPREFIXAction={Terminate|Warning}]
   [MINAsoriginated={DEFault|0..3600}]
   [MINRouteadvert={DEFault|0..3600}]
   [NEXthopself={NO|YES}] [OUTFilter={NONE|300..399}]
   [OUTPathfilter={NONE|1..99}] [OUTRoutemap=routemap]
   [PASSword=password] [SENdcommunity={NO|YES}]

ADD BGP PEer=ipadd POLICYTemplate=1..30 REMoteas=1..65534
   [AUthentication={MD5|NONE}] [DESCription=description]
   [EHOps={DEFault|1..255}] [PASSword=password]
```

Description

This command adds a BGP peer to the router. This command adds the peer in the disabled state; the router does not attempt to communicate with the peer until the **enable bgp peer** command on page 49-85 command is entered. This allows you to fully configure the peer entry before starting to communicate with it.

Parameter	Description	
PEer	The IP address of the new peer, in dotted decimal notation. This address should be the address that this router uses when communicating with the peer; that is, the address of the interface on the peer that is closest to this router. Default: no default	
REMoteas	The remote Autonomous System to which this peer belongs. If the remote AS number is the same as this router's AS number, the peer is an internal BGP (IBGP) peer. If the remote AS number is different from this router's AS number, the peer is an external BGP (EBGP) peer. If the remote AS numbers are different but the routers have the same confederation peer, the peer is a confederation BGP peer. The AS number is assigned by the IANA. Default: no default	
AUthentication	Whether to use MD5 authentication for the BGP peer. If you specify **md5**, you must also specify **password**. Default: **none**	
	MD5	An MD5 digest is added to every BGP packet sent over the TCP connection and is authenticated at the other end. If any part of the digest cannot be verified, the packet is dropped with no response sent.
	NONE	The BGP session is not authenticated.

Parameter	Description	
CONnectretry	The time interval between attempts to establish a BGP connection to the peer, in seconds. Default: **120**	
	0	The router does not repeat an attempt to establish a BGP connection.
	1..4294967295	The router waits the specified number of seconds between attempts.
	DEFault	The router waits 120 seconds between attempts.
DESCription	A description of the peer, which has no effect on its operation. A string 1 to 63 characters long. All printable characters are valid except the question mark and double quotes. If *description* contains spaces, the string must be in double quotes. Default: no default	
EHOps	The number of hops put in the *TTL* (Time To Live) field of BGP messages for external BGP. Normally, EBGP requires that BGP peers be connected to a common network, which means they are separated by a single hop. Setting **ehops** to a value greater than 1 indicates that multihop EBGP is allowed. Default: **1**	
	1..255	The specified number of hops is put into the TTL field.
	DEFault	The number of hops put in the TTL field is 1.
HOLdtime	The value in seconds that this router proposes for the time interval between reception of keepalive and/or update messages from this peer. The actual hold time used on a peer connection is negotiated when the connection is opened, as the lower of the hold times proposed. Default: **90**	
	0	This router proposes not to have a hold time on this BGP connection.
	3..65535	This router proposes the specified number of seconds as hold time.
	DEFault	This router proposes a hold time of 90 seconds.
INFilter	The IP routing filter that acts as a prefix filter to filter any prefixes advertised via incoming BGP update messages from this peer. You can use a prefix filter to exclude routes to particular networks from the update message. The filter must already exist. To create a filter use the **add ip filter** command on page 14-68 of Chapter 14, Internet Protocol (IP) and create a filter with a number from 300 to 399. If you specify more than one of **inpathfilter**, **infilter** and **inroutemap**, the router applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the router stops checking after the first filter entry that excludes the prefix, so a prefix is only included if all the applied filters result in it being included. Default: **none**	

Parameter	Description
INPathfilter	The AS path list that filters the BGP update messages from this peer. You can use an AS path list to exclude or include update messages that have traversed particular ASs or paths. If the path list does not already exist, it is created. To create a path list and/or add entries to it, use the **add ip aspathlist** command on page 49-62. If you specify more than one of **inpathfilter**, **infilter** and **inroutemap**, the router applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the router stops checking after the first filter entry that excludes the update, so an update is only included if all the applied filters result in it being included. Default: **none**
INRoutemap	The route map that filters and/or modifies prefixes from this peer. You can use a route map to include or exclude update messages or a subset of an update message's routes, on the basis of a range of BGP attributes, and/or to modify attributes. The route map must already exist. To create a route map use the **add ip routemap** command on page 49-68. If you specify more than one of **inpathfilter**, **infilter** and **inroutemap**, the router applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the router stops checking after the first filter entry that excludes the update, so an update is only included if all the applied filters result in it being included. Default: **none**
KEEpalive	The time in seconds that this router would prefer to leave between keepalive messages to this peer. This time should be one third of the **holdtime** parameter. The actual value used for the keep alive interval is determined once the BGP connection is opened, because the hold time interval is calculated as part of the BGP connection opening. The actual keep alive interval is calculated so that the ratio: configured keep alive interval: configured hold time interval is the same as the ratio: actual keep alive interval: negotiated hold time interval. If the hold time is negotiated at 0 seconds, then the keep alive interval is also 0 seconds, and keepalive messages are not sent. Default: one third of **holdtime**
1..21845	This router prefers the specified number of seconds as keepalive interval.
DEFault	This router prefers a keepalive interval of one third the hold time.
LOCal	The local interface. In certain circumstances, the router uses this address as the source for BGP packets it generates and sends to this BGP peer. For a description of when the router uses the local interface, see "How to Set the IP Address By Which the Router Identifies Itself" on page 49-38. Default: **none**

Parameter	Description	
MAXPREFIX	The maximum number of network prefixes that the router expects to receive from this peer. This parameter provides a safety mechanism in case the peer sends more prefixes than you might normally expect to receive. Default: **off**	
	1..4294967295	The maximum number of prefixes the router expects to receive from this peer. Once this number is exceeded, the action you specify in **maxprefixaction** is carried out.
	OFF	No maximum prefix checking.
MAXPREFIXAction	The action to take when a peer has sent a number of prefixes that exceeds the number specified by **maxprefix**. Default: **warning**	
	Warning	The router logs warnings when the maximum number of prefixes is exceeded.
	Terminate	The router resets the peer connections and logs warnings.
MINAsoriginated	The minimum time in seconds between advertisements, from the router to this peer, of routes that originate in the router's autonomous system. Default: **15**	
	0..3600	The interval is the specified number of seconds.
	DEFault	The interval is 15 seconds.
MINRouteadvert	The minimum time in seconds between advertisements, from the router to this peer, of routes that originate outside the router's autonomous system. Default: **30**	
	0..3600	The interval is the specified number of seconds.
	DEFault	The interval is 30 seconds.
NEXthopself	Whether this router advertises to this peer that the next hop for all routes is itself. Default: **no**	
	YES	All updates that the router sends to this peer specify this router as the next hop.
	NO	The next hop is specified as described in RFC 1771.
OUTFilter	The routing filter that acts as a prefix filter to filter the prefixes sent in BGP update messages to this peer. You can use a prefix filter to exclude routes to particular networks from the update message. The filter must already exist. To create a filter use the **add ip filter** command on page 14-68 of Chapter 14, Internet Protocol (IP) and create a filter with a number from 300 to 399. If you specify more than one of **outpathfilter**, **outfilter** and **outroutemap**, the router applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the router stops checking after the first filter entry that excludes the prefix, so a prefix is only included if all the applied filters result in it being included. Default: **none**	

Parameter	Description
OUTPathfilter	The AS path list that filters the BGP update messages sent to this peer. You can use an AS path list to exclude or include update messages that have traversed particular ASs or paths. If the path list does not already exist, it is created. To create a path list and/or add entries to it, use the **add ip aspathlist** command on page 49-62. If you specify more than one of **outpathfilter**, **outfilter** and **outroutemap**, the router applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the router stops checking after the first filter entry that excludes the update, so an update is only included if all the applied filters result in it being included. Default: **none**
OUTRoutemap	The route map that filters and/or modifies prefixes sent to this peer. You can use a route map to include or exclude update messages or a subset of an update message's routes, on the basis of a range of BGP attributes, and/or to modify attributes. The route map must already exist. To create a route map use the **add ip routemap** command on page 49-68. If you specify more than one of **outpathfilter**, **outfilter** and **outroutemap**, the router applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the router stops checking after the first filter entry that excludes the update, so an update is only included if all the applied filters result in it being included. Default: **none**
PASSword	The key used by the authentication algorithm. Two BGP peers can only communicate with each other if they have the same key. *password* is a character string from 1 to 80 characters long. All printable characters are valid except the question mark and double quotes. If *password* contains spaces, it must be in double quotes. Only valid if **authentication=md5** Default: no default
POLICYTemplate	The ID number of the peer policy template that applies to this peer. The specified policy template must already exist. To create a template, use the **add bgp peertemplate** command on page 49-57. You can only specify **remoteas**, **description**, **authentication**, **password**, and **ehops** at the same time as **policytemplate**. The template provides all other configuration values.

Parameter	Description	
SENdcommunity	Whether the router includes the community attribute in update messages that it sends to this peer. Default: no	
	YES	The community attribute is set in update messages to this peer. To set the value of the community attribute, create a route map with a **set** clause to set the community, and use the **outroutemap** parameter to apply it to update messages to this peer. To create a route map use the **add ip routemap** command on page 49-68.
	NO	The community attribute is not set in update messages to this peer, even if it is set in the route map used by the peer.

Tip The shortest string you can enter is shown in capital letters.

Examples

To add a BGP peer whose IP address is 192.168.1.1 and whose AS number is 54321, use the command:

```
add bgp pe=192.168.1.1 rem=54321 desc="test remote bgp peer"
```

Related Commands

add ip aspathlist
add ip filter in Chapter 14, Internet Protocol (IP)
add ip routemap
delete bgp peer
disable bgp peer
enable bgp peer
reset bgp peer
set bgp peer
show bgp peer

add bgp peertemplate

Syntax

```
ADD BGP PEERTemplate=1..30
   [CONnectretry={DEFault|0..4294967295}]
   [DESCription=description]
   [HOLdtime={DEFault|0|3..65535}]
   [INFilter={NONE|300..399}] [INPathfilter={NONE|1..99}]
   [INRoutemap=routemap] [KEEpalive={DEFault|1..21845}]
   [LOCal={NONE|1..15}] [MAXPREFIX={OFF|1..4294967295}]
   [MAXPREFIXAction={Terminate|Warning}]
   [MINAsoriginated={DEFault|0..3600}]
   [MINRouteadvert={DEFault|0..3600}]
   [NEXthopself={NO|YES}] [OUTFilter={NONE|300..399}]
   [OUTPathfilter={NONE|1..99}] [OUTRoutemap=routemap]
   [SENdcommunity={NO|YES}]
```

Description This command creates a template for use on BGP peers.

Parameter	Description	
PEERTemplate	The ID number of the template. Default: no default	
CONnectretry	The time interval between attempts to establish a BGP connection to peers that use the template, in seconds. Default: **120**	
	0	The router does not repeat an attempt to establish a BGP connection.
	1..4294967295	The router waits the specified number of seconds between attempts.
	DEFault	The router waits 120 seconds between attempts.
DESCription	A description for the peers that use the template, which has no effect on their operation. A string 1 to 63 characters long. All printable characters are valid except the question mark and double quotes. If *description* contains spaces, the string must be in double quotes. Default: no default	
HOLdtime	The value in seconds that this router proposes for the time interval between reception of keepalive and/or update messages from peers that use the template. The actual hold time used on a peer connection is negotiated when the connection is opened, and is the lower of the hold times proposed. Default: **90**	
	0	This router proposes not to have a hold time on this BGP connection.
	3..65535	This router proposes the specified number of seconds as hold time.
	DEFault	This router proposes a hold time of 90 seconds.

Parameter	Description
INFilter	The routing filter that acts as a prefix filter and filters any prefixes advertised via incoming BGP update messages from peers that use the template. You can use a prefix filter to exclude routes to particular networks from update messages. The filter must already exist. To create a filter use the **add ip filter** command on page 14-68 of Chapter 14, Internet Protocol (IP). The filter number must be in the range 300 to 399. If you specify more than one of **inpathfilter**, **infilter** and **inroutemap**, the router applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the router stops checking after the first filter entry that excludes the prefix, so a prefix is only included if all the applied filters result in it being included. Default: **none**
INPathfilter	The AS path list that filters the BGP update messages from peers that use the template. You can use an AS path list to exclude or include update messages that have traversed particular ASs or paths. If the path list does not already exist, it is created. To create a path list and/or add entries to it, use the **add ip aspathlist** command on page 49-62. If you specify more than one of **inpathfilter**, **infilter** and **inroutemap**, the router applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the router stops checking after the first filter entry that excludes the update, so an update is only included if all the applied filters result in it being included. Default: **none**
INRoutemap	The route map that filters and/or modifies prefixes from peers that use the template. You can use a route map to include or exclude update messages or a subset of an update message's routes, on the basis of a range of BGP attributes, and/or to modify attributes. The route map must already exist. To create a route map use the **add ip routemap** command on page 49-68. If you specify more than one of **inpathfilter**, **infilter** and **inroutemap**, the router applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the router stops checking after the first filter entry that excludes the update, so an update is only included if all the applied filters result in it being included. Default: **none**

Parameter	Description	
KEEpalive	The time in seconds that this router would prefer to leave between keepalive messages to peers that use the template. This time should be one third of the **holdtime** parameter. The actual value used for the keep alive interval is determined once the BGP connection is opened, because the hold time interval is calculated as part of the BGP connection opening. The actual keep alive interval is calculated so that the ratio: configured keep alive interval: configured hold time interval is the same as the ratio: actual keep alive interval: negotiated hold time interval. If the hold time is negotiated at 0 seconds, then the keep alive interval is also 0 seconds, and keepalive messages are not sent. Default: one third of **holdtime**	
	1..21845	This router prefers the specified number of seconds as keepalive interval.
	DEFault	This router prefers a keepalive interval of one third the hold time.
LOCal	The local interface. In certain circumstances, the router uses this address as the source for BGP packets it generates and sends to peers that use this template. For a description of when the router uses the local interface, see "How to Set the IP Address By Which the Router Identifies Itself" on page 49-38. Default: **none**	
MAXPREFIX	The maximum number of network prefixes that the router expects to receive from peers that use the template. This parameter provides a safety mechanism in case the peer sends more prefixes than you might normally expect to receive. Default: **off**	
	1..4294967295	The maximum number of prefixes the router expects to receive from peers that use the template. Once this number is exceeded, the action you specify in **maxprefixaction** is carried out.
	OFF	No maximum prefix checking.
MAXPREFIXAction	The action to take when a peer has sent a number of prefixes that exceeds the number specified by **maxprefix**. Default: **warning**	
	Warning	The router logs warnings when the maximum number of prefixes is exceeded.
	Terminate	The router resets the peer connections and logs warnings.
MINAsoriginated	The minimum time in seconds between advertisements, from the router to peers that use the template, of routes that originate in the router's autonomous system. Default: **15**	
	0..3600	The interval is the specified number of seconds.
	DEFault	The interval is 15 seconds.

Parameter	Description	
MINRouteadvert	The minimum time in seconds between advertisements, from the router to peers that use the template, of routes that originate outside the router's autonomous system. Default: **30**	
	0..3600	The interval is the specified number of seconds.
	DEFault	The interval is 30 seconds.
NEXthopself	Whether this router advertises to peers that use the template that the next hop for all routes is itself. Default: **no**	
	YES	All updates that the router sends to peers that use this template specify this router as the next hop.
	NO	The next hop is specified as described in RFC 1771.
OUTFilter	The routing filter that acts as a prefix filter and filters the prefixes sent in BGP update messages to peers that use this template. You can use a prefix filter to exclude routes to particular networks from the update message. The filter must already exist. To create a filter use the **add ip filter** command on page 14-68 of Chapter 14, Internet Protocol (IP) and create a filter with a number from 300 to 399. If you specify more than one of **outpathfilter**, **outfilter** and **outroutemap**, the router applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the router stops checking after the first filter entry that excludes the prefix, so a prefix is only included if all the applied filters result in it being included. Default: **none**	
OUTPathfilter	The AS path list that filters the BGP update messages sent to peers that use this template. You can use an AS path list to exclude or include update messages that have traversed particular ASs or paths. If the path list does not already exist, it is created. To create a path list and/or add entries to it, use the **add ip aspathlist** command on page 49-62. If you specify more than one of **outpathfilter**, **outfilter** and **outroutemap**, the router applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the router stops checking after the first filter entry that excludes the update, so an update is only included if all the applied filters result in it being included. Default: **none**	
OUTRoutemap	The route map that filters and/or modifies prefixes sent to peers that use this template. You can use a route map to include or exclude update messages or a subset of an update message's routes, on the basis of a range of BGP attributes, and/or to modify attributes. The route map must already exist. To create a route map use the **add ip routemap** command on page 49-68. If you specify more than one of **outpathfilter**, **outfilter** and **outroutemap**, the router applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the router stops checking after the first filter entry that excludes the update, so an update is only included if all the applied filters result in it being included. Default: **none**	

Parameter	Description	
SENdcommunity	Whether the router includes the community attribute in update messages that it sends to peers that use this template. Default: no	
	YES	The community attribute is set in update messages to peers that use this template. To set the value of the community attribute, create a route map with a **set** clause to set the community, and use the **outroutemap** parameter to apply it to update messages to peers that use this template. To create a route map use the **add ip routemap** command on page 49-68.
	NO	The community attribute is not set in update messages to this peer, even if it is set in the route map used by the peers that use this template.

Tip The shortest string you can enter is shown in capital letters.

Examples

To create a new peer policy template with a hold time of 30 seconds, and assign it to a peer, use the commands:

```
add bgp peert=1 hol=30
add bgp pe=192.168.1.0/24 policyt=1
```

Related Commands

add bgp peer
set bgp peer
set bgp peertemplate
show bgp peer

add ip aspathlist

Syntax

```
ADD IP ASPATHlist=1..99 [ENTry=1..4294967295]
   INCLude=aspath-reg-exp

ADD IP ASPATHlist=1..99 [ENTry=1..4294967295]
   EXCLude=aspath-reg-exp
```

Description

This command adds an entry to an AS path list, and creates the list if it does not already exist. You must specify the index number of the AS path list, and may also specify the position of the entry in the list.

When the router searches through an AS path list, the first entry that causes a match stops the search, returning the result **include** or **exclude** depending on the type of entry. A totally empty AS path list is identical to an AS path list that matches all AS paths and is of type **include**. Any non-empty AS path list has an implicit entry at the end that matches all AS paths and is of type **exclude**. For more information about using AS path lists, see "How to Configure AS Path Filters" on page 49-26.

Parameter	Description
ASPATHlist	The ID number of the AS path list. You can create up to 99 AS path lists. Default: no default
ENTry	The desired position of the new entry in the AS path list once the entry has been added. Entries are numbered from 1 to the number of entries in the list. Default: The entry is added to the end of the list
INCLude	An AS path regular expression, which specifies the AS path values that this entry includes in this AS path list. When you use the AS path list in a route map or filter, the map or filter carries out its specified action on update messages with a matching AS path attribute value. Regular expressions are a list of one or more AS numbers, separated by spaces. To match from the first number in the list, start the expression with the ^ character. To match the last number, end with the $ character. If the expression contains spaces, surround it with double quotes. For more information about valid syntax, see Table 49-8 on page 49-27. For example: • **include="23334 45634 88988"** includes any path containing these numbers • **include="^23334 45634 88988$"** includes only that exact path • **include=^23334** includes any path that begins with 23334 Default: no default

Parameter	Description
EXCLude	An AS path regular expression, which specifies the AS path values that this entry excludes from this AS path list. When you use the AS path list in a route map or filter, the map or filter does not carry out its specified action on update messages with a matching AS path attribute value. Regular expressions are a list of one or more AS numbers, separated by spaces. To match from the first number in the list, start the expression with the ^ character. To match the last number, end with the $ character. If the expression contains spaces, surround it with double quotes. For more information about valid syntax, see Table 49-8 on page 49-27. For example: • **exclude="23334 45634 88988"** excludes any path containing these numbers • **exclude="^23334 45634 88988$"** excludes only that exact path • **exclude=23334$** excludes any path that ends with 23334 Default: no default
Tip The shortest string you can enter is shown in capital letters.	

Examples

To add an entry to AS path list 1 that matches all AS paths and excludes them, use the command:

```
add ip aspath=1 excl=.*
```

To add an entry to AS path list 2 that matches an empty AS path and includes it, use the command:

```
add ip aspath=2 incl=^$
```

Related Commands

add ip routemap
delete ip aspathlist
show ip aspathlist

add ip communitylist

Syntax

```
ADD IP COMmunitylist=1..99 [ENTry=1..4294967295]
   INCLude={INTernet|NOExport|NOAdvertise|
   NOEXPORTSubconfed|1..4294967295}[,...]

ADD IP COMmunitylist=1..99 [ENTry=1..4294967295]
   EXCLude={INTernet|NOExport|NOAdvertise|
   NOEXPORTSubconfed|1..4294967295}[,...]
```

Description This command adds an entry to a community list, and creates the list if it does not already exist. You must specify the index number of the community list, and may also specify the position of the entry in the list.

Parameter	Description	
COMmunitylist	The ID number of the community list. You can create up to 99 lists. Default: no default	
ENTry	The desired position of the new entry in the community list once the entry has been added. Entries are numbered from 1 to the number of entries in the list. Default: The entry is added to the end of the list	
INCLude	A community name, community number, or comma-separated list of names and numbers, which specifies the communities that this entry includes in this community list. When you use the community list in a route map or filter, the map or filter carries out its specified action on update messages with a matching community attribute value. Default: no default	
	INTernet	The community of routes that can be advertised to all BGP peers.
	NOExport	The community of routes that must not be advertised outside a BGP confederation boundary (a standalone autonomous system that is not part of a confederation should be considered a confederation itself).
	NOAdvertise	The community of routes that must not be advertised to other BGP peers.
	NOEXPORTSubconfed	The community of routes that must not be advertised to external BGP peers (this includes peers in other members' autonomous systems inside a BGP confederation).
	1..4294967295	The number of a community.

Parameter	Description	
EXCLude	A community name, community number, or comma-separated list of names and numbers, which specifies the communities that this entry excludes from this community list. When you use the community list in a route map or filter, the map or filter does not carry out its specified action on update messages with a matching community attribute value. Default: no default	
	INTernet	The community of routes that can be advertised to all BGP peers.
	NOExport	The community of routes that must not be advertised outside a BGP confederation boundary (a standalone autonomous system that is not part of a confederation should be considered a confederation itself).
	NOAdvertise	The community of routes that must not be advertised to other BGP peers.
	NOEXPORTSubconfed	The community of routes that must not be advertised to external BGP peers (this includes peers in other members' autonomous systems inside a BGP confederation).
	1..4294967295	The number of a community.
Tip The shortest string you can enter is shown in capital letters.		

Examples

To add an entry to community list 1 that matches communities attributes that contain the communities NOEXPORT and 70000 and excludes them, use the command:

```
add ip com=1 excl=noe 70000
```

Related Commands

add ip routemap
delete ip communitylist
show ip communitylist

add ip prefixlist

Syntax

```
ADD IP PREFIXList=name ENTry=1..65535
   [ACTion={MATch|NOMatch}] [MASklength=range]
   [PREfix=ipadd]
```

Description

This command adds a numbered *entry* to a prefix list. If the prefix list does not already exist, this command first creates it. You can create up to 400 prefix lists, with up to 1000 entries in each list.

Parameter	Description
PREFIXList	A name to identify the prefix list. A string 1 to 15 characters long. Valid characters are uppercase letters (A-Z), lowercase letters (a-z), digits (0-9) and the underscore character ("_"). If *name* contains spaces, it must be in double quotes. Default: no default
ENTry	An integer to specify the position of the new entry in the prefix list. When a BGP peer uses a prefix list, it checks the entries in order, starting with the lowest, until it finds a match. Therefore, give more specific entries lower numbers than general entries. If you leave gaps between entry numbers, you can add future entries between existing entries. Each prefix list has an implicit final entry that matches all addresses, with an action of **nomatch**. Default: no default
ACTion	Whether matching prefixes are included or excluded by the process that is using the prefix list. You can use multiple entries in a prefix list with actions of **match** and **nomatch** to build up a list of prefixes. Prefixes with **action=match** are included in the list. Then to use this list of prefixes, create a route map that matches it and apply the route map to a peer. The route map also has an **action** parameter, which determines whether the peer includes or excludes the prefixes in the list. Default: **match**
MATch	The prefix list includes the prefix.
NOMatch	The prefix list excludes the prefix.

Parameter	Description
MASklength	The range of prefix mask lengths matched by this entry in the prefix list. The *range* is either a single CIDR mask from 0 to 32, or two masks separated by a hyphen. These options are valid for setting the mask length: • as a mask length range (**masklength=*a*-*b*)**. For a route to match against this entry, its prefix mask length must be between *a* and *b* inclusive. *a* must be less than *b*. • as a single mask length (**masklength=*a***). For a route to match against this entry, its prefix mask length must be exactly *a*. • as an implicit mask length, by not specifying **masklength** (for example, **prefix=192.168.0.0**). For a route to match against this entry, its prefix mask length must correspond exactly to the mask for the class of the given address; in this example, 24. Default: The natural mask for the prefix, based on whether it is a class A, B, or C network
PREfix	The network address matched by this entry in the prefix list, specified in dotted decimal notation. If you do not specify a prefix, the router sets it to 0.0.0.0. This is correct if you are matching all routes or the default route. Default: 0.0.0.0
Tip The shortest string you can enter is shown in capital letters.	

Examples

To match only routes from the 192.168.0.0/16 network, use the command:

```
add ip prefixlist=sample1 entry=1 action=match
   prefix=192.168.0.0 masklength=16
```

To match all routes in all 192.168.0.0 networks, except those in the 192.168.7.0 network, use the commands:

```
add ip prefixlist=sample2 entry=1 action=nomatch
   prefix=192.168.7.0 masklength=24-32
```

```
add ip prefixlist=sample2 entry=2 action=match
   prefix=192.168.0.0 masklength=16-32
```

To exclude the default route, use the command:

```
add ip prefixlist=sample3 entry=1 action=nomatch masklength=0
```

To include all routes, use the command:

```
add ip prefixlist=sample4 entry=1 action=match
   masklength=0-32
```

Related Commands

add ip routemap
delete ip prefixlist
set ip routemap
show ip prefixlist

add ip routemap

Syntax for an empty entry

```
ADD IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion={INCLude|EXCLude}]
```

Syntax for a *match* clause

```
ADD IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion={INCLude|EXCLude}] MAtch ASPath=1..99

ADD IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion={INCLude|EXCLude}] MAtch COMmunity=1..99
   [EXAct={NO|YES}]

ADD IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion={INCLude|EXCLude}] MAtch NEXThop=ipadd

ADD IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion={INCLude|EXCLude}] MAtch
   ORIGin={EGP|IGP|INCOmplete}

ADD IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion={INCLude|EXCLude}] MAtch PREFIXList=name

ADD IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion={INCLude|EXCLude}] MAtch TAG=1..65535
```

Syntax for a *set* clause

```
ADD IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion=INCLude] SET ASPath={1..65534[,...]}

ADD IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion=INCLude] SET
   COMmunity={NOExport|NOAdvertise|NOEXPORTSubconfed|
   1..4294967295}[,...]} [ADD={NO|YES}]

ADD IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion=INCLude] SET LOCalpref=0..4294967295

ADD IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion=INCLude] SET MED={0..4294967295|REMOVE}

ADD IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion=INCLude] SET ORIGin={IGP|EGP|INCOmplete}
```

Description

This command adds a numbered *entry* to a route map, or adds a *clause* to an existing entry in a route map. If the route map does not already exist, this command first creates it.

Route maps are made up of a list of entries. Each entry contains:

- zero or one **match** clause, to determine which update messages the entry applies to. If an entry does not have a match clause, the effect is that it matches everything.
- one **action**, to determine whether matching update messages are included or excluded by the process that is using the route map (by default matching items are included).
- zero, one, or more **set** clauses. Most **set** clauses change the attributes of matching update messages. Each entry can have at most one **set** clause of a given type. For example, you can only set the MED once in an entry.

Parameters for both *match* and *set* clauses

Parameter	Description
ROUTEMap	The name of the route map to add the entry or clause to. The *routemap* is a character string 0 to 15 characters long. Valid characters are uppercase and lowercase letters, digits (0-9), and the underscore character. Default: no default
ENTry	An integer to specify the position of the new entry in the route map. When a BGP peer uses a route map, it checks the entries in order, starting with the lowest, until it finds a match. If you leave gaps between entry numbers, you can add future entries between existing entries. Be careful when specifying the entry number. If you make an error in the number (for example, enter entry=11 instead of entry=1), the router adds a new entry to the route map. Default: no default
ACtion	Whether matching prefixes or update messages are included or excluded by the process that is using the route map. The **action** parameter applies to the entire entry, but you can change it at the same time as you add a clause. The most recently entered value of this parameter applies to the entire entry. It is not meaningful to have **action=exclude** in an entry with a set clause. Default: the current setting. If there is no current setting, **include**

Tip The shortest string you can enter is shown in capital letters.

Parameters for *match* clauses

Parameter	Description
MAtch	Adds a **match** clause to the entry, to determine which update messages the entry applies to. A route map entry can have zero or one **match** clauses. A entry without a **match** clause matches all update messages.
ASPath	The ID number of an AS path list. An update message matches the route map entry if its AS path attribute matches the AS path list. To configure an AS path list use the **add ip aspathlist** command on page 49-62. Default: no default
COMmunity	The ID number of a community list. An update message matches the route map entry if its community attribute matches the community list.To configure a community list use the **add ip communitylist** command on page 49-64. Default: no default

Parameter	Description	
EXAct	Whether the community attribute in an update message must precisely match the route map's community list. Only valid when you specify both **match** and **community**. Default: **no**	
	YES	An update message only matches the route map entry if its community attribute contains all the communities specified in the community list and only those communities.
	NO	An update message still matches the route map entry if its community attribute contains all the communities specified in the community list plus extra communities.
NEXThop	The IP address of the next node in the path to the route's destination, specified in dotted decimal notation. An update message matches the route map entry if its next_hop attribute matches this address. Default: no default	
ORIGin	An origin attribute value, which indicates BGP's source for the routes at their originating AS. An update message matches the route map entry if its origin attribute matches this value. Default: no default	
	IGP	The original source of the route was IGP.
	EGP	The original source of the route was EGP.
	INCOmplete	The original source of the route was neither IGP or EGP. This includes statically-configured routes.
PREFIXList	The name of a prefix list. A route matches the route map entry if the prefix list contains that route. To create a list use the **add ip prefixlist** command on page 49-66. Default: no default	
TAG	A tag that identifies a particular static route. A route matches this route map entry if it has been tagged with this value by using the **tag** parameter of the **add ip route** command on page 14-88 of Chapter 14, Internet Protocol (IP). You can only use a route map that matches on **tag** when you use the **add bgp network** and **add bgp import** commands to import static routes from IP to BGP. You cannot use **tag** to filter routes that are sent to BGP peers or to match update messages that are received from BGP peers. Default: no default	

Tip The shortest string you can enter is shown in capital letters.

Parameters for *set* clauses

Parameter	Description	
SET	Adds a **set** clause to the entry, to modify an attribute in update messages that match the entry. A route map entry can have zero, one or more **set** clauses, but can only modify each attribute once. A entry without a **set** clause does not modify any attributes.	
ASPath	A comma-separated list of 1 to 10 AS numbers. These numbers are added to the beginning of the update message's AS path attribute. Default: no default	
COMmunity	A comma-separated list of 1 to 10 communities, identified by name or number. If the **add** parameter is **yes**, these communities are added to the update message's community attribute. If the **add** parameter is **no** (its default), these communities replace the update message's community attribute. Note that you must also set the peer's **sendcommunity** parameter to **yes** if you want the peer to include the community attribute in the update messages it sends. By default, peers do not include the community attribute in outgoing updates. Default: no default	
	INTernet	The community of routes that can be advertised to all BGP peers.
	NOExport	The community of routes that must not be advertised outside a BGP confederation boundary (a standalone autonomous system that is not part of a confederation should be considered a confederation itself).
	NOAdvertise	The community of routes that must not be advertised to other BGP peers.
	NOEXPORTSubconfed	The community of routes that must not be advertised to external BGP peers (this includes peers in other members' autonomous systems inside a BGP confederation).
	1..4294967295	The number of a community.
ADD	Whether the list of communities specified by the **community** parameter is added to the community attribute, or replaces the community attribute. Only valid when you specify both **set** and **community**. Default: **no**	
	YES	The communities are added to the update message's community attribute.
	NO	The communities replace the update message's community attribute.
LOCalpref	The metric to write into the update message's local_preference attribute. IBGP uses the local preference to determine which path it should use inside the AS to reach the advertised prefix. A lower metric indicates a preferred path. EBGP does not use this attribute. Default: no default	

Parameter	Description	
MED	The metric to write into the update message's Multi_Exit_Discriminator attribute. EBGP uses the MED to determine the optimal path for reaching the advertised prefixes. A lower metric indicates a preferred path. IBGP does not use this attribute. Default: no default	
	0..4294967295	This value is written into the MED attribute of the matched update message.
	REMOVE	The MED attribute is removed from the matched update message.
ORIGin	The value to write into the update message's origin attribute. The origin indicates BGP's source for the routes at their originating AS. Default: no default	
	IGP	The original source of the route was IGP.
	EGP	The original source of the route was EGP.
	INCOmplete	The original source of the route was neither IGP or EGP. This includes statically-configured routes.
Tip The shortest string you can enter is shown in capital letters.		

Examples

To add a route map entry that sets the community attribute to 489816064 for all BGP routes, use the command:

```
add ip routem=set_comm ent=10 set com=489816064
```

This command creates the route map, adds an entry to it, and adds a set clause to the entry. No match clause is required because we wish to match all routes. To use this route map for routes being sent to BGP peer 192.168.1.1, use the command:

```
set bgp peer=192.168.1.1 outr=set_comm
```

Related Commands

delete ip routemap
set ip routemap
show ip routemap

delete bgp aggregate

Syntax `DELete BGP AGGregate=`*`prefix`* `[MASK=`*`ipadd`*`]`

Description This command deletes an aggregate entry from BGP. BGP no longer advertises the aggregate entry. If the aggregate entry is currently in the BGP route table, BGP also sends an update message to all peers to withdraw the route. Associated aggregate suppressed routes are then reconsidered for advertisement.

Parameter	Description
AGGregate	The network prefix of the aggregate entry to delete. This is expressed as the base IP address of the network, in dotted decimal notation, optionally followed by a "/" character and the number of bits in the network mask. If you do not specify the CIDR mask, the router uses the value from the **mask** parameter, if present, or otherwise the natural mask for the network, based on whether it is a class A, B, or C network. Default: no default
MASK	The network mask for the aggregate entry. This parameter is provided for compatibility with other router commands that specify an IP address and mask; we recommend that you instead specify the mask in the **aggregate** parameter. If you specify a mask in this parameter and the **aggregate** parameter, an error results unless the two masks agree. Default: The natural mask for the network, based on whether it is a class A, B, or C network.

Tip The shortest string you can enter is shown in capital letters.

Examples To delete the aggregate entry for the network 192.168.8.0/21, use the command:

```
del bgp agg=192.168.8.0/21
```

Related Commands

add bgp aggregate
set bgp aggregate
show bgp aggregate

delete bgp confederationpeer

Syntax `DELete BGP CONFEDerationpeer=1..65534`

Description This command deletes an Autonomous System from the AS confederation to which this router belongs. An AS confederation is a group of Autonomous Systems that communicate between themselves using confederation BGP, and communicate to Autonomous Systems outside the confederation as if they were a single Autonomous System. For more information about AS confederations, see "AS Confederations" .

The **confederationpeer** parameter specifies the number of an Autonomous System that is no longer to be treated as one of the Autonomous Systems in the confederation. The specified AS number must be an already existing AS confederation peer.

Examples To remove AS 60003 from the AS confederation to which this router belongs, use the command:

```
del bgp confed=60003
```

Related Commands
add bgp confederationpeer
set bgp
show bgp confederation

delete bgp import

Syntax `DELete BGP IMPort={INTerface|OSPF|RIP|STAtic}`

Description This command deletes an import entry from BGP. Routes from the source specified are no longer imported into the BGP route table. Routes already in the BGP route table are removed.

Parameter	Description	
IMPort	The source of routing information for the routes that are no longer to be imported into BGP. Default: no default	
	INTerface	Stops the import of interface routes.
	OSPF	Stops the import of OSPF routes.
	RIP	Stops the import of RIP or RIP2 routes.
	STAtic	Stops the import of statically configured routes.
Tip The shortest string you can enter is shown in capital letters.		

Examples To stop importing OSPF routes into BGP, use the command:

```
del bgp imp=ospf
```

Related Commands

add bgp import
set bgp import
show bgp import

delete bgp network

Syntax `DELete BGP NETwork=`*`prefix`* `[MASK=`*`ipadd`*`]`

Description This command deletes a network from the list of networks that the router can advertise to remote BGP peers. If the router had previously advertised the route, BGP sends an update message to all peers to withdraw the network.

Parameter	Description
NETwork	The network to remove from the list of networks that can be advertised. This is expressed as the base IP address of the network, in dotted decimal notation, optionally followed by a "/" character and the number of bits in the network mask. If you do not specify the CIDR mask, the router uses the value from the **mask** parameter, if present, or otherwise the natural mask for the network, based on whether it is a class A, B, or C network. Default: no default
MASK	The network mask for the network. This parameter is provided for compatibility with other router commands that specify an IP address and mask; we recommend that you instead specify the mask in the **network** parameter. If you specify a mask in this parameter and the **network** parameter, an error results unless the two masks agree. Default: The natural mask for the network, based on whether it is a class A, B, or C network

Tip The shortest string you can enter is shown in capital letters.

Examples To delete the network 192.169.2.0 from the list of networks advertised by BGP, use the command:

```
del bgp net=192.169.2.0/24
```

Related Commands
add bgp network
show bgp network

delete bgp peer

Syntax `DELete BGP PEer=`*`ipadd`*

Description This command deletes a BGP peer from the router. The BGP peer must be in a disabled state: either never enabled, or previously enabled and subsequently disabled with the **disable bgp peer** command on page 49-83.

The **peer** parameter specifies the IP address of the peer to be deleted, in dotted decimal notation. The peer must be an existing BGP peer on this router.

Examples To delete a BGP peer whose IP address is 192.168.1.1, use the command:

```
del bgp pe=192.168.1.1
```

Related Commands **disable bgp peer**
show bgp peer

delete bgp peertemplate

Syntax `DELete BGP PEERTemplate=1..30`

Description This command deletes an existing BGP peer policy template from the router. All peers that have been assigned the specified peer template receive their own copies of the current peer template settings. You can subsequently modify these peers.

The **peertemplate** parameter specifies the ID number of the template to be deleted.

Examples To delete BGP peer template 1, use the command:

```
del bgp peert=1
```

Related Commands **add bgp peertemplate**
show bgp peer

delete ip aspathlist

Syntax

```
DELete IP ASPATHlist=1..99 [ENTry=1..4294967295]
```

Description

This command deletes an entry from an AS path list or deletes an entire AS path list. You cannot delete an AS path list if a route map is using it, or if a peer is using it as a filter. First use the **match** parameter of the **delete ip routemap** command on page 49-80 to delete the route map entry, or the **set bgp peer** command on page 49-94 to remove the filter association.

Parameter	Description
ASPATHlist	The ID number of the AS path list to delete, or to remove an entry from. Default: no default
ENTry	The number of the entry to delete. If you do not specify an entry, the whole AS path list is deleted. Default: no default

Tip The shortest string you can enter is shown in capital letters.

Examples

To delete the third entry in AS path list 1, use the command:

```
del ip aspath=1 ent=3
```

To delete AS path list 1 and all its entries, use the command:

```
del ip aspath=1
```

Related Commands

add ip aspathlist
show ip aspathlist

delete ip communitylist

Syntax `DELete IP COMmunitylist=1..99 [ENTry=1..4294967295]`

Description This command deletes an entry from a community list or the entire list. You cannot delete a community list if a route map is using it. First use the **match** parameter of the **delete ip routemap** command on page 49-80 to delete the route map entry.

Parameter	Description
COMmunitylist	The ID number of the community list to delete, or to remove an entry from. Default: no default
ENTry	The number of the entry to delete. If you do not specify an entry, the whole community list is deleted. Default: no default

Tip The shortest string you can enter is shown in capital letters.

Examples To delete the entire community list 1, use the command:

```
del ip com=1
```

Related Commands
add ip communitylist
show ip communitylist

delete ip prefixlist

Syntax `DELete IP PREFIXList[=name] [ENTry=1..65535]`

Description This command deletes:

- an entry from a particular prefix list if you specify a name in the **prefixlist** parameter and an **entry** number
- a prefix list if you specify a name in the **prefixlist** parameter but do not specify an **entry** number
- all prefix lists if you do not specify a name in the **prefixlist** parameter or an **entry** number

You cannot delete a prefix list if a route map is using it. Delete the route map entry first.

Examples To delete entry 2 from the prefix list “office”, use the command:

```
del ip prefixl=office entry=2
```

Related Commands
add ip prefixlist
delete ip routemap
set ip routemap
show ip prefixlist

delete ip routemap

Syntax

```
DELete IP ROUTEMap=routemap

DELete IP ROUTEMap=routemap ENTry=1..4294967295

DELete IP ROUTEMap=routemap ENTry=1..4294967295
   MAtch={ASPath|COMmunity|NEXThop|ORIGin|PREFIXList|TAG}

DELete IP ROUTEMap=routemap ENTry=1..4294967295
   SET={ASPath|COMmunity|LOCalpref|MED|ORIGin|TAG}
```

Description

This command deletes one of:

- an entire route map
- a single entry in a route map, or
- a match or set clause in an entry in a route map

For information on route maps, see "Route Maps" on page 49-13 and "How to Configure Route Maps" on page 49-29.

You cannot delete a whole route map if a BGP peer is using it, or if an aggregate, network or import process is using it.

Parameter	Description
ROUTEMap	The name of the route map to be deleted or the name of the route map from which an entry, match clause, or set clause is to be deleted. Default: no default
ENTry	The number of the entry in the route map to be deleted, or the number of the entry from which a match clause or set clause is to be deleted. The entry must already exist in the route map. If you do not specify an entry, the whole route map is deleted. Default: no default
MAtch	The type of match clause to be deleted from the route map entry. Since only one match clause is allowed in a route map entry, this uniquely identifies the clause. Default: no default
SET	The type of set clause to be deleted from the route map entry. Since only one set clause of each type is allowed in a route map entry, this uniquely identifies the clause. Default: no default
Tip The shortest string you can enter is shown in capital letters.	

Examples

To delete the localpref set clause from entry 10 in route map "set_loc_pref", use the command:

```
del ip routem=set_loc_pref ent=10 set=loc
```

Related Commands

add ip routemap
set ip routemap
show ip routemap

disable bgp autosoftupdate

Syntax `DISable BGP AUTOsoftupdate`

Description This command disables automatic updating of modified BGP peers. Changes to a peer take effect only when the peer next receives or sends an update message, unless you manually trigger the update by using the **reset bgp peer soft** command on page 49-86. Automatic updating is disabled by default.

Examples To disable automatic updating, use the command:

```
dis bgp auto
```

Related Commands

reset bgp peer
set bgp peer
show bgp peer

disable bgp debug

Syntax

```
DISable BGP DEBug[={ALL|MSG|STAte|UPdate}[,...]]
   [PEer=ipadd]
```

Description

This command disables one or more forms of BGP debugging, optionally for a given BGP peer.

You can direct BGP debugging to only one manager device at a time. This means that if someone is debugging BGP on another terminal device, you cannot enable debugging on the current terminal device. However, you can use this command to disable debugging for the other device, and then enable debugging for the current device.

Parameter	Description	
DEBug	The debugging options to disable; a single option or a comma-separated list of options. Default: **all**	
	ALL	All debugging options.
	MSG	Message reception and transmission. There are four message types: open, update, keepalive, and notify. The output of the debug messages consists of the timestamp, the direction of the message, incoming or outgoing, the IP address of the peer, the message type and the details. For open, keepalive, and notify messages, the details consist of the decoded contents of the packet. For update messages, the details consist of the lengths of the withdrawn routes and NRLI fields, and a complete decode of the path attributes.
	STAte	State machine events and transitions. The output of the debug messages consists of the timestamp, the IP address of the peer, the event that causes the state change, and the old and new states.
PEer	The IP address of the peer for which debugging is no longer required, in dotted decimal notation. The peer must be an existing BGP peer on this router. Default: no default (debugging is turned off for all BGP peers)	

Tip The shortest string you can enter is shown in capital letters.

Examples

To disable all debugging for all BGP peers, use the command:

```
dis bgp deb
```

Related Commands

enable bgp debug

disable bgp peer

Syntax `DISable BGP PEer={ALL|`*ipadd*`}`

Description This command disables a given BGP peer, or all BGP peers. The router destroys its TCP connection to the peer, and the associated BGP session. The router and the peer withdraw any routes they learned during that session.

The **peer** parameter specifies the IP address of the peer to disable, in dotted decimal notation. If you specify **all**, all BGP peers are disabled.

Examples To disable the BGP peer 192.168.1.1, use the command:

```
dis bgp pe=192.168.1.1
```

Related Commands **enable bgp peer**
show bgp peer

enable bgp autosoftupdate

Syntax `ENAble BGP AUTOsoftupdate`

Description This command enables the router to automatically update BGP peers after you modify them. Automatic updating is disabled by default.

Examples To enable automatic updating, use the command:

```
ena bgp auto
```

Related Commands **reset bgp peer**
set bgp peer
show bgp peer

enable bgp debug

Syntax `ENAble BGP DEBug={ALL|MSG|STAte|UPdate}[,...] [PEer=`*`ipadd`*`]`

Description This command enables one or more forms of BGP debugging, optionally for a given BGP peer.

You can direct BGP debugging to only one manager device at a time. This means that if someone is debugging BGP on another terminal device, you cannot enable debugging on the current terminal device. However, you can use the **disable bgp debug** command on page 49-82 to disable debugging for the other device, and then enable debugging for the current device.

Parameter	Description	
DEBug	The debugging options to enable; a single option or a comma-separated list of options. Default: **all**	
	ALL	All debugging options.
	MSG	Message reception and transmission. There are four message types: open, update, keepalive, and notify. The output of the debug messages consists of the timestamp, the direction of the message, incoming or outgoing, the IP address of the peer, the message type and the details. For open, keepalive, and notify messages, the details consist of the decoded contents of the packet. For update messages, the details consist of the lengths of the withdrawn routes and NRLI fields, and a complete decode of the path attributes.
	STAte	State machine events and transitions. The output of the debug messages consists of the timestamp, the IP address of the peer, the event that causes the state change, and the old and new states.
PEer	The IP address of the peer for which debugging is required, in dotted decimal notation. Default: no default (debugging is turned on for all BGP peers)	

Tip The shortest string you can enter is shown in capital letters.

Examples To enable message and state debugging for BGP peer 192.168.1.1, use the command:

```
ena bgp deb=msg,state peer=192.168.1.1
```

Related Commands **disable bgp debug**

enable bgp peer

Syntax `ENAble BGP PEer={ALL|`*ipadd*`}`

Description This command enables a specific BGP peer or all BGP peers. The router initialises BGP resources, initiates a TCP connection to the peer, and listens for connections initiated by the peer.

The **peer** parameter specifies the IP address of the peer to enable, in dotted decimal notation. The peer must already exist. If you specify **all**, all BGP peers are enabled.

Examples To enable all BGP peers, use the commands:

```
ena bgp pe=all
```

Related Commands
disable bgp peer
reset bgp peer
show bgp peer

reset bgp peer

Syntax `RESET BGP PEer={ALL|`*ipadd*`}`

Description This command resets a specific BGP peer or all BGP peers. This is the equivalent of disabling the peer, then immediately enabling it. The router destroys its TCP connection to the peer and the associated BGP session. The router and the peer withdraw routes they learned from that session. Then the router initialises BGP resources, initiates a TCP connection to the peer, and listens for connections initiated by the peer.

The **peer** parameter specifies the IP address of the peer to reset, in dotted decimal notation. If you specify **all**, all BGP peers are reset.

Examples To reset BGP peer 192.168.1.1, use the command:

```
reset bgp pe=192.168.1.1
```

Related Commands
enable bgp peer
disable bgp peer
show bgp peer

reset bgp peer soft

Syntax `RESET BGP PEer={ALL|`*ipadd*`} SOft={IN|OUt|ALl}`

Description This command updates all BGP peers or a specific one after you have modified the peer. You do not need to use this command if automatic updating is enabled (see the **enable bgp autosoftupdate** command on page 49-83).

Parameter	Description	
PEer	The peer or peers to update. Default: no default	
	ALL	All peers are updated.
	ipadd	Only the specified peer is updated. The peer is identified by its IP address in dotted decimal notation.
SOft	The direction to update. Default: no default	
	IN	Updates routes that the peer receives. To trigger this update, the peer sends a Route Refresh message to the remote peers it receives routes from. The Route Refresh message triggers the remote peers to resend a BGP Update message.
	OUt	Updates routes that the peer sends. To reset these, the peer simply sends a BGP Update message to the affected BGP peers.
	ALl	Updates routes the peer sends and receives.

Tip The shortest string you can enter is shown in capital letters.

Examples To update BGP peer 192.168.1.1 after you have changed the filter it uses on incoming routes, use the command:

```
reset bgp pe=192.168.1.1 soft=in
```

Related Commands
enable bgp peer
disable bgp peer
show bgp peer

set bgp

Syntax

```
SET BGP [CONfederationid={NONE|1..65534}]
   [LOCalpref={DEFault|0..4294967295}]
   [MED={NONE|0..4294967294}] [PREFExt={DEFault|1..255}]
   [PREFInt={DEFault|1..255}] [ROuterid=ipadd]
   [SELEction_timer=3..60] [TABlemap[=routemap]
```

Description This command sets global BGP parameters in the router.

Parameter	Description
CONfederationid	The AS number of the AS confederation to which this router belongs. This AS number is used as the AS number for this router when communicating with BGP peers outside the confederation. A peer is outside the confederation if its AS number is different from this router's AS number, and it is not one of this router's confederation peers. For more information about AS confederations, see "AS Confederations" on page 49-15. You need to specify this parameter if the router has peer relationships with any BGP routers outside the confederation. When you create the peer relationship using the **add bgp peer** command on page 49-51, specify the peer's confederation ID in the **remoteas** parameter. Default: **none**
LOCalpref	The local preference value used in all update messages sent to internal peers when this is not overridden by changes due to the actions in a route map. The local preference is used in internal BGP to decide which BGP route to put into the main routing table. The route with the highest value of local preference is used. Default: **100** DEFault — The local preference is 100. 0..4294967295 — The local preference is the specified number.
MED	The Multi-Exit Discriminator (MED) value that is placed in update messages to external BGP peers from this router when not overridden by the action of a route map. The MED attribute is used by routers in one AS to distinguish between different exit points in the same neighbouring AS. A route with a lower value of MED is used, all other things being equal. For more information about the use of MED in route selection, see "BGP Route Selection" on page 49-8. Default: **none** (no MED attribute is put in update messages)

Parameter	Description
PREFExt	The route preference BGP gives to routes that it learns from external peers. The router uses routes with a lower preference value before routes with a higher preference. This parameter has been superseded by the **set ip route preference** command on page 14-160 of Chapter 14, Internet Protocol (IP), but is still accepted for backwards compatibility. The **set ip route preference** command allows a wider range of preference values. When you save your configuration using the **create config** command on page 1-70 of Chapter 1, Operation converts the **prefext** parameter to a **set ip route preference** command. Default: **170** DEFault — The route preference is 170. 1..255 — The route preference is the specified number.
PREFInt	The route preference BGP gives to routes that it learns from internal peers. The router uses routes with a lower preference value before routes with a higher preference. This parameter has been superseded by the **set ip route preference** command on page 14-160 of Chapter 14, Internet Protocol (IP), but is still accepted for backwards compatibility. The **set ip route preference** command allows a wider range of preference values. When you save your configuration using the **create config** command on page 1-70 of Chapter 1, Operation converts the **prefint** parameter to a **set ip route preference** command. Default: **170** DEFault — The route preference is 170. 1..255 — The route preference is the specified number.
ROuterid	A 4-byte number that uniquely identifies the router in a network system in certain circumstances, specified as an IP address in dotted decimal notation. For a description of when the router uses the router ID, see "How to Set the IP Address By Which the Router Identifies Itself" on page 49-38. Default: The default local interface's IP address, if it is configured. Otherwise, the highest interface IP address on the router.
TABlemap	A route map for BGP routes to be passed through before installing the route into the routing table. If you specify **tablemap** without specifying a routemap, you clear the **tablemap** setting and the router does not pass routes through a route map before writing them into the routing table. Default: no default (the router does not pass routes through a route map before writing them into the routing table)
SELEction_timer	The time in seconds that the router waits, after changes to its BGP routing information database, before it determines whether the changes need to be propagated to its BGP peers. By deferring the operation, multiple changes may be able to be aggregated into a single update. Therefore accepting slightly longer convergence times may reduce the BGP messaging overhead. Default: **15** seconds.

Tip The shortest string you can enter is shown in capital letters.

Examples To set the preference of routes learned by internal BGP to be better than the preference of routes learned by OSPF (which is 10), use the command:

```
set bgp prefi=9
```

Related Commands

add bgp confederationpeer
delete bgp confederationpeer
show bgp
show bgp confederation

set bgp aggregate

Syntax

```
SET BGP AGGregate=prefix [MASK=ipadd] [SUMmary={NO|YES}]
   [ROUTEMap[=routemap]]
```

Description

This command modifies the parameters of an existing aggregate entry in the BGP route table.

Parameter	Description	
AGGregate	The aggregate entry to modify, which is identified by its network prefix. This is expressed as the base IP address of the network, in dotted decimal notation, optionally followed by a "/" character and the number of bits in the network mask. If you do not specify the CIDR mask, the router uses the value from the **mask** parameter, if present, or otherwise the natural mask for the network, based on whether it is a class A, B, or C network. Default: no default	
MASK	The network mask for the aggregate entry. This parameter is provided for compatibility with other router commands that specify an IP address and mask; we recommend that you instead specify the mask in the **aggregate** parameter. If you specify a mask in this parameter and the **aggregate** parameter, an error results unless the two masks agree. Default: The natural mask for the network, based on whether it is a class A, B, or C network	
SUMmary	Whether the router advertises only the aggregate route, or also the more specific routes that make up the aggregate. Default: **no**	
	no	The router advertises the more specific routes that make up the aggregate.
	yes	The router only advertises the aggregate route. Note that unadvertised routes are still displayed in the output of the **show bgp route** command, but are marked with an "s".
ROUTEMap	The route map used to filter the more specific routes that make up the aggregate, or to set attributes for the aggregate route. The *routemap* is the name of the appropriate pre-existing map. Default: no route map (routes are not filtered and attributes are not set)	

Tip The shortest string you can enter is shown in capital letters.

Examples

To set the route map for the aggregate entry for the network 192.168.8.0/21 to be route map *agg_map1*, use the command:

```
set bgp agg=192.168.8.0/21 routem=agg_map1
```

Related Commands

add bgp aggregate
delete bgp aggregate
show bgp aggregate

set bgp backoff

Syntax

```
SET BGP BACkoff[=0..100] [BASEtime=0..100]
   [CONSecutive=1..20] [MULtiplier=0..1000] [STep=0..1000]
   [TOTallimit=0..1000]
```

Description

This command configures BGP backoff, which stops BGP processing when system memory is heavily used by another process.

Parameter	Description
BACkoff	The percentage utilisation of system memory that causes BGP to back off, from 0 to 100. Default: 95
BASEtime	The time value in seconds used to calculate the total backoff time for the first backoff iteration, from 0 to 100. The first backoff time is calculated as: `basetime x multiplier/100` Default: 10
CONSecutive	The number of consecutive backoffs that causes BGP to disable all peers, from 1 to 20. Default: 20
MULtiplier	A multiplier for increasing or decreasing the backoff time at each backoff iteration, from 0 to 1000. The change in backoff time at each step is calculated as: `current backoff time x multiplier/100` Default: 100
STep	The number of backoff iterations after which the backoff time is recalculated. Default: 1
TOTallimit	The total number of backoffs that may occur until all peers are disabled. Default: 0 (no limit)

Tip The shortest string you can enter is shown in capital letters.

Examples

To back BGP processing off when the system memory is 90% utilised, use the command:

```
set bgp bac=90
```

Related Commands

set bgp memlimit
show bgp backoff
show bgp memlimit

set bgp import

Syntax

```
SET BGP IMPort={INTerface|OSPF|RIP|STAtic}
   [ROUTEMap[=routemap]]
```

Description

This command associates a different route map with a BGP import entry, or disassociates a route map. The import entry instructs BGP to import routes from that route source into the BGP route table, and the route map allows filtering of routes and setting of BGP attributes.

Parameter	Description	
IMPort	The BGP import entry to be modified. This must already have been added to BGP by using the **add bgp import** command on page 49-49. Default: no default	
	INTerface	Imports interface routes.
	OSPF	Imports OSPF routes.
	RIP	Imports RIP or RIP2 routes.
	STAtic	Imports statically configured routes.
ROUTEMap	The route map used to filter the routes imported into BGP and to set attributes for the routes as advertised by BGP. The *routemap* is the name of the appropriate pre-existing map. The route map can **match** on origin, next hop, prefix list or tag, and can use any of the **set** parameters. If you specify the **routemap** parameter without specifying a routemap name, the existing route map is removed from the import entry. Default: no route map (routes are not filtered and attributes are not set)	

Tip The shortest string you can enter is shown in capital letters.

Examples

To change the route map for the import entry for importing OSPF routes into BGP to route map *ospf_bgp_map1*, use the command:

```
set bgp imp=ospf routem=ospf_bgp_map1
```

Related Commands

add bgp import
delete bgp import
show bgp import

set bgp memlimit

Syntax `SET BGP MEMlimit[=0..100]`

Description This command limits the percentage of system memory available to BGP.

The **memlimit** parameter specifies the maximum percentage of system memory that BGP can use. When BGP exceeds this percentage, the router shuts down BGP peers and drops all routes learnt from the peers. The default is 95.

Example To limit BGP to 90% of system memory, use the command:

```
set bgp mem=90
```

Related Commands

set bgp backoff
show bgp memlimit
show bgp memlimit scan
show buffer in Chapter 1, Operation

set bgp peer

Syntax

```
SET BGP PEer=ipadd [AUthentication={MD5|NONE}]
   [CONnectretry={DEFault|0..4294967295}]
   [DESCription=description] [EHOps={DEFault|1..255}]
   [HOLdtime={DEFault|0|3..65535}]
   [INFilter={NONE|300..399}] [INPathfilter={NONE|1..99}]
   [INRoutemap=[routemap]] [KEEpalive={DEFault|1..21845}]
   [LOCal={NONE|1..15}] [MAXPREFIX={OFF|1..4294967295}]
   [MAXPREFIXAction={Terminate|Warning}]
   [MINAsoriginated={DEFault|0..3600}]
   [MINRouteadvert={DEFault|0..3600}]
   [NEXthopself={NO|YES}] [OUTFilter={NONE|300..399}]
   [OUTPathfilter={NONE|1..99}] [OUTRoutemap=[routemap]]
   [PASSword=password] [REMoteas=1..65534]
   [SENdcommunity={NO|YES}]

SET BGP PEer=ipadd POLICYTemplate=

SET BGP PEer=ipadd [POLICYTemplate=1..30]
   [AUthentication={MD5|NONE}] [DESCription=description]
   [EHOps={DEFault|1..255}] [PASSword=password]
   [REMoteas=1..65534]
```

Description

This command modifies parameters for an existing BGP peer in the router. The changes take effect immediately if you have enabled automatic updating by using the **enable bgp autosoftupdate** command on page 49-83. Otherwise, you need to trigger an update by using the **reset bgp peer soft** command on page 49-86.

Parameter	Description	
PEer	The IP address of the peer, in dotted decimal notation. Default: no default	
AUthentication	Whether or not to use MD5 authentication for the BGP peer. If you specify **md5**, you must also specify **password**. Default: **none**	
	MD5	An MD5 digest is added to every BGP packet sent over the TCP connection and is authenticated at the other end. If any part of the digest cannot be verified, the packet is dropped with no response sent.
	NONE	The BGP session is not authenticated.
CONnectretry	The time interval between attempts to establish a BGP connection to the peer, in seconds. Default: **120**	
	0	The router does not repeat an attempt to establish a BGP connection.
	1..4294967295	The router waits the specified number of seconds between attempts.
	DEFault	The router waits 120 seconds between attempts.

Parameter	Description	
DESCription	A description of the peer, which has no effect on its operation. A string 1 to 63 characters long. All printable characters are valid except the question mark and double quotes. If *description* contains spaces, the string must be in double quotes. Default: no default	
EHOps	The number of hops put in the *TTL* (Time To Live) field of BGP messages for external BGP. Normally, EBGP requires that BGP peers be connected to a common network, which means they are separated by a single hop. Setting **ehops** to a value greater than 1 indicates that multihop EBGP is allowed. Default: **1**	
	1..255	The specified number of hops is put into the TTL field.
	DEFault	The number of hops put in the TTL field is 1.
HOLdtime	The value in seconds that this router proposes for the time interval between reception of keepalive and/or update messages from this peer. The actual hold time used on a peer connection is negotiated when the connection is opened, as the lower of the hold times proposed. Default: **90**	
	0	This router proposes not to have a hold time on this BGP connection.
	3..65535	This router proposes the specified number of seconds as hold time.
	DEFault	This router proposes a hold time of 90 seconds.
INFilter	The IP routing filter that acts as a prefix filter to filter any prefixes advertised via incoming BGP update messages from this peer. You can use a prefix filter to exclude routes to particular networks from the update message. The filter must already exist. To create a filter use the **add ip filter** command on page 14-68 of Chapter 14, Internet Protocol (IP) and create a filter with a number from 300 to 399. If you specify more than one of **inpathfilter**, **infilter** and **inroutemap**, the router applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the router stops checking after the first filter entry that excludes the prefix, so a prefix is only included if all the applied filters result in it being included. Default: **none**	
INPathfilter	The AS path list that filters the BGP update messages from this peer. You can use an AS path list to exclude or include update messages that have traversed particular ASs or paths. If the path list does not already exist, it is created. To create a path list and/or add entries to it, use the **add ip aspathlist** command on page 49-62. If you specify more than one of **inpathfilter**, **infilter** and **inroutemap**, the router applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the router stops checking after the first filter entry that excludes the update, so an update is only included if all the applied filters result in it being included. Default: **none**	

Parameter	Description
INRoutemap	The route map that filters and/or modifies prefixes from this peer. You can use a route map to include or exclude update messages or a subset of an update message's routes, on the basis of a range of BGP attributes, and/or to modify attributes. If you specify the inroutemap parameter without specifying a route map name, the current route map is disassociated from the peer. The route map must already exist. To create a route map use the **add ip routemap** command on page 49-68. If you specify more than one of **inpathfilter**, **infilter** and **inroutemap**, the router applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the router stops checking after the first filter entry that excludes the update, so an update is only included if all the applied filters result in it being included. Default: **none**
KEEpalive	The time in seconds that this router would prefer to leave between keepalive messages to this peer. This time should be one third of the **holdtime** parameter. The actual value used for the keep alive interval is determined once the BGP connection is opened, because the hold time interval is calculated as part of the BGP connection opening. The actual keep alive interval is calculated so that the ratio: configured keep alive interval: configured hold time interval is the same as the ratio: actual keep alive interval: negotiated hold time interval. If the hold time is negotiated at 0 seconds, then the keep alive interval is also 0 seconds, and keepalive messages are not sent. Default: one third of **holdtime**
1..21845	This router prefers the specified number of seconds as keepalive interval.
DEFault	This router prefers a keepalive interval of one third the hold time.
LOCal	The local interface. In certain circumstances, the router uses this address as the source for BGP packets it generates and sends to this BGP peer. For a description of when the router uses the local interface, see "How to Set the IP Address By Which the Router Identifies Itself" on page 49-38. Default: **none**
MAXPREFIX	The maximum number of network prefixes that the router expects to receive from this peer. This parameter provides a safety mechanism in case the peer sends more prefixes than you might normally expect to receive. Default: **off**
1..4294967295	The maximum number of prefixes the router expects to receive from this peer. Once this number is exceeded, the action you specify in **maxprefixaction** is carried out.
OFF	No maximum prefix checking.

Parameter	Description	
MAXPREFIXAction	The action to take when a peer has sent a number of prefixes that exceeds the number specified by **maxprefix**. Default: **warning**	
	Warning	The router logs warnings when the maximum number of prefixes is exceeded.
	Terminate	The router resets the peer connections and logs warnings.
MINAsoriginated	The minimum time in seconds between advertisements, from the router to this peer, of routes that originate in the router's autonomous system. Default: **15**	
	0..3600	The interval is the specified number of seconds.
	DEFault	The interval is 15 seconds.
MINRouteadvert	The minimum time in seconds between advertisements, from the router to this peer, of routes that originate outside the router's autonomous system. Default: **30**	
	0..3600	The interval is the specified number of seconds.
	DEFault	The interval is 30 seconds.
NEXthopself	Whether this router advertises to this peer that the next hop for all routes is itself. Default: **no**	
	YES	All updates that the router sends to this peer specify this router as the next hop.
	NO	The next hop is specified as described in RFC 1771.
OUTFilter	The routing filter that acts as a prefix filter to filter the prefixes sent in BGP update messages to this peer. You can use a prefix filter to exclude routes to particular networks from the update message. The filter must already exist. To create a filter use the **add ip filter** command on page 14-68 of Chapter 14, Internet Protocol (IP) and create a filter with a number from 300 to 399. If you specify more than one of **outpathfilter**, **outfilter** and **outroutemap**, the router applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the router stops checking after the first filter entry that excludes the prefix, so a prefix is only included if all the applied filters result in it being included. Default: **none**	

Parameter	Description
OUTPathfilter	The AS path list that filters the BGP update messages sent to this peer. You can use an AS path list to exclude or include update messages that have traversed particular ASs or paths. If the path list does not already exist, it is created. To create a path list and/or add entries to it, use the **add ip aspathlist** command on page 49-62. If you specify more than one of **outpathfilter**, **outfilter** and **outroutemap**, the router applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the router stops checking after the first filter entry that excludes the update, so an update is only included if all the applied filters result in it being included. Default: **none**
OUTRoutemap	The route map that filters and/or modifies prefixes sent to this peer. You can use a route map to include or exclude update messages or a subset of an update message's routes, on the basis of a range of BGP attributes, and/or to modify attributes. The route map must already exist. To create a route map use the **add ip routemap** command on page 49-68. If you specify the **outroutemap** parameter without specifying a route map name, the current route map is disassociated from the peer. If you specify more than one of **outpathfilter**, **outfilter** and **outroutemap**, the router applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the router stops checking after the first filter entry that excludes the update, so an update is only included if all the applied filters result in it being included. Default: **none**
PASSword	The key used by the authentication algorithm. Two BGP peers can communicate with each other only if they have the same key. *password* is a character string from 1 to 80 characters long. All printable characters are valid except the question mark and double quotes. If *password* contains spaces, it must be in double quotes. Only valid if **authentication=md5** Default: no default
POLICYTemplate	The ID number of the peer policy template that applies to this peer. The specified policy template must already exist. To create a template, use the **add bgp peertemplate** command on page 49-57. You can only specify **remoteas**, **description**, **authentication**, **password**, and **ehops** at the same time as **policytemplate**. The template provides all other configuration values. Specifying **policytemplate=** with no number disassociates the template and the peer. The peer retains the template's settings. You can then modify the peer.
REMoteas	The remote Autonomous System to which this peer belongs. If the remote AS number is the same as this router's AS number, the peer is an internal BGP (IBGP) peer. If the remote AS number is different from this router's AS number, the peer is an external BGP (EBGP) peer. If the remote AS numbers are different but the routers have the same confederation peer, the peer is a confederation BGP peer. The AS number is assigned by the IANA. Default: no default

Parameter	Description	
SENdcommunity	Whether the router includes the community attribute in update messages that it sends to this peer. Default: no	
	YES	The community attribute is set in update messages to this peer. To set the value of the community attribute, create a route map with a **set** clause to set the community, and use the **outroutemap** parameter to apply it to update messages to this peer. To create a route map use the **add ip routemap** command on page 49-68.
	NO	The community attribute is not set in update messages to this peer, even if it is set in the route map used by the peer.

Tip The shortest string you can enter is shown in capital letters.

Examples

To set authentication for a BGP peer whose IP address is 192.168.1.1, use the command:

```
set bgp pe=192.168.1.1 au=md5 password=a-very-secret-password
```

To modify the keepalive time and hold time for a BGP peer whose IP address is 192.168.1.1, use the command:

```
set bgp pe=192.168.1.1 kee=10 hol=30
```

Related Commands

add bgp peer
add ip aspathlist
add ip filter
add ip routemap
delete bgp peer
disable bgp peer
enable bgp peer
reset bgp peer
show bgp peer

set bgp peertemplate

Syntax

```
SET BGP PEERTemplate=1..30
   [CONnectretry={DEFault|0..4294967295}]
   [DESCription=description]
   [HOLdtime={DEFault|0|3..65535}]
   [INFilter={NONE|300..399}] [INPathfilter={NONE|1..99}]
   [INRoutemap=routemap] [KEEpalive={DEFault|1..21845}]
   [LOCal={NONE|1..15}] [MAXPREFIX={OFF|1..4294967295}]
   [MAXPREFIXAction={Terminate|Warning}]
   [MINAsoriginated={DEFault|0..3600}]
   [MINRouteadvert={DEFault|0..3600}]
   [NEXthopself={NO|YES}] [OUTFilter={NONE|300..399}]
   [OUTPathfilter={NONE|1..99}] [OUTRoutemap=routemap]
   [SENdcommunity={NO|YES}]
```

Description This command modifies a template for use on BGP peers. BGP applies the changes to all peers that are using the template.

Parameter	Description	
PEERTemplate	The ID number of the template. Default: no default	
CONnectretry	The time interval between attempts to establish a BGP connection to peers that use the template, in seconds. Default: **120**	
	0	The router does not repeat an attempt to establish a BGP connection.
	1..4294967295	The router waits the specified number of seconds between attempts.
	DEFault	The router waits 120 seconds between attempts.
DESCription	A description for the peers that use the template, which has no effect on their operation. A string 1 to 63 characters long. All printable characters are valid except the question mark and double quotes. If *description* contains spaces, the string must be in double quotes. Default: no default	
HOLdtime	The value in seconds that this router proposes for the time interval between reception of keepalive and/or update messages from peers that use the template. The actual hold time used on a peer connection is negotiated when the connection is opened, as the lower of the hold times proposed. Default: **90**	
	0	This router proposes not to have a hold time on this BGP connection.
	3..65535	This router proposes the specified number of seconds as hold time.
	DEFault	This router proposes a hold time of 90 seconds.

Parameter	Description
INFilter	The routing filter that acts as a prefix filter and filters any prefixes advertised via incoming BGP update messages from peers that use the template. You can use a prefix filter to exclude routes to particular networks from update messages. The filter must already exist. To create a filter use the **add ip filter** command on page 14-68 of Chapter 14, Internet Protocol (IP). The filter number must be in the range 300 to 399. If you specify more than one of **inpathfilter**, **infilter** and **inroutemap**, the router applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the router stops checking after the first filter entry that excludes the prefix, so a prefix is only included if all the applied filters result in it being included. Default: **none**
INPathfilter	The AS path list that filters the BGP update messages from peers that use the template. You can use an AS path list to exclude or include update messages that have traversed particular ASs or paths. If the path list does not already exist, it is created. To create a path list and/or add entries to it, use the **add ip aspathlist** command on page 49-62. If you specify more than one of **inpathfilter**, **infilter** and **inroutemap**, the router applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the router stops checking after the first filter entry that excludes the update, so an update is only included if all the applied filters result in it being included. Default: **none**
INRoutemap	The route map that filters and/or modifies prefixes from peers that use the template. You can use a route map to include or exclude update messages or a subset of an update message's routes, on the basis of a range of BGP attributes, and/or to modify attributes. The route map must already exist. To create a route map use the **add ip routemap** command on page 49-68. If you specify more than one of **inpathfilter**, **infilter** and **inroutemap**, the router applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the router stops checking after the first filter entry that excludes the update, so an update is only included if all the applied filters result in it being included. Default: **none**

Parameter	Description	
KEEpalive	The time in seconds that this router would prefer to leave between keepalive messages to peers that use the template. This time should be one third of the **holdtime** parameter. The actual value used for the keep alive interval is determined once the BGP connection is opened, because the hold time interval is calculated as part of the BGP connection opening. The actual keep alive interval is calculated so that the ratio: configured keep alive interval: configured hold time interval is the same as the ratio: actual keep alive interval: negotiated hold time interval. If the hold time is negotiated at 0 seconds, then the keep alive interval is also 0 seconds, and keepalive messages are not sent. Default: one third of **holdtime**	
	1..21845	This router prefers the specified number of seconds as keepalive interval.
	DEFault	This router prefers a keepalive interval of one third the hold time.
LOCal	The local interface. In certain circumstances, the router uses this address as the source for BGP packets it generates and sends to peers that use this template. For a description of when the router uses the local interface, see "How to Set the IP Address By Which the Router Identifies Itself" on page 49-38. Default: **none**	
MAXPREFIX	The maximum number of network prefixes that the router expects to receive from peers that use the template. This parameter provides a safety mechanism in case the peer sends more prefixes than you might normally expect to receive. Default: **off**	
	1..4294967295	The maximum number of prefixes the router expects to receive from peers that use the template. Once this number is exceeded, the action you specify in **maxprefixaction** is carried out.
	OFF	No maximum prefix checking.
MAXPREFIXAction	The action to take when a peer has sent a number of prefixes that exceeds the number specified by **maxprefix**. Default: **warning**	
	Warning	The router logs warnings when the maximum number of prefixes is exceeded.
	Terminate	The router resets the peer connections and logs warnings.
MINAsoriginated	The minimum time in seconds between advertisements, from the router to peers that use the template, of routes that originate in the router's autonomous system. Default: **15**	
	0..3600	The interval is the specified number of seconds.
	DEFault	The interval is 15 seconds.

Parameter	Description	
MINRouteadvert	The minimum time in seconds between advertisements, from the router to peers that use the template, of routes that originate outside the router's autonomous system. Default: **30**	
	0..3600	The interval is the specified number of seconds.
	DEFault	The interval is 30 seconds.
NEXthopself	Whether this router advertises to peers that use the template that the next hop for all routes is itself. Default: **no**	
	YES	All updates that the router sends to peers that use this template specify this router as the next hop.
	NO	The next hop is specified as described in RFC 1771.
OUTFilter	The routing filter that acts as a prefix filter and filters the prefixes sent in BGP update messages to peers that use this template. You can use a prefix filter to exclude routes to particular networks from the update message. The filter must already exist. To create a filter use the **add ip filter** command on page 14-68 of Chapter 14, Internet Protocol (IP) and create a filter with a number from 300 to 399. If you specify more than one of **outpathfilter**, **outfilter** and **outroutemap**, the router applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the router stops checking after the first filter entry that excludes the prefix, so a prefix is only included if all the applied filters result in it being included. Default: **none**	
OUTPathfilter	The AS path list that filters the BGP update messages sent to peers that use this template. You can use an AS path list to exclude or include update messages that have traversed particular ASs or paths. If the path list does not already exist, it is created. To create a path list and/or add entries to it, use the **add ip aspathlist** command on page 49-62. If you specify more than one of **outpathfilter**, **outfilter** and **outroutemap**, the router applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the router stops checking after the first filter entry that excludes the update, so an update is only included if all the applied filters result in it being included. Default: **none**	
OUTRoutemap	The route map that filters and/or modifies prefixes sent to peers that use this template. You can use a route map to include or exclude update messages or a subset of an update message's routes, on the basis of a range of BGP attributes, and/or to modify attributes. The route map must already exist. To create a route map use the **add ip routemap** command on page 49-68. If you specify more than one of **outpathfilter**, **outfilter** and **outroutemap**, the router applies them in that order: first the AS path filter, then the prefix filter, then the route map. Note that the router stops checking after the first filter entry that excludes the update, so an update is only included if all the applied filters result in it being included. Default: **none**	

Parameter	Description	
SENdcommunity	Whether the router includes the community attribute in update messages that it sends to peers that use this template. Default: no	
	YES	The community attribute is set in update messages to peers that use this template. To set the value of the community attribute, create a route map with a **set** clause to set the community, and use the **outroutemap** parameter to apply it to update messages to peers that use this template. To create a route map use the **add ip routemap** command on page 49-68.
	NO	The community attribute is not set in update messages to this peer, even if it is set in the route map used by the peers that use this template.

Tip The shortest string you can enter is shown in capital letters.

Examples

To modify a peer policy template 1 to have a hold time of 30 seconds, use the command:

```
set bgp peert=1 hol=30
```

Related Commands

add bgp peer
add bgp peertemplate
set bgp peer
show bgp peer

set ip autonomous

Syntax `SET IP AUtonomous=1..65534`

Description This command sets the router's autonomous system number (ASN). The router cannot be configured to use BGP-4 until it is part of an AS and therefore has an ASN.

There are two types of ASNs:

- public ASNs (1 to 64511)

  These are globally unique and assigned by the IANA. They identify the router's AS when the router exchanges routes with external organisations.

- private ASNs (64512 to 65534)

  These are non-assigned numbers. You can use them when you are running BGP in an AS confederation, for example. The individual AS numbers in the confederation can be non-assigned numbers.

Caution If the router has a peer relationship with a public peer, always use an assigned autonomous system number rather than inventing one.

Related Commands

add bgp peer
enable bgp peer
show bgp

set ip routemap

Syntax to change the action

```
SET IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion={INCLude|EXCLude}]
```

Syntax to change a *match* clause

```
SET IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion={INCLude|EXCLude}] MAtch ASPath=1..99

SET IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion={INCLude|EXCLude}] MAtch COMmunity=1..99
   [EXAct={NO|YES}]

SET IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion={INCLude|EXCLude}] MAtch NEXThop=ipadd

SET IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion={INCLude|EXCLude}] MAtch
   ORIGin={EGP|IGP|INCOmplete}

SET IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion={INCLude|EXCLude}] MAtch PREFIXList=name

SET IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion={INCLude|EXCLude}] MAtch TAG=1..65535
```

Syntax to change a *set* clause

```
SET IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion=INCLude] SET ASPath={1..65534[,...]}

SET IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion=INCLude] SET
   COMmunity={NOExport|NOAdvertise|NOEXPORTSubconfed|
   1..4294967295}[,...] [ADD={NO|YES}]

SET IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion=INCLude] SET LOCalpref=0..4294967295

SET IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion=INCLude] SET MED={0..4294967295|REMOVE}

SET IP ROUTEMap=routemap ENTry=1..4294967295
   [ACtion=INCLude] SET ORIGin={IGP|EGP|INCOmplete}
```

Description

This command does one of the following:

- changes the **action** of an entry in a route map
- modifies an entry's **match** clause
- modifies an entry's **set** clause

This command does not create or delete an entry or clause. To create a new entry or clause, use the **add ip routemap** command on page 49-68. To delete an entry or clause, use the **delete ip routemap** command on page 49-80.

Parameters for both *match* and *set* clauses

Parameter	Description
ROUTEMap	The name of the route map that the entry or clause belongs to. Default: no default
ENTry	The ID number of the entry to change. Default: no default
ACtion	Whether matching prefixes or update messages are included or excluded by the process that is using the route map. The **action** parameter applies to the entire entry. It is not meaningful to have **action=exclude** in an entry with a set clause. Default: the current setting. If there is no current setting, **include**

Tip The shortest string you can enter is shown in capital letters.

Parameters for *match* clauses

Parameter	Description
MAtch	Modifies a **match** clause in the entry. The match clause determines which update messages the entry applies to.
ASPath	The ID number of an AS path list. An update message matches the route map entry if its AS path attribute matches the AS path list. To configure an AS path list use the **add ip aspathlist** command on page 49-62. Default: no default
COMmunity	The ID number of a community list. An update message matches the route map entry if its community attribute matches the community list.To configure a community list use the **add ip communitylist** command on page 49-64. Default: no default
EXAct	Whether the community attribute in an update message must precisely match the route map's community list. Valid only when you specify both **match** and **community**. Default: **no**
NEXThop	The IP address of the next node in the path to the route's destination, specified in dotted decimal notation. An update message matches the route map entry if its next_hop attribute matches this address. Default: no default

YES	An update message only matches the route map entry if its community attribute contains all the communities specified in the community list and only those communities.
NO	An update message still matches the route map entry if its community attribute contains all the communities specified in the community list plus extra communities.

Parameter	Description	
ORIGin	An origin attribute value, which indicates BGP's source for the routes at their originating AS. An update message matches the route map entry if its origin attribute matches this value. Default: no default	
	IGP	The original source of the route was IGP.
	EGP	The original source of the route was EGP.
	INCOmplete	The original source of the route was neither IGP or EGP. This includes statically-configured routes.
PREFIXList	The name of a prefix list. A route matches the route map entry if the prefix list contains that route. To create a list use the **add ip prefixlist** command on page 49-66. Default: no default	
TAG	A tag that identifies a particular static route. A route matches this route map entry if it has been tagged with this value by using the **tag** parameter of the **add ip route** command on page 14-88 of Chapter 14, Internet Protocol (IP). Use a route map that matches on **tag** when you use the **add bgp network** and **add bgp import** commands to import static routes from IP to BGP. You cannot use **tag** to filter routes that are sent to BGP peers or to match update messages that are received from BGP peers. Default: no default	
Tip The shortest string you can enter is shown in capital letters.		

Parameters for *set* clauses

Parameter	Description	
SET	Modifies a **set** clause in the entry. The set clause changes route attributes.	
ASPath	A comma-separated list of 1 to 10 AS numbers. These numbers are added to the beginning of the update message's AS path attribute. Default: no default	
COMmunity	A comma-separated list of 1 to 10 communities identified by name or number. If the **add** parameter is **yes**, these communities are added to the update message's community attribute. If the **add** parameter is **no** (its default), these communities replace the update message's community attribute. Note that you must also set the peer's **sendcommunity** parameter to **yes** if you want the peer to include the community attribute in the update messages it sends. By default, peers do not include the community attribute in outgoing updates. Default: no default	
	INTernet	The community of routes that can be advertised to all BGP peers.
	NOExport	The community of routes that must not be advertised outside a BGP confederation boundary (a standalone autonomous system that is not part of a confederation should be considered a confederation itself).
	NOAdvertise	The community of routes that must not be advertised to other BGP peers.
	NOEXPORTSubconfed	The community of routes that must not be advertised to external BGP peers (this includes peers in other members' autonomous systems inside a BGP confederation).
	1..4294967295	The number of a community.
ADD	Whether the list of communities specified by the **community** parameter is added to the community attribute, or replaces the community attribute. Only valid when you specify both **set** and **community**. Default: **no**	
	YES	The communities are added to the update message's community attribute.
	NO	The communities replace the update message's community attribute.
LOCalpref	The metric to write into the update message's local_preference attribute. IBGP uses the local preference to determine which path it should use inside the AS to reach the advertised prefix. A lower metric indicates a preferred path. EBGP does not use this attribute. Default: no default	

Parameter	Description	
MED	The metric to write into the update message's Multi_Exit_Discriminator attribute. EBGP uses the MED to determine the optimal path for reaching the advertised prefixes. A lower metric indicates a preferred path. IBGP does not use this attribute. Default: no default	
	0..4294967295	This value is written into the MED attribute of the matched update message.
	REMOVE	The MED attribute is removed from the matched update message.
ORIGin	The value to write into the update message's origin attribute. The origin indicates BGP's source for the routes at their originating AS.	
	IGP	The original source of the route was IGP.
	EGP	The original source of the route was EGP.
	INCOmplete	The original source of the route was neither IGP or EGP. This includes statically-configured routes.

Tip The shortest string you can enter is shown in capital letters.

Examples

To change the MED for an existing set MED clause in entry 10 of the route map called set_med, use the command:

```
set ip routem=set_med ent=10 set med=234
```

Related Commands

add ip routemap
delete ip routemap
show ip routemap

show bgp

Syntax `SHow BGP`

Description This command displays information about BGP global configuration and operation (Figure 49-11, Table 49-13).

Figure 49-11: Example output from the **show bgp** command

```
BGP router ID ................. 192.168.1.1
Local autonomous system ....... 123
Confederation ID .............. 1234
Local preference .............. 100 (default)
Multi Exit Discriminator ...... -
Route table route map ......... -
Auto soft reconfiguration ..... Disabled

Number of peers
  Defined ..................... 4
  Established ................. 2
BGP route table
  Iteration ................... 231
  Number of routes ............ 12654
  Route table memory .......... 431872
```

Table 49-13: Parameters in the output of the **show bgp** command

Parameter	Meaning
BGP router ID	The ID for BGP for this router. This is the router ID if one has been set using the **set bgp** command. Otherwise it is the local interface if one has been configured. Otherwise it is the highest IP address configured on any of the router's interfaces. For more information, see "How to Set the IP Address By Which the Router Identifies Itself" on page 49-38.
Local autonomous system	The number of the Autonomous System to which this router belongs.
Confederation ID	The AS number of the AS confederation to which this router belongs.
Local preference	The value of the local preference for this router. This is sent to internal BGP peers to help in the decision process for deciding which BGP routes go into the main routing table.
Multi-Exit Discriminator	The value of the multi-exit discriminator for this router. This is sent to external BGP peers to help in the decision process for deciding which BGP routes go into the main routing table.
Route table route map	The name of the route map that BGP uses before entering a route into the route table.
Auto soft reconfiguration	Whether the router automatically updates modified peers.
Number of peers	Counters giving the number of configured and established peers.
Defined	The number of peers currently configured on the router.
Established	The number of peers currently in the established state.
BGP route table	Information about the BGP route table.

Table 49-13: Parameters in the output of the **show bgp** command (continued)

Parameter	Meaning
Iteration	The number of times the BGP route table has been modified.
Number of routes	The number of routes in the BGP route table.
Route table memory	The amount of router memory currently used in the BGP route table.

Examples To show general BGP parameters and a summary of BGP operations, use the command:

```
sh bgp
```

Related Commands
show bgp aggregate
show bgp import
show bgp network
show bgp peer
show bgp route

show bgp aggregate

Syntax `SHow BGP AGGregate`

Description This command displays information about the BGP aggregate entries configured in this router (Figure 49-12, Table 49-14).

Figure 49-12: Example output from the **show bgp aggregate** command

```
BGP aggregate entries

Prefix               Summary   Route map
-----------------------------------------------
192.168.248.0/21     Yes       aggregate_map
192.168.16.0/21      No        -
-----------------------------------------------
```

Table 49-14: Parameters in the output of the **show bgp aggregate** command

Parameter	Meaning
Prefix	Prefix for this aggregate entry. This is the prefix that BGP advertises for this aggregate as long as a route that is a subset of this prefix is present in the BGP routing table.
Summary	Either Yes or No. If yes, the aggregate route is advertised and no subset routes. If no, subset routes are also advertised.
Route map	Name of the route map used to filter routes for this aggregate entry and to set the BGP attributes for the aggregate route entry.

Examples To display information about BGP aggregates, use the command:

```
sh bgp agg
```

Related Commands

add bgp aggregate
delete bgp aggregate
set bgp aggregate
show bgp
show ip routemap

show bgp confederation

Syntax `SHow BGP CONfederation`

Description This command displays information about the BGP confederation setup of this router (Figure 49-13, Table 49-15).

Figure 49-13: Example output from the **show bgp confederation** command

```
BGP confederation information

Local AS ................ 60001
Confederation ID ........ 1234
Confederation peers ..... 60002
                          60003
Peers ................... 192.168.1.1 (AS 60001, IBGP)
                          192.169.3.2 (AS 60002, CBGP)
                          192.170.4.5 (AS 7658, EBGP)
```

Table 49-15: Parameters in the output of the **show bgp confederation** command

Parameter	Meaning	
Local AS	The AS number of the AS to which this router belongs.	
Confederation ID	The AS number of the AS confederation to which this router belongs. The AS confederation behaves as this AS to external BGP peers.	
Confederation peers	The AS numbers of Autonomous Systems in the router's AS confederation.	
Peers	A list of the configured BGP peers from the point of view of AS confederation configuration. For each peer, the peer address, peer AS, and BGP type is given. BGP type is:	
	EBGP	External BGP. The peer is in a different AS and outside the AS confederation.
	CBGP	Confederation BGP. The peer is in a different AS but inside the AS confederation.
	IBGP	Internal BGP. The peer is in the same AS as this router.

Examples To display information concerning the AS confederation to which this router belongs, use the command:

```
sh bgp con
```

Related Commands

add bgp confederationpeer
delete bgp confederationpeer
set bgp
set ip autonomous
show bgp

show bgp backoff

Syntax `SHow BGP BACkoff`

Description This command displays BGP backoff details (Figure 49-14, Table 49-16).

Figure 49-14: Example output of the **show bgp backoff** command

```
BGP Backoff Stats:
 Stat                          Value
---------------------------------
state                          NORMAL
total hist backOffs               0
total backOffs                    0
total backOff Limit               0
was backedOff                  FALSE
consecutive backOffs              0
consecutive backOffs limit        5
base Timeout                     60
Timeout multiplier              100%
Timeout step                      1
Timeout length (sec)             60
Trigger on Mem use               95%
Current Mem use                   7%
---------------------------------
```

Table 49-16: Parameters in the output of the **show bgp backoff** command

Parameter	Meaning
state	Whether the BGP backoff state is NORMAL, BACKING OFF, or DISABLING PEERS.
total hist backOffs	The total number of backoffs that have occurred. Unlike **total backOffs**, this value is not reset when BGP disables peers because it reaches the total or consecutive backoff limit. You can use this value to determine the optimal setting for the total backoff limit.
total backOffs	The number of times that BGP has backed off since it last reached the total backoff limit. Note that this counter is not reset when BGP disables its peers because the consecutive backoff limit is reached.
total backoff limit	The total number of backoffs that cause BGP to disconnect its peers.
was backedOff	Whether BGP was backed off during the last operation.This is used to detect consecutive backoffs.
consecutive backoffs	The number of times in a row that BGP has reached the end of its backoff time and found that system memory is high enough that it backs off again immediately.
consecutive backoffs limit	The number of consecutive backoffs that causes BGP to disable all peers.
base Timeout	The number of seconds used to calculate the total backoff time for the first backoff iteration. The first backoff time is calculated as: `basetime x multiplier/100`

Table 49-16: Parameters in the output of the **show bgp backoff** command (continued)

Parameter	Meaning
Timeout multiplier	A multiplier for increasing or decreasing the backoff time at each backoff iteration. The change in backoff time at each step is calculated as: `current backoff time x multiplier/100`
Timeout step	The number of backoff iterations after which the backoff time is recalculated.
Timeout length	The current backoff time.
Trigger on Mem use	The percentage of system memory use that triggers BGP to back off.
Current Mem use	The amount of memory used by the system at the moment the command was executed.

Example To see the existing BGP backoff settings, use the command:

```
sh bgp bac
```

Related Commands

set bgp backoff
show bgp memlimit

show bgp import

Syntax `SHow BGP IMPort`

Description This command displays information about the BGP import entries present in the router (Figure 49-15, Table 49-17).

Figure 49-15: Example output from the **show bgp import** command

```
BGP import entries

Proto      Route map
--------------------------
OSPF       ospf_proto_map
RIP        rip_proto_map
--------------------------
```

Table 49-17: Parameters in the output of the **show bgp import** command

Parameter	Meaning
Protocol	The routing protocol whose routes are to be imported into BGP; either Interface, OSPF, RIP, or Static.
Route map	The name of the route map used to filter routes and set attributes for routes imported into BGP.

Examples To show the BGP import entries on this router, use the command:

```
sh bgp imp
```

Related Commands

add bgp import
delete bgp import
set bgp import
show ip routemap

show bgp memlimit

Syntax `SHow BGP MEMlimit`

Description This command displays the percentage of system memory that BGP is limited to, and the current actual memory use by BGP (Figure 49-16).

Figure 49-16: Example output of the **show bgp memlimit** command

```
BGP Memory Limit:  95%,   Actual Use:  0%
```

Example To display the amount of memory BGP is currently using, and its limit, use the command:

```
sh bgp mem
```

Related Commands
set bgp backoff
set bgp memlimit
show bgp memlimit scan
show buffer in Chapter 1, Operation

show bgp memlimit scan

Syntax `SHow BGP MEMlimit SCAN`

Description This command displays information about the freelists that are registered to a given module. This output is useful to display a detailed state of BGP memory use, at a given moment in time. (Figure 49-17, Table 49-18 on page 49-120).

Figure 49-17: Example output of the **show bgp memlimit scan** command

```
BGP Memory Limit:  65%,   Actual Use:  0%
Module Freelist Stats: moduleId = 5
module buffer use:      3
module percent use:     0%
         list  unitSize     freeUsed  buffersUsed
-------------------------------------------------
     008418d0        80            0            0
     0084fae0        12            0            0
     0084f104        88            0            0
     00841784        68            0            0
     00841900        48            2            1
     0084fb70        12            0            0
     00855398        32            3            1
     0085b8c4       228            2            1
-------------------------------------------------
Module Freelist Stats: moduleId = 103
module buffer use:      4
module percent use:     0%
         list  unitSize     freeUsed  buffersUsed
-------------------------------------------------
     0081306c        24            0            0
     0081300c        12            0            0
     008131c0        32            0            0
     0081361c        44            0            0
     00813220         8            0            0
     00812fdc       512            0            0
     00813124         8            0            0
     0081353c         8            0            0
     00813190         8            0            0
     00812654        40            0            0
     00813154       520            0            0
     008135c4       112            0            0
     00813350        32            0            0
     00812684       128            0            0
     00811e6c        24            0            0
     008120dc        56            0            0
     008130bc       268            0            0
     008126b4        20            0            0
     00813380        16            0            0
     0081356c        52            0            0
     008131f0        36            0            0
     008123a4        16            0            0
     0081364c        20            1            1
     00811f9c        40            0            0
     00812374      1060            1            1
     008136a4        40            0            0
     0081303c        16            0            0
     0085b8c4       228            2            1
 -------------------------------------------------
```

Table 49-18: Parameters in the output of the **show bgp memlimit scan** command

Parameter	Meaning
BGP Memory Limit	Percentage of system memory that limits BGP.
Actual Use	Percentage of memory BGP currently uses.
Module Freelist Stats	Statistics relating to the freelists for each module. Freelists divide memory buffers into small segments to increase efficiency of memory use.
moduleId	ID number of the software module (see "Module Identifiers and Names" on page C-2 of Appendix C, Reference Tables).
module buffer use	Number of buffers currently in use by the module.
module percent use	Number of buffers currently used by the module, as a percentage of the total number of buffers.
list	Freelists (as a hexadecimal address) registered to the module.
unitSize	Number of bytes each freelist segment uses.
freeUsed	Number of segments of the freelist currently used by the module.
buffersUsed	Number of memory buffers the freelist is currently using.

Example To display the detailed state of current BGP memory use, use the command:

```
sh bgp mem scan
```

Related Commands

set bgp backoff
set bgp memlimit
show bgp memlimit
show buffer in Chapter 1, Operation

show bgp network

Syntax `SHow BGP NETwork`

Description This command displays information about the BGP network entries configured in this router (Figure 49-18, Table 49-19).

Figure 49-18: Example output from the show bgp network command

```
BGP network entries

Prefix                Route map
-----------------------------------
192.168.248.0/21      network_map
192.168.16.0/21       -
-----------------------------------
```

Table 49-19: Parameters in the output of the **show bgp network** command

Parameter	Meaning
Prefix	Prefix for this network entry. This is the prefix that BGP advertises for this network as long as a route that matches this prefix exactly is present in the BGP routing table.
Route map	Name of the route map that is used to filter routes and set the BGP attributes for this network entry.

Examples To display information about BGP networks, use the command:

```
sh bgp net
```

Related Commands

add bgp network
delete bgp network
show bgp
show ip routemap

show bgp peer

Syntax `SHow BGP PEer[=ipadd]`

Description This command displays:

- summary information about all BGP peers if you do not specify a peer address (Figure 49-19, Table 49-20)
- detailed information about a peer, if you specify the IP address of the peer (Figure 49-20 on page 49-123, Table 49-21 on page 49-124). Addresses are specified in dotted decimal notation.

Figure 49-19: Example summary output from the **show bgp peer** command

```
BGP peer entries
Peer          State     AS     InMsg  OutMsg  Template
-------------------------------------------------------
192.168.2.254 Estab     12345  23456  3245    -
192.168.3.16  Idle(D)   123    2      3       2
```

Table 49-20: Example summary output from the **show bgp peer** command

Parameter	Meaning
Peer	IP address of the BGP peer.
State	BGP peer state, one of: • Idle • Idle (D)—Idle and also disabled • Connect • Active • OpenSent • OpenConf—OpenConfirm • Estab—Established For more information about the states, see "BGP Operation" on page 49-5.
AS	Number of the autonomous system to which this peer belongs.
InMsg	Number of messages received from this peer since the TCP connection opened.
OutMsg	Number of messages sent to this peer since the TCP connection opened.
Template	ID number of the peer policy template that provides the peer with its settings.

Figure 49-20: Example output from the **show bgp peer** command for a specific peer

```
Peer ................ 192.168.10.1
Description ......... -
State ............... Idle
Policy Template ..... 4
  Description ....... Test Template 1
Remote AS ........... 3
BGP Identifier ...... 172.20.25.2
Authentication ...... None
  Password .......... -
Connect retry ....... 120s
Hold time ........... 90s
Keep alive .......... 30s
Min AS originated ... 15
Min route advert .... 30
Local Interface ..... Not defined

Filtering
  In filter ......... -
  In path filter .... -
  In route map ...... -
  Out filter ........ -
  Out path filter ... -
  Out route map ..... -

Max prefix .......... OFF
External hops ....... 1 (EBGP multihop disabled)
Next hop self ....... No
Send community ...... No
Messages In/Out ..... 0/0
Debugging ........... -
  Device ............ -

Capabilties ......... Route Refresh

Established transitions ........... 0

Session Message counters:
  inOpen ........................... 0       outOpen ........................... 0
  inKeepAlive ...................... 0       outKeepAlive ...................... 0
  inUpdate ......................... 0       outUpdate ......................... 0
  inNotification ................... 0       outNotification ................... 0
  inRouteRefresh ................... 0       outRouteRefresh ................... 0

Total Message counters:
  inOpen ........................... 0       outOpen ........................... 0
  inKeepAlive ...................... 0       outKeepAlive ...................... 0
  inUpdate ......................... 0       outUpdate ......................... 0
  inNotification ................... 0       outNotification ................... 0
  inRouteRefresh ................... 0       outRouteRefresh ................... 0
```

Table 49-21: Parameters in output of the **show bgp peer** command for a specific peer

Parameter	Meaning
Peer	IP address of the BGP peer.
Description	Description of the peer if it has one.
State	BGP peer state, one of: • Idle • Idle (D)—Idle and also disabled • Connect • Active • OpenSent • OpenConfirm • Established For more information about the states, see "BGP Operation" on page 49-5.
Policy Template	ID number of the peer policy template that provides the peer with its settings.
Description	Description of the peer policy template if it has one.
Remote AS	Number of the autonomous system to which this peer belongs.
BGP identifier	The ID this router uses to identify itself to the peer. For more information, see "How to Set the IP Address By Which the Router Identifies Itself" on page 49-38.
Authentication	Authentication type used for communication with this BGP peer, or None if the connection is not using authentication.
Password	Password for this peer if the connection uses authentication.
Connect Retry	The time interval for retrying the initial TCP connection to this peer in the event of a connection failure.
Hold time	The configured and actual hold times for this peer. The actual hold time is the lower of the configured hold times of the peer and this router.
Keep alive	The configured and actual keepalive times for this peer. The actual keepalive time is set by the actual hold time in such a way that the ratio of actual keepalive to hold time is the same as the ratio of configured keepalive to hold time.
Min AS originated	Minimum time between advertisements of routes that originate in this autonomous system.
Min route advert	Minimum time between advertisements of routes that originate outside this autonomous system.
Filtering	Settings for inward and outward filtering of routing information via BGP.
In filter	Traffic filter used for filtering incoming routes from this peer.
In path filter	AS path filter used for filtering incoming routes from this peer.
In route map	Route map used for filtering incoming routes from this peer.
Out filter	Traffic filter used for filtering outgoing routes to this peer.
Out path filter	AS path filter used for filtering outgoing routes to this peer.
Out route map	Route map used for filtering outgoing routes to this peer.
Max prefix	Maximum number of route prefixes that may be received from this peer, and the action taken when this number is exceeded. The action is WARNING or TERMINATE.

Table 49-21: Parameters in output of the **show bgp peer** command for a specific peer

Parameter	Meaning
External hops	Number of hops that can be used to reach this peer when it is an EBGP peer. Having this number exceed 1 allows multihop EBGP.
Next hop self	Whether this router advertises to this peer that the next hop for all routes is itself.
Send community	Whether this router sends the community attribute in the path attributes of update messages.
Messages In/Out	Number of incoming/outgoing BGP messages from/to this peer.
Debugging	Debugging types enabled for this peer; one or more of MSG, STATE, UPDATE, or ALL.
Device	Device where debugging output is sent.
Local Interface	The local interface. In certain circumstances, the router uses this address as the source for BGP packets it generates and sends to the peer. For a description of when the router uses the local interface, see "How to Set the IP Address By Which the Router Identifies Itself" on page 49-38.
Capabilities	Extra capabilities negotiated between the peer and the router. "Route Refresh" indicates that the router automatically sends route refresh messages to the peer and process route refresh messages from the peer. Route refresh messages request a new update message, and are used after a BGP peer has been modified, to reset its routes.
Established transitions	Number of times the peer session has become established (entered the Established state).
Message Counters	**Session Message Counters** give the number of messages received or sent for the most recently established session with the peer. If the peer session is torn down at any time, the session counters are reset and accumulate from zero when a new session is established. **Total Message Counters** give the number of messages received or sent for all sessions ever established with the peer. For more information about messages, see "BGP Operation" on page 49-5.
InOpen	Number of *open* messages received from this peer. BGP peers use open messages to identify themselves to each other and negotiate settings.
OutOpen	Number of *open* messages sent to this peer. BGP peers use open messages to identify themselves to each other and negotiate settings.
InKeepAlive	Number of keepalive messages received from this peer. Keepalive messages maintain the BGP session when the peer has not needed to send update messages.
OutKeepAlive	Number of keepalive messages sent to this peer. Keepalive messages maintain the BGP session when the router has not needed to send update messages.
InUpdate	Number of update messages received from this peer. BGP peers use update messages to inform each other of route changes.
OutUpdate	Number of update messages sent to this peer. BGP peers use update messages to inform each other of route changes.
InNotification	Number of notification messages received from this peer. BGP peers use notification messages to inform each other of errors.

Table 49-21: Parameters in output of the **show bgp peer** command for a specific peer

Parameter	Meaning
OutNotification	Number of notification messages sent to this peer. BGP peers use notification messages to inform each other of errors.
InRouteRefresh	Number of route refresh messages received from this peer.
OutRouteRefresh	Number of route refresh messages sent to this peer.

Example To display summary information for all BGP peers, use the command:

```
sh bgp pe
```

To display detailed information for the BGP peer 192.168.1.1, use the command:

```
sh bgp pe=192.168.1.1
```

Related Commands

add bgp peer
delete bgp peer
set bgp peer
show bgp
show ip routemap

show bgp peertemplate

Syntax `SHow BGP PEERTemplate[=1..30]`

Description This command displays information about all BGP peer policy templates, or about the specified BGP peer policy template (Figure 49-21, Table 49-22).

The **peertemplate** parameter specifies the identification number of the BGP peer policy template. If you do not specify a value, information is displayed for all BGP peers.

Figure 49-21: Example output from the **show bgp peertemplate** command

```
BGP Peer Template Information
------------------------------------------------------------
Template.............. 1
Description ......... -
Connect retry ....... 120s
Hold time ........... 90s
Keep alive .......... 30s
Min AS originated ... 15
Min route advert .... 30

Filtering
  In filter ......... -
  In path filter .... -
  In route map ...... -
  Out filter ........ -
  Out path filter ... -
  Out route map ..... -

Max prefix .......... OFF
Next hop self ....... No
Send community ...... No

------------------------------------------------------------
```

Table 49-22: Parameters in output of the **show bgp peertemplate** command

Parameter	Meaning
Template	ID number of the template.
Description	Description for peers that use the template.
Connect Retry	Interval for retrying the initial TCP connection to peers that use the template in the event of a connection failure.
Hold time	The configured hold time for peers that use the template. The actual hold time is the lower of the configured hold times of the peer and this router.
Keep alive	The configured keepalive time for peers that use the template. The actual keepalive time is set by the actual hold time in such a way that the ratio of actual keepalive to hold time is the same as the ratio of configured keepalive to hold time.
Min AS originated	Minimum seconds between advertisements from the router to peers that use the template for routes that originate in this autonomous system.

Table 49-22: Parameters in output of the **show bgp peertemplate** command (continued)

Parameter	Meaning
Min route advert	Minimum seconds between advertisements from the router to peers that use the template for routes that originate outside this autonomous system.
Filtering	Settings for inward and outward filtering of routing information via BGP.
In filter	Traffic filter used for filtering incoming routes from peers that use the template.
In path filter	AS path filter used for filtering incoming update messages from peers that use the template.
In route map	Route map used for filtering incoming routes or update messages from peers that use the template, and/or for setting their attributes.
Out filter	Traffic filter used for filtering outgoing routes to peers that use the template.
Out path filter	AS path filter used for filtering outgoing update messages to peers that use the template.
Out route map	Route map used for filtering outgoing routes or update messages to peers that use the template, and/or for setting their attributes.
Max prefix	Maximum number of route prefixes that may be received from peers that use the template, and the action taken when this number is exceeded. The action is WARNING or TERMINATE.
Next hop self	Whether this router advertises to peers using the template that the next hop for all routes is itself.
Send community	Whether this router sends the community attribute in the path attributes of update messages to peers that use the template.

Example To display the settings of peer template 1, use the command:

```
sh bgp peert=1
```

Related Commands

add bgp peer
add bgp peertemplate
delete bgp peertemplate
set bgp peer
show bgp

show bgp route

Syntax

```
SHow BGP ROUte[=prefix] [REGexp=aspathregexp]
   [COMmunity={INTernet|NOAdvertise|NOExport|
   NOEXPORTSubconfed|1..4294967295}]
```

Description

This command displays information about some or all routes in the BGP routing table (Figure 49-22 on page 49-130 and Table 49-23 on page 49-130).

Parameter	Description	
ROUte	The network prefix of the routes to display. The *prefix* is an IP address in dotted decimal notation, optionally followed by the CIDR mask. All routes that match the prefix or that are subnets of the prefix are displayed. If you do not specify a prefix, the router displays all BGP routes that match the **regexp** and **community** specified. Default: no default	
REGexp	An AS path regular expression. The router only displays routes with an AS path attribute that matches the regular expression. Regular expressions are a list of one or more AS numbers separated by spaces. To match from the first number in the list, start the expression with the ^ character. To match the last number, end with the $ character. If the expression contains spaces, surround it with double quotes. For more information about valid syntax, see Table 49-8 on page 49-27. For example: • **regexp="23334 45634 88988"** displays any route with a path containing these numbers • **regexp="^23334 45634 88988$"** displays any route with that exact path • **regexp=^23334** displays any route with a path beginning with 23334 Default: no default	
COMmunity	A community name or number. The router only displays routes with this value of the community attribute. Note that if you specify a community, routes that do not contain a community attribute are not displayed. Default: no default	
	INTernet	The community of routes that can be advertised to all BGP peers.
	NOExport	The community of routes that must not be advertised outside a BGP confederation boundary (a standalone autonomous system that is not part of a confederation should be considered a confederation itself).
	NOAdvertise	The community of routes that must not be advertised to other BGP peers.
	NOEXPORTSubconfed	The community of routes that must not be advertised to external BGP peers (this includes peers in other members' autonomous systems inside a BGP confederation).
	1..4294967295	The number of a community.

Tip The shortest string you can enter is shown in capital letters.

Figure 49-22: Example output from the **show bgp route** command

```
BGP route table
-------------------------------------------------------------------------------
   Prefix              Next hop          Origin      MED           Local pref
   Path
-------------------------------------------------------------------------------
 > 10.0.0.0/8          0.0.0.0           INCOMPLETE  0             100
   EMPTY

 * 12.12.0.0/16        11.0.0.1          INCOMPLETE  0             100
   SEQ 1023 1024 1025;

 > 192.168.1.0/24      10.89.0.1         IGP         0             100
   EMPTY
-------------------------------------------------------------------------------
```

Table 49-23: Parameters in the output of the **show bgp route** command

Parameter	Meaning
Prefix	Network prefix for this route.
Next hop	IP address of the next hop for this route. A next hop of 0.0.0.0 indicates it is an interface route.
Origin	Origin attribute for this route; one of IGP, EGP, or Incomplete.
MED	Multi Exit Discriminator attribute for this route.
Local pref	Local preference attribute for this route.
Path	The AS path attribute for this route.
>	The best route for the prefix.
*	This route has an unreachable next hop and has been suppressed from selection.
A	An aggregate route.
S	This route has been suppressed as part of aggregation.

Examples To display the BGP route table for routes that pass through AS 1234, use the command:

```
sh bgp rou reg=1234
```

Related Commands

show bgp
show bgp aggregate
show bgp import
show bgp network
show bgp peer

show ip aspathlist

Syntax `SHow IP ASPATHlist[=1..99]`

Description This command displays information about a specific AS path list or all lists in the router (Figure 49-23, Table 49-24).

Figure 49-23: Example output from the **show ip aspathlist** command

```
IP AS path lists

List    Entry    Regular expression
----------------------------------------
1       1        Include ^$
        2        Exclude .*
----------------------------------------
34      1        Exclude ^123
        2        Include 345 234.+123
        3        Exclude .*
----------------------------------------
```

Table 49-24: Parameters in the output of the **show ip aspathlist** command

Parameter	Meaning
List	AS path list number from 1 to 99.
Entry	Entry in the AS path list from 1 to the number of entries in the list.
Regular expression	AS path regular expression for this entry. This is preceded by "exclude" or "include" to indicate what the router does when there is a a match. For a description of regular expressions, see Table 49-8 on page 49-27.

Examples To display AS path list number 23, use the command:

```
sh ip aspath=23
```

Related Commands

add ip aspathlist
delete ip aspathlist

show ip communitylist

Syntax `SHow IP COMmunitylist[=1..99]`

Description This command displays information about a specific community list or all lists in the router (Figure 49-24, Table 49-25).

The **communitylist** parameter specifies the community list to display. If a list is not specified, all are displayed.

Figure 49-24: Example output from the **show ip communitylist** command

```
IP community lists

List     Entry     Community list
-----------------------------------------------
 1       1         Include noexport,1234,2345
         2         Exclude 34567,123
 -----------------------------------------------
 23      1         Exclude 1
         2         Include internet
 -----------------------------------------------
```

Table 49-25: Parameters in the output of the **show ip communitylist** command

Parameter	Meaning
List	Number of community list from 1 to 99.
Entry	Entry in the community list from 1 to the number of entries in the list.
Community list	The community list for this entry, preceded by "exclude" or "include" to indicate whether a match means that the community attribute should be excluded or included in the function for which the community list is being used.

Examples To display all IP community lists, use the command:

```
sh ip com
```

Related Commands
add ip communitylist
delete ip communitylist

show ip prefixlist

Syntax `SHow IP PREFIXList[=name]`

Description This command displays information about prefix lists on the router. If you specify a prefix list name, detailed information about that prefix list and its entries is displayed (Figure 49-26, Table 49-27). Otherwise, summary information about all existing prefix lists is displayed (Figure 49-25, Table 49-26).

Figure 49-25: Example summary output from the **show ip prefixlist** command

```
IP Prefix Lists
Name              Entries     In Use
-----------------------------------
Sample                 11     Yes
Test                    3     No
-----------------------------------
```

Table 49-26: Parameters in the output of the **show ip prefixlist** command

Parameter	Meaning
Name	The name of the prefix list.
Entries	The number of entries in the prefix list.
In Use	Whether the prefix list is currently assigned to a route map.

Figure 49-26: Example detailed output from the **show ip prefixlist** command

```
IP Prefix List
-------------------------------------------------------------------
 Name ................... Sample
 In Use ................. Yes
 Entries:
   Number       Action         Prefix            Length Range
   ---------------------------------------------------------------
        1       Match          192.168.0.0                 16
        3       No Match       0.0.0.0                  25-30
       10       No Match       10.10.10.0               24-30
   ---------------------------------------------------------------
```

Table 49-27: Parameters in the detailed output of the **show ip prefixlist** command

Parameter	Meaning
Name	Name of the prefix list.
In Use	Whether the prefix list is currently assigned to a route map.
Number	The entry number of the prefix list entry. The router checks entries in order, starting with the lowest entry number.
Action	Whether the prefix list includes (“match”) or excludes (“nomatch”) any prefix that is within the entry’s prefix range.
Prefix	IP network address for the entry to match on.
Length Range	Range of CIDR mask lengths that the entry can match on.

Examples To see the entries in prefix list “office”, use the command:

```
sh ip prefixl=office
```

Related Commands

add ip prefixlist
add ip routemap
delete ip prefixlist
set ip routemap

show ip routemap

Syntax `SHow IP ROUTEMap[=`*routemap*`]`

where *routemap* is a character string 0 to 15 characters long. Valid characters are uppercase and lowercase letters, digits (0-9), and the underscore character ("_").

Description This command displays information about all IP route maps or a specific one (Figure 49-27, Table 49-28).

The **routemap** parameter specifies the name of the route map to display. If one is not specified, information about all route maps is displayed.

Figure 49-27: Example output from the **show ip routemap** command

```
IP route maps

Map name
  Entry        Action
      Clauses
-----------------------------------------------------
test
  1            Include
     match     Community   12 Exact=no
       set     LocalPref   3245
       set     Med         8726
       set     Origin      incomplete
  12345        Include
       set     Community   12 noadvertise Add=yes
  4294967295   Include
       set     AS-path     44
       set     Local Pref  3245
       set     Med         8762
       set     Origin      igp
-----------------------------------------------------
```

Table 49-28: Parameters in the output of the **show ip routemap** command

Parameter	Meaning
Map name	Name of the route map.
Entry	Entry number for the route map entry. Entry numbers can be any number, but all entries within a route map are sorted by entry number.
Action	Whether the action for this route map entry is include or exclude.
Clauses	The match and set clauses for this route map entry. Each entry can have only one match clause, and only one set clause of a given type.

Examples To display the IP route map with the name "ospf_bgp_map", use the command:

```
sh ip routem=ospf_bgp_map
```

Related Commands

add ip routemap
delete ip routemap
set ip routemap