# sendmail address rewriting mini−HOWTO

# Table of Contents

# sendmail address rewriting mini−HOWTO


## Thomas Roessler, roessler@guug.de



v0.0, 6 May 1998


---


*This document is a brief description of how to set up sendmail's configuration file for the home user's dial−up access.*


---



---

## 1. Introduction

We assume that you have the kind of Internet access which seems to be most common at universities and online services nowadays: You dial into your provider's network using PPP over a serial connection. Your incoming mail is spooled at the provider's POP or IMAP server, while outgoing messages are to be sent via SMTP. You don't have a domain name of your own, so everything has to use *one* address.

We assume that you have already installed a fairly recent version of Eric Allman's sendmail (version 8.8.8 is current at the time of this writing and should work fine).

This document is partially referring to specific properties of Debian GNU/Linux systems; users of different distributions will have to take some care.

Make sure you have the following information at hand:

- Your ISP's mail server
- Your Internet mail address

The configuration we are planning has two main goals:

1. Sending mail between various local users must be possible.
2. The outside world must see the local users' ISP mail addresses, not the local ones.

To achieve this, we will make use of sendmail's `genericstable` feature.

## 2. File Roadmap

We will put all of sendmail's configuration files in a separate directory under `etc`: `etcmail`. Usually, sendmail will expect these files to reside directly under `etc`. To avoid problems, `etcsendmail.cf` should be a symbolic link to `etcmailsendmail.cf`.

The following files will populate `etcmail`:

- =20
- `aliases` – contains additional local addresses
- `genericsdomain` – contains some information on your local host's configuration
- `genericstable` – contains the actual rewriting rules.
- `sendmail.cf` – sendmail's configuration file
- `sendmail.mc` – the source of `sendmail.cf`.

Some of these files will be accompanied by `.db` files. They contain hashed databases for sendmail's direct use.

We assume that the `cf` part of sendmail's source tree resides under a directory named `usrlibsendmail.cf`. This is the case on Debian GNU/Linux systems. Other distributions will put this stuff at different places. Please refer to your distribution's documentation for details.

## 3. Configuring sendmail

### 3.1 The main configuration file

Sendmail uses a highly complex rule system for it's configuration. While you can do lots of neat tricks with this stuff, writing a `sendmail.cf` file from scratch is rather unusual and time−consuming. If you are interested in doing so, you should stop reading this document right now and instead read the "Bat Book" from O'Reilly.

Instead of hand−crafting these rules, we will rely on the `m4` macro processor to put together our configuration file from ready−made pieces which are distributed together with sendmail.

Let's look at the first lines of the `sendmail.mc` file:

```
include(/usr/lib/sendmail.cf/m4/cf.m4)
VERSIONID(`sendmail.mc - roessler@guug.de')
OSTYPE(debian)
define(`ALIAS_FILE',`/etc/mail/aliases')
```

In the beginning, `cf.m4` is included. This m4 macro file contains lots of macro definitions for the rest of the file. Be sure that the path you give here is correct – the one we are representing in our example is typical for

Debian GNU/Linux. The `OSTYPE` macro is used to give some useful defaults for certain configuration values. If you aren't using a Debian system, you should replace the word "debian" by "linux" here. `ALIAS_FILE` tells sendmail where to look for the list of aliases.

The following lines tell sendmail to use the `genericstable` feature, and where to find the configuration files needed to use it:

```
FEATURE(masquerade_envelope) FEATURE(genericstable, `hash
-o /etc/mail/genericstable')
GENERICS_DOMAIN_FILE(`/etc/mail/genericsdomain')
```

The `masquerade_envelope` feature tells sendmail to apply header rewriting to the *envelope* sender of a message. This is the mail address to which external mail delivery subsystems will direct their delivery failure reports and warning messages. The `generics*` files will be explained below.

Now, we have to define a so−called smart host, that is, a machine which will handle outgoing mail for your system. Note that this machine may be different from your ISP's POP and IMAP servers. If in doubt, contact the hotline. The code in the master configuration file:

```
define(`SMART_HOST',`mail-out.your.provider')
```

Please replace *mail−out.your.provider* by the fully qualified hostname of your internet service provider.

The final two lines include the "mailer" definitions which are needed by sendmail to find out how to handle various types of mail:

```
MAILER(local)
MAILER(smtp)
```

To generate the `sendmail.cf` file from this `sendmail.mc`, type the following commands (as root):

```
# m4 sendmail.mc > _sendmail.cf
# mv −f _sendmail.cf sendmail.cf
```

Note the technique of writing `m4`'s output to a temporary file which is thereafter moved to the proper place. This helps us to prevent sendmail from reading partially written configuration files.

## 3.2 Address rewriting

First, we have to tell sendmail what addresses are to be considered local (and thus should be subjected to the rewriting). This is quite simple: Just put the fully qualified host name of your machine into the file `etcmailgenericsdomain`. To get your host's fully qualified name, type the following command:

```
 $ hostname −f
```

Now, let's come to the rewriting table proper: `etcmailgenericstable`. This file consists of two white−space separated columns. The first column contains the local address, the second column contains the e−mail address which should be used instead. The file may look like this:

```
harry   harryx@your.isp
maude   maudey@her.isp
root    fredx@your.isp
news    fredx@your.isp
```

Note that there should be one entry for *each* account on the local machine, so that automatically generated mail which leaks out of the local system carries correct header information.

For performance reasons, sendmail won't use this text file directly, but rely on a "hashed" version instead. To generate it, type the following command:

```
# makemap −r hash genericstable.db < genericstable
```

Note that the rewriting rules from the `genericstable` will *not* apply to local mail or to messages you receive from outside − the mapping is only used if a message leaves your local system for your ISP's smart host.

## 3.3 Aliases

The aliases file contains additional local names which are only valid for local messages. This is useful for administrative accounts like `root` which receive automatically generated messages from your system.

A reasonable start for `etcmailaliases` could look like the following file:

```
root: fred
news: root
postmaster: root
mail: root
www: root

nobody: devnull
MAILER−DAEMON: nobody
```

This example will forward local mail for the `root`, `news`, `postmaster`, `mail`, and `www` users to `fred`, while messages for `nobody` and `MAILER−DAEMON` will be redirected to `devnull`.

Just like the `genericstable`, `aliases` may contain *lots* of entries. Thus, it would once again be inefficient for sendmail to use the text file we just described. The same mechanism as with `genericstable` is used for `aliases`: A hashed database is generated. Instead of using `makemap` directly, you can type in the command `newaliases` this time. It will automatically take care of all what's needed.

# 4. Further reading

The sendmail source distribution includes quite a bit of documentation. Read it, especially the file `cfREADME`.

If you are interested to dive deeper into sendmail's configuration options, you want to get the "Bat Book" from O'Reilly: Bryan Costales, Eric Allman, and Neil Rickert: "sendmail". O'Reilly, 1993.