Chapter 2

Overview and Statement Summary

2.1 What is GateD?

GateD is a modular software program consisting of

- core services
    - Check sum
    - AS Path Regular Expression parsing
    - Multiple RIBs
    - GateD Interactive Interface (GII)
    - Task and Timer functionality
    - Weighted Route Damping
- protocol modules supporting multiple routing protocols
    - RIP versions 1 and 2
    - OSPF
    - IS-IS
    - BGP
    - MP-BGP
    - DVMRP
    - IGMP
    - PIM-SM
    - MSDP
    - IS-IS for v6
    - RIPng

GateD was first used to interconnect the NSFNet and emerging regional networks and to implement policy-based dynamic routing. GateD provides complete policy control for your layer 3 IP routing, which allows a network administrator to control import and export of routing information by

- individual protocol
- source and destination autonomous system
- origin attribute
- source and destination interface
- previous and next hop router
- specific destination address

The network administrator can specify a preference level for each combination of routing information that is imported by using a flexible masking capability. Once the preference levels are assigned, GateD makes a decision about which route to use independent of the protocols involved.

GateD relies on the underlying operating system to provide some facilities, such as forwarding and layer 2 management. It was designed with porting in mind and has been ported to most popular operating systems.

2.2 How to Configure GateD

GateD reads its configuration from a file (`gated.conf`) which consists of a sequence of statements, each terminated by a semi-colon (";"). Statements are composed of commands (and sometimes variables) separated by white space, which can be any combination of blanks, tabs, and newlines. This structure makes it easy to identify parts of the configuration that are associated with each other and with specific protocols. Comments can be specified in either of two forms. One form begins with a pound sign ("#") and runs to the end of the line. The other form, C style, starts with "/*" and continues until it reaches "*/ ."

2.3 Statement Grouping

The configuration statements and the order in which these statements appear divide a `gated.conf` file into six groups:

1. options group (See "Chapter 6 Options Statements" on page 21 for more information.)
2. interface group (See "Chapter 7 Interface Statement" on page 23 for more information.)
3. definition group (See "Chapter 8 Definition Statements" on page 29 for more information.)
4. protocol group (See Chapters 11 through 27 for unicast, multicast, and IPv6 protocol statements.
5. static group (See "Chapter 19 Static Routes" on page 101 for more information.)
6. control and aggregate group (See "Chapter 33 Route Aggregation and Generation" on page 155 for more information.)

See "Table 1: Summary of GateD Configuration Statements" on page 7 for more information about these groupings.

If you enter a grouping out of order, an error occurs when the configuration file is parsed.

Two other types of statements do not fit into these categories: directive statements and trace statements. These statements provide instructions to the parser and control tracing. They do not relate to the configuration of any protocol, and they can occur anywhere in the `gated.conf` file. (See "Chapter 5 Directive Statements" on page 19 for more information about directive statements and "Chapter 4 Trace Statements" on page 15 for more information about trace statements.)

2.4 Address and Prefix Formats

GateD allows configuration of both IPv4 and IPv6 address types. Normally GateD can recognize which type of address is being configured in a particular instance by the format of the address.

IPv4 addresses are 32 bits long. The formats of Ipv4 addresses recognized by GateD are:

d

d.d

d.d.d

d.d.d.d

where *d* represents a number in the range 0-255 inclusive. Each *d* specifies 8 bits of the address. If fewer than four *d* values are provided then the values provided specify the high order values of the address. For example, **`192.168`** is equivalent to **`192.168.0.0`**.

For IPv6 addresses, the forms recognized are those specified in RFC 237. IPv6 addresses are 128 bits long and can be specified in one of three forms. In the preferred form, the address is specified as 8 hexadecimal values separated by '`:`'s. Each hexadecimal value specifies 16 bits. The hexadecimal digits a-f can be specified in either upper or lower case. An example of the preferred form is:

`3FFd:201:1:7fff:0:0:0:a`

The compressed form allows consecutive 0 bits in an IPv6 address to be specified as '`::`'. The `::` may only appear once in a given address. Examples of the compressed form are:

`::` (equivalent to **`0:0:0:0:0:0:0:0`**)

`3ffd:201:1:7fff::a` (equivalent to the preferred address example above)

`::1` (equivalent to **`0:0:0:0:0:0:0:1`**, the IPv6 loopback address).

The third form provides a way to specify IPv4 addresses to be embedded in an IPv6 address. In this case, the last 32 bits of the address are specified in dotted-quad form (all four *d* values are required). Examples of this address form are:

`::192.168.0.0`

`::ffff:64.32.1.0`

In the case where an IPv6 Link-Local address is being configured, GateD allows the interface associated with the Link-Local address to be specified as follows:

`fe80::%`*`interface_name`*

where *`interface_name`* is the name of a physical interface on the machine on which GateD is running. Gated inserts the interface index associated with *`interface_name`* in the Link-Local address. This form is only valid for Link-Local addresses.

In many cases IPv4 and IPv6 addresses are combined with masks to configure prefixes. There are two methods for specifying the mask: It can be specified as an IPv4 or IPv6 address proceeded by the **`mask`** keyword; or it can be specified as a length proceeded by the **`masklen`** keyword or, more conventionally, by a '`/`'. In the "**`mask`**" case, the address type of the mask must match the address type. Currently only contiguous bit masks are allowed in GateD. Any non-zero address bits in positions that are covered by the specified mask cause a parse error. Example prefix specifications are:

`10/8`

`10.0.0.0 mask 255.0.0.0` (equivalent to 10/8)

`10 masklen 8` (equivalent to 10/8)

`3ffd::/16`

`3ffd:: mask ffff::` (equivalent to 3ffd::/16)

`3ffd:: masklen 16`	(equivalent to 3ffd::/16)
`::/0`	(IPv6 default address)
`0/0`	(IPv4 default address)
`192.168.1/16`	(invalid because the `.1` is not covered by the mask)

When configuring route filters (See “Chapter 28 Route Filtering” on page 129) the keywords **`default`** and **`all`** are used to indicate an address that matches only the default address and the address that matches any address, respectively. In contexts where either an IPv4 or IPv6 address filter could be specified it is sometimes necessary to proceed either of these keywords with an **`inet`** or **`inet6`** to indicate the desired address family. In such contexts, specifying **`all`** or **`default`** by itself is taken to mean both address families. For example, within a BGP import or export statement (when MPBGP is enabled)

```
{ all ; } ;
```

Is equivalent to

```
{inet all ; inet6 all ; } ;
```

Note that **`ipv4`** and **`inet4`** are synonyms for **`inet`** and **`ipv6`** is a synonym for **`inet6`**.

Although its use usually isn't recommended, GateD provides a feature where host names can be used to configure certain addresses. The syntax for this type of configuration is

```
host [ inet | inet6 ] host_name
```

The optional **`inet`** or **`inet6`** keyword indicates the type of address desired as a result of the name resolution. If neither is specified, the type of address sought is determined by context.

The **`host`** keyword can also proceed an IPv4 or IPv6 address instead of a *`host_name`*. In this case the mask associated with the prefix will be the host mask for the address family of the specified address. For example

```
host 1.2.3.4
```

is equivalent to

```
1.2.3.4/32
```

In contexts where an IPv4 prefix is allowed, GateD currently allows just an IPv4 address to be specified. It then uses the "natural mask" as the mask for the prefix. While this "feature" is still provided by GateD, its use is strongly discouraged.

2.5 Route Preference and Route Selection

Preference is the value GateD uses to order preference of routes from one protocol or peer over another. Preference can be set in the GateD configuration file in several different configuration statements. Preference can be set based on one network interface over another, from one protocol over another, or from one remote gateway over another. Flexibility of preference configuration varies with the type of protocol being configured. (See “Chapter 3 Preferences and Route Selection” on page 11 for more information about Route Preference.)

2.6 Statement Summary

Table 1 lists each GateD configuration statement by name, indicates the chapter in which the statement is described, identifies the statement group, provides a short synopsis of the statement's function, and lists the type of protocol. More detailed definitions and descriptions of each of the eight groups of GateD statements follow.

These statements must appear in statement group order in the configuration file. (For example, if definition and protocol statements are both to be included, definition statements must precede the protocol statements.)

Table 1: Summary of GateD Configuration Statements

Statement Name	Chapter in which the Statement is Described	Statement Group	Statement Function	Type of Protocol
`traceoptions`	Chapter 4	Trace	Global tracing parameters	n/a
`%directory`	Chapter 5	Directive	Sets the directory for include files	n/a
`%include`	Chapter 5	Directive	Includes a file in `gated.conf`	n/a
`options`	Chapter 6	Option	Sets GateD options	n/a
`interfaces`	Chapter 7	Interface	Defines GateD interfaces	n/a
`autonomous-system`	Chapter 8	Definition	Sets the AS number for this router	n/a
`routerid`	Chapter 8	Definition	Sets the router ID for BGP and OSPF	n/a
`martians`	Chapter 8	Definition	Defines invalid destination addresses	n/a
`rip`	Chapter 11	Protocol	Configures RIP protocol	Unicast
`ospf`	Chapter 12	Protocol	Configure OSPF protocol	Unicast
`isis`	Chapter 13	Protocol	Configures IS-IS protocol	Unicast
`bgp`	Chapter 14	Protocol	Configures BGP protocol	Unicast

Table 1: Summary of GateD Configuration Statements

Statement Name	Chapter in which the Statement is Described	Statement Group	Statement Function	Type of Protocol
`routerdiscovery`	Chapter 15	Protocol	Configures the Router Discovery protocol	Pseudo-protocol
`icmp`	Chapter 16	Protocol	Configures the processing of general ICMP packets	Pseudo-protocol
`redirect`	Chapter 17	Protocol	Configures the processing of ICMP redirects	Pseudo-protocol
`kernel`	Chapter 18	Protocol	Configures kernel interaction options	Pseudo-protocol
`dvmrp`	Chapter 20	Protocol	Configures DVMRP protocol	Multicast
`pim`	Chapter 21	Protocol	Configures PIM protocol	Multicast
`msdp`	Chapter 23	Protocol	Configures MSDP protocol	Multicast
`igmp`	Chapter 24	Protocol	Enables IGMP	Multicast
`multicast`	Chapter 25	Protocol	Sets interface-specific multicast options	n/a
`ripng`	Chapter 27	Protocol	Configures RIP next generation protocol	Unicast (v6)
`static`	Chapter 19	Static	Configures static routes	n/a
`mstatic`	Chapter 26	Static	Configures static group joins	n/a
`import`	Chapter 31	Control	Sets policy for the routes installed in the RIBs	n/a

Table 1: Summary of GateD Configuration Statements

Statement Name	Chapter in which the Statement is Described	Statement Group	Statement Function	Type of Protocol
`export`	Chapter 32	Control	Sets the policy for the routes to export from one protocol to another	n/a
`aggregate`	Chapter 33	Control	Sets the aggregation policy	n/a
`generate`	Chapter 33	Control	Sets conditions for generating a default route	n/a