Practical Modern Sendmail Configuration

by David Bank CNE, CCSE, CCNA

v1.50 (2007-Nov-11)

Myths of sendmail

The sendmail mail router is one of the oldest software stacks still in wide use on the 'Net. It should be little wonder that there's a wide variety of myths about it. The two most prevalent seem to be:

* sendmail is insecure
* configuration is a nightmare for mere mortals

As with all long-lived myths, the staying power derives from a grain of truth.

For example, in 1995, sendmail was practically the only widely-deployed SMTP MTA. Those were the days of sendmail v8.7, when it was still a monolithic process that ran as *root* and the code was rife with buffer overruns. By default, it was an open relay.

Moreover, configuration was a black art involving hacking *sendmail.cf*, and the uninitiated sendmail administrator usually asked a friend for a copy of a working *sendmail.cf* and hacked it until it worked in the new environment. Then they were afraid to touch it, because when they touched it, it almost always broke.

That was then - this is now.

Modern sendmail gives up its *root* privileges as soon as is practical, meaning that exploits have a harder time compromising a sendmail process in a privileged state. The modern program architecture separates the more-vulnerable Mail Submission Program (MSP) component, which can run un-privileged, from the Message Transfer Agent (MTA) component. By default, sendmail refuses to run when presented with group- or world-writable paths or files, a regrettably common sysadmin mis-step that was the actual cause for many compromises blamed on sendmail[1]. A custom restricted shell (**smrsh**) can contain the code even further, and newer tools, such as Novell's AppArmor[2], can provide even more protection (and not just for sendmail).

Configuration is much easier with the **m4** macro-based *.mc* files, and it's now possible to document configurations and transport them reliably between versions and platforms. The default configuration prohibits relaying, which must now be explicitly enabled. As someone who spent 9 years hacking *sendmail.cf*, I have no desire to return to that method - using the macros is much easier.

So when evaluating modern sendmail, don't fall for the old myths and legends. Consider its modern incarnation and how it works today.

Configuration with the .mc configuration files

The heart of modern sendmail configuration is the *.mc* configuration files, *sendmail.mc* and *submit.mc*. These are written using the **m4** macro syntax, which is also the syntax for *syslog.conf*, SELinux and other configuration files. They are used to generate *sendmail.cf* and *submit.cf*, respectively.

Modern sendmail has two components, one being the Message Transfer Agent, or MTA, which listens on TCP/25 and exchanges E-Mail with other MTAs. The Mail Submission Program, or MSP, is a special MTA instance that listens (by default) on TCP/587 and is designed for Mail User Agents (MUAs), like PINE, Thunderbird and Evolution, to submit E-Mail for delivery by the MTA. The MTA and its MSP instance can operate independently. The term MUA, as used here, is generic and includes "Mail Submission Agents", or MSAs.

sendmail.cf is the configuration file associated with the MTA instance of sendmail, and is the default configuration file read by the sendmail executable, absent any parameters to the contrary. *submit.cf* is the configuration file for the MSP instance, and the **`-Ac`** parameter on the sendmail invocation instructs sendmail to read *submit.cf* instead of *sendmail.cf*.

Assumptions and Standards

This paper assumes that you are running a modern version of sendmail on a Linux, UNIX or UNIX-like (*e.g.* AIX) platform. "modern" is defined as sendmail v8.12.10 or later[3]. Also, sendmail has been successfully built and installed, or was installed from a package or as part of the OS install. Finally, however you got sendmail, you have the necessary **m4** macro files and programs needed to generate the *.cf* files.

The sendmail software stack has its own standards as to where it wants to be installed. Maintainers of pre-built packages may or may not override those. I prefer to build sendmail from source, and I let sendmail install where its defaults for the particular OS tell it to install. There's too many things that expect sendmail to be a particular place, and sendmail is so sensitive to directory modes and file ownerships, that it's generally best to let it install where it wants to. In the modern sendmail environment, the configuration files are usually in */etc/mail*.

Solaris Admins:	I note in passing that the Sun-supplied builds of sendmail usually have NIS/NIS+ support enabled. This is a compile-time option and there's no handy way to turn it off in the run-time configuration. Sun also seems to hard-code things like the *service.switch* file, and trying to set that at run-time may not be effective. Further, Sun-supplied packages are usually outdated (especially for Solaris v8 and earlier). My advice is that you **pkgrm** the Sun-supplied sendmail packages and build/install new ones. Pre-compiled sendmail in Solaris package format can be found at Sun Freeware[4].

Linux Admins:	Some modern Linux environments overlay sendmail configuration with other tools, such as **SuSEconfig** or **WebMin**. This paper assumes you're directly configuring sendmail, and not relying on an overarching configuration tool. If you prefer to use those other tools, then consult their documentation concerning sendmail configuration options. Note that some tools may not offer you the ability to configure sendmail fully, or make use of all the capabilities shown in this paper.

Before You Begin

While the purpose of this paper is to show you how to configure sendmail, it is not an all-inclusive or exhaustive guide to doing so. The configuration options highlighted in this paper, and in the sample files, are generally selected as reasonable candidates for adjustment; there are many more options than are shown.

However, it is inadvisable to adjust any configuration item unless the reason for doing so is fully understood, along with the possible consequences of the change. The fact that sendmail is highly configurable is simultaneously the source of its greatest strength and its most humbling weakness. If you aren't sure what a configuration option does, then leave the setting at the default.

Vendor-supplied Sendmail Configurations

Few vendors do sendmail any favors when it comes to the default configuration files included in their distributions. Most vendors that include a sendmail build provide either a *sendmail.cf* without the corresponding *sendmail.mc* file, or provide a *sendmail.mc* file that is poorly written, and usually undocumented. Frequently, the vendor-provided configurations omit important options, mis-order the macros (see the next section), and/or fail to properly utilize configuration parameters.

If you choose to use a vendor-supplied build of sendmail rather than building your own, use the example configuration files associated with this paper as an alternative to the vendor-supplied files. Few vendors, if any, supply reasonably intelligent sendmail configurations.

.mc file blueprint

The *.mc* files are a series of macro definitions. These are used to build the corresponding *.cf* files, using the **m4** macro processor. Detailing the mechanics of **m4** is outside of the scope of this paper (some helpful information is in Sendmail, Chapter 4.1.2, Page 147), but with respect to sendmail, one important thing is the order of the macros in the *sendmail.mc* and *submit.mc* files. Quoting the Sendmail text (Chapter 4.2.3, Page 155), you should use the following order (the relevant Chapter for the macro or macros is shown in parentheses):

* **VERSIONID** (4.2.3.1)
* **OSTYPE** (4.2.2.1)
* **DOMAIN** (4.2.2.3)
* option definitions (24.4)
* **FEATURE** (4.8)
* macro definitions (21.7)
* **MAILER** (4.2.2.2)
* ruleset definitions (19.1.7)

This paper will begin by exploring the architecture of a *sendmail.mc* with some example entries from each type.

The Header

Everyone documents their configuration files and tracks changes, right? This is a simple documentation header.

```
divert(-1)dnl
dnl # * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
dnl # Author: Your name here
dnl # File: sendmail.mc for the hostname
dnl # Change Log:
dnl # Who  When        What
dnl # ---- ----------- ----------------------------------------
divert(0)dnl
```

Preamble

These are the first "real" statements in the *sendmail.mc*.

The **`VERSIONID`** macro allows you to put a string into the *sendmail.cf* (and *submit.cf*) to track its version. It can be almost anything you want - one limit is that the string cannot contain a **`<CR>`** character. This macro is not required, but revision control should be as important as commenting. This is an exemplar:

```
VERSIONID(`$Id: sendmail.mc, vsendmail version  year/month/day  HH:MM:SS    your name  Exp $')dnl
```

The **`OSTYPE`** macro is required, and selects from a number of pre-defined macro files, located in the *cf/ostype* subdirectory. Most common OSes are already pre-defined. While this is handy, be aware that some of the definitions may contain settings that you may not want. For example, **`OSTYPE(linux)`**defines ProcMail-related paths and FEATURE statements - while most Linux distributions include the ProcMail package, if you have removed it (or didn't install it), then this may cause problems. Similarly, **`OSTYPE(solaris8)`** defines IPv6 support. If you build sendmail without IPv6 support, this may cause problems.

```
OSTYPE(your OS selection here)dnl
```

Next, the optional **`DOMAIN`** macro allows you to define Domain-specific configurations. The **`generic`** Domain is pre-defined and is a simple boilerplate. You may define one or more **`DOMAIN`** macros. For each custom one that you define, a corresponding file must exist in the *cf/domain* directory.

```
DOMAIN(generic)dnl
```

You may also omit the **`DOMAIN`** macro entirely. The sample *sendmail.mc* includes it and shows the **`generic`** value for illustrative purposes only.

Option Definitions

There is a wide and sometimes confusing range of Options available in sendmail. The bulk of these are defined in Chapter 24 of Sendmail, starting on Page 921 (new ones introduced in sendmail v8.13 are in the Companion). Additional Options have been added since the publication of the Companion, and are documented in the various documentation files included with the source code, and on the sendmail website.

Typically, you would use Options to tweak sendmail's internal defaults. For example, some **`OSTYPE`** macros may place the PID files in locations other than where you keep yours, or even don't create PID files. With the **`confPID_FILE`** Option, the existence, name and location of the sendmail daemon's PID file is specified, overriding any settings defined by the **`OSTYPE`** macro:

```
dnl # Sendmail, Chap 24.9.78, Page 1027
dnl # Define the name and path of the daemon's PID file
define(`confPID_FILE',`/var/run/sendmail-mta.pid')dnl
```

If necessary, sendmail will create its PID file(s), but the directory must already exist or sendmail will exit with an error.

Another common need is to limit the size of E-Mails. By default, sendmail will handle (or attempt to handle) any size of E-Mail. This creates the possibility of a denial-of-service attack, where someone shovels an impossibly huge E-Mail, or series of E-mails, at your mailserver and attempts to crash the MTA or fill up the disk space. While not absolute protection, the **`confMAX_MESSAGE_SIZE`** Option will limit the maximum size of an E-Mail that sendmail will attempt to process:

```
dnl # Sendmail, Chap 24.9.63, Page 1013
dnl # Limit maximum size, in bytes, of any given E-Mail to 10 MB
dnl # (10485760 bytes) - checked if sender reports; again at end of DATA
define(`confMAX_MESSAGE_SIZE',`10485760')dnl
```

Over the limit and sendmail will reject the message with a **PERMFAIL** error.

Finally, some Options affect the behavior of FEATURE statements defined in the next section. For example, it is possible to increase the time window during which sendmail will retain the connection records for a given host (IP), records that are used by the v8.13-specific **`ratecontrol`** and

`conncontrol` FEATURE statements:

```
        dnl # Sendmail v8.13 Companion, Chap 4.1.3, Page 17
        dnl # Instruct daemon to maintain record of host connections for 120
        dnl # seconds (default is 60; for use with ratecontrol and conncontrol)
        define(`confCONNECTION_RATE_WINDOW_SIZE',`120')dnl
```

Options that are specific to certain FEATURE statements may be defined in *sendmail.mc* (or *submit.mc*) even if the associated FEATURE is not enabled in the configuration file.

FEATURES

Using FEATURE macros, you turn on extra functionality in sendmail. You can also turn off functions with the **`undefine`** statement. Sendmail, Chapter 7, starting on Page 287, documents the various FEATURE macros included in sendmail, and the Companion documents those added in v8.13. FEATUREs added after publication of the Companion are documented with the sendmail source and on the website.

The configuration presented in this paper is based on the principle of "least access", which translates into the sysadmin world as "least pieces parts" - or, "If I don't need it, turn it off; or don't turn it on". As shown here, if you don't use certain mail protocols, you can disable support for them (does *anyone* still use UUCP mail?):

```
        dnl # Disable the following features
        undefine(`UUCP_RELAY')dnl
        undefine(`BITNET_RELAY')dnl
        undefine(`DECNET_RELAY')dnl
        undefine(`FAX_RELAY')dnl
```

Modern sendmail is FEATURE-rich, and useful new capabilities are often introduced in new versions. One of the most useful is the connection control FEATURE, which is an effective anti-SPAM/anti-mailbomb defense. It must be explicitly enabled in *sendmail.mc*, like so:

```
    dnl # Sendmail v8.13 Companion, Chap 4.1.8, Page 20
    dnl # Enable access map DB feature to control the number of simultaneous
    dnl # connections other hosts may have to this server; "nodelay"
    dnl # causes this feature to bypass "delay_checks" and work at connection
    dnl # time instead of after RCPT_TO; "terminate" means that sendmail
    dnl # will immediately drop a violating connection instead of waiting
    dnl # for other server to drop it; MUST appear AFTER "delay_checks"
    FEATURE(`conncontrol', `nodelay', `terminate')dnl
```

Note that it's RFC-impolite to simply terminate connections rather than waiting for the remote host to acknowledge the disconnect. Realistically, however, this FEATURE only affects spammers, and who cares about being polite to them?

Realtime Blackhole Lists, or **RBLs**, are also a FEATURE. Here, sendmail is configured to check connections against the NJABL blacklist service, and, if the remote host in blacklisted, to display a custom error message as opposed to the default:

```
dnl # Sendmail, Chap 7.2.2, Page 297
dnl # Add NJABL BL with custom reject message
FEATURE(`enhdnsbl',`dnsbl.njabl.org',`"ACCESS DENIED. Mail from
      " $&{client_addr} " refused based on information from
      http://njabl.org"')dnl
```

Defined Macros

There are three types of Defined Macros, including compilation and configuration macros. The third type, **sendmail macros**, are used to represent strings of text symbolically within the *.cf* configuration files. You may occasionally wish to define different values from the defaults. Here, the configuration re-defines the **`CF_VERSION`** macro, which is used in the sendmail greeting banner and in the (default) **`Received: from`** header added by sendmail:

```
dnl # Sendmail, Chap 21.9.100, Page 834
dnl # Set the config file version in format < host >-< serial # >
define(`confCF_VERSION',`host-serial #')dnl
```

MAILER macros

`MAILER` macros define delivery agents that sendmail will use to deliver E-Mail. The stock sendmail environment supports all manner of delivery agents, including SMTP, UUCP, Decnet and ProcMail, just to name a few.

The **`smtp`** delivery agent transmits E-Mail to other hosts using ESMTP (the default), SMTP, DSMTP, and/or SMTP8. This macro also defines the **`relay`** agent. See the file *mailer/smtp.m4*.

The **`local`** delivery agent is used to direct E-mail for local delivery (*i.e.* to a mailbox on the local machine). The **`prog`** delivery agent (used to direct E-mail to a program) is also defined with this macro. The **`local`** delivery agent usually defaults to the *mail.local* program, but if the **`MAILER(procmail)`** macro is used, then **`local`** will usually refer to the ProcMail program.

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

Important!

Sendmail, Chapter 4.2.2.2, page 152, discusses the order in which the **`MAILER`** macros should appear. The configuration is sensitive to this order. Also, generally, **`MAILER(local)`** is a required **`MAILER`**.

Local configurations

Lastly, and optionally, you may define macros and Rulesets specific to an instance of sendmail. Such definitions can include MILTERs, setting values that are not set elsewhere (due to lack of an associated **conf** macro, or if you do not want to use the associated macro), and custom Rulesets, such as **`Check_local`**. See Sendmail, Chapter 4.2.2.2, page 153, concerning the location of this section.

While a full discussion of Mail Filters, or MILTERs, is outside the scope of this paper, briefly, a MILTER is an external program that participates in the SMTP conversation that sendmail has with a connecting host. An example is **MIMEDefang**[5], which might be added like so:

```
INPUT_MAIL_FILTER(`mimedefang',
     `S=unix:/var/spool/defang/MIMEDefang/mimedefang.sock, F=T,
     T=C:30m;S:30m;R:30m;E:30m')dnl
```

Sometimes, it is necessary to alter the value of a standard sendmail Class macro, without using the "normal" macro defined for that purpose. As an example, consider *F{VirtHost}*, which defines the virtual user Domains file. Ordinarily, to specify the value of Class *F{VirtHost}*, the configuration would use the **`VIRTUSER_DOMAIN_FILE`** macro (Sendmail, Chapter 4.8.51.2, page 203). However, in the specific configuration presented in this paper, there are issues with the normal process.

Because the sample *sendmail.mc* file includes **`FEATURE(blacklist_recipients)`** to enable per-recipient accept/reject in the *access* table (see below), the fact that the standard macro also adds the Domains specified to Class *{R}*, which defines Domains for which sendmail will relay, becomes a problem. When used with per-recipient blacklisting, the unintended result is that RBL checks are bypassed for all Domains in Class *{R}*. If you want to use RBLs, per-recipient blacklisting, and you also want to use the Virtual User capabilities of modern sendmail, then you can't use the standard **`VIRTUSER_DOMAIN_FILE`** macro to populate Class *F{VirtHost}*.

You could give up per-recipient blacklisting, use the standard macro, and implement recipient checks some other way, perhaps using MIMEDefang, or a custom sendmail Ruleset. In keeping with a philosophy of rejecting obvious SPAM as early and often as possible, however, it helps to have the option of per-recipient blacklisting in the *access* table, so alternative methods must be explored.

One way to avoid the problem would be to hack *cf/m4/proto.m4* and change the macros that copy the contents of *F{VirtHost}* into Class *{R}*. The trouble with that approach is maintenance – the hack would have to be replicated to every sendmail install, and also replicated between versions.

A better method, however, is to simply define Class *F{VirtHost}* without using the standard macro. This approach is easier to write, easier to port between platforms and versions, and sidesteps the usual copy of the Domains into Class *{R}*. It's also easily done with this *sendmail.mc* entry:

```
dnl # Define a file for Virtual Hosting - but do not use the
dnl # built-in VIRTUSER_DOMAIN_FILE macro, as that also
dnl # adds the Domains to RELAY_DOMAINS, and we don't want that
LOCAL_CONFIG
F{VirtHost}/etc/mail/virtuser.domains
```

Finally, you can define your own entire macros. This is most frequently done to have custom Rulesets, like **Check_local**, where you perform your own relay checking. For more information, see Sendmail, Pages 158, 159 and Chapters 19 and 25.

Important!

Your custom ruleset names should generally start with a capital letter, to avoid any possibility of name collision with sendmail's internal Rulesets. It's also important to note that TAB is the delimiter between keys, spaces will not work.

Other Configuration Sources

Once you have a working *sendmail.mc* (and *submit.mc*) and have built *sendmail.cf* (and *submit.cf*), you can use various other databases and files to supply run-time information to sendmail. This next section discusses those sources of information.

Database Maps

Database maps are files that are consulted as sendmail processes an E-Mail. To speed lookups, the files that sendmail references are usually in a binary format, such as Berkeley DB, or GNU DB. These are built from simple text files that the sysadmin can edit, and then use a tool like **makemap**, or the built-in sendmail **newaliases** functionality, to convert the data in the text file to a file of the database type that sendmail expects.

Note that database maps are read on an as-needed basis. This means they can be updated without restarting sendmail. However, the two-step update process means that simply changing the source text file doesn't do anything. The change does not take effect until the sendmail-readable database file is rebuilt.

While purposes and keywords vary between the different database map files, the syntax of the source text files is consistent. Each has a **L**eft-**H**and **S**ide (**LHS**), or key; separating whitespace; and a **R**ight-**H**and **S**ide (**RHS**), or value. The whitespace may be spaces, TABs, or a combination of the two.

When sendmail consults a database map, it is looking for a particular string, trying to match it against a key, or LHS, of the database map. When it finds a matching *key*, the associated *value*, or RHS, is returned to the calling function. Generally, sendmail ceases searching the database map when the first matching key is found, so as a rule, you should construct your source text files with more-specific LHS entries first, and then generic, or default, entries further down. When converting from the source text files, the database records are added in the order they appear in the source text file.

You may insert comments into database map files, and I recommend you take advantage of this capability. **makemap** and **newaliases** both ignore blank lines, and lines beginning with a **#** are treated as comments and likewise ignored.

The **Reference** section below contains documented samples of all the files discussed here.

The *access* database

The *access* database file is most useful for an Internet-facing relay host, but also can help protect internal mailservers. This database may be used to specify hosts and/or (not recommended) Domains that are exempt from RBL checks, create a specific E-Mail address to which RBLed senders can write to request whitelisting, reject or discard E-Mail to and/or from specific addresses/Domains, and permit relaying for specific Domains and/or hosts. If you're running v8.13 or later, you can also specify overrides for the **`GreetPause`** `FEATURE`, and specify per-IP, per-network and/or per Domain values to override connection limits for "friendly" hosts/Domains.

Your specific environment or host may not require all of these things, but at a minimum, if you use RBLs, you should have entries using the **`Connect`** *tag* (see below) to bypass RBL checks and permit relaying for your own internal hosts. Otherwise, sendmail will constantly make DNS requests to the RBL server(s), checking for your own mailservers, which is fairly pointless.

The *access* database is easily the most complex database map. The LHS (*keys*) are fronted by *tags* that apply the RHS (*value*) to a specific **`FEATURE`**. Prior to v8.14, these *tags* were optional; now, untagged entries in the *access* database are considered deprecated (they will function, but run the risk of producing unexpected or ambiguous results).

The following table lists the more-common *tags* associated with the LHS entries, and their purpose/reference. This table is not all-inclusive, nor is all possible functionality shown in the example *access* entries.

<u>*access* database map LHS tag reference</u>

Tag	Associated **FEATURE**(s)	Description/Reference	Example *access* entry

Connect	**enhdnsbl** (Chap 7.2.2, Page 297), **delay_checks** (Chap 7.5.6, Page 318)	Specifies LHS entries checked when a foreign host connects, and action taken for that specific host (can also specify subnets on class boundaries)	`Connect:10.0.0.1     RELAY`
			`Connect:192.168.0    REJECT`
			`Connect:192.168.111 ERROR:5.7.1:554 This subnet not allowed to relay`

Greetpause	**greetpause** (Companion, Chap 7.1.3, Page 51)	These LHS entries are checked to see if a time delay different from the default specified in the **FEATURE** statement should be applied to the connecting host before the SMTP greeting banner is displayed	`GreetPause:127.0.0.1    0`

ClientConn	***conncontrol*** (Companion, Chap 4.1.8, Page 20)	With these LHS entries, the sendmail admin may limit the number of simultaneous connections another host (or host on a subnet) may have to the host running sendmail. A given host (or subnet) entry is kept in the table of entries for the time window specified in **confCONNECTION_RATE_WINDOW_SIZE** (Companion, Chap 21.1.13, Page 150). If the LHS does not specify an address/network, then the limit applies to all hosts/networks without an entry. A RHS of 0 indicates no limit.	`ClientConn:127.0.0.1    0`
			`ClientConn:192.168.100  0`
			`ClientConn:             2`

ClientRate	**rateconn** (Companion, Chap 4.1.7, Page 18)	When this **FEATURE** is enabled and appropriate entries are included in the *access* database, the LHS entries define, by IP address, the hosts (or networks) to which the numerical limit specified in the corresponding RHS. The limit is the number of connections the remote host may have open at the same time, during the time window specified by **confCONNECTION_RATE_WINDOW_SIZE** (Companion, Chap 21.1.13, Page 150). If the LHS does not specify an address/network, then the limit applies to all hosts/networks without a specific entry. A RHS of **0** indicates no limit.[6]	`ClientRate:127.0.0.1    0`
			`ClientRate:192.168.9    5`
			`ClientRate:             2`

Spam	**enhdnsbl** (Chap 7.2.2, Page 297), **delay_checks** (Chap 7.5.6, Page 318)	If the E-Mail is being sent to the LHS address, and the sending host is listed in an RBL, then the rule specified by the RHS is used to determine the disposition of the E-Mail (rather than the RBL result). If the RHS is **FRIEND** then the E-Mail is permitted to proceed, which is useful to for creating an E-Mail address to which even RBLed senders can write (to request white-listing). Note that **FEATURE(delay_checks)** must be enabled.	`Spam:whitelist.request@domain.tld    FRIEND`

Note that in most cases, host/Domain names are submitted for matching before any attempt is made to match an IP address. Thus, it may be possible for an LHS entry that uses an IP address to be ignored in favor of another entry that uses a hostname, Domain name or E-Mail address.

Remember that using **RELAY** in the RHS will bypass all further checks for the current step in the SMTP conversation. It is generally safer to use **OK** in the RHS.

If you use the RBL features, it is highly recommend that you have an E-Mail address that even RBLed senders can reach - this E-Mail address should **not** appear on any web-pages or anywhere else it might be easily harvested; and should probably be an alias on an interior machine. Spammers rarely pay attention to RBL error messages, so they won't see the address. Legitimate senders should get the information if their E-Mail system isn't brain-dead and strips it.

Important!

Your *sendmail.mc* must include **`FEATURE(`delay_checks',`friend')`** in order for such an override to work.

Finally, don't forget to allow relay for the Domain(s) you host. You can do this with the **`To:`** *tag*, which will check during the **`RCPT TO:`** step in the SMTP conversation. This should not be a problem, because if the sending host is legitimately able to send E-Mail **`From:`** the Domain, it would (or should) have been permitted to relay based on an earlier **`Connect`** *tag*. This helps prevent spammers from relaying by pretending to be sending from a Domain you host.

The *aliases* database

The *aliases* database map is consulted by sendmail after it has determined that an E-Mail should be delivered locally. In this paper, the sendmail host is a relay host, and doesn't have any local end-user accounts. Thus, the sample *aliases* database is quite small. A host with user mailboxes could have an extensive *aliases* file, however.

The most important thing to remember about the *aliases* database is that it is only consulted after sendmail has determined that an E-Mail is destined for a local mailbox (*i.e.* a mailbox hosted on the local system). While the RHS of an *aliases* entry can result in an E-Mail that is no longer destined for the local mailbox, this database is not consulted for E-Mails being relayed.

The *domaintable* database

domaintable is generally used when moving from one Domain Name to another. As a rule, most installations do not need to use *domaintable*, although there is no harm in creating a blank one.

The *genericstable* database

The *genericstable* database file is used to instruct sendmail to re-write the SMTP headers for outgoing E-Mail. There are several possible reasons you might wish to do this; for example, if your internal usernames are limited to 8 characters (many older *NIXs have this limit), but users want a **firstname.lastname** format for their E-Mail addresses, it's easy enough to do that for incoming E-Mail using *virtualusertable* or an *aliases* database, but outgoing E-Mail is not affected by those things. Another scenario would be if a mail recipient is planning to move from one Domain you host to another, this can make the change seem to have taken place before it actually does.

The *mailertable* database

A *mailertable* instructs sendmail on how to route an E-mail based on the destination Domain. *mailertable* is consulted when sendmail has determined that an E-mail is destined for a Domain for which it relays, and after *virtualusertable* has been consulted (so if *userA@DomainOne.com* maps to *userZ@domainX.org*, that will happen first).

The syntax is similar to other databases - the left-hand side, or *key* that is used to match, and then the right-hand side, or *value* that determines what it to be done. In the case of *mailertable*, the *keys* are Domain names or FQDNs. The *values* consist of two parts, a **mailer** specification and a **hostname** (FQDN), separated by a colon (however, no whitespace should be on either side of the colon, only between the *key* and *value*).

Normally, when the **hostname** portion of the *value* is an FQDN, sendmail will perform an **MX** lookup in DNS to resolve the IP address. However, by enclosing the **hostname** in square brackets, you instruct sendmail to not perform an **MX** record lookup for the host, but instead resolve the **A** record. This helps prevent mail loops. For example, if **mail1.somedomain.tld** is registered in DNS as the **M**ail e**X**changer for **somedomain.tld**, and individual hosts in **somedomain.tld** do not have **MX** records in DNS, then when an E-Mail arrives at **mail1.somedomain.tld** and *mailertable* indicates that the final destination is **userbox.somedomain.tld**, the normal address resolution process would result in the mail being relayed to **mail1.somedomain.tld** - a loop. By configuring sendmail to lookup the **A** record instead of the **MX** record, the loop is avoided.

The *value* can also be an IP address, and if this is done, it should be enclosed in square brackets. When the value is an IP address, sendmail does not perform any DNS lookup.

The **`mailer`** portion of the *value* can be any delivery agent that sendmail configuration supports: **`smtp`**, **`esmtp`**, **`uucp`**, whatever.

The *virtusertable* database

The *virtusertable* instructs sendmail how to map inbound E-mail from one address to another. It's consulted after *aliases* (if *aliases* was consulted) but before *mailertable*. So *virtusertable* tells sendmail where an E-Mail needs to go, but not how to get it there.

The *keys* can be either full E-Mail addresses, or Domains (when preceded with @). Note that lookups in *virtusertable* are unusual in that when a match is not found on the first try, sendmail will pare down the hostname and try again. So if **bob@server.mail.somedomain.tld** did not match on the first pass, sendmail would chop off the **server.** and try to match **bob@mail.somedomain.tld**. This continues until the host portion of the address is pared down to a Domain and TLD; if no match occurs at that point, the lookup fails (which may or may not result in a delivery failure depending on the other tables).

The *value* consists of the new E-Mail address for delivery. It is important to note that the headers are not re-written. So the receiving host must not be too picky about the **To:** E-Mail address in the message headers. Also, some Domain substitutions are possible, such as mapping **joe@hosteddomain.tld** to **joe@userbox.somedomain.com**.

Finally, the syntax of *virtusertable* supports *wildcarding*. This is demonstrated in the sample file in the **Reference** section.

Configuration Files

Unlike database maps, configuration files are simple text files. They contain one entry per line, and the nature of that entry depends on the specific file. Configuration files are read only when sendmail initializes - if they are changed, sendmail must be shut down and restarted (a **SIGHUP** won't do).

It is important to note that these file formats do not support comments. They are interpreted literally, and extraneous characters will only cause problems.

These files are, by default, in */etc/mail*, although links may exist in */etc*. The location and names of the files may be changed with various configuration macros.

The *genericdomains* file

This file contains all Domains, including FQDNs, for which entries exist in *genericstable*. If a Domain/FQDN is not listed in this file, then *genericstable* will not be consulted for the Domain.

The *local-host-names* file

The *local-host-names* file defines the Domains and/or FQDNs that the sendmail daemon will consider as "local". E-Mail destined for these Domains will be delivered locally, and *mailertable* will not be consulted, even if local delivery fails. *aliases* and *virtusertable* are consulted, however.

The *relay-domains* file

Similar to *local-host-names*, the *relay-domains* file defines the Domains and FQDNs for which sendmail will relay, without question. E-Mail destined to these Domains will not undergo RBL checks, if they are configured. *mailertable* and *virtusertable* are consulted normally.

The *trusted-users* file

A *trusted* user is a user ID (the text, not the numeric value) allowed to bypass certain security checks in sendmail. This is useful for programs that submit E-Mail directly to a sendmail queue, such as mailing list managers. Usually, only *root* is listed in this file, but application package documentation may instruct you to add other UIDs.

The *virtuser.domains* file

Similar to *genericdomains*, this file defines all Domains and FQDNs for which entries exist in the *virtusertable* file. Unless a Domain/FQDN is listed here, *virtusertable* will not be consulted for E-Mail destined for that Domain/FQDN. Note that this filename is specific to the configuration presented in this paper. If you use the “standard” **`VIRTUSER_DOMAIN_FILE`** macro, then whatever argument you give in that macro is the proper file.

Summary

While perhaps more maligned than warranted, sendmail is a complex software package. It offers the mail system administrator an unparalleled amount of power and flexibility. The cost of those capabilities is a potentially steep learning curve - for both configuration and operation.

Like any other complex environment, you can simplify the administration by following the basics of good system management: document the environment and any changes you make; make changes incrementally; and always have a backout plan. In this respect, sendmail is no different than any other powerful software package.

Despite the fact that the **m4** configuration method has been available for years, it is still regrettably common to see advice, especially from vendors, instructing mail system administrators to modify *sendmail.cf* directly, instead of making changes to *sendmail.mc*. It's best to resist such suggestions - if necessary, convert the suggestions into macros in the **`LOCAL_CONFIG`** portion of *sendmail.mc*. Insist that vendors update their documents and provide instructions for modern sendmail.

Finally, learn to leverage sendmail and make it work for you. sendmail offers a great deal of power, and has a wealth of software designed to plug-in and expand its already considerable functionality.

Helpful reference materials

These files are available at the URLs in the Footnotes:

Sample *sendmail.mc* configuration file[7]

Sample *submit.mc* configuration file[8]

Sample *access* map database source[9]

Sample *aliases* map database source[10]

Sample *domaintable* map database source[11]

Sample *genericstable* map database source[12]

Sample *mailertable* map database source[13]

Sample v*irtusertable* map database source[14]

The following books have been referenced in this paper:

Sendmail, Third Edition by Bryan Costales, ISBN 1-56592-839-3 (O'Reilly)[15]
At just over 1,200 pages, this is an intimidating book, but don't let it scare you. It's not a "How to" or "For Dummies" sort of thing; instead, it's a reference you to consult while building or configuring sendmail. It will take some time to learn how to extract information from it, but once you understand how it's put together, it's very handy. My biggest issue with this book is the inordinate amount of time/text it spends on discussing configuration of obsolete versions.

Sendmail 8.13 Companion by Bryan Costales *et. al.*, ISBN 0-596-00845-7 (O'Reilly)[16]
If you're running sendmail v8.13.x, this reference book documents the new features and options available. It's a bit more concise than the original Costales book, and does a better job of explaining the nuts-and-bolts.

Sendmail Cookbook by Craig Hunt, ISBN 0-596-00471-0 (O'Reilly)[17]
A general reference for sendmail problem-solving. A bit dated now (when it was written in 2001, sendmail was at v8.12.9), it still has good information, and does a good job of explaining how sendmail "thinks" about E-Mail. While Linux-centric, it contains references to the sendmail documentation that help the reader relate functions to the sendmail documentation.

Footnotes

[1] The tendency of some sysadmins to run such insecure configurations gave rise to the aptly-named **`confDONT_BLAME_SENDMAIL`** configuration option.
[2] `http://en.opensuse.org/AppArmor_Detail`
[3] **sendmail v8.12.10** should be considered the absolute oldest version that should be used on any Internet-connected host. As of this writing, the current version is **v8.14.2**. The reader is encouraged, in the strongest possible terms, to upgrade anything older than **v8.13**.
[4] `http://www.sunfreeware.com`
[5] `http://www.mimedefang.org`
[6] A minor bug prior to v8.14.2 caused the rate condition to be met when the specified number was *reached*, rather than *exceeded* (the latter is the documented behavior). This bug did not affect how a RHS of 0 was handled.
[7] `http://www.hiredavidbank.com/sendmail.mc`
[8] `http://www.hiredavidbank.com/submit.mc`
[9] `http://www.hiredavidbank.com/access`
[10] `http://www.hiredavidbank.com/aliases`
[11] `http://www.hiredavidbank.com/domaintable`
[12] `http://www.hiredavidbank.com/genericstable`
[13] `http://www.hiredavidbank.com/mailertable`
[14] `http://www.hiredavidbank.com/virtusertable`
[15] `http://www.oreilly.com/catalog/sendmail3/index.html`
[17] `http://www.oreilly.com/catalog/sendmailcomp/index.html`
[18] `http://www.oreilly.com/catalog/sendmailckbk/index.html`

Change Log

Version	Date	Change
1.00	2005-Feb-20	Initial Creation
1.05	2006-Feb-16	Updated with v8.13-specific information
1.10	2006-Mar-12	Added Myths section; re-published
1.11	2006-Mar-28	Fixed typos; documented changes
1.12	2006-Mar-31	Began adding access database map reference table
1.15	2006-Apr-11	Minor additions and corrections
1.17	2006-Apr-16	Additions to access database map table, and to Reference materials
1.18	2006-Apr-18	Fixed typos and formatting errors
1.19	2006-Apr-20	Added section for Summary; minor text edits
1.20	2006-Apr-22	Completed Summary; expanded aliases description (1st PDF version)
1.25	2006-May-11	Added clarification to why the sample configuration avoids the standard **`VIRTUSER`** macro in favor of the custom one
1.30	2006-Jun-16	Reconciled with HTML version; added Footnotes
1.40	2007-Jan-19	Updated with information about new sendmail versions; fixed minor typos; clarified some language; added headers/footers
1.41	2007-Mar-19	Fixed URL in Footnote #13; deleted stray comma in headers
1.45	2007-Jun-07	Updated for v8.14; updated sample files; fixed minor typos; added Before You Begin section
1.50	2007-Nov-11	Updated for v8.14.2; added Vendor section; minor additions/corrections