*Chapter 8*

# IRIX sendmail

This chapter describes IRIX *sendmail,* a facility for routing mail across an internetwork. This chapter is for system administrators who set up and maintain the mail system on a station or network. It provides the information necessary for a straightforward implementation of *sendmail*. The following topics are covered:

- "The Mail System" on page 171
- "An Overview of sendmail" on page 173
- "How sendmail Works" on page 175
- "Aliases Database" on page 181
- "sendmail Network Configurations" on page 184
- "User-Configurable Macros and Classes" on page 186
- "sendmail Planning Checklist" on page 188
- "Configuring sendmail" on page 189
- "Managing sendmail" on page 201
- "Sendmail Questions, Problems, and Troubleshooting" on page 207
- "Notes to Current sendmail Users" on page 208

For sites already using *sendmail*, refer directly to "Notes to Current sendmail Users" on page 208.

For additional reference material on IRIX *sendmail*, see Appendix B, "IRIX sendmail Reference."

## The Mail System

The mail system is a group of programs that you can use to send messages to and receive messages from other users on the network. You can send mail through either UUCP or

TCP/IP. The IRIX operating system uses MediaMail, System V */bin/mail,* 4.3BSD */usr/sbin/Mail,* and *sendmail* for its mail implementation.

The process of delivering mail involves four elements:

| | |
|---|---|
| User Interface | The user interface creates new messages and reads, removes, and archives received messages. MediaMail, System V */bin/mail*, and 4.3BSD */usr/sbin/Mail* are the user interfaces provided with IRIX. Reference pages are available to fully describe the features of these interfaces, and MediaMail has an extensive online help system. |
| Mail Routing | A mail router examines each message and routes it through the network to the appropriate station. The *sendmail* program not only routes messages, but also formats them appropriately for their recipient stations. |
| Mail Transfer | A mail transfer program transmits messages from one station to another. *sendmail* implements the Simple Mail Transfer Protocol (SMTP) over TCP/IP. For TCP/IP mail, *sendmail* acts as an integrated routing and transfer program. In all cases, mail transfer has a counterpart: mail reception. In most cases, a single program provides both functions. UUCP is a mail transfer program that uses its own protocols and runs over serial lines. |
| Mail Delivery | A mail delivery program deposits mail into a data file for later perusal by a user or another program. The */bin/mail -d* program delivers local mail. |

After you compose a message by using MediaMail, */bin/mail*, or */usr/sbin/Mail*, the message is sent to *sendmail,* which attempts to determine the destination of the message. *sendmail* either calls */bin/mail* (for mail to a user on the local station) or passes the message to the appropriate mail transfer program (for mail to a user on a remote station).

When *sendmail* receives a message from another station, it analyzes the recipient address; then, it either calls */bin/mail* to complete the delivery if the local station is acting as a relay, or passes the message to the mail transfer program. For TCP/IP SMTP, *sendmail* also performs the mail transfer.

When you send a mail message on a network that uses TCP/IP, several layers of network software are involved. Figure 8-1 shows the layers of TCP/IP mail network software.

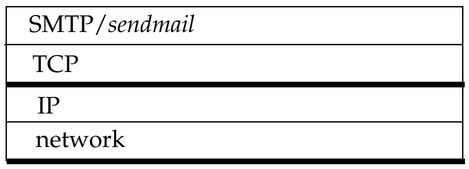

**Figure 8-1** Layers of TCP/IP Mail Software

## An Overview of sendmail

The Transmission Control Protocol (TCP) layer supports SMTP, which *sendmail* uses to transfer mail to other TCP/IP stations. *sendmail* is responsible for calling local delivery programs, mail routing, and TCP/IP mail transfer; it may also call other mail transfer programs. For example, *sendmail* uses the UUCP transmission program to handle messages sent to UUCP stations.

*sendmail*'s implementation features aliasing, forwarding, automatic routing to network gateways, and flexible configuration.

In a simple network, each node has an address, and resources can be identified with a host-resource pair. For example, a mail system can refer to users with a host–user-name pair. Station names and numbers must be administered by a central authority, but user names can be assigned locally to each station.

In an internetwork, multiple networks with different characteristics and management must communicate. In particular, the syntax and semantics of resource identification change. You can handle certain simple cases by using improvised techniques, such as providing network names that appear local to stations on other networks. However, the general case is extremely complex. For example, some networks require point-to-point routing, which simplifies the database update problem, because only adjacent stations are entered into the system tables; others use end-to-end addressing. Some networks use a left-associative syntax; others use a right-associative syntax, causing ambiguity in mixed addresses.

Internetwork standards seek to eliminate these problems. Initially, these standards proposed expanding the address pairs to address triples, consisting of *network*, *station*, *resource*. Network numbers must be universally agreed upon; stations can be assigned locally on each network. The user-level presentation was quickly expanded to address domains, composed of a local resource identification and a hierarchical domain specification with a common static root, as defined in RFC 1034. The domain technique separates the issue of physical versus logical addressing. For example, an address of the form "jane@iris1.company.com" describes only the logical organization of the address space.

*sendmail* bridges the gap between the world of totally isolated networks that know nothing of each other and the clean, tightly coupled world of unique network numbers. *sendmail* can accept old arbitrary address syntaxes, resolving ambiguities by using heuristics specified by the network administrator, as well as domain-based addressing. *sendmail* helps guide the conversion of message formats between disparate networks. In short, *sendmail* is designed to assist a graceful transition to consistent internetwork addressing schemes.

## System Organization

The design goals for *sendmail* included the following:

1. Message delivery should be reliable, guaranteeing that every message is correctly delivered or at least brought to the attention of a human for correct disposal; no message should ever be completely lost.
2. Existing software should be used to do actual message delivery whenever possible.
3. *sendmail* should be easy to expand to fairly complex environments.
4. Configuration should not be compiled into the code.
5. *sendmail* should let various groups maintain their own mailing lists, and let individuals specify their own forwarding, without modifying the station's alias file.
6. Each user should be able to specify the mailer to execute to process mail being delivered. This feature allows users with specialized mailers that use a different format to build their environments without changing the system, and facilitates specialized functions (such as returning an "I am on vacation" message).
7. To minimize network traffic, addresses should be batched to a single station where possible, without assistance from the user.

Figure 8-2 illustrates the *sendmail* system structure that is based on the original design goals for *sendmail.*

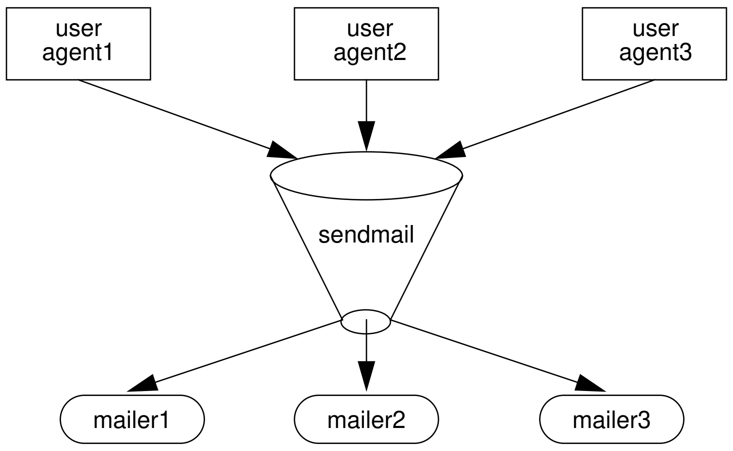

**Figure 8-2** sendmail System Structure

*sendmail* neither interfaces with the user nor does actual mail delivery. Rather, it collects a message generated by a user agent program such as Berkeley *Mail*, edits the message as required by the destination network, and calls appropriate mailers to do mail delivery or queueing for network transmission. The exception is mail sent to a file; in this case, *sendmail* delivers the mail directly.

This discipline allows the insertion of new mailers at minimum cost.

Because some of the senders may be network servers and some of the mailers may be network clients, *sendmail* can be used as an internetwork mail gateway.

## How sendmail Works

Understanding the *sendmail* programs requires understanding a variety of components. Some of these components are daemons, scripts, files, and commands. This section describes the various *sendmail* components.

## The sendmail Daemon

For *sendmail* to process incoming mail, a daemon must be running. The *sendmail* daemon is the *sendmail* program with specific flags. (Appendix E describes the *sendmail* command-line flags in detail.) The daemon is automatically started by the */etc/init.d/mail* script at station startup. The default command for the *sendmail* daemon is

```
/usr/lib/sendmail –bd –q15m
```

The **-bd** flag causes *sendmail* to run in daemon mode. The **-q15m** flag causes *sendmail* to fork a subdaemon for queue processing every fifteen minutes. The **-bd** and **-q** flags can be combined in one call.

## sendmail Scripts

There are two scripts provided with your system that perform common functions in *sendmail*. Use these scripts whenever possible, as they have been tested and are known to perform the task correctly.

### /etc/init.d/mail

Under rare circumstances, a user may need to stop or start the *sendmail* daemon manually. For example, to implement changes to the configuration file, you must stop all running *sendmail* processes, "refreeze" the configuration file, and restart the *sendmail* daemon before the new configuration will take effect. To simplify the task of starting and stopping *sendmail,* IRIX provides a shell script called */etc/init.d/mail*.

This script takes a single argument, either **start** or **stop**, which starts or stops the *sendmail* daemon respectively. You must be superuser (root) to use this script. For example, to stop *sendmail*, use the following command:

```
/etc/init.d/mail stop
```

When */etc/init.d/mail* is called with the **start** argument, it verifies the existence and permissions of various *sendmail* related files and directories (see "sendmail Related Files and Directories" on page 177). If a required component such as the */var/spool/mqueue* directory is missing, the script creates it. For more complex components, such as */etc/aliases*, the script exits with a message.

When the */etc/init.d/mail* script is called with the **stop** argument, it kills all running *sendmail* processes with a SIGTERM signal.

**Note:** Station start-up includes an automatic call to the */etc/init.d/mail* script with the `start` argument. If station start-up runs in verbose mode (that is, */etc/chkconfig* verbose on), the following message appears, verifying that *sendmail* has been started:

```
Mailer daemons: sendmail
```

For more information, examine the */etc/init.d/mail* script.

### /usr/etc/configmail

The */usr/etc/configmail* script provides an interface between command line input and the *sendmail.cf* file. For more information, see "sendmail Related Files and Directories" on page 177. It pipes the macro and class definitions into the *sendmail.params* file. This script simplifies the *sendmail* configuration process.

The *configmail* script allows the user to interact with several *sendmail* parameters. These parameters are equivalent to *sendmail.cf* macros and classes. You can verify the current parameter settings, set specific parameters, issue a quick setup command, and get some basic online help. *configmail* stores your changes in the *sendmail.params* file, which is read by *sendmail* at startup time.

## sendmail Related Files and Directories

The *sendmail* configuration files and directories are

- */etc/sendmail.cf*
- */etc/sendmail.fc*
- */etc/sendmail.hf*
- */etc/sendmail.st*
- */etc/aliases*
- */var/spool/mqueue*
- */var/mail*

### /etc/sendmail.cf

At the heart of the *sendmail* program is the *sendmail* configuration file */etc/sendmail.cf*. The *sendmail.cf* file is an ASCII file that contains most of the configuration information and is read at run time. This file encodes options, header declarations, mailer declarations, trusted user declarations, message precedences, address-rewriting rules, macro definitions, and class definitions.

As the mail administrator and in order for you to successfully set up *sendmail*, you must know which *sendmail.cf* macros and variables to change.
The *sendmail.cf* file takes advantage of *sendmail*'s ability to read macro and class definitions from pipes, thereby simplifying and automating the *sendmail* configuration process. This file takes command line input from the *sendmail.params* file and */usr/etc/configmail* script and incorporates the input into the appropriate macros and classes.

### /etc/sendmail.fc

The *sendmail.fc* file is a frozen configuration file. A frozen configuration file is an image of the data space that belongs to *sendmail* when the configuration file is read. The *sendmail.fc* file is not present by default. You can create the *sendmail.fc* file using the *touch* command. After the *sendmail.fc* file is created, it is used in place of */etc/sendmail.cf.* This process improves start-up speed.

**Note:** All modifications to *sendmail* macros and classes should be made to *sendmail.cf*.

However, if the */etc/sendmail.fc* file exists, changes to it are not honored until you rebuild */etc/sendmail.fc*. The mail script */etc/init.d/mail* automatically rebuilds the frozen configuration file if the sendmail.cf file exists. *Always* use the mail script because it automatically rebuilds the *sendmail.fc* file. If you need to rebuild the frozen configuration file manually, the command is

```
/usr/lib/sendmail –bz
```

### /etc/sendmail.hf

The *sendmail.hf* file is the Simple Mail Transfer Protocol (SMTP) help file. It contains some brief information about the various SMTP commands.

### /etc/sendmail.st

The *sendmail.st* file is used to collect statistics related to *sendmail*. By default, the file is not present. You can create the file using the *touch* command. If the file is present, *sendmail* automatically updates the file with relevant *sendmail* statistics.

### /etc/aliases

The *aliases* file contains the text form of the alias database used by the *sendmail* program. The alias database contains aliases for local mail recipients. For example, the following alias delivers mail addressed to *jd* on the local station to *johndoe@company.com*:

```
jd:johndoe@company.com
```

When sendmail starts up, it automatically processes the aliases file into the files */etc/aliases.dir* and */etc/aliases.pag*. The *aliases.dir* and *aliases.pag* are DBM versions of the *aliases* database. The DBM format improves *sendmail* performance.

**Note:** The *newaliases* program must be run after modifying the alias database file. See "Building the Aliases Database" on page 181 for more information about building the alias database.

### /var/spool/mqueue

The mail queue, */var/spool/mqueue*, is the directory in which the mail queue and temporary files reside. The messages are stored in various queue files that exist under the */var/spool/mqueue* directory. Queue files take these forms:

- qf*—control (queue) files for messages
- df*—data files
- tf*—temporary files
- nf*—a file used when a unique ID is created
- xf*—transcript file of the current session

Normally, a *sendmail* subdaemon processes the messages in this queue periodically, attempting to deliver each message. (The */etc/init.d/mail* script starts the *sendmail* daemon so that it forks a subdaemon every 15 minutes to process the mail queue.) Each time *sendmail* processes the queue, it reads and sorts the queue, then attempts to run all jobs in order.

### /var/mail

*/var/mail* is the directory that houses all incoming mail. Each user on a local station receives his or her mail in a file in the directory */var/mail*. For example, the user *guest* receives mail in the file */var/mail/guest*.

## sendmail Commands

These section describes some of the related *sendmail* programs and commands. The programs and commands discussed in this section are

- *sendmail*
- *newaliases*
- *mailq*

### sendmail

*sendmail* is the program that implements *sendmail*'s routing and transfer service. As a program it has many flags that can be set on the command line to tailor the sendmail environment. Appendix B, "IRIX sendmail Reference," provides complete descriptions of the various command line flags and options.

### /usr/bsd/newaliases

*newaliases* is the program used to rebuild the DBM version of the *aliases* database. This program *must* be run any time the text version of the *aliases* file is modified. If *newaliases* is not run after making changes to the *aliases* file, the changes are not incorporated into the DBM *alias* database and are not seen by the *sendmail* program. See "Aliases Database" on page 181 for more details about the *alias* database.

### /usr/bin/mailq

The *mailq* command prints a current listing of the mail queue.

## Aliases Database

The aliases database is an *ndbm* database that contains mail aliases to be used by the *sendmail* program. The text form of the database is maintained in the file */etc/aliases.* The aliases are of this form:

```
name: name1 [, name2, ...]
```

For example, the following command delivers mail addressed to *jd* to *johndoe@company.com*:

```
jd:johndoe@company.com
```

**Note:** Only the local part of an address can be aliased. For example, the following command is wrong and does not have the desired effect:

```
jd@big.university.edu:jd@company.com
```

*sendmail* consults the alias database only after deciding that the message (as originally addressed) should be delivered locally, and after it has rewritten the address to contain only the local part.

An alias continuation line must start with a space or a tab. Blank lines and lines beginning with the number sign (#) are treated as comments.

If you are running NIS, *sendmail* can use the contents of the NIS alias database with the local *aliases* database by adding the following special alias to the */etc/aliases* file:

```
+:+
```

This special alias tells *sendmail* to consult the NIS alias database if the alias cannot be found in the local alias database. When the same alias is specified in both the local and NIS aliases file, the local alias supersedes the NIS alias.

### Building the Aliases Database

At startup, *sendmail* automatically uses the *ndbm* library to process the */etc/aliases* file into the files */etc/aliases.dir* and */etc/aliases.pag.* Using these files to resolve aliases improves performance.

To rebuild the DBM version of the database without restarting *sendmail,* execute this command:

```
newaliases
```

Executing this command is equivalent to giving *sendmail* the **-bi** flag:

```
/usr/lib/sendmail –bi
```

When building the DBM version of the database, *sendmail* checks the left-hand side of each entry to make sure that it is a local address. *sendmail* issues a warning for each entry in */etc/aliases* with a non-local left-hand side. Such entries are not entered into the DBM version of the database.

If the NIS alias database is used with the local *usr/lib/aliases* database, the special "+:+" alias is entered into the DBM version of the database. If *sendmail* cannot find an alias in the DBM version of the database, it looks for the special "+:+" alias. If it finds the special alias, *sendmail* then queries the NIS alias database. This query permits you to change the global NIS alias database without having to rebuild the local alias database. However, the left-hand sides of the NIS alias are *not* checked by *sendmail* to ensure that they contain only local addresses.

If the configuration or the command line specifies the **D** option, *sendmail* will automatically try to rebuild the alias database when it is out of date.

*sendmail* rebuilds the alias database if either of the following conditions exists:

- The DBM version of the database is mode 666.
- *sendmail* is running *setuid* to root.

Auto-rebuild can be dangerous on heavily loaded stations with large alias files. If it takes more than five minutes to rebuild the database, there is a chance that several processes will start the rebuild process simultaneously.

## Testing the Aliases Database

You can test the alias database with the **-bv** flag of the *sendmail* program. See "sendmail Command-Line Flags" on page 201 for more details.

## Alias Database Problems

Problems can occur with the alias database, especially if a *sendmail* process accesses the DBM version before it is completely rebuilt. Two circumstances can cause this problem:

- One process accesses the database while another process is rebuilding it.
- The process rebuilding the database dies because it has been killed, or a station crash has occurred before completing the rebuild.

*sendmail* has two techniques for trying to relieve these problems. First, to avoid the problem of a partially rebuilt database, *sendmail* ignores interrupts while rebuilding the database. Second, at the end of the rebuild it adds an alias of the following form (which is not normally legal):

```
@: @
```

Before *sendmail* accesses the database, it ensures that this entry exists. For this action to occur, the configuration file must contain the **-a** option.

If the @:@ entry does not exist, *sendmail* waits for it to appear. After the specified waiting period elapses, *sendmail* itself forces a rebuild. For this action to occur, the configuration file must include the **D** option. If the **D** option is not specified, a warning message is generated and *sendmail* continues.

Another alias problem can arise for stations incorporating the NIS alias database in */etc/aliases* through the use of the +:+ alias. If the NIS alias server goes down or is otherwise nonresponsive to NIS queries, *sendmail* will not see the aliases normally obtained from the NIS server. This situation may result in mail being returned, marked `User unknown`.

## List Owners

If an error occurs when mail is sent to a certain address (*x*, for example), *sendmail* looks for an alias of the following form to receive the errors:

```
owner-x
```

This scheme is typically useful for a mailing list where a user mailing to the list has no control over the maintenance of the list itself; in this case the list maintainer would be the owner of the list. For example, the following would cause *jd@1company.com* to get the error that occurs when someone sends mail to *unix-hackers,* and *sendmail* finds the phony user *nosuchuser* on the list.

```
unix-hackers: jd@company1.com, ed@big.university.edu,nosuchuser,
jane@company2.com
owner-unix-hackers: jd@company1.com
```

## sendmail Network Configurations

This section explains the functions of domains, forwarders, and relays in a mail network. It also explains how each of these components is designated in the */usr/etc/configmail* script and in the *sendmail.cf* file. It is important to understand these designations, since you will be expected to enter this information in the working copy of these files for your station.

### Mail Domains

Within the sendmail environment, a domain is an administratively defined area of control with logical rather than physical boundaries.

You can configure three general types of domains:

- root: top-level domain of the local domain space. For example, *horses.com* is the top level domain for the domain *pintos.horses.com*.
- direct: the domain(s) a station can send mail to directly (no forwarders or relays involved).
- local: the domain to which a station belongs.

The following parameters designate domains in the */usr/etc/configmail* script and the */etc/sendmail.cf* file:

- configmail parameters: *rootdomain, directdomain,* and *localdomain*
- *sendmail.cf* macros and classes:
  - root domain: *T macro*
  - direct domain: *D class*
  - local domain: *D macro*

## Mail Forwarders

A forwarder station is a station that acts as a mail gateway into another network. Typically, stations on either side of a gateway cannot connect directly to each other, making the forwarder station a physical (not just administrative) necessity.

Forwarder stations are not necessarily “smarter” about mail routing than other stations in the network, but they are “better connected.” Forwarder stations deliver mail to “all points beyond” some point in the domain name space.

The designation of the forwarder station is primarily determined by the physical topology of the network; the default *sendmail.cf* file can designate only a single forwarder, and the name of that station must be hard-coded in the configuration file.

The following parameters designate a mail forwarder in the */usr/etc/configmail* script and the */etc/sendmail.cf* file:

- *configmail* parameter: *forwarder*
- *sendmail.cf* macro and class: **F**

## Mail Relays

A relay station is a station that acts as a collection point for mail destined for a specified domain or group of domains. In the absence of MX records and mail exchangers, relay stations provide a mechanism whereby mail can be concentrated onto centralized locations prior to actual delivery. For more information about MX records and mail exchangers, see Appendix B, “IRIX sendmail Reference.”

Relay stations are not necessarily “better connected” than other stations in the network, but they are “smarter” about mail routing. They deliver mail to “all points within” some point in the domain name space.

For example, a company with a domain *company.com* has configured *sendmail* to treat *alpha.company.com* as the forwarder station and *omega.company.com* as the relay station. *sendmail* assumes that *alpha.company.com* is ultimately responsible for all mail to domains other than *company.com* and that station *omega.company.com* is ultimately responsible for all mail to the *company.com* domain itself. Note that there is nothing to prevent the relay and forwarder functions from residing on the same station.The designation of a relay station is primarily determined by administrative decision. *sendmail* can recognize a number of relay stations.

The relay station name is a special name used to identify relay stations in the network. This special name is defined by means of the **R** macro and is typically the name “relay.” A relay station is so designated by being aliased to the name “relay.” The default *sendmail.cf* file probes for a station named or aliased to the special relay station name and delivers mail to any such station in preference to the actual destination station. Mail is also sent to relay stations whenever the local station cannot determine the proper routing.

The following parameters designate a mail relay in the */usr/etc/configmail* script and the */etc/sendmail.cf* file:

- *configmail* parameter: *relayname*
- *sendmail.cf* macro: **R**

## User-Configurable Macros and Classes

The *sendmail.cf* file defines your mail network by assigning each element in the network a *macro* or *class* value. The default values in your distribution *sendmail.cf* file will not work without some modification.

Instead of modifying your *sendmail.cf* file directly, you can use the */usr/etc/configmail* script. It takes your input, saves it in *sendmail.params,* and configures the appropriate macros and classes according to your *sendmail* environment.

### Domain Name Macro and Class (D)

The **D** macro defines the local domain name. Be sure that the macro contains the name of the domain in which this station resides. If domains are not used, you can leave this macro empty or comment it out.

- The **D** class explicitly lists all domains for which this station should send mail directly by means of the local area network mailer.
- Mail for domains listed in the **D** class is sent directly to the destination station, if possible. No attempt is made to send the mail by means of a relay station.
- Mail for domains not listed in the **D** class is sent by means of a relay station associated with the destination station, if possible, rather than directly to the recipient station.

- If this station is a relay for a particular domain, you must enter that domain name in the **D** class. It is recommended that you always enter the domain name, regardless of whether the station is a relay or not.

### Forwarder Station Name Macro and Class (F)

The **F** macro defines the station name or alias of the station to which this station forwards mail for unknown stations or domains.

- The **F** class contains all known names for the station defined in the **F** macro.
- Mail is sent to the forwarder as a last resort only if one of these conditions exists:
  - This station cannot make the destination station name canonical or determine an appropriate relay or mail exchanger for the mail.
  - The appropriate station, relay, or exchanger for the mail exists outside the top-level domain. (See the description of the **T** macro later in this section.)
- If either condition for sending mail to this forwarder station exists, this station puts messages to unknown stations or domains "on the wire" and hopes for the best.
- If no such station exists in your environment, leave the **F** macro and class empty. If this station is the forwarder station, put this station's name in the **F** macro. Put all known names for the station in the **F** class.

### Relay Station Name Macro (R)

The **R** macro defines the station name (or an alias) used by all stations that act as relay stations. Relay stations are forwarders to known internal domains and are defined by the use of this relay station name as their station name or alias. This macro comes preconfigured as "*relay*," a name that is strongly suggested.

- Do not leave this macro blank, even if your network has no relay stations configured.
- Mail relay stations provide an alternative to an MX scheme, and can also be useful as an emergency backup to the use of MX records for internal mail routing.

### Top-Level Domain Macro (T)

The **T** macro defines the name of the top level of the local domain space. For example, if this station resides in a subdomain named *bar.foo.com* under the *foo.com* domain, and if all stations under the *foo.com* domain or any subdomain under the *foo.com* domain are considered to be internal stations, the **T** macro contains *foo.com.*

The top-level domain is used with the forwarder station (**F**) macro and class described earlier in this section. All mail sent to stations outside the top level domain is sent by means of the forwarder station.

### Killed Stations Class (K)

The **K** class is a list of all known "killed" or "dead" stations in the local domain. This is defined only on mail forwarders, to detect mail to stations that no longer exist. Any mail directed to a "dead" station is automatically sent to the mail forwarder.

### Pathalias Database Macro (P)

The **P** macro defines the location of the pathalias database that is used by *sendmail* for UUCP mail routing.

## sendmail Planning Checklist

Here is a list of items to consider before configuring your *sendmail* environment.

- What is the layout of your sendmail network? (domains, forwarders, relays)
- If you are using the */usr/etc/configmail* script, do you have the values for the parameters you need to modify?
- If you are manually modifying the */etc/sendmail.cf* file, do you have the values for the macros and classes you need to modify?
- Are you setting up any custom sendmail aliases? If you are, have aliases ready for the aliases database file.

- Are the */etc/hosts* and */etc/passwd* files up to date? The files should include the special *relay* station name for the mail forwarder, any station aliases, and user accounts for mail users, and so on. If you are using domain names, the */etc/hosts* file should use the following format:

  *ip_address fully_qualified_domain_name alias1, alias2, ...*

  A common problem with *sendmail* is that administrators place the aliases before the fully qualified domain name in the */etc/hosts* file.
- If you are using the Network Information Service (NIS) or BIND, are the sendmail-related files configured correctly (*aliases, hosts, passwd*)?

## Configuring sendmail

Configuring *sendmail* involves these tasks:

1. Customize the *sendmail.cf* file.
2. Modify the *aliases* file.
3. Start the *sendmail* daemon.

This section provides an example for configuring a fictitious *sendmail* environment. The fictitious environment includes:

- a stand-alone station where users can send mail locally *(solitaire)*
- a simple isolated sendmail network with one domain *(lab.fictitious.com)*
- a hierarchical sendmail network with relays and one domain *(eng.fictitious.com)*
- a hierarchical sendmail network with relays and multiple domains *(corp.fictitious.com* and *fin.fictitious.com)*
- a complex hierarchical sendmail network with forwarders, relays, and multiple domains (*corp.fictitious.com* and *eng.fictitious.com*)
- a UUCP sendmail connection *(uk.com*)

**Note:** In the following examples, an *empty* macro or class has no values assigned to it. The macro or class is left blank.

Figure 8-3 illustrates the fictitious sendmail environment to be used for configuring *sendmail*.

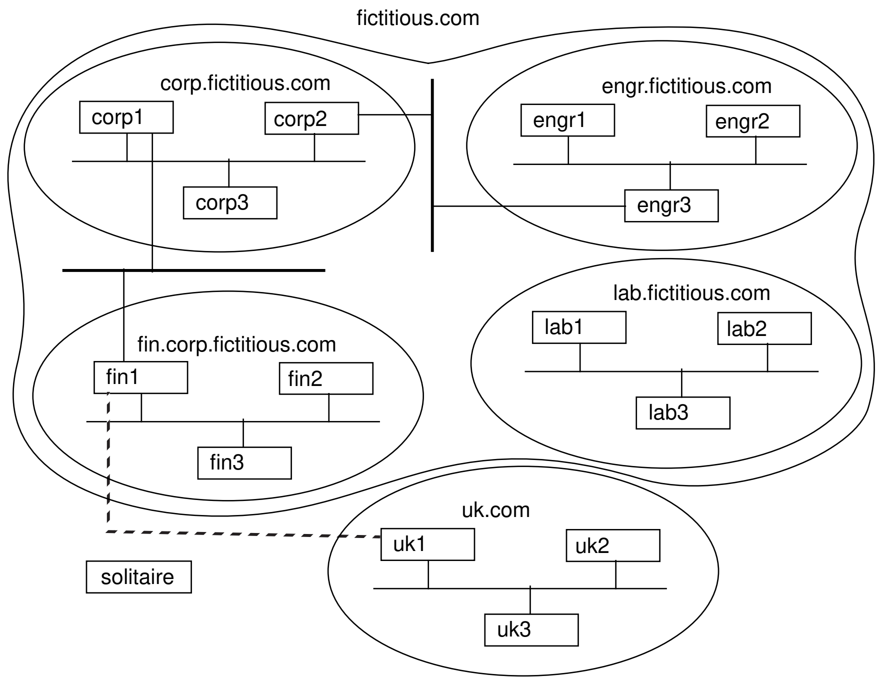

**Figure 8-3** sendmail Configuration Environment (Fictitious)

## Customizing the sendmail.cf File

The configuration file describes mailers, tells *sendmail* how to parse addresses and rewrite message headers, and sets various *sendmail* options. The standard configuration file shipped with IRIX supports a wide variety of mail configurations and *does not* work "out of the box."

All examples are based on the default *sendmail.cf* configuration file as it is shipped with IRIX. Note that *sendmail* macros and classes are both case sensitive. See "User-Configurable Macros and Classes" on page 186.

### Standalone Station

In the sample configuration in Figure 8-3, there is a single, isolated station named *solitaire*. Mail is sent only from one user on the station to another user on the same station. No mail is sent to any other station, and no mail is received from any other station.

Using the *configmail* script, set up mail like this:

```
/usr/etc/configmail set directdomains NULL
/usr/etc/configmail set localdomain NULL
/usr/etc/configmail set forwarder NULL
/usr/etc/configmail set rootdomain NULL
```

If you are configuring the *sendmail.cf* file by hand, make the required adjustments to the following macros and classes:

The **D** macro and class:
Make sure that both the **D** macro and class are empty.

The **F** macro and class:
Make sure that both the **F** macro and class are empty.

The **T** macro: Make sure that the **T** macro is empty.

**Note:** If this is the only station you are configuring for sendmail, proceed to "Modifying the Aliases Database" on page 199.

### Simple Isolated Network

This is the simplest network mail environment. A number of stations reside on a private network and send mail to each other on a peer-to-peer basis. All stations exist in the same domain; no subdomains exist. There is no connection or gateway between this private network and the outside world. No station in the network has greater responsibility for mail delivery than any other station. No relay or forwarder stations exist. The stations in the network are named *lab1, lab2,* and *lab3.* All stations exist under the *lab.fictitious.com* domain.

Each station in the network uses the same configuration. Using *configmail* on each station, make the changes shown below. For example, use *configmail* to set up mail on station *lab1*:

```
/usr/etc/configmail set directdomains lab.fictitious.com
/usr/etc/configmail set localdomain lab.fictitious.com
/usr/etc/configmail set forwarder NULL
/usr/etc/configmail set rootdomain NULL
```

If you are configuring the *sendmail.cf* file by hand, make the required adjustments to the following macros and classes:

The **D** macro and class:
Change the **D** macro and class to contain the *lab.fictitious.com* domain name.

The **F** macro and class:
Make sure that both the **F** macro and class are empty.

The **T** macro: Make sure that the **T** macro is empty.

If you modified *sendmail.cf* by hand, you can copy the modified *sendmail.cf* file to all other stations in the *lab.fictitious.com* domain. When complete, proceed to "Modifying the Aliases Database" on page 199.

#### Hierarchical (Relay) Network With a Single Domain

In this example, all stations do not bear the same responsibility for mail delivery. One or more stations are designated as mail relay stations, where mail is concentrated for further processing or queueing before delivery.

This scheme has particular advantages if some stations frequently are powered off or are otherwise unable to communicate. In such a situation, one or more relay stations are more reliable; they are never or infrequently out of communication with the network and are designated as mail concentration points. When mail is sent to a station that is down, the mail travels to the relay station, where it is queued for later delivery, rather than being queued on the originating station. When the destination station returns to operation, it is more likely that the relay station will be up than the originating station. Therefore, the mail will be delivered to the destination station in a timely manner.

This hierarchical scheme also offers administrative advantages. For example, if a single station goes down for an extended period of time, or is simply failing to accept mail, the situation is easier to detect when there is a central mail queue. An administrator can check the mail queue on the relay station to see which stations are not accepting mail. If there were no relay station, mail to the down station would be queued on stations throughout the network, and the problem could be harder to spot.

All stations exist under the *engr.fictitious.com* domain. The stations in the network are named *engr1, engr2,* and *engr3.* The mail relay station is *engr1.* The other stations in the network are expected to send mail through *engr1* rather than delivering it directly.

Each of the non-relay stations runs the same *sendmail.cf* configuration file. The *sendmail.cf* file on relay station *engr1* has a slightly different **D** class definition.

For example, using the *configmail* script, configure mail on the relay station *engr1*:

```
/usr/etc/configmail set directdomains engr.fictitious.com
/usr/etc/configmail set localdomain engr.fictitious.com
/usr/etc/configmail set forwarder NULL
/usr/etc/configmail set rootdomain engr.fictitious.com
```

Using the *configmail* script, set up mail on the remaining stations in the *engr.fictitious.com* domain. Note that the *directdomains* parameter is set to NULL on all stations except the relay station *engr1*.

```
/usr/etc/configmail set directdomains NULL
/usr/etc/configmail set localdomain engr.fictitious.com
/usr/etc/configmail set forwarder NULL
/usr/etc/configmail set rootdomain engr.fictitious.com
```

If you are configuring the *sendmail.cf* file by hand, make the required adjustments to the following macros and classes:

The **D** macro and class

On all stations, change the **D** macro to contain the *engr.fictitious.com* domain name.

On the relay station *engr1,* make sure that the **D** *class* contains the *engr.fictitious.com* domain name so that *engr1* sends mail directly to all stations in the *engr.fictitious.com* domain.

On the remaining stations, make sure that the **D** *class* is empty so that they *do not* send mail directly to other stations. (They are to send the mail to *engr1.*)

The **F** macro and class

Make sure that the **F** macro and class are empty.

The **T** macro

On all stations, change the **T** macro to contain the *engr.fictitious.com* domain name.

For *engr1* to be recognized as the mail relay station, the special relay station name “relay” (as defined by the **R** macro) must be one of the station aliases that belongs to *engr1.* Include the station name “relay” in the entry for *engr1* in */etc/hosts* or the DNS or NIS equivalent.

When you have completed this exercise, proceed to “Modifying the Aliases Database” on page 199.

### Hierarchical (Relay) Network With Multiple Domains

In this example, the hierarchical model is extended to multiple subdomains. This type of environment is a logical extension of the preceding one and is probably the easiest model to expand as the number of stations on the network increases. The environment requires that domain names be used for proper mail addressing.

The entire local domain is named *corp.fictitious.com.* There is one subdomain under the *corp.fictitious.com* domain: *fin.corp.fictitious.com.* The stations in the *corp.fictitious.com* domain are *corp1, corp2,* and *corp3.* The stations in the *fin.corp.fictitious.com* domain are *fin1, fin2,* and *fin3. corp3* is the relay for the *corp.fictitious.com* domain; *fin3* is the relay for the *fin.corp.fictitious.com* domain.

The stations in each of the two domains *(corp.fictitious.com* and *fin.corp.fictitious.com)* are configured much like those described in the preceding subsections.

Using the *configmail* script, set up mail on the relay station *corp3*:

```
/usr/etc/configmail set directdomains corp.fictitious.com
/usr/etc/configmail set localdomain corp.fictitious.com
/usr/etc/configmail set forwarder NULL
/usr/etc/configmail set rootdomain corp.fictitious.com
```

Using the *configmail* script, set up mail on the relay station *fin3*:

```
/usr/etc/configmail set directdomains fin.corp.fictitious.com
/usr/etc/configmail set localdomain fin.corp.fictitious.com
/usr/etc/configmail set forwarder NULL
/usr/etc/configmail set rootdomain corp.fictitious.com
```

Using the *configmail* script, set up mail on the remaining non-relay stations in the *corp.fictitious.com* domain. Note that the *directdomains* parameter is set to NULL on non-relay stations.

```
/usr/etc/configmail set directdomains NULL
/usr/etc/configmail set localdomain corp.fictitious.com
/usr/etc/configmail set forwarder NULL
```

```
/usr/etc/configmail set rootdomain corp.fictitious.com
```

Using the *configmail* script, set up mail on the remaining non-relay stations in the *fin.corp.fictitious.com* domain. Note that the *directdomains* parameter is set to NULL on non-relay stations.

```
/usr/etc/configmail set directdomains NULL
/usr/etc/configmail set localdomain fin.corp.fictitious.com
/usr/etc/configmail set forwarder NULL
/usr/etc/configmail set rootdomain corp.fictitious.com
```

If you are configuring the *sendmail.cf* file by hand, make the required adjustments to the following macros and classes:

| | |
|---|---|
| The **D** macro | For all stations in the *corp.fictitious.com* domain, change the **D** macro to contain the *corp.fictitious.com* domain name.<br><br>For all stations in the *fin.corp.fictitious.com* domain, change the **D** macro to contain the *fin.corp.fictitious.com d*omain name. |
| The **D** class: | On the relay station *corp3,* make sure that the **D** class contains the *corp.fictitious.com* domain name so that *corp3* will send mail directly to all stations in the *corp.fictitious.com* domain.<br><br>On the relay station *fin3,* make sure that the **D** class contains the *fin.corp.fictitious.com d*omain name so that *fin3* will send mail directly to all stations in the *fin.corp.fictitious.com* domain.<br><br>On the remaining stations in the network, make sure that the **D** class is empty so that they *do not* send mail directly to other stations. |
| The **F** macro and class: | Make sure that the **F** macro and class are empty. |
| The **T** macro: | On each of the stations, change the **T** macro to contain the *corp.fictitious.com* domain name. |

For the relay stations *corp3* and *fin3 t*o be recognized as such, the special relay station name "relay" (as defined by the **R** macro) must be an alias for each of them. There can be only one *relay* alias in the */etc/hosts* file. Here is how to set up each alias:

- For *corp3,* the alias *relay.corp.fictitious.com (*and optionally "relay") should be included in its entry in */etc/hosts* or the DNS or NIS equivalent.

- For *fin3,* the alias *relay.fin.corp.fictitious.com s*hould be included in its entry in */etc/hosts* or the DNS or NIS equivalent.

When you have completed this procedure, proceed to “Modifying the Aliases Database” on page 199.

### Complex (Forwarder) Hierarchical (Relay) Network With Domains

This section explains how to configure a station to act as the forwarder station; a forwarder station can be added to any of the scenarios described in the preceding subsections. Please see “sendmail Network Configurations” on page 184 for an explanation of the forwarder station concept as used by IRIX *sendmail.*

This discussion applies to mail environments of all types. Whatever the form of your internal mail environment, whenever you want to use a mail gateway station between your internal mail network and the external world, a forwarder station is required. The *sendmail* configuration for using this forwarder station is exactly the same for all stations on the internal side of the gateway.

The internal mail environment consists of the domain *corp.fictitious.com* and any or all domains under *corp.fictitious.com (fin.corp.fictitious.com).* All stations within the *corp.fictitious.com* domain are capable of communicating with each other. For example, there is no physical restriction to prevent station *fin1.fin.corp.fictitious.com* from sending mail to *corp1.corp.fictitious.com.*

Station *corp2.corp.fictitious.com* can connect to all stations within the *corp.fictitious.com* domain and can also connect to stations in other domains, such as *engr.fictitious.com.* Station *corp2.corp.fictitious.com* is therefore the forwarder station to all domains beyond the *corp.fictitious.com* domain. In this example, station *corp2.corp.fictitious.com* is aliased to *corp2* for ease of addressing in the internal mail environment.

In addition to the changes to *sendmail.cf* required for the internal mail environment, make the following changes to the *sendmail.cf* file on all stations within the *corp.fictitious.com* domain. Using the *configmail* script, change the appropriate parameter:

```
/usr/etc/configmail set forwarder corp2.corp.fictitious.com corp2
```

If configuring the *sendmail.cf* file by hand, make the required adjustments to the following macros and classes:

| | |
|---|---|
| The **F** macro: | Make sure that the **F** macro contains the station name *corp2.corp.fictitious.com*. |

The **F** class: Make sure that the **F** class contains the two names *corp2* and *corp2.corp.fictitious.com,* by which the forwarder station is known.

When you have completed the procedure, proceed to "Modifying the Aliases Database" on page 199.

### UUCP Mail

The default *sendmail.cf* file shipped with IRIX includes support for sending mail through UUCP. This section discusses these capabilities and explains how to integrate UUCP mail into the local mail environment. UUCP support can be added to any of the scenarios described in the preceding subsections.

The *sendmail.cf* file directs *sendmail* to read the */etc/uucp/Systems* file on startup. All UUCP station names are read from this file, and stations marked "domain-machine" are noted. If a UUCP pathalias database is maintained on the station, the location of the database is set with the **P** macro.

If the */etc/uucp/Systems* file indicates that there are stations connected to the local station through UUCP, *sendmail* sends mail received on the local station, and addressed to one of the stations described in the */etc/uucp/Systems* file, on to the proper place. If the **P** macro is set and points to a valid UUCP pathalias database, *sendmail* will attempt to find a UUCP path to a station for which it cannot find an address or *MX* record. If the database returns a good UUCP path to the destination station, *sendmail* attempts to send the mail to the left-most station on the path.

Depending upon the network environment, UUCP mail may range from the only form of network mail to one part of a much larger network mail environment. The following sections describe a common technique for adding UUCP to an existing local area mail network.

#### Sample Environment

The local domain is named *uk.com.* Station *uk1.uk.com is the* forwarder station, as described in "Complex (Forwarder) Hierarchical (Relay) Network With Domains" on page 196 in "Customizing the sendmail.cf File" on page 190.

To avoid forwarder loops, the default *sendmail.cf* file permits only one forwarder station to be configured. Therefore, station *uk1.uk.com* is also the UUCP forwarder station.

#### Changes to sendmail.cf

No changes to *sendmail.cf* are necessary beyond those required to configure station *uk1.uk.com* as the forwarder station. (See “Complex (Forwarder) Hierarchical (Relay) Network With Domains” on page 196.) If station *uk1.uk.com* maintains a pathalias database, the **P** macro should be set to the pathname of the pathalias database.

**Other Changes**

The */etc/uucp/Systems* file must also be configured before *sendmail* will see any of the UUCP-connected stations. For more information see Chapter 7, “UUCP.”

When you have completed this procedure, look ahead to “Modifying the Aliases Database” on page 199.

### Non-Domain Addressing

This section discusses issues related to a mail network that does not use domain addressing. Note that all of the previously discussed mail environments, with the exception of the “hierarchical multi-domain” environment, are possible in a network that does not implement domains.

If a network does not use domain addressing, specific changes are required in the *sendmail.cf* file on all stations in the network. First, make changes to the *sendmail.cf* file as described in the appropriate examples in this section. Next, make the following changes, even if they replace changes you have just made.

Using the *configmail* script, set up mail on each station on the network.

```
/usr/etc/configmail set directdomains NULL
/usr/etc/configmail set localdomain NULL
/usr/etc/configmail set rootdomain NULL
```

If you are configuring the *sendmail.cf* file by hand, make the required adjustments to the following macros and classes:

| | |
|---|---|
| The **D** macro and class | Make sure that the **D** macro and class are both empty. |
| The **T** macro | Make sure that the **T** macro is empty. |

In an isolated network, you must create the file */etc/resolv.conf* and add this line:

```
hostresorder local bind yp
```

For more information about the *resolv.conf* file, see the resolv.conf(4) reference page.

When you have completed this procedure, proceed to "Modifying the Aliases Database" on page 199.

## Modifying the Aliases Database

After modifying a station's *sendmail.cf* file (see "Customizing the sendmail.cf File" on page 190), the alias database file should also be modified to reflect your *sendmail* environment. If you don't have any "private" company aliases, you still need to modify the *aliases* file to provide it with a valid *postmaster* alias.

### Creating the Aliases File

Continuing with the fictitious example used in "Customizing the sendmail.cf File" on page 190, assume that the administrator for the domain *corp.fictitious.com* has derived a list of aliases for the *corp.fictitious.com* domain. The list of aliases is shown in Table 8-1.

**Table 8-1** Sample *aliases* File Entries

| Alias Name | Member | Station Name |
|---|---|---|
| finance | john | fin1 |
| finance | paul | fin2 |
| finance | mary | fin3 |
| corp | sharon | corp1 |
| corp | pam | corp2 |
| corp | peter | corp3 |
| all | finance and corp | N/A |
| postmaster | mailmgr | corp1 |

The */etc/aliases* file entries based on the list of aliases generated by the administrator would look like this:

```
##############################################################
# Aliases in this file will NOT be expanded in the header
# from Mail, but WILL be visible over networks or from
# /bin/mail.
# >>>>>>>>> The program "newaliases" must be run after
# >> NOTE >> this file is updated for any changes to
# >>>>>>>>>> show through to sendmail.
##############################################################
start of common aliases--do not remove this line
# Add the following alias to enable Yellow Page aliases. If
# enabled, the YP database defines anything not defined in
# this file.
#+:+
# Alias for mailer daemon
MAILER-DAEMON:postmaster
# send mail likely to be lost to the mail server
rootcsh:postmaster
rootsh:postmaster
.
.
.
games:postmaster
# Following alias is required by RFC 822
# You should change 'root' in the first line below to
# the administrator of this machine, and un-comment the
# following line.
postmaster:root
root:mailmgr@corp1
# aliases to handle mail to msgs and news
nobody: /dev/null
# end of common aliases--do not remove this line
# corp.fictitious.com aliases
finance:john@fin1,paul@fin2,mary@fin3
corp:sharon@corp1,pam@corp2,peter@corp3
all:finance,corp
```

### Updating the aliases Database

After you modify the */etc/aliases* text database file, run the *newaliases* program to incorporate the text changes into the DBM files, */etc/aliases.dir* and */etc/aliases.pag*.

Update the aliases database:

**`/usr/bsd/newaliases`**

If there is nothing wrong with your aliases database, *newaliases* lists the number of aliases and then return your prompt. If you see any other message, most likely there is a problem with your aliases file. See “Debugging Flags” on page 203 for hints on troubleshooting the alias file.

### Starting the sendmail Daemon

After customizing the *sendmail.cf* files and modifying the *aliases* database, you are ready to start *sendmail*.

By default, IRIX automatically starts *sendmail* at station start-up by using the shell script */etc/init.d/mail.* However, if you are configuring and testing *sendmail* and don’t want to reboot the station, you can run the */etc/init.d/mail* script manually. You should always use the *mail* script to stop and start *sendmail*. It processes and checks *sendmai*l related files and programs and in the correct order.

Start the *sendmail* daemon:

```
/etc/init.d/mail start
```

If you need to stop *sendmail*, enter the following command:

```
/etc/init.d/mail stop
```

## Managing sendmail

This section describes some of the tasks related to managing the *sendmail* environment.

### sendmail Command-Line Flags

You can include one or more flags on the command line to tailor a *sendmail* session. This section describes some of the more frequently used flags. For a complete description of command-line flags, see Appendix B, “IRIX sendmail Reference.”

#### Changing the Values of Configuration Options

The **-o** flag overrides an option in the configuration file. The override is for the current session only. In the following example, the **T** (timeout) option becomes two minutes for this session only:

```
/usr/lib/sendmail -oT2m
```

For a complete discussion of configuration options, see Appendix B, "IRIX sendmail Reference."

### Delivery Mode

One configuration option frequently overridden on the command line is the **d** option, which specifies the *sendmail* delivery mode. The delivery mode determines how quickly mail is delivered:

| | |
|---|---|
| **i** | deliver interactively (synchronously) |
| **b** | deliver in background (asynchronously) |
| **q** | queue only (don't deliver) |

There are trade-offs. Mode **i** passes the maximum amount of information to the sender, but is rarely necessary.

Mode **q** puts the minimum load on your station, but if you use it, delivery may be delayed for up to the queue interval.

Mode **b** is probably a good compromise. However, in this mode, *sendmail* may initiate a large number of processes if you have a mailer that takes a long time to deliver a message.

### Queue Mode

The **-q** flag causes *sendmail* to process the mail queue at regular intervals. The syntax is as follows, where *time* defines the interval between instances of queue processing:

**-q** [*time*]

Time is expressed in number of minutes: 15m sets the interval to 15 minutes. If *time* is omitted, *sendmail* processes the queue once and returns. The **-q** flag is often used in conjunction with daemon mode, described in the next subsection.

### Daemon Mode

To process incoming mail over sockets, a daemon must be running. The **-bd** flag causes *sendmail* to run in daemon mode. The **-bd** and **-q** flags can be combined in one call, as in the following example:

```
/usr/lib/sendmail -bd -q30m
```

This command causes *sendmail* to run in daemon mode and to fork a subdaemon for queue processing every half hour.

The script for starting *sendmail* that is provided with IRIX includes the following command line:

```
/usr/lib/sendmail -bd -q15m
```

### Verify Mode

Using the **-bv** flag directs *sendmail* to validate addresses, aliases, and mailing lists. In this mode, *sendmail* performs verification only. It does not try to collect or deliver a message. *sendmail* expands all aliases, suppresses duplicates, and displays the expanded list of names. For each name, *sendmail* indicates if it knows how to deliver a message to that destination.

### Test Mode

The **-bt** flag places *sendmail* in test mode so that it describes how the current configuration rewrites addresses. Test mode is extremely useful for debugging modifications to the */etc/sendmail.cf* configuration file. For more information, see Appendix B, "IRIX sendmail Reference."

## Debugging Flags

Several debugging flags are built into *sendmail.* Each flag includes a number and a level. The number identifies the debugging flag. The level, which defaults to 1, dictates how much information is printed. A low level causes minimal information to print; a high level causes more comprehensive information to print. By convention, levels greater than 9 are not recommended, since so much information prints that it is of limited value. Debugging flags use the following syntax:

**-d** *debug-list*

- Set flag 13 to level 1.

  **-d13**
- Set flag 13 to level 3.

  **-d13.3**
- Set flags 5 though 18 to level 1.

  **-d5-18**
- Set flags 5 through 18 to level 4.

  **-d5-18.4**

Many debugging flags are of little use to the average *sendmail* user. Some are occasionally useful for helping to track down obscure problems. Appendix B, "IRIX sendmail Reference," includes a complete list of debugging flags.

## Using a Different Configuration File

The **-C** flag directs *sendmail* to use an alternate configuration file. For example, the following line directs *sendmail* to use the *test.cf* file instead of the default */etc/sendmail.cf* file:

```
/usr/lib/sendmail –Ctest.cf
```

If the **-C** flag appears without a filename, *sendmail* uses the file *sendmail.cf* in the current directory. Thus, the **-C** flag directs *sendmail* to ignore any */etc/sendmail.fc* ("frozen") file that may be present.

## The Mail Queue

This section discusses how to print and force the mail queue.

### Listing the Queue

You can list the contents of the queue by using the *mailq* command or by specifying the **-bp** flag to *sendmail.* The list includes a listing of the queue IDs, the size of each message, the date the message entered the queue, and the sender and recipients.

### Forcing the Queue

The **-q** flag (with no value) forces *sendmail* to process the queue. It is sometimes useful to use the **-v** flag (verbose) also when running the queue manually, as follows:

**/usr/lib/sendmail -q -v**

In verbose mode, *sendmail* displays the SMTP chatter with other stations as well as messages indicating any delivery errors and final message disposition.

Because of the locking algorithm, it is impossible for one job to freeze the queue. However, an uncooperative recipient station or a program recipient that never returns can consume many station resources. Unfortunately, there is no way to resolve this situation without violating the SMTP protocol used by *sendmail.*

In some cases, if a major station goes down for a couple of days, a prohibitively large queue may be created. As a result, *sendmail* spends an inordinate amount of time sorting the queue. You can remedy this situation by moving the queue to a temporary location and creating a new queue. The old queue can be run later when the offending station returns to service.

Use the following commands to move the entire queue directory. The mail queue should be owned by *root* and belong to the *mail* group.

```
cd /var/spool

mv mqueue omqueue

mkdir mqueue

chmod 755 mqueue
```

Then kill the existing *sendmail* daemon (because it will still be processing in the old queue directory) and create a new daemon:

```
/etc/init.d/mail stop

/etc/init.d/mail start
```

To run the old mail queue, use the following command:

```
/usr/lib/sendmail –oQ/var/spool/omqueue –q
```

The **-oQ** flag specifies an alternate queue directory, and the **-q** flag causes *sendmail* to run every job in the queue once and then return. Use the **-v** (verbose) flag to watch what is going on. It may be necessary to run the old mail queue a number of times before all of the messages can be delivered.

When the queue is finally emptied, the directory can be removed:

```
rmdir /var/spool/omqueue
```

## The .forward File

As an alternative to the alias database, users can put a file with the name *.forward* in their home directories. If the *.forward* file exists, in a user's home directory *sendmail* redirects mail for that user to the list of recipients in the file. The recipients are separated by commas or new lines. For example, if the home directory for user *jane* has a *.forward* file with the following contents, any mail arriving for *jane* is redirected to the specified accounts:

```
zippy@state.edu
bongo@widgets.com
```

The *.forward* file also allows the user to redirect mail to files or programs. A *.forward* file with the following contents redirects any incoming messages to *jd@company.com,* appends a copy of the message to the file */var/tmp/mail.log,* and pipes a copy of the message to *stdin* of the */usr/bin/mymailer* program:

```
jd@company.com
/var/tmp/mail.log
| /usr/bin/mymailer
```

In general, file-type recipients must be writable by everyone. However, if *sendmail* is running as *root* and the file has *setuid* or *setgid* bits set, then the message will be written to the file.

Users can redirect mail to themselves in addition to sending it to other recipients. This feature is particularly useful if the users want to continue to receive mail in their own mailboxes while passing copies of each incoming message to some alternative destination. For example, say that the home directory for user *john* contains a *.forward* file with the following contents:

```
\john, |/usr/sbin/vacation
```

*sendmail* behaves as follows:

- It sends each incoming message to john's regular mailbox; the backslash (\) preceding the name indicates that no further aliasing is to occur.
- It pipes a copy of each message to *stdin* of the */usr/sbin/vacation* program. (The vertical bar [|] is the standard UNIX pipe symbol.)

## Sendmail Questions, Problems, and Troubleshooting

1. What are MX records?

   MX records are resource records in the BIND database. Each record contains the name of a target station, a preference level, and the name of an exchanger station that handles mail for the target station. (The exchanger station may be the target station itself.)

   The BIND database can contain several MX records for each target station; the record with the lowest preference level is tried first.

   MX records provide a way to direct mail to alternative stations. Using MX records lets you eliminate static routes from your *sendmail* configuration file. For more details about setting up MX records, see Appendix B, "IRIX sendmail Reference."

2. *sendmail* doesn't seem to see my changes to *sendmail.cf.*

   Remember to save your changes to the configuration file by issuing the following commands:

   ```
   /etc/init.d/mail stop
   /etc/init.d/mail start
   ```

   The stopping and starting of the daemon forces *sendmail* to reconfigure the file. Otherwise, *sendmail* runs using the old, "frozen" version of the configuration file, thus ignoring your changes. For more information on "frozen" configuration files, see Appendix B, "IRIX sendmail Reference."

3. Can I put comments in macro definitions?

   Do *not* include comments on macro or class definition lines in the *sendmail.cf* file. For example,

   ```
   DDfoo.com # my domain name
   ```

   would define the **D** macro as:

   ```
   "foo.com # my domain name"
   ```

Likewise,

```
CD foo.com bar.com # my local domains
```

would define the **D** class as containing “foo.com,” “bar.com,” “#,” “my,” “local,” and “domains.”

4. Where can I find information to help me troubleshoot my *sendmail* installation?

   See Appendix B, “IRIX sendmail Reference.”

## Notes to Current sendmail Users

In general, the current version of *sendmail* should be backward-compatible with previous versions. Most users should be able to replace their existing *sendmail* with the new IRIX 4.0 *sendmail* and run as before without any changes. However, there are certain differences between this and previous versions of *sendmail* that may cause compatibility problems. These differences are described in the subsections that follow.

### MX Record Support

For each destination station contacted by means of an IPC-type mailer (P=[IPC] in the mailer definition line), *sendmail* queries the DNS database for MX records associated with the destination station. If the MX query succeeds, mail will be routed through the appropriate exchanger station found among the returned MX records as described in RFC 974 and required by RFC 1123.

The result is that this version of *sendmail* and previous versions may use different methods to route mail to stations for which MX records are available. See Appendix B, “IRIX sendmail Reference,” for information regarding mailer definitions.

With the advent of *MX* records, you may want to edit your *sendmail.cf* file to remove previously required static routes.

### Multi-Token Class Match

Some *sendmail.cf* implementations inadvertently rely on the inability of *sendmail* to do multi-token class-matching. One such implementation is the standard *sendmail.cf* file distributed with IRIX Releases 3.2 and 3.3. If your *sendmail.cf* file is based upon one of those standard *sendmail.cf* files, you should read this section. If your *sendmail.cf* file is *not* based on one of those standard files, this section may serve as an example if you encounter odd behavior with class-matching while running the new *sendmail.*

The standard IRIX Release 3.2 and 3.3 *sendmail.cf* files define an **S** class that is scanned in from the */etc/hosts* file. This class is used to detect single-token station names that appear in the local */etc/hosts* file. With the advent of multi-token class-matching, the **S** class no longer operates as intended.

The problem is that station names appearing in the */etc/hosts* file are scanned into the **S** class whether they are single- or multi-token station names (that is, whether or not they contain dots). The **S** class still worked as intended with previous versions of *sendmail,* because if an attempt was made to match a multi-token station name in the class, the match would always fail. With the new *sendmail,* that same match will (incorrectly) succeed. This problem is observed when rules such as this one began matching qualified station names such as *foo.bar.sgi.com*:

```
# Assume that unqualified names are local.
R$*<@$=S>$* $1<@$2.$D>$3
```

The result was that stations such as *foo.bar.sgi.com* that appeared on the LHS of the rewrite rule shown here were being rewritten to *foo.bar.sgi.com.bar.sgi.com*, which is obviously wrong.

The problem was not the use of the **S** class, but rather the practice of scanning multi-token station names from the */etc/hosts* file into the class in the first place.

An examination of the *sendmail.cf* file shows that the **S** class is being scanned in by the following scan sets:

```
# Directly-connected SMTP hosts
FS/etc/hosts %*[.0-99] %[-_.a-zzA-ZZ0-99]
FS/etc/hosts %*[.0-99] %*[-._a-zzA-ZZ0-99] %[-_.a-zzA-Z0-99]
```

These scan sets read in the two left-most station names from */etc/hosts* regardless of whether they contain dots. To correct the situation, modify the scan sets to read in only the station names from */etc/hosts* in their single-token, unqualified form, as follows:

```
# Directly-connected SMTP hosts
FS/etc/hosts %*[.0-99] %[-_a-zzA-ZZ0-99]
FS/etc/hosts %*[.0-99] %*[-._a-zzA-ZZ0-99] %[-_a-zzA-ZZ0-99]
```

Note the removal of the dots from the right-most patterns.

Depending on your use of class-matching, this incompatibility may not affect you. If you suspect there might be a problem, you should examine your use of classes and your class definitions. If you are currently using a *sendmail.cf* file supplied by Silicon Graphics, you should examine the **S** class scan sets and make the corrections indicated here. If you use the default *sendmail.cf* as supplied in this release, you should be free from any such problems.