Installing GateD

V.9.3.2

Copyright Notice

Trademark Notice

Contents

Chapter 1
Autoconfiguration

1.1 About This Manual

Installation options for GateD depend on which version of GateD you are using and on how many architectures GateD will run.

You can use the autoconfiguration option if you are running on one of the supported operating systems. Supported operating systems include:

- Solaris 8 (32 and 64 bit)
- NETBSD 1.5
- FreeBSD 4.2
- BSD/OS 4.2
- Linux 2.4
- VxWorks 5.4.2

For a complete list, contact support@nexthop.com.

This manual gives examples of how to use autoconfiguration for a single-architecture build or for a multiple-architecture build. It also gives information on using manual configuration.

1.2 Overview

GateD uses GNU autoconf to detect platform-specific configuration options. Future versions will also support autoconfiguration. You do not need to install autoconf to build GateD. By default, GateD is compiled with all protocols available. You can disable support for some protocols using command-line flags.

1.3 Getting Started

To compile GateD for a single architecture with all supported protocols, use the following from the top level of the GateD release:

```
./configure

make depend

make
```

1.4 Compiling GateD for Multiple Platforms

To compile GateD for multiple platforms, do not build GateD in the original location. If you have already compiled for a single architecture, you can clean out the original location by typing:

```
make distclean
```

`make distclean` may be used to clean the program binaries and object files from the build directories. This target also removes files that **`configure`** created.

The example below shows how to build GateD for a single architecture, using **`gated-u`** as the source base for the directory:

```
cd gated-u

mkdir -p obj/myarch

cd obj/myarch

../../configure

make depend

make
```

New directories should be created under **`obj`** for each supported architecture by reiterating the above steps for different **`myarch`**'s

Note: When **`make distclean`** is used, the Makefiles generated in directories that were not entered during the build process are not removed. This set of directories is configured based on user request (e.g., flags to '**`configure`**') and operating system support.

1.5 Disabling Support for Specific Protocols

A list of compile time options can be generated by typing:

```
configure --help
```

Protocols may be disabled at compile time to prevent them from being compiled into the GateD binary. The **`--disable-all`** flag can be used to disable all protocols, allowing fine control over the enabled subset. For example, to build a GateD binary with only the OSPF protocol use:

```
configure --disable-all --enable-ospf
```

Protocol dependencies are automatically resolved by **`configure`**. For example, enabling the BGP module causes the ASPATH's module to be enabled.

Chapter 2
System Installation Notes

2.1 Overview

GateD relies on certain services of the UNIX kernel for the TCP/IP stack. These services include:

- IP forwarding table
- Setting of UDP checksum in some packets
- SNMP support (via snmpd/smux)
- Interfaces with the kernel to query interface status, routes, and timers
- Multicast support

GateD runs on a large number of platforms using the system functions specified above. To select the correct set of system functions, GateD uses autoconf-generated configuration files to build compile-time configurations. Refer to the Core Porting Guide for details on required functions and instructions on handling their absence.

This section provides notes on how to configure UNIX system functions that are needed on various systems.

2.2 UDP Checksums

RIP will refuse to run if it determines that UDP checksums are disabled in the kernel. Running without UDP checksums can lead GateD to propagate incorrect routing information, especially on serial links. This check does not help you determine if the RIP packets you receive are missing a checksum, but at least it prevents you from generating these packets and calls attention to the problem.

2.3 IP Multicast Support

The OSPF and RIP implementations make use of IP multicasting facilities. If these facilities are not present, functionality is reduced.

In order to support legacy operating systems, a route is installed in the unicast FIB with a next hop of the loopback interface, for each multicast group that is joined. This ensures that GateD receives copies of packets that it sends to a given multicast group.

2.3.1 IP Multicast Routing Support

GateD includes support for multicast routing protocols. It implements the router portion of the IGMP protocol to determine local group membership information. Check your license agreement to see what multicast protocols are available in your GateD package. GateD is generally compatible with the 3.3 and 3.5 releases of the IP Multicast distribution from Xerox Parc.

2.3.2 KRT_IPMULTI_TTL0

With this model, the kernel also acts as a protocol independent multicast forwarding cache. Forwarding cache entries are generated on demand as data traffic is forwarded. It uses messages on the IGMP socket to communicate with the routing daemon and request that a forwarding cache entry be calculated. Examples of the use of this model include the *Xerox PARC IP Multicast 3.3 and 3.5* kernel releases.

2.4 Interfacing to the SNMP SMUX Interface

2.4.1 SMUX

GateD supports SNMP via the RFC 1227 SMUX interface. It is compiled and enabled by default. In order to retrieve SNMP information from GateD, a master agent must be running and listening for SMUX connections on TCP port 199.

GateD is known to interoperate with the net-snmp (UC-Davis) SNMP agent, which can be found at:
net-snmp.sourceforge.net

Instructions for building the master agent with SMUX support can be found in the distribution.

2.5 Running GateD on Various Systems

2.5.1 Turn off ICMP Redirect in Kernel

HPUX and any other system that lacks a routing socket must have ICMP when building RIP and BGP. Turn off ICMP-REDIRECT in the kernel (with **`ndd-sst /dev/ip ip-ignore_redirect`**). GateD handles the ICMP redirects.

2.5.2 Running GateD on BSD

For *BSD* variants, make sure ip-forwarding is enabled.

2.6 Compiling GateD on Systems with Shared Libraries

If an assertion failure occurs in **`task_stdio_read()`** it is because a file descriptor was improperly closed. This can occur when the named resolver libraries are improperly installed in the system shared libraries. If the socket used by the shared libraries is not statically initialized to `-1`, file descriptor zero will be closed when GateD calls **`endhostent()`**. The solution is to fix the shared libraries. A workaround would be to avoid using any symbolic names in the config file and specify **`options noresolv ;`**.

Chapter 3
Customizing Your Build Configuration

3.1 Overview

This release of GateD uses GNU autoconf to detect platform-specific configuration options. You do not need to install autoconf to build GateD. By default, GateD is compiled with all protocols available in the license under which it was distributed.

This section documents how the build process may be configured to set the protocols and features compiled in the GateD binary and change various compile-time parameters.

Several binaries may be produced by a compile of GateD, including standalone utilities such as 'ripquery' and 'gdc'. These binaries will be present in a directory off of 'src' named the same. The 'gated' binary is produced in src/gated.

Most build configuration is performed by passing command-line flags to the 'configure' script. For a complete list of options, use:

```
./configure --help
```

3.2 Build Tools

The autoconf system provides some built-in flags for setting the tools to be used for compilation. The GNU web site, www.gnu.org, is the most complete source of documentation for these options. A few important ones are documented here.

3.2.1 C Compiler

The path to the C compiler to be used may be set with either the **`--with-cc`** flag or by setting the $CC environment variable. For example, to build GateD using cc, use:

```
./configure --with-cc=cc
```

3.2.2 Parser Generator

GateD uses a yacc-based parser for parsing the configuration file. This requires a tool such as yacc or bison to generate the C source code from a set of rules. The path to this tool may be set by using the **`--with-yacc flag`** or by setting the $YACC environment variable. For example, to use bison, use:

```
./configure --with-yacc=bison
```

3.2.3 Lex Scanner

GateD requires a lexer generator, which generates a lexer C source file from a set of rules. To set the path to this tool, use the **`--with-lex flag`** or set the $LEX environment variable. For example, to use flex, use:

```
./configure --with-lex=flex
```

3.2.4 Installation Paths

See the GNU autoconf manual for a complete list of options concerning the installation paths. Compilation of GateD will typically result in several standalone binaries, which may be installed in different locations. The GateD binary is installed in the 'sbin' directory specified by 'configure'.

3.3 Protocols

The 'configure' script may be used to configure a set of protocols to be built in this GateD binary. Most protocol modules have **`--enable`** and **`--disable`** flags, which may be used to enable or disable the protocol, respectively. The **`--disable-all`** flag is also useful because it allows the user to disable all protocols and then enable a small subset. For example, to build a GateD binary with only RIP, use:

```
./configure --disable-all --enable-rip
```

Dependencies are automatically resolved by 'configure'. Enabling the BGP module, for example, also enables the ASPATHS module.

3.4 Standard Features

Some features of GateD can be enabled or disabled at compile time. This is in addition to the protocols themselves; the features documented here are optionally enabled to support some particular functionality or different mode of operation.

All of the features defined here can be enabled or disabled by passing flags to the 'configure' script, which is run before building GateD.

3.5 Additional Features

In addition to this standard set of features, there are some other parameters that can be manipulated to fine-tune the protocols' compile-time configuration. These parameters are documented here. All parameters are disabled by default.

Feature: GII_DEBUG_MENU

Flag: --enable-developer

The GateD Interactive Interface (GII) contains some protocol debugging functions. For example, one function allows a neighbor's Router-LSA to be sequence-wrapped and reflooded. These functions are available in the 'debug' submenu when GII_DEBUG_MENU is enabled.

In addition to enabling GII_DEBUG_MENU, the --enable-developer flag also turns on some strict compiler warning flags when certain types of compilers are used (gcc is one of them).

Feature: NOSPF_ALWAYS_CYCLE_MEMBERSHIP

Flag: --enable-ospf_dropfirst

Some versions of BSD exhibit a bug that prevents a multicast group from being dropped on a logical interface when that interface has been deleted. This option forces GateD to always drop and join when attempting to join a multicast group.

Feature: FEATURE_NOSPF_ALTQ_TE

Flag: --enable-ospf_altq_te

In the new_ospf protocol module there exists code to support learning bandwidth information from the /dev/cbq device in a BSD kernel with ALTQ patches. This code requires that the ALTQ header files exist in /usr/include/altq.

Feature: FEATURE_NOSPF_FAST_SPF

Flag: --enable-ospf_fast_spf

This feature expands the size of the **`vertex_t`** structure (the structure used to represent a Link State Advertisement) in order to gain speed in the SPF. An extra pointer is added, increasing the size by one machine word (typically 32 bits). The result is an approximate twenty-five percent reduction of computation time required by the SPF. The pointer is used to keep track of a candidate list entry for a vertex.

Feature: FEATURE_INTERFACE_AGING

Flag: --enable-interface_aging

Interface aging is a legacy behavior of GateD. In older code bases, unless the 'passive' option was specified on an interface, an interface that did not receive routing protocol traffic for a certain period of time was downed. This feature is now disabled unless explicitly requested here.