

SENDMAIL CONFIGURATION FILES

This document describes the sendmail configuration files. It explains how to create a sendmail.cf file for use with sendmail. It also describes how to set options for sendmail which are explained in the Sendmail Installation and Operation guide (doc/op/op.me).

To get started, you may want to look at tcppproto.mc (for TCP-only sites) and clientproto.mc (for clusters of clients using a single mail host), or the generic-*.mc files as operating system-specific examples.

Table of Content:

- INTRODUCTION AND EXAMPLE
- A BRIEF INTRODUCTION TO M4
- FILE LOCATIONS
- OSTYPE
- DOMAINS
- MAILERS
- FEATURES
- HACKS
- SITE CONFIGURATION
- USING UUCP MAILERS
- TWEAKING RULESETS
- MASQUERADING AND RELAYING
- USING LDAP FOR ALIASES, MAPS, AND CLASSES
- LDAP ROUTING
- ANTI-SPAM CONFIGURATION CONTROL
- STARTTLS
- SMTP AUTHENTICATION
- ADDING NEW MAILERS OR RULESETS
- ADDING NEW MAIL FILTERS
- QUEUE GROUP DEFINITIONS
- NON-SMTP BASED CONFIGURATIONS
- WHO AM I?
- ACCEPTING MAIL FOR MULTIPLE NAMES
- USING MAILERTABLES
- USING USERDB TO MAP FULL NAMES
- MISCELLANEOUS SPECIAL FEATURES
- SECURITY NOTES
- TWEAKING CONFIGURATION OPTIONS
- MESSAGE SUBMISSION PROGRAM
- FORMAT OF FILES AND MAPS
- DIRECTORY LAYOUT
- ADMINISTRATIVE DETAILS

```
+-----+
| INTRODUCTION AND EXAMPLE |
+-----+
```

Configuration files are contained in the subdirectory "cf", with a suffix ".mc". They must be run through "m4" to produce a ".cf" file. You must pre-load "cf.m4":

```
m4 ${CFDIR}/m4/cf.m4 config.mc > config.cf
```

Alternatively, you can simply:

```
cd ${CFDIR}/cf
./Build config.cf
```

where \${CFDIR} is the root of the cf directory and config.mc is the name of your configuration file. If you are running a version of M4 that understands the `__file__` builtin (versions of GNU m4 ≥ 0.75 do this, but the versions distributed with 4.4BSD and derivatives do not) or the `-I` flag (ditto), then \${CFDIR} can be in an arbitrary directory. For "traditional" versions, \${CFDIR} **MUST** be `".."`, or you **MUST** use `-D_CF_DIR=/path/to/cf/dir/` -- note the trailing slash! For example:

```
m4 -D_CF_DIR=${CFDIR}/ ${CFDIR}/m4/cf.m4 config.mc > config.cf
```

Let's examine a typical .mc file:

```
divert(-1)
#
# Copyright (c) 1998-2002 Sendmail, Inc. and its suppliers.
# All rights reserved.
# Copyright (c) 1983 Eric P. Allman. All rights reserved.
# Copyright (c) 1988, 1993
# The Regents of the University of California. All rights reserved.
#
# By using this file, you agree to the terms and conditions set
# forth in the LICENSE file which can be found at the top level of
# the sendmail distribution.
#
#
# This is a Berkeley-specific configuration file for HP-UX 9.x.
# It applies only to the Computer Science Division at Berkeley,
# and should not be used elsewhere. It is provided on the sendmail
# distribution as a sample only. To create your own configuration
# file, create an appropriate domain file in ../domain, change the
# 'DOMAIN' macro below to reference that file, and copy the result
# to a name of your own choosing.
#
divert(0)
```

The `divert(-1)` will delete the crud in the resulting output file. The copyright notice can be replaced by whatever your lawyers require; our lawyers require the one that is included in these files. A copyleft is a copyright by another name. The `divert(0)` restores regular output.

```
VERSIONID('<SCCS or RCS version id>')
```

VERSIONID is a macro that stuffs the version information into the resulting file. You could use SCCS, RCS, CVS, something else, or omit it completely. This is not the same as the version id included in SMTP greeting messages -- this is defined in `m4/version.m4`.

```
OSTYPE('hpux9')dnl
```

You must specify an OSTYPE to properly configure things such as the pathname of the help and status files, the flags needed for the local mailer, and other important things. If you omit it, you will get an error when you try to build the configuration. Look at the ostype directory for the list of known operating system types.

```
DOMAIN('CS.Berkeley.EDU')dnl
```

This example is specific to the Computer Science Division at Berkeley. You can use "DOMAIN('generic')" to get a sufficiently bland definition that may well work for you, or you can create a customized domain definition appropriate for your environment.

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

These describe the mailers used at the default CS site. The local mailer is always included automatically. Beware: MAILER declarations should always be at the end of the configuration file. The general rules are that the order should be:

```
VERSIONID
OSTYPE
DOMAIN
FEATURE
local macro definitions
MAILER
LOCAL_CONFIG
LOCAL_RULE_*
LOCAL_RULESETS
```

There are a few exceptions to this rule. Local macro definitions which influence a FEATURE() should be done before that feature. For example, a define('PROCMAIL_MAILER_PATH', ...) should be done before FEATURE('local_procmail').

```
*****
***  BE SURE YOU CUSTOMIZE THESE FILES!  They have some      ***
***  Berkeley-specific assumptions built in, such as the name ***
***  of their UUCP-relay.  You'll want to create your own    ***
***  domain description, and use that in place of            ***
***  domain/Berkeley.EDU.m4.                                  ***
*****
```

```
+-----+
| A BRIEF INTRODUCTION TO M4 |
+-----+
```

Sendmail uses the M4 macro processor to ``compile'' the configuration files. The most important thing to know is that M4 is stream-based, that is, it doesn't understand about lines. For this reason, in some places you may see the word ``dnl'', which stands for ``delete through newline''; essentially, it deletes all characters starting at the ``dnl'' up to and including the next newline character. In most cases sendmail uses this only to avoid lots of unnecessary blank lines in the output.

Other important directives are define(A, B) which defines the macro ``A'' to have value ``B''. Macros are expanded as they are read, so one normally quotes both values to prevent expansion. For example,

```
define('SMART_HOST', 'smart.foo.com')
```

One word of warning: M4 macros are expanded even in lines that appear to be comments. For example, if you have

```
# See FEATURE('foo') above
```

it will not do what you expect, because the FEATURE('foo') will be expanded. This also applies to

```
# And then define the $X macro to be the return address
```

because ``define`` is an M4 keyword. If you want to use them, surround them with directed quotes, 'like this'.

Notice:

This package requires a post-V7 version of m4; if you are running the 4.2bsd, SysV.2, or 7th Edition version. SunOS's /usr/5bin/m4 or BSD-Net/2's m4 both work. GNU m4 version 1.1 or later also works. Unfortunately, the M4 on BSDI 1.0 doesn't work -- you'll have to use a Net/2 or GNU version. GNU m4 is available from <ftp://ftp.gnu.org/pub/gnu/m4/m4-1.4.tar.gz> (check for the latest version). EXCEPTIONS: DEC's m4 on Digital UNIX 4.x is broken (3.x is fine). Use GNU m4 on this platform.

```
+-----+
| FILE LOCATIONS |
+-----+
```

sendmail 8.9 has introduced a new configuration directory for sendmail related files, /etc/mail. The new files available for sendmail 8.9 -- the class {R} /etc/mail/relay-domains and the access database /etc/mail/access -- take advantage of this new directory. Beginning with 8.10, all files will use this directory by default (some options may be set by OSTYPE() files). This new directory should help to restore uniformity to sendmail's file locations.

Below is a table of some of the common changes:

Old filename	New filename
-----	-----
/etc/bitdomain	/etc/mail/bitdomain
/etc/domaintable	/etc/mail/domaintable
/etc/genericstable	/etc/mail/genericstable
/etc/uudomain	/etc/mail/uudomain
/etc/virtusertable	/etc/mail/virtusertable
/etc/userdb	/etc/mail/userdb
/etc/aliases	/etc/mail/aliases
/etc/sendmail/aliases	/etc/mail/aliases
/etc/ucbmail/aliases	/etc/mail/aliases
/usr/adm/sendmail/aliases	/etc/mail/aliases
/usr/lib/aliases	/etc/mail/aliases
/usr/lib/mail/aliases	/etc/mail/aliases
/usr/ucblib/aliases	/etc/mail/aliases
/etc/sendmail.cw	/etc/mail/local-host-names
/etc/mail/sendmail.cw	/etc/mail/local-host-names
/etc/sendmail/sendmail.cw	/etc/mail/local-host-names

/etc/sendmail.ct	/etc/mail/trusted-users
/etc/sendmail.oE	/etc/mail/error-header
/etc/sendmail.hf	/etc/mail/helpfile
/etc/mail/sendmail.hf	/etc/mail/helpfile
/usr/ucblib/sendmail.hf	/etc/mail/helpfile
/etc/ucbmail/sendmail.hf	/etc/mail/helpfile
/usr/lib/sendmail.hf	/etc/mail/helpfile
/usr/share/lib/sendmail.hf	/etc/mail/helpfile
/usr/share/misc/sendmail.hf	/etc/mail/helpfile
/share/misc/sendmail.hf	/etc/mail/helpfile
/etc/service.switch	/etc/mail/service.switch
/etc/sendmail.st	/etc/mail/statistics
/etc/mail/sendmail.st	/etc/mail/statistics
/etc/mailer/sendmail.st	/etc/mail/statistics
/etc/sendmail/sendmail.st	/etc/mail/statistics
/usr/lib/sendmail.st	/etc/mail/statistics
/usr/ucblib/sendmail.st	/etc/mail/statistics

Note that all of these paths actually use a new m4 macro MAIL_SETTINGS_DIR to create the pathnames. The default value of this variable is '/etc/mail/'. If you set this macro to a different value, you MUST include a trailing slash.

Notice: all filenames used in a .mc (or .cf) file should be absolute (starting at the root, i.e., with '/'). Relative filenames most likely cause surprises during operations (unless otherwise noted).

```
+-----+
| OSTYPE |
+-----+
```

You MUST define an operating system environment, or the configuration file build will puke. There are several environments available; look at the "ostype" directory for the current list. This macro changes things like the location of the alias file and queue directory. Some of these files are identical to one another.

It is IMPERATIVE that the OSTYPE occur before any MAILER definitions. In general, the OSTYPE macro should go immediately after any version information, and MAILER definitions should always go last.

Operating system definitions are usually easy to write. They may define the following variables (everything defaults, so an ostype file may be empty). Unfortunately, the list of configuration-supported systems is not as broad as the list of source-supported systems, since many of the source contributors do not include corresponding ostype files.

ALIAS_FILE	[/etc/mail/aliases] The location of the text version of the alias file(s). It can be a comma-separated list of names (but be sure you quote values with commas in them -- for example, use <pre>define('ALIAS_FILE', 'a,b')</pre> to get "a" and "b" both listed as alias files; otherwise the define() primitive only sees "a").
HELP_FILE	[/etc/mail/helpfile] The name of the file

	containing information printed in response to the SMTP HELP command.
QUEUE_DIR	[/var/spool/mqueue] The directory containing queue files. To use multiple queues, supply a value ending with an asterisk. For example, /var/spool/mqueue/qd* will use all of the directories or symbolic links to directories beginning with 'qd' in /var/spool/mqueue as queue directories. The names 'qf', 'df', and 'xf' are reserved as specific subdirectories for the corresponding queue file types as explained in doc/op/op.me. See also QUEUE GROUP DEFINITIONS.
MSP_QUEUE_DIR	[/var/spool/clientmqueue] The directory containing queue files for the MSP (Mail Submission Program, see sendmail/SECURITY).
STATUS_FILE	[/etc/mail/statistics] The file containing status information.
LOCAL_MAILER_PATH	[/bin/mail] The program used to deliver local mail.
LOCAL_MAILER_FLAGS	[Prmn9] The flags used by the local mailer. The flags lsDFMAw5:/ @q are always included.
LOCAL_MAILER_ARGS	[mail -d \$u] The arguments passed to deliver local mail.
LOCAL_MAILER_MAX	[undefined] If defined, the maximum size of local mail that you are willing to accept.
LOCAL_MAILER_MAXMSG	[undefined] If defined, the maximum number of messages to deliver in a single connection. Only useful for LMTP local mailers.
LOCAL_MAILER_CHARSET	[undefined] If defined, messages containing 8-bit data that ARRIVE from an address that resolves to the local mailer and which are converted to MIME will be labeled with this character set.
LOCAL_MAILER_EOL	[undefined] If defined, the string to use as the end of line for the local mailer.
LOCAL_MAILER_DSN_DIAGNOSTIC_CODE	[X-Unix] The DSN Diagnostic-Code value for the local mailer. This should be changed with care.
LOCAL_SHELL_PATH	[/bin/sh] The shell used to deliver piped email.
LOCAL_SHELL_FLAGS	[eu9] The flags used by the shell mailer. The flags lsDFM are always included.
LOCAL_SHELL_ARGS	[sh -c \$u] The arguments passed to deliver "prog" mail.
LOCAL_SHELL_DIR	[\$z:/] The directory search path in which the shell should run.
LOCAL_MAILER_QGRP	[undefined] The queue group for the local mailer.
USENET_MAILER_PATH	[/usr/lib/news/inews] The name of the program used to submit news.
USENET_MAILER_FLAGS	[rsDFMmn] The mailer flags for the usenet mailer.
USENET_MAILER_ARGS	[-m -h -n] The command line arguments for the usenet mailer. NOTE: Some versions of inews (such as those shipped with newer versions of INN) use different flags. Double check the defaults against the inews man page.
USENET_MAILER_MAX	[100000] The maximum size of messages that will be accepted by the usenet mailer.
USENET_MAILER_QGRP	[undefined] The queue group for the usenet mailer.
SMTP_MAILER_FLAGS	[undefined] Flags added to SMTP mailer. Default flags are 'mDFMuX' for all SMTP-based mailers; the "esmtplib" mailer adds 'a'; "smtp8" adds '8'; and "dsmtplib" adds '%'. [undefined] Flags added to the relay mailer. Default
RELAY_MAILER_FLAGS	

	flags are 'mDFMuX' for all SMTP-based mailers; the relay mailer adds 'a8'. If this is not defined, then SMTP_MAILER_FLAGS is used.
SMTP_MAILER_MAX	[undefined] The maximum size of messages that will be transported using the smtp, smtp8, esmtp, or dsmtmp mailers.
SMTP_MAILER_MAXMSGSGS	[undefined] If defined, the maximum number of messages to deliver in a single connection for the smtp, smtp8, esmtp, or dsmtmp mailers.
SMTP_MAILER_ARGS	[TCP \$h] The arguments passed to the smtp mailer. About the only reason you would want to change this would be to change the default port.
ESMTP_MAILER_ARGS	[TCP \$h] The arguments passed to the esmtp mailer.
SMTP8_MAILER_ARGS	[TCP \$h] The arguments passed to the smtp8 mailer.
DSMTMP_MAILER_ARGS	[TCP \$h] The arguments passed to the dsmtmp mailer.
RELAY_MAILER_ARGS	[TCP \$h] The arguments passed to the relay mailer.
SMTP_MAILER_QGRP	[undefined] The queue group for the smtp mailer.
ESMTP_MAILER_QGRP	[undefined] The queue group for the esmtp mailer.
SMTP8_MAILER_QGRP	[undefined] The queue group for the smtp8 mailer.
DSMTMP_MAILER_QGRP	[undefined] The queue group for the dsmtmp mailer.
RELAY_MAILER_QGRP	[undefined] The queue group for the relay mailer.
RELAY_MAILER_MAXMSGSGS	[undefined] If defined, the maximum number of messages to deliver in a single connection for the relay mailer.
SMTP_MAILER_CHARSET	[undefined] If defined, messages containing 8-bit data that ARRIVE from an address that resolves to one of the SMTP mailers and which are converted to MIME will be labeled with this character set.
UUCP_MAILER_PATH	[/usr/bin/uux] The program used to send UUCP mail.
UUCP_MAILER_FLAGS	[undefined] Flags added to UUCP mailer. Default flags are 'DFMhuU' (and 'm' for uucp-new mailer, minus 'U' for uucp-dom mailer).
UUCP_MAILER_ARGS	[uux - -r -z -a\$g -gC \$h!rmail (\$u)] The arguments passed to the UUCP mailer.
UUCP_MAILER_MAX	[100000] The maximum size message accepted for transmission by the UUCP mailers.
UUCP_MAILER_CHARSET	[undefined] If defined, messages containing 8-bit data that ARRIVE from an address that resolves to one of the UUCP mailers and which are converted to MIME will be labeled with this character set.
UUCP_MAILER_QGRP	[undefined] The queue group for the UUCP mailers.
FAX_MAILER_PATH	[/usr/local/lib/fax/mailfax] The program used to submit FAX messages.
FAX_MAILER_ARGS	[mailfax \$u \$h \$f] The arguments passed to the FAX mailer.
FAX_MAILER_MAX	[100000] The maximum size message accepted for transmission by FAX.
POP_MAILER_PATH	[/usr/lib/mh/spop] The pathname of the POP mailer.
POP_MAILER_FLAGS	[Penu] Flags added to POP mailer. Flags lsDFMq are always added.
POP_MAILER_ARGS	[pop \$u] The arguments passed to the POP mailer.
POP_MAILER_QGRP	[undefined] The queue group for the pop mailer.
PROCMail_MAILER_PATH	[/usr/local/bin/procmail] The path to the procmail program. This is also used by FEATURE('local_procmail').
PROCMail_MAILER_FLAGS	[SPPhnu9] Flags added to Procmail mailer. Flags DFM are always set. This is NOT used by FEATURE('local_procmail'); tweak LOCAL_MAILER_FLAGS instead.
PROCMail_MAILER_ARGS	[procmail -Y -m \$h \$f \$u] The arguments passed to

	the Procmail mailer. This is NOT used by FEATURE('local_procmail'); tweak LOCAL_MAILER_ARGS instead.
PROCMail_MAILER_MAX	[undefined] If set, the maximum size message that will be accepted by the procmail mailer.
PROCMail_MAILER_QGRP	[undefined] The queue group for the procmail mailer.
Mail11_MAILER_PATH	[/usr/etc/maill1] The path to the maill1 mailer.
Mail11_MAILER_FLAGS	[nsFx] Flags for the maill1 mailer.
Mail11_MAILER_ARGS	[maill1 \$g \$x \$h \$u] Arguments passed to the maill1 mailer.
Mail11_MAILER_QGRP	[undefined] The queue group for the maill1 mailer.
PH_MAILER_PATH	[/usr/local/etc/phquery] The path to the phquery program.
PH_MAILER_FLAGS	[ehmu] Flags for the phquery mailer. Flags nrDFM are always set.
PH_MAILER_ARGS	[phquery -- \$u] -- arguments to the phquery mailer.
PH_MAILER_QGRP	[undefined] The queue group for the ph mailer.
CYRUS_MAILER_FLAGS	[Ah5@/:] The flags used by the cyrus mailer. The flags lsDFMnPq are always included.
CYRUS_MAILER_PATH	[/usr/cyrus/bin/deliver] The program used to deliver cyrus mail.
CYRUS_MAILER_ARGS	[deliver -e -m \$h -- \$u] The arguments passed to deliver cyrus mail.
CYRUS_MAILER_MAX	[undefined] If set, the maximum size message that will be accepted by the cyrus mailer.
CYRUS_MAILER_USER	[cyrus:mail] The user and group to become when running the cyrus mailer.
CYRUS_MAILER_QGRP	[undefined] The queue group for the cyrus mailer.
CYRUS_BB_MAILER_FLAGS	[u] The flags used by the cyrusbb mailer. The flags lsDFMnP are always included.
CYRUS_BB_MAILER_ARGS	[deliver -e -m \$u] The arguments passed to deliver cyrusbb mail.
confEBINDIR	[/usr/libexec] The directory for executables. Currently used for FEATURE('local_lmtp') and FEATURE('smrsh').
QPAGE_MAILER_FLAGS	[mDFMs] The flags used by the qpage mailer.
QPAGE_MAILER_PATH	[/usr/local/bin/qpage] The program used to deliver qpage mail.
QPAGE_MAILER_ARGS	[qpage -l0 -m -P\$u] The arguments passed to deliver qpage mail.
QPAGE_MAILER_MAX	[4096] If set, the maximum size message that will be accepted by the qpage mailer.
QPAGE_MAILER_QGRP	[undefined] The queue group for the qpage mailer.
LOCAL_PROG_QGRP	[undefined] The queue group for the prog mailer.

Note: to tweak Name_MAILER_FLAGS use the macro MODIFY_MAILER_FLAGS:
MODIFY_MAILER_FLAGS('Name', 'change') where Name is the first part of
the macro Name_MAILER_FLAGS and change can be: flags that should
be used directly (thus overriding the default value), or if it
starts with '+' ('-') then those flags are added to (removed from)
the default value. Example:

```
MODIFY_MAILER_FLAGS('LOCAL', '+e')
```

will add the flag 'e' to LOCAL_MAILER_FLAGS. Notice: there are
several smtp mailers all of which are manipulated individually.
See the section MAILERS for the available mailer names.
WARNING: The FEATURES local_lmtp and local_procmail set LOCAL_MAILER_FLAGS
unconditionally, i.e., without respecting any definitions in an
OSTYPE setting.


```
+-----+
| DOMAINS |
+-----+
```

You will probably want to collect domain-dependent defines into one file, referenced by the DOMAIN macro. For example, the Berkeley domain file includes definitions for several internal distinguished hosts:

UUCP_RELAY	The host that will accept UUCP-addressed email. If not defined, all UUCP sites must be directly connected.
BITNET_RELAY	The host that will accept BITNET-addressed email. If not defined, the .BITNET pseudo-domain won't work.
DECNET_RELAY	The host that will accept DECNET-addressed email. If not defined, the .DECNET pseudo-domain and addresses of the form node::user will not work.
FAX_RELAY	The host that will accept mail to the .FAX pseudo-domain. The "fax" mailer overrides this value.
LOCAL_RELAY	The site that will handle unqualified names -- that is, names without an @domain extension. Normally MAIL_HUB is preferred for this function. LOCAL_RELAY is mostly useful in conjunction with FEATURE('stickyhost') -- see the discussion of stickyhost below. If not set, they are assumed to belong on this machine. This allows you to have a central site to store a company- or department-wide alias database. This only works at small sites, and only with some user agents.
LUSER_RELAY	The site that will handle lusers -- that is, apparently local names that aren't local accounts or aliases. To specify a local user instead of a site, set this to 'local:username'.

Any of these can be either 'mailer:hostname' (in which case the mailer is the internal mailer name, such as 'uucp-new' and the hostname is the name of the host as appropriate for that mailer) or just a 'hostname', in which case a default mailer type (usually 'relay', a variant on SMTP) is used. WARNING: if you have a wildcard MX record matching your domain, you probably want to define these to have a trailing dot so that you won't get the mail diverted back to yourself.

The domain file can also be used to define a domain name, if needed (using "DD<domain>") and set certain site-wide features. If all hosts at your site masquerade behind one email name, you could also use MASQUERADE_AS here.

You do not have to define a domain -- in particular, if you are a single machine sitting off somewhere, it is probably more work than it's worth. This is just a mechanism for combining "domain dependent knowledge" into one place.

```
+-----+
| MAILERS |
+-----+
```

There are fewer mailers supported in this version than the previous version, owing mostly to a simpler world. As a general rule, put the MAILER definitions last in your .mc file.

local	The local and prog mailers. You will almost always need these; the only exception is if you relay ALL your mail to another site. This mailer is included automatically.
smtp	The Simple Mail Transport Protocol mailer. This does not hide hosts behind a gateway or another other such hack; it assumes a world where everyone is running the name server. This file actually defines five mailers: "smtp" for regular (old-style) SMTP to other servers, "esmtplib" for extended SMTP to other servers, "smtp8" to do SMTP to other servers without converting 8-bit data to MIME (essentially, this is your statement that you know the other end is 8-bit clean even if it doesn't say so), "dsmtplib" to do on demand delivery, and "relay" for transmission to the RELAY_HOST, LUSER_RELAY, or MAIL_HUB.
uucp	The UNIX-to-UNIX Copy Program mailer. Actually, this defines two mailers, "uucp-old" (a.k.a. "uucp") and "uucp-new" (a.k.a. "suucp"). The latter is for when you know that the UUCP mailer at the other end can handle multiple recipients in one transfer. If the smtp mailer is included in your configuration, two other mailers ("uucp-dom" and "uucp-uudom") are also defined [warning: you MUST specify MAILER('smtp') before MAILER('uucp')]. When you include the uucp mailer, sendmail looks for all names in class {U} and sends them to the uucp-old mailer; all names in class {Y} are sent to uucp-new; and all names in class {Z} are sent to uucp-uudom. Note that this is a function of what version of rmail runs on the receiving end, and hence may be out of your control. See the section below describing UUCP mailers in more detail.
usenet	Usenet (network news) delivery. If this is specified, an extra rule is added to ruleset 0 that forwards all local email for users named ``group.usenet`` to the ``inews`` program. Note that this works for all groups, and may be considered a security problem.
fax	Facsimile transmission. This is experimental and based on Sam Leffler's HylaFAX software. For more information, see http://www.hylafax.org/ .
pop	Post Office Protocol.
procmail	An interface to procmail (does not come with sendmail). This is designed to be used in mailertables. For example, a common question is "how do I forward all mail for a given domain to a single person?". If you have this mailer defined, you could set up a mailertable reading:

host.com procmail:/etc/procmailrcs/host.com

with the file /etc/procmailrcs/host.com reading:

```
:0      # forward mail for host.com
! -oi -f $1 person@other.host
```

This would arrange for (anything)@host.com to be sent to person@other.host. Within the procmail script, \$1 is the name of the sender and \$2 is the name of the recipient. If you use this with FEATURE('local_procmail'), the FEATURE should be listed first.

Of course there are other ways to solve this particular problem, e.g., a catch-all entry in a virtusertable.

maill11 The DECnet maill11 mailer, useful only if you have the maill11 program from gatekeeper.dec.com:/pub/DEC/gwtools (and DECnet, of course). This is for Phase IV DECnet support; if you have Phase V at your site you may have additional problems.

phquery The phquery program. This is somewhat counterintuitively referenced as the "ph" mailer internally. It can be used to do CCSO name server lookups. The phquery program, which this mailer uses, is distributed with the ph client.

cyrus The cyrus and cyrusbb mailers. The cyrus mailer delivers to a local cyrus user. this mailer can make use of the "user+detail@local.host" syntax (see FEATURE('preserve_local_plus_detail')); it will deliver the mail to the user's "detail" mailbox if the mailbox's ACL permits. The cyrusbb mailer delivers to a system-wide cyrus mailbox if the mailbox's ACL permits. The cyrus mailer must be defined after the local mailer.

qpage A mailer for QuickPage, a pager interface. See <http://www.qpage.org/> for further information.

The local mailer accepts addresses of the form "user+detail", where the "+detail" is not used for mailbox matching but is available to certain local mail programs (in particular, see FEATURE('local_procmail')). For example, "eric", "eric+sendmail", and "eric+swm" all indicate the same user, but additional arguments <null>, "sendmail", and "swm" may be provided for use in sorting mail.

```
+-----+
| FEATURES |
+-----+
```

Special features can be requested using the "FEATURE" macro. For example, the .mc line:

```
FEATURE('use_cw_file')
```

tells sendmail that you want to have it read an /etc/mail/local-host-names file to get values for class {w}. A FEATURE may contain up to 9 optional parameters -- for example:

```
FEATURE('mailertable', 'dbm /usr/lib/mailertable')
```

The default database map type for the table features can be set with

```
define('DATABASE_MAP_TYPE', 'dbm')
```

which would set it to use ndbm databases. The default is the Berkeley DB hash database format. Note that you must still declare a database map type if you specify an argument to a FEATURE. DATABASE_MAP_TYPE is only used if no argument is given for the FEATURE. It must be specified before any feature that uses a map.

Also, features which can take a map definition as an argument can also take the special keyword 'LDAP'. If that keyword is used, the map will use the LDAP definition described in the ''USING LDAP FOR ALIASES, MAPS, AND CLASSES'' section below.

Available features are:

use_cw_file	Read the file /etc/mail/local-host-names file to get alternate names for this host. This might be used if you were on a host that MXed for a dynamic set of other hosts. If the set is static, just including the line "Cw<name1> <name2> ..." (where the names are fully qualified domain names) is probably superior. The actual filename can be overridden by redefining confCW_FILE.
use_ct_file	Read the file /etc/mail/trusted-users file to get the names of users that will be ''trusted'', that is, able to set their envelope from address using -f without generating a warning message. The actual filename can be overridden by redefining confCT_FILE.
redirect	Reject all mail addressed to "address.REDIRECT" with a ''551 User has moved; please try <address>'' message. If this is set, you can alias people who have left to their new address with ".REDIRECT" appended.
nouucp	Don't route UUCP addresses. This feature takes one parameter: 'reject': reject addresses which have "!" in the local part unless it originates from a system that is allowed to relay. 'nospecial': don't do anything special with "!". Warnings: 1. See the notice in the anti-spam section. 2. don't remove "!" from OperatorChars if 'reject' is given as parameter.
nocanonify	Don't pass addresses to \${ ... \$} for canonification by default, i.e., host/domain names are considered canonical, except for unqualified names, which must not be used in this mode (violation of the standard). It can be changed by setting the DaemonPortOptions modifiers (M=). That is, FEATURE('nocanonify') will be overridden by setting the 'c' flag. Conversely, if FEATURE('nocanonify') is not used, it can be emulated by setting the 'C' flag (DaemonPortOptions=Modifiers=C). This would generally only be used by sites that only act as mail gateways or which have user agents that do full canonification themselves. You may also want to use "define('confBIND_OPTS', '-DNSRCH -DEFNAMES')" to turn off the usual resolver options that do a similar thing.

An exception list for FEATURE('nocanonify') can be specified with CANONIFY_DOMAIN or CANONIFY_DOMAIN_FILE, i.e., a list of domains which are nevertheless passed to \$[... \$] for canonification. This is useful to turn on canonification for local domains, e.g., use CANONIFY_DOMAIN('my.domain my') to canonify addresses which end in "my.domain" or "my". Another way to require canonification in the local domain is CANONIFY_DOMAIN('\$=m').

A trailing dot is added to addresses with more than one component in it such that other features which expect a trailing dot (e.g., virtusertable) will still work.

If 'canonify_hosts' is specified as parameter, i.e., FEATURE('nocanonify', 'canonify_hosts'), then addresses which have only a hostname, e.g., <user@host>, will be canonified (and hopefully fully qualified), too.

stickyhost

This feature is sometimes used with LOCAL_RELAY, although it can be used for a different effect with MAIL_HUB.

When used without MAIL_HUB, email sent to "user@local.host" are marked as "sticky" -- that is, the local addresses aren't matched against UDB, don't go through ruleset 5, and are not forwarded to the LOCAL_RELAY (if defined).

With MAIL_HUB, mail addressed to "user@local.host" is forwarded to the mail hub, with the envelope address still remaining "user@local.host". Without stickyhost, the envelope would be changed to "user@mail_hub", in order to protect against mailing loops.

mailertable

Include a "mailer table" which can be used to override routing for particular domains (which are not in class {w}, i.e. local host names). The argument of the FEATURE may be the key definition. If none is specified, the definition used is:

```
hash /etc/mail/mailertable
```

Keys in this database are fully qualified domain names or partial domains preceded by a dot -- for example, "vangogh.CS.Berkeley.EDU" or ".CS.Berkeley.EDU". As a special case of the latter, "." matches any domain not covered by other keys. Values must be of the form:

```
mailer:domain
```

where "mailer" is the internal mailer name, and "domain" is where to send the message. These maps are not reflected into the message header. As a special case, the forms:

```
local:user
```

will forward to the indicated user using the local mailer,

```
local:
```

will forward to the original user in the e-mail address

using the local mailer, and
error:code message
error:D.S.N:code message
will give an error message with the indicated SMTP reply
code and message, where D.S.N is an RFC 1893 compliant
error code.

domaintable Include a "domain table" which can be used to provide
domain name mapping. Use of this should really be
limited to your own domains. It may be useful if you
change names (e.g., your company changes names from
oldname.com to newname.com). The argument of the
FEATURE may be the key definition. If none is specified,
the definition used is:

hash /etc/mail/domaintable

The key in this table is the domain name; the value is
the new (fully qualified) domain. Anything in the
domaintable is reflected into headers; that is, this
is done in ruleset 3.

bitdomain Look up bitnet hosts in a table to try to turn them into
internet addresses. The table can be built using the
bitdomain program contributed by John Gardiner Myers.
The argument of the FEATURE may be the key definition; if
none is specified, the definition used is:

hash /etc/mail/bitdomain

Keys are the bitnet hostname; values are the corresponding
internet hostname.

uucpdomain Similar feature for UUCP hosts. The default map definition
is:

hash /etc/mail/uudomain

At the moment there is no automagic tool to build this
database.

always_add_domain Include the local host domain even on locally delivered
mail. Normally it is not added on unqualified names.
However, if you use a shared message store but do not use
the same user name space everywhere, you may need the host
name on local names. An optional argument specifies
another domain to be added than the local.

allmasquerade If masquerading is enabled (using MASQUERADE_AS), this
feature will cause recipient addresses to also masquerade
as being from the masquerade host. Normally they get
the local hostname. Although this may be right for
ordinary users, it can break local aliases. For example,
if you send to "localalias", the originating sendmail will
find that alias and send to all members, but send the
message with "To: localalias@masqueradehost". Since that
alias likely does not exist, replies will fail. Use this
feature ONLY if you can guarantee that the ENTIRE
namespace on your masquerade host supersedes all the

local entries.

limited_masquerade

Normally, any hosts listed in class {w} are masqueraded. If this feature is given, only the hosts listed in class {M} (see below: MASQUERADE_DOMAIN) are masqueraded. This is useful if you have several domains with disjoint namespaces hosted on the same machine.

masquerade_entire_domain

If masquerading is enabled (using MASQUERADE_AS) and MASQUERADE_DOMAIN (see below) is set, this feature will cause addresses to be rewritten such that the masquerading domains are actually entire domains to be hidden. All hosts within the masquerading domains will be rewritten to the masquerade name (used in MASQUERADE_AS). For example, if you have:

```
MASQUERADE_AS('masq.com')
MASQUERADE_DOMAIN('foo.org')
MASQUERADE_DOMAIN('bar.com')
```

then *foo.org and *bar.com are converted to masq.com. Without this feature, only foo.org and bar.com are masqueraded.

NOTE: only domains within your jurisdiction and current hierarchy should be masqueraded using this.

local_no_masquerade

This feature prevents the local mailer from masquerading even if MASQUERADE_AS is used. MASQUERADE_AS will only have effect on addresses of mail going outside the local domain.

genericstable

This feature will cause unqualified addresses (i.e., without a domain) and addresses with a domain listed in class {G} to be looked up in a map and turned into another ("generic") form, which can change both the domain name and the user name. Notice: if you use an MSP (as it is default starting with 8.12), the MTA will only receive qualified addresses from the MSP (as required by the RFCs). Hence you need to add your domain to class {G}. This feature is similar to the userdb functionality. The same types of addresses as for masquerading are looked up, i.e., only header sender addresses unless the allmasquerade and/or masquerade_envelope features are given. Qualified addresses must have the domain part in class {G}; entries can be added to this class by the macros GENERICS_DOMAIN or GENERICS_DOMAIN_FILE (analogously to MASQUERADE_DOMAIN and MASQUERADE_DOMAIN_FILE, see below).

The argument of FEATURE('genericstable') may be the map definition; the default map definition is:

```
hash /etc/mail/genericstable
```

The key for this table is either the full address, the domain (with a leading @; the localpart is passed as first argument) or the unqualified username (tried in the order mentioned); the value is the new user address. If the new user address does not include a domain, it will be qualified in the standard manner, i.e., using \$j or the masquerade name. Note that the

address being looked up must be fully qualified. For local mail, it is necessary to use FEATURE('always_add_domain') for the addresses to be qualified. The "+detail" of an address is passed as %1, so entries like

old+*@foo.org	new+%1@example.com
gen+*@foo.org	%1@example.com

and other forms are possible.

generics_entire_domain

If the genericstable is enabled and GENERICS_DOMAIN or GENERICS_DOMAIN_FILE is used, this feature will cause addresses to be searched in the map if their domain parts are subdomains of elements in class {G}.

virtusertable

A domain-specific form of aliasing, allowing multiple virtual domains to be hosted on one machine. For example, if the virtuser table contained:

info@foo.com	foo-info
info@bar.com	bar-info
joe@bar.com	error:nouser 550 No such user here
jax@bar.com	error:5.7.0:550 Address invalid
@baz.org	jane@example.net

then mail addressed to info@foo.com will be sent to the address foo-info, mail addressed to info@bar.com will be delivered to bar-info, and mail addressed to anyone at baz.org will be sent to jane@example.net, mail to joe@bar.com will be rejected with the specified error message, and mail to jax@bar.com will also have a RFC 1893 compliant error code 5.7.0.

The username from the original address is passed as %1 allowing:

@foo.org	%1@example.com
----------	----------------

meaning someone@foo.org will be sent to someone@example.com. Additionally, if the local part consists of "user+detail" then "detail" is passed as %2 and "+detail" is passed as %3 when a match against user+* is attempted, so entries like

old+*@foo.org	new+%2@example.com
gen+*@foo.org	%2@example.com
+*@foo.org	%1%3@example.com
X++@foo.org	Z%3@example.com
@bar.org	%1%3

and other forms are possible. Note: to preserve "+detail" for a default case (@domain) %1%3 must be used as RHS. There are two wildcards after "+": "+" matches only a non-empty detail, "*" matches also empty details, e.g., user+@foo.org matches +*@foo.org but not ++@foo.org. This can be used to ensure that the parameters %2 and %3 are not empty.

All the host names on the left hand side (foo.com, bar.com, and baz.org) must be in class {w} or class {VirtHost}. The latter can be defined by the macros VIRTUSER_DOMAIN or

VIRTUSER_DOMAIN_FILE (analogously to MASQUERADE_DOMAIN and MASQUERADE_DOMAIN_FILE, see below). If VIRTUSER_DOMAIN or VIRTUSER_DOMAIN_FILE is used, then the entries of class {VirtHost} are added to class {R}, i.e., relaying is allowed to (and from) those domains. The default map definition is:

```
hash /etc/mail/virtusertable
```

A new definition can be specified as the second argument of the FEATURE macro, such as

```
FEATURE('virtusertable', 'dbm /etc/mail/virtusers')
```

virtuser_entire_domain

If the virtusertable is enabled and VIRTUSER_DOMAIN or VIRTUSER_DOMAIN_FILE is used, this feature will cause addresses to be searched in the map if their domain parts are subdomains of elements in class {VirtHost}.

ldap_routing

Implement LDAP-based e-mail recipient routing according to the Internet Draft draft-lachman-laser-ldap-mail-routing-01. This provides a method to re-route addresses with a domain portion in class {LDAPRoute} to either a different mail host or a different address. Hosts can be added to this class using LDAPROUTE_DOMAIN and LDAPROUTE_DOMAIN_FILE (analogously to MASQUERADE_DOMAIN and MASQUERADE_DOMAIN_FILE, see below).

See the LDAP ROUTING section below for more information.

nodns

If you aren't running DNS at your site (for example, you are UUCP-only connected). It's hard to consider this a "feature", but hey, it had to go somewhere. Actually, as of 8.7 this is a no-op -- remove "dns" from the hosts service switch entry instead.

nullclient

This is a special case -- it creates a configuration file containing nothing but support for forwarding all mail to a central hub via a local SMTP-based network. The argument is the name of that hub.

The only other feature that should be used in conjunction with this one is FEATURE('nocanonify'). No mailers should be defined. No aliasing or forwarding is done.

local_lmtp

Use an LMTP capable local mailer. The argument to this feature is the pathname of an LMTP capable mailer. By default, mail.local is used. This is expected to be the mail.local which came with the 8.9 distribution which is LMTP capable. The path to mail.local is set by the confEBINDIR m4 variable -- making the default LOCAL_MAILER_PATH /usr/libexec/mail.local. WARNING: This feature sets LOCAL_MAILER_FLAGS unconditionally, i.e., without respecting any definitions in an OSTYPE setting.

local_procmail

Use procmail or another delivery agent as the local mailer. The argument to this feature is the pathname of the delivery agent, which defaults to PROCMAIL_MAILER_PATH. Note that this does NOT use PROCMAIL_MAILER_FLAGS or PROCMAIL_MAILER_ARGS for the local mailer; tweak

LOCAL_MAILER_FLAGS and LOCAL_MAILER_ARGS instead, or specify the appropriate parameters. When procmail is used, the local mailer can make use of the "user+indicator@local.host" syntax; normally the +indicator is just tossed, but by default it is passed as the -a argument to procmail.

This feature can take up to three arguments:

1. Path to the mailer program
[default: /usr/local/bin/procmail]
2. Argument vector including name of the program
[default: procmail -Y -a \$h -d \$u]
3. Flags for the mailer [default: SPfhn9]

Empty arguments cause the defaults to be taken.

For example, this allows it to use the maildrop (<http://www.flounder.net/~mrsam/maildrop/>) mailer instead by specifying:

```
FEATURE('local_procmail', '/usr/local/bin/maildrop',  
        'maildrop -d $u')
```

or scanmails using:

```
FEATURE('local_procmail', '/usr/local/bin/scanmails')
```

WARNING: This feature sets LOCAL_MAILER_FLAGS unconditionally, i.e., without respecting any definitions in an OSTYPE setting.

bestmx_is_local Accept mail as though locally addressed for any host that lists us as the best possible MX record. This generates additional DNS traffic, but should be OK for low to medium traffic hosts. The argument may be a set of domains, which will limit the feature to only apply to these domains -- this will reduce unnecessary DNS traffic. THIS FEATURE IS FUNDAMENTALLY INCOMPATIBLE WITH WILDCARD MX RECORDS!!! If you have a wildcard MX record that matches your domain, you cannot use this feature.

smrsh Use the SendMail Restricted SHell (smrsh) provided with the distribution instead of /bin/sh for mailing to programs. This improves the ability of the local system administrator to control what gets run via e-mail. If an argument is provided it is used as the pathname to smrsh; otherwise, the path defined by confEBINDIR is used for the smrsh binary -- by default, /usr/libexec/smrsh is assumed.

promiscuous_relay By default, the sendmail configuration files do not permit mail relaying (that is, accepting mail from outside your local host (class {w}) and sending it to another host than your local host). This option sets your site to allow mail relaying from any site to any site. In almost all cases, it is better to control relaying more carefully with the access map, class {R}, or authentication. Domains can be added to class {R} by the macros RELAY_DOMAIN or RELAY_DOMAIN_FILE (analogously to MASQUERADE_DOMAIN and

MASQUERADE_DOMAIN_FILE, see below).

relay_entire_domain

By default, only hosts listed as RELAY in the access db will be allowed to relay. This option also allows any host in your domain as defined by class {m}.

relay_hosts_only

By default, names that are listed as RELAY in the access db and class {R} are domain names, not host names. For example, if you specify ``foo.com'', then mail to or from foo.com, abc.foo.com, or a.very.deep.domain.foo.com will all be accepted for relaying. This feature changes the behaviour to lookup individual host names only.

relay_based_on_MX

Turns on the ability to allow relaying based on the MX records of the host portion of an incoming recipient; that is, if an MX record for host foo.com points to your site, you will accept and relay mail addressed to foo.com. See description below for more information before using this feature. Also, see the KNOWNBUGS entry regarding bestmx map lookups.

FEATURE('relay_based_on_MX') does not necessarily allow routing of these messages which you expect to be allowed, if route address syntax (or %-hack syntax) is used. If this is a problem, add entries to the access-table or use FEATURE('loose_relay_check').

relay_mail_from

Allows relaying if the mail sender is listed as RELAY in the access map. If an optional argument 'domain' is given, relaying can be allowed just based on the domain portion of the sender address. This feature should only be used if absolutely necessary as the sender address can be easily forged. Use of this feature requires the "From:" tag be prepended to the key in the access map; see the discussion of tags and FEATURE('relay_mail_from') in the section on anti-spam configuration control.

relay_local_from

Allows relaying if the domain portion of the mail sender is a local host. This should only be used if absolutely necessary as it opens a window for spammers. Specifically, they can send mail to your mail server that claims to be from your domain (either directly or via a routed address), and you will go ahead and relay it out to arbitrary hosts on the Internet.

accept_unqualified_senders

Normally, MAIL FROM: commands in the SMTP session will be refused if the connection is a network connection and the sender address does not include a domain name. If your setup sends local mail unqualified (i.e., MAIL FROM: <joe>), you will need to use this feature to accept unqualified sender addresses. Setting the DaemonPortOptions modifier 'u' overrides the default behavior, i.e., unqualified addresses are accepted even without this FEATURE. If this FEATURE is not used, the DaemonPortOptions modifier

'f' can be used to enforce fully qualified addresses.

accept_unresolvable_domains

Normally, MAIL FROM: commands in the SMTP session will be refused if the host part of the argument to MAIL FROM: cannot be located in the host name service (e.g., an A or MX record in DNS). If you are inside a firewall that has only a limited view of the Internet host name space, this could cause problems. In this case you probably want to use this feature to accept all domains on input, even if they are unresolvable.

access_db

Turns on the access database feature. The access db gives you the ability to allow or refuse to accept mail from specified domains for administrative reasons. Moreover, it can control the behavior of sendmail in various situations. By default, the access database specification is:

```
hash -T<TMPF> /etc/mail/access
```

See the anti-spam configuration control section for further important information about this feature. Notice:

"-T<TMPF>" is meant literal, do not replace it by anything.

blacklist_recipients

Turns on the ability to block incoming mail for certain recipient usernames, hostnames, or addresses. For example, you can block incoming mail to user nobody, host foo.mydomain.com, or guest@bar.mydomain.com. These specifications are put in the access db as described in the anti-spam configuration control section later in this document.

delay_checks

The rulesets check_mail and check_relay will not be called when a client connects or issues a MAIL command, respectively. Instead, those rulesets will be called by the check_rcpt ruleset; they will be skipped under certain circumstances. See "Delay all checks" in the anti-spam configuration control section. Note: this feature is incompatible to the versions in 8.10 and 8.11.

dnsbl

Turns on rejection of hosts found in an DNS based rejection list. If an argument is provided it is used as the domain in which blocked hosts are listed; otherwise it defaults to blackholes.mail-abuse.org. An explanation for an DNS based rejection list can be found at <http://mail-abuse.org/rbl/>. A second argument can be used to change the default error message. Without that second argument, the error message will be

Mail from IP-ADDRESS refused by blackhole site SERVER where IP-ADDRESS and SERVER are replaced by the appropriate information. By default, temporary lookup failures are ignored. This behavior can be changed by specifying a third argument, which must be either 't' or a full error message. See the anti-spam configuration control section for an example. The dnsbl feature can be included several times to query different DNS based rejection lists. See also enhdnsbl for an enhanced version.

NOTE: The default DNS blacklist, blackholes.mail-abuse.org,

is a service offered by the Mail Abuse Prevention System (MAPS). As of July 31, 2001, MAPS is a subscription service, so using that network address won't work if you haven't subscribed. Contact MAPS to subscribe (<http://mail-abuse.org/>).

enhdnsbl Enhanced version of dnsbl (see above). Further arguments (up to 5) can be used to specify specific return values from lookups. Temporary lookup failures are ignored unless a third argument is given, which must be either 't' or a full error message. By default, any successful lookup will generate an error. Otherwise the result of the lookup is compared with the supplied argument(s), and only if a match occurs an error is generated. For example,

```
FEATURE('enhdnsbl', 'dnsbl.example.com', '', 't', '127.0.0.2.')
```

will reject the e-mail if the lookup returns the value '127.0.0.2.', or generate a 451 response if the lookup temporarily failed. The arguments can contain metasympols as they are allowed in the LHS of rules. As the example shows, the default values are also used if an empty argument, i.e., '', is specified. This feature requires that sendmail has been compiled with the flag DNSMAP (see sendmail/README).

lookupdotdomain Look up also .domain in the access map. This allows to match only subdomains. It does not work well with FEATURE('relay_hosts_only'), because most lookups for subdomains are suppressed by the latter feature.

loose_relay_check Normally, if % addressing is used for a recipient, e.g. user%site@othersite, and othersite is in class {R}, the check_rcpt ruleset will strip @othersite and recheck user@site for relaying. This feature changes that behavior. It should not be needed for most installations.

authinfo Provide a separate map for client side authentication information. See SMTP AUTHENTICATION for details. By default, the authinfo database specification is:

```
hash /etc/mail/authinfo
```

preserve_luser_host Preserve the name of the recipient host if USER_RELAY is used. Without this option, the domain part of the recipient address will be replaced by the host specified as USER_RELAY. This feature only works if the hostname is passed to the mailer (see mailer triple in op.me). Note that in the default configuration the local mailer does not receive the hostname, i.e., the mailer triple has an empty hostname.

preserve_local_plus_detail Preserve the +detail portion of the address when passing address to local delivery agent. Disables alias and .forward +detail stripping (e.g., given user+detail, only that address will be looked up in the alias file; user+* and user will not be looked up). Only use if the local delivery agent in use supports +detail addressing.

compat_check Enable ruleset check_compat to look up pairs of addresses with the Compat: tag -- Compat:sender<@>recipient -- in the access map. Valid values for the RHS include
 DISCARD silently discard recipient
 TEMP: return a temporary error
 ERROR: return a permanent error
In the last two cases, a 4xy/5xy SMTP reply code should follow the colon.

no_default_msa Don't generate the default MSA daemon, i.e.,
 DAEMON_OPTIONS('Port=587,Name=MSA,M=E')
To define a MSA daemon with other parameters, use this FEATURE and introduce new settings via DAEMON_OPTIONS().

misp Defines config file for Message Submission Program.
See sendmail/SECURITY for details and cf/cf/submit.mc how to use it. An optional argument can be used to override the default of 'localhost' to use as host to send all e-mails to. If 'MSA' is specified as second argument then port 587 is used to contact the server. Example:

```
FEATURE('misp', '', 'MSA')
```

Some more hints about possible changes can be found below in the section MESSAGE SUBMISSION PROGRAM.

queuegroup A simple example how to select a queue group based on the full e-mail address or the domain of the recipient. Selection is done via entries in the access map using the tag QGRP:, for example:

```
QGRP:example.com        main
QGRP:friend@some.org    others
QGRP:my.domain          local
```

where "main", "others", and "local" are names of queue groups. If an argument is specified, it is used as default queue group.

```
+-----+
| HACKS |
+-----+
```

Some things just can't be called features. To make this clear, they go in the hack subdirectory and are referenced using the HACK macro. These will tend to be site-dependent. The release includes the Berkeley-dependent "cssubdomain" hack (that makes sendmail accept local names in either Berkeley.EDU or CS.Berkeley.EDU; this is intended as a short-term aid while moving hosts into subdomains.

```
+-----+
| SITE CONFIGURATION |
+-----+
```

```
*****
* This section is really obsolete, and is preserved *
* only for back compatibility. You should plan on *
```

```

* using mailertables for new installations. In      *
* particular, it doesn't work for the newer forms  *
* of UUCP mailers, such as uucp-uudom.             *
*****

```

Complex sites will need more local configuration information, such as lists of UUCP hosts they speak with directly. This can get a bit more tricky. For an example of a "complex" site, see cf/ucbvax.mc.

The SITECONFIG macro allows you to indirectly reference site-dependent configuration information stored in the siteconfig subdirectory. For example, the line

```
SITECONFIG('uucp.ucbvax', 'ucbvax', 'U')
```

reads the file uucp.ucbvax for local connection information. The second parameter is the local name (in this case just "ucbvax" since it is locally connected, and hence a UUCP hostname). The third parameter is the name of both a macro to store the local name (in this case, {U}) and the name of the class (e.g., {U}) in which to store the host information read from the file. Another SITECONFIG line reads

```
SITECONFIG('uucp.ucbarpa', 'ucbarpa.Berkeley.EDU', 'W')
```

This says that the file uucp.ucbarpa contains the list of UUCP sites connected to ucbarpa.Berkeley.EDU. Class {W} will be used to store this list, and \$W is defined to be ucbarpa.Berkeley.EDU, that is, the name of the relay to which the hosts listed in uucp.ucbarpa are connected. [The machine ucbarpa is gone now, but this out-of-date configuration file has been left around to demonstrate how you might do this.]

Note that the case of SITECONFIG with a third parameter of ``U'' is special; the second parameter is assumed to be the UUCP name of the local site, rather than the name of a remote site, and the UUCP name is entered into class {w} (the list of local hostnames) as \$U.UUCP.

The siteconfig file (e.g., siteconfig/uucp.ucbvax.m4) contains nothing more than a sequence of SITE macros describing connectivity. For example:

```

SITE('cnmat')
SITE('sgi olympus')

```

The second example demonstrates that you can use two names on the same line; these are usually aliases for the same host (or are at least in the same company).

```

+-----+
| USING UUCP MAILERS |
+-----+

```

It's hard to get UUCP mailers right because of the extremely ad hoc nature of UUCP addressing. These config files are really designed for domain-based addressing, even for UUCP sites.

There are four UUCP mailers available. The choice of which one to use is partly a matter of local preferences and what is running at the other end of your UUCP connection. Unlike good protocols that

define what will go over the wire, UUCP uses the policy that you should do what is right for the other end; if they change, you have to change. This makes it hard to do the right thing, and discourages people from updating their software. In general, if you can avoid UUCP, please do.

The major choice is whether to go for a domainized scheme or a non-domainized scheme. This depends entirely on what the other end will recognize. If at all possible, you should encourage the other end to go to a domain-based system -- non-domainized addresses don't work entirely properly.

The four mailers are:

uucp-old (obsolete name: "uucp")

This is the oldest, the worst (but the closest to UUCP) way of sending messages accros UUCP connections. It does bangify everything and prepends \$U (your UUCP name) to the sender's address (which can already be a bang path itself). It can only send to one address at a time, so it spends a lot of time copying duplicates of messages. Avoid this if at all possible.

uucp-new (obsolete name: "suucp")

The same as above, except that it assumes that in one rmail command you can specify several recipients. It still has a lot of other problems.

uucp-dom

This UUCP mailer keeps everything as domain addresses. Basically, it uses the SMTP mailer rewriting rules. This mailer is only included if MAILER('smtp') is specified before MAILER('uucp').

Unfortunately, a lot of UUCP mailer transport agents require bangified addresses in the envelope, although you can use domain-based addresses in the message header. (The envelope shows up as the From_ line on UNIX mail.) So....

uucp-uudom

This is a cross between uucp-new (for the envelope addresses) and uucp-dom (for the header addresses). It bangifies the envelope sender (From_ line in messages) without adding the local hostname, unless there is no host name on the address at all (e.g., "wolf") or the host component is a UUCP host name instead of a domain name ("somehost!wolf" instead of "some.dom.ain!wolf"). This is also included only if MAILER('smtp') is also specified earlier.

Examples:

On host grasp.insa-lyon.fr (UUCP host name "grasp"), the following summarizes the sender rewriting for various mailers.

Mailer	sender	rewriting in the envelope
-----	-----	-----
uucp-{old,new}	wolf	grasp!wolf
uucp-dom	wolf	wolf@grasp.insa-lyon.fr
uucp-uudom	wolf	grasp.insa-lyon.fr!wolf

uucp-{old,new}	wolf@fr.net	grasp!fr.net!wolf
uucp-dom	wolf@fr.net	wolf@fr.net
uucp-uudom	wolf@fr.net	fr.net!wolf
uucp-{old,new}	somehost!wolf	grasp!somehost!wolf
uucp-dom	somehost!wolf	somehost!wolf@grasp.insa-lyon.fr
uucp-uudom	somehost!wolf	grasp.insa-lyon.fr!somehost!wolf

If you are using one of the domainized UUCP mailers, you really want to convert all UUCP addresses to domain format -- otherwise, it will do it for you (and probably not the way you expected). For example, if you have the address foo!bar!baz (and you are not sending to foo), the heuristics will add the @uucp.relay.name or @local.host.name to this address. However, if you map foo to foo.host.name first, it will not add the local hostname. You can do this using the uucpdomain feature.

```
+-----+
| TWEAKING RULESETS |
+-----+
```

For more complex configurations, you can define special rules. The macro LOCAL_RULE_3 introduces rules that are used in canonicalizing the names. Any modifications made here are reflected in the header.

A common use is to convert old UUCP addresses to SMTP addresses using the UUCPSMTP macro. For example:

```
LOCAL_RULE_3
UUCPSMTP('decvax',      'decvax.dec.com')
UUCPSMTP('research',    'research.att.com')
```

will cause addresses of the form "decvax!user" and "research!user" to be converted to "user@decvax.dec.com" and "user@research.att.com" respectively.

This could also be used to look up hosts in a database map:

```
LOCAL_RULE_3
R$* < @ $+ > $*      $: $1 < @ $(hostmap $2 $) > $3
```

This map would be defined in the LOCAL_CONFIG portion, as shown below.

Similarly, LOCAL_RULE_0 can be used to introduce new parsing rules. For example, new rules are needed to parse hostnames that you accept via MX records. For example, you might have:

```
LOCAL_RULE_0
R$+ <@ host.dom.ain.>  $#uucp $@ cnmat $: $1 < @ host.dom.ain.>
```

You would use this if you had installed an MX record for cnmat.Berkeley.EDU pointing at this host; this rule catches the message and forwards it on using UUCP.

You can also tweak rulesets 1 and 2 using LOCAL_RULE_1 and LOCAL_RULE_2. These rulesets are normally empty.

A similar macro is LOCAL_CONFIG. This introduces lines added after the boilerplate option setting but before rulesets. Do not declare rulesets in

the LOCAL_CONFIG section. It can be used to declare local database maps or whatever. For example:

```
LOCAL_CONFIG
Khostmap hash /etc/mail/hostmap
Kyplocal nis -m hosts.byname
```

```
+-----+
| MASQUERADING AND RELAYING |
+-----+
```

You can have your host masquerade as another using

```
MASQUERADE_AS('host.domain')
```

This causes mail being sent to be labeled as coming from the indicated host.domain, rather than \$j. One normally masquerades as one of one's own subdomains (for example, it's unlikely that Berkeley would choose to masquerade as an MIT site). This behaviour is modified by a plethora of FEATURES; in particular, see masquerade_envelope, allmasquerade, limited_masquerade, and masquerade_entire_domain.

The masquerade name is not normally canonified, so it is important that it be your One True Name, that is, fully qualified and not a CNAME. However, if you use a CNAME, the receiving side may canonify it for you, so don't think you can cheat CNAME mapping this way.

Normally the only addresses that are masqueraded are those that come from this host (that is, are either unqualified or in class {w}, the list of local domain names). You can augment this list, which is realized by class {M} using

```
MASQUERADE_DOMAIN('otherhost.domain')
```

The effect of this is that although mail to user@otherhost.domain will not be delivered locally, any mail including any user@otherhost.domain will, when relayed, be rewritten to have the MASQUERADE_AS address. This can be a space-separated list of names.

If these names are in a file, you can use

```
MASQUERADE_DOMAIN_FILE('filename')
```

to read the list of names from the indicated file (i.e., to add elements to class {M}).

To exempt hosts or subdomains from being masqueraded, you can use

```
MASQUERADE_EXCEPTION('host.domain')
```

This can come handy if you want to masquerade a whole domain except for one (or a few) host(s). If these names are in a file, you can use

```
MASQUERADE_EXCEPTION_FILE('filename')
```

Normally only header addresses are masqueraded. If you want to masquerade the envelope as well, use

```
FEATURE('masquerade_envelope')
```

There are always users that need to be "exposed" -- that is, their internal site name should be displayed instead of the masquerade name. Root is an example (which has been "exposed" by default prior to 8.10). You can add users to this list using

```
EXPOSED_USER('usernames')
```

This adds users to class {E}; you could also use

```
EXPOSED_USER_FILE('filename')
```

You can also arrange to relay all unqualified names (that is, names without @host) to a relay host. For example, if you have a central email server, you might relay to that host so that users don't have to have .forward files or aliases. You can do this using

```
define('LOCAL_RELAY', 'mailer:hostname')
```

The ``mailer:'' can be omitted, in which case the mailer defaults to "relay". There are some user names that you don't want relayed, perhaps because of local aliases. A common example is root, which may be locally aliased. You can add entries to this list using

```
LOCAL_USER('usernames')
```

This adds users to class {L}; you could also use

```
LOCAL_USER_FILE('filename')
```

If you want all incoming mail sent to a centralized hub, as for a shared /var/spool/mail scheme, use

```
define('MAIL_HUB', 'mailer:hostname')
```

Again, ``mailer:'' defaults to "relay". If you define both LOCAL_RELAY and MAIL_HUB _AND_ you have FEATURE('stickyhost'), unqualified names will be sent to the LOCAL_RELAY and other local names will be sent to MAIL_HUB. Note: there is a (long standing) bug which keeps this combination from working for addresses of the form user+detail. Names in class {L} will be delivered locally, so you MUST have aliases or .forward files for them.

For example, if you are on machine mastodon.CS.Berkeley.EDU and you have FEATURE('stickyhost'), the following combinations of settings will have the indicated effects:

email sent to....	eric	eric@mastodon.CS.Berkeley.EDU
LOCAL_RELAY set to	mail.CS.Berkeley.EDU	(delivered locally)
mail.CS.Berkeley.EDU	(no local aliasing)	(aliasing done)
MAIL_HUB set to	mammoth.CS.Berkeley.EDU	mammoth.CS.Berkeley.EDU
mammoth.CS.Berkeley.EDU	(aliasing done)	(aliasing done)
Both LOCAL_RELAY and	mail.CS.Berkeley.EDU	mammoth.CS.Berkeley.EDU
MAIL_HUB set as above	(no local aliasing)	(aliasing done)

If you do not have `FEATURE('stickyhost')` set, then `LOCAL_RELAY` and `MAIL_HUB` act identically, with `MAIL_HUB` taking precedence.

If you want all outgoing mail to go to a central relay site, define `SMART_HOST` as well. Briefly:

```
LOCAL_RELAY applies to unqualified names (e.g., "eric").
MAIL_HUB applies to names qualified with the name of the
    local host (e.g., "eric@mastodon.CS.Berkeley.EDU").
SMART_HOST applies to names qualified with other hosts or
    bracketed addresses (e.g., "eric@mastodon.CS.Berkeley.EDU"
    or "eric@[127.0.0.1]").
```

However, beware that other relays (e.g., `UUCP_RELAY`, `BITNET_RELAY`, `DECNET_RELAY`, and `FAX_RELAY`) take precedence over `SMART_HOST`, so if you really want absolutely everything to go to a single central site you will need to unset all the other relays -- or better yet, find or build a minimal config file that does this.

For duplicate suppression to work properly, the host name is best specified with a terminal dot:

```
define('MAIL_HUB', 'host.domain.')
    note the trailing dot ---^
```

```
+-----+
| USING LDAP FOR ALIASES, MAPS, AND CLASSES |
+-----+
```

LDAP can be used for aliases, maps, and classes by either specifying your own LDAP map specification or using the built-in default LDAP map specification. The built-in default specifications all provide lookups which match against either the machine's fully qualified hostname (`${j}`) or a "cluster". The cluster allows you to share LDAP entries among a large number of machines without having to enter each of the machine names into each LDAP entry. To set the LDAP cluster name to use for a particular machine or set of machines, set the `confLDAP_CLUSTER` m4 variable to a unique name. For example:

```
define('confLDAP_CLUSTER', 'Servers')
```

Here, the word 'Servers' will be the cluster name. As an example, assume that `smtp.sendmail.org`, `etrn.sendmail.org`, and `mx.sendmail.org` all belong to the Servers cluster.

Some of the LDAP LDIF examples below show use of the Servers cluster. Every entry must have either a `sendmailMTAHost` or `sendmailMTACluster` attribute or it will be ignored. Be careful as mixing clusters and individual host records can have surprising results (see the CAUTION sections below).

See the file `cf/sendmail.schema` for the actual LDAP schemas. Note that this schema (and therefore the lookups and examples below) is experimental at this point as it has had little public review. Therefore, it may change in future versions. Feedback via `sendmail@sendmail.org` is encouraged.

```
-----
Aliases
-----
```

The ALIAS_FILE (O AliasFile) option can be set to use LDAP for alias lookups. To use the default schema, simply use:

```
define('ALIAS_FILE', 'ldap:')
```

By doing so, you will use the default schema which expands to a map declared as follows:

```
ldap -k (&(objectClass=sendmailMTAAliasObject)
        (sendmailMTAAliasGrouping=aliases)
        (|(sendmailMTACluster=${sendmailMTACluster})
          (sendmailMTAHost=$j))
        (sendmailMTAKey=%0))
-v sendmailMTAAliasValue
```

NOTE: The macros shown above `${sendmailMTACluster}` and `$j` are not actually used when the binary expands the `'ldap:'` token as the AliasFile option is not actually macro-expanded when read from the sendmail.cf file.

Example LDAP LDIF entries might be:

```
dn: sendmailMTAKey=sendmail-list, dc=sendmail, dc=org
objectClass: sendmailMTA
objectClass: sendmailMTAAlias
objectClass: sendmailMTAAliasObject
sendmailMTAAliasGrouping: aliases
sendmailMTAHost: etrn.sendmail.org
sendmailMTAKey: sendmail-list
sendmailMTAAliasValue: ca@example.org
sendmailMTAAliasValue: eric
sendmailMTAAliasValue: gshapiro@example.com

dn: sendmailMTAKey=owner-sendmail-list, dc=sendmail, dc=org
objectClass: sendmailMTA
objectClass: sendmailMTAAlias
objectClass: sendmailMTAAliasObject
sendmailMTAAliasGrouping: aliases
sendmailMTAHost: etrn.sendmail.org
sendmailMTAKey: owner-sendmail-list
sendmailMTAAliasValue: eric

dn: sendmailMTAKey=postmaster, dc=sendmail, dc=org
objectClass: sendmailMTA
objectClass: sendmailMTAAlias
objectClass: sendmailMTAAliasObject
sendmailMTAAliasGrouping: aliases
sendmailMTACluster: Servers
sendmailMTAKey: postmaster
sendmailMTAAliasValue: eric
```

Here, the aliases sendmail-list and owner-sendmail-list will be available only on etrn.sendmail.org but the postmaster alias will be available on every machine in the Servers cluster (including etrn.sendmail.org).

CAUTION: aliases are additive so that entries like these:

```
dn: sendmailMTAKey=bob, dc=sendmail, dc=org
objectClass: sendmailMTA
objectClass: sendmailMTAAlias
```

```

objectClass: sendmailMTAAliasObject
sendmailMTAAliasGrouping: aliases
sendmailMTACluster: Servers
sendmailMTAKey: bob
sendmailMTAAliasValue: eric

dn: sendmailMTAKey=bob, dc=sendmail, dc=org
objectClass: sendmailMTA
objectClass: sendmailMTAAlias
objectClass: sendmailMTAAliasObject
sendmailMTAAliasGrouping: aliases
sendmailMTAHost: etrn.sendmail.org
sendmailMTAKey: bob
sendmailMTAAliasValue: gshapiro

```

would mean that on all of the hosts in the cluster, mail to bob would go to eric EXCEPT on etrn.sendmail.org in which case it would go to BOTH eric and gshapiro.

If you prefer not to use the default LDAP schema for your aliases, you can specify the map parameters when setting ALIAS_FILE. For example:

```
define('ALIAS_FILE', 'ldap:-k (&(objectClass=mailGroup)(mail=%0)) -v mgrpRFC822MailMen
```

```

----
Maps
----

```

FEATURE()'s which take an optional map definition argument (e.g., access, mailertable, virtusertable, etc.) can instead take the special keyword 'LDAP', e.g.:

```

FEATURE('access_db', 'LDAP')
FEATURE('virtusertable', 'LDAP')

```

When this keyword is given, that map will use LDAP lookups consisting of the objectClass sendmailMTAClassObject, the attribute sendmailMTAMapName with the map name, a search attribute of sendmailMTAKey, and the value attribute sendmailMTAMapValue.

The values for sendmailMTAMapName are:

FEATURE()	sendmailMTAMapName
-----	-----
access_db	access
authinfo	authinfo
bitdomain	bitdomain
domaintable	domain
genericstable	generics
mailertable	mailer
uucpdomain	uucpdomain
virtusertable	virtuser

For example, FEATURE('mailertable', 'LDAP') would use the map definition:

```

Kmailertable ldap -k (&(objectClass=sendmailMTAMapObject)
                      (sendmailMTAMapName=mailer)
                      (|(sendmailMTACluster=${sendmailMTACluster})
                      (sendmailMTAHost=${j}))
                      (sendmailMTAKey=%0))

```

```
-l -v sendmailMTAMapValue
```

An example LDAP LDIF entry using this map might be:

```
dn: sendmailMTAMapName=mailer, dc=sendmail, dc=org
objectClass: sendmailMTA
objectClass: sendmailMTAMap
sendmailMTACluster: Servers
sendmailMTAMapName: mailer

dn: sendmailMTAKey=example.com, sendmailMTAMapName=mailer, dc=sendmail, dc=org
objectClass: sendmailMTA
objectClass: sendmailMTAMap
objectClass: sendmailMTAMapObject
sendmailMTAMapName: mailer
sendmailMTACluster: Servers
sendmailMTAKey: example.com
sendmailMTAMapValue: relay:[smtp.example.com]
```

CAUTION: If your LDAP database contains the record above and **ALSO** a host specific record such as:

```
dn: sendmailMTAKey=example.com@etrn, sendmailMTAMapName=mailer, dc=sendmail, dc=org
objectClass: sendmailMTA
objectClass: sendmailMTAMap
objectClass: sendmailMTAMapObject
sendmailMTAMapName: mailer
sendmailMTAHost: etrn.sendmail.org
sendmailMTAKey: example.com
sendmailMTAMapValue: relay:[mx.example.com]
```

then these entries will give unexpected results. When the lookup is done on etrn.sendmail.org, the effect is that there is **NO** match at all as maps require a single match. Since the host etrn.sendmail.org is also in the Servers cluster, LDAP would return two answers for the example.com map key in which case sendmail would treat this as no match at all.

If you prefer not to use the default LDAP schema for your maps, you can specify the map parameters when using the FEATURE(). For example:

```
FEATURE('access_db', 'ldap:-l -k (&(objectClass=mapDatabase)(key=%0)) -v value')
```

```
-----
Classes
-----
```

Normally, classes can be filled via files or programs. As of 8.12, they can also be filled via map lookups using a new syntax:

```
F{ClassName}mapkey@mapclass:mapspec
```

mapkey is optional and if not provided the map key will be empty. This can be used with LDAP to read classes from LDAP. Note that the lookup is only done when sendmail is initially started. Use the special value '@LDAP' to use the default LDAP schema. For example:

```
RELAY_DOMAIN_FILE('@LDAP')
```

would put all of the attribute sendmailMTAClassValue values of LDAP records with objectClass sendmailMTAClass and an attribute sendmailMTAClassName of

'R' into class $\$=\{R\}$. In other words, it is equivalent to the LDAP map specification:

```
F{R}@ldap:-k (&(objectClass=sendmailMTAClass)
              (sendmailMTAClassName=R)
              (|(sendmailMTACluster=${sendmailMTACluster})
                (sendmailMTAHost=$j)))
-v sendmailMTAClassValue
```

NOTE: The macros shown above $\${sendmailMTACluster}$ and $\$j$ are not actually used when the binary expands the '@LDAP' token as class declarations are not actually macro-expanded when read from the sendmail.cf file.

This can be used with class related commands such as RELAY_DOMAIN_FILE(), MASQUERADE_DOMAIN_FILE(), etc:

Command	sendmailMTAClassName
-----	-----
CANONIFY_DOMAIN_FILE()	Canonify
EXPOSED_USER_FILE()	E
GENERIC_DOMAIN_FILE()	G
LDAPROUTE_DOMAIN_FILE()	LDAPRoute
LDAPROUTE_EQUIVALENT_FILE()	LDAPRouteEquiv
LOCAL_USER_FILE()	L
MASQUERADE_DOMAIN_FILE()	M
MASQUERADE_EXCEPTION_FILE()	N
RELAY_DOMAIN_FILE()	R
VIRTUSER_DOMAIN_FILE()	VirtHost

You can also add your own as any 'F'ile class of the form:

```
F{ClassName}@LDAP
^^^^^^^^^^
```

will use "ClassName" for the sendmailMTAClassName.

An example LDAP LDIF entry would look like:

```
dn: sendmailMTAClassName=R, dc=sendmail, dc=org
objectClass: sendmailMTA
objectClass: sendmailMTAClass
sendmailMTACluster: Servers
sendmailMTAClassName: R
sendmailMTAClassValue: sendmail.org
sendmailMTAClassValue: example.com
sendmailMTAClassValue: 10.56.23
```

CAUTION: If your LDAP database contains the record above and *ALSO* a host specific record such as:

```
dn: sendmailMTAClassName=R@etrn.sendmail.org, dc=sendmail, dc=org
objectClass: sendmailMTA
objectClass: sendmailMTAClass
sendmailMTAHost: etrn.sendmail.org
sendmailMTAClassName: R
sendmailMTAClassValue: example.com
```

the result will be similar to the aliases caution above. When the lookup is done on etrn.sendmail.org, $\$=\{R\}$ would contain all of the entries (from both the cluster match and the host match). In other words, the effective is additive.

If you prefer not to use the default LDAP schema for your classes, you can specify the map parameters when using the class command. For example:

```
VIRTUSER_DOMAIN_FILE('@ldap:-k (&(objectClass=virtHosts)(host=*)) -v host')
```

Remember, macros can not be used in a class declaration as the binary does not expand them.

```
+-----+
| LDAP ROUTING |
+-----+
```

FEATURE('ldap_routing') can be used to implement the IETF Internet Draft LDAP Schema for Intranet Mail Routing (draft-lachman-laser-ldap-mail-routing-01). This feature enables LDAP-based rerouting of a particular address to either a different host or a different address. The LDAP lookup is first attempted on the full address (e.g., user@example.com) and then on the domain portion (e.g., @example.com). Be sure to setup your domain for LDAP routing using LDAPROUTE_DOMAIN(), e.g.:

```
LDAPROUTE_DOMAIN('example.com')
```

Additionally, you can specify equivalent domains for LDAP routing using LDAPROUTE_EQUIVALENT() and LDAPROUTE_EQUIVALENT_FILE(). 'Equivalent' hostnames are mapped to \$M (the masqueraded hostname for the server) before the LDAP query. For example, if the mail is addressed to user@host1.example.com, normally the LDAP lookup would only be done for 'user@host1.example.com' and '@host1.example.com'. However, if LDAPROUTE_EQUIVALENT('host1.example.com') is used, the lookups would also be done on 'user@example.com' and '@example.com' after attempting the host1.example.com lookups.

By default, the feature will use the schemas as specified in the draft and will not reject addresses not found by the LDAP lookup. However, this behavior can be changed by giving additional arguments to the FEATURE() command:

```
FEATURE('ldap_routing', <mailHost>, <mailRoutingAddress>, <bounce>, <detail>)
```

where <mailHost> is a map definition describing how to lookup an alternative mail host for a particular address; <mailRoutingAddress> is a map definition describing how to lookup an alternative address for a particular address; the <bounce> argument, if present and not the word "passthru", dictates that mail should be bounced if neither a mailHost nor mailRoutingAddress is found; and <detail> indicates what actions to take if the address contains +detail information -- 'strip' tries the lookup with the +detail and if no matches are found, strips the +detail and tries the lookup again; 'preserve', does the same as 'strip' but if a mailRoutingAddress match is found, the +detail information is copied to the new address.

The default <mailHost> map definition is:

```
ldap -l -v mailHost -k (&(objectClass=inetLocalMailRecipient)
                        (mailLocalAddress=%0))
```

The default <mailRoutingAddress> map definition is:

```
ldap -l -v mailRoutingAddress -k (&(objectClass=inetLocalMailRecipient)
                                   (mailLocalAddress=%0))
```

Note that neither includes the LDAP server hostname (-h server) or base DN (-b o=org,c=COUNTRY), both necessary for LDAP queries. It is presumed that your .mc file contains a setting for the confLDAP_DEFAULT_SPEC option with these settings. If this is not the case, the map definitions should be changed as described above.

The following possibilities exist as a result of an LDAP lookup on an address:

mailHost is	mailRoutingAddress is	Results in
-----	-----	-----
set to a "local" host	set	mail delivered to mailRoutingAddress
set to a "local" host	not set	delivered to original address
set to a remote host	set	mailRoutingAddress relayed to mailHost
set to a remote host	not set	original address relayed to mailHost
not set	set	mail delivered to mailRoutingAddress
not set	not set	delivered to original address *OR* bounced as unknown user

The term "local" host above means the host specified is in class {w}. If the result would mean sending the mail to a different host, that host is looked up in the mailertable before delivery.

Note that the last case depends on whether the third argument is given to the FEATURE() command. The default is to deliver the message to the original address.

The LDAP entries should be set up with an objectClass of inetLocalMailRecipient and the address be listed in a mailLocalAddress attribute. If present, there must be only one mailHost attribute and it must contain a fully qualified host name as its value. Similarly, if present, there must be only one mailRoutingAddress attribute and it must contain an RFC 822 compliant address. Some example LDAP records (in LDIF format):

```
dn: uid=tom, o=example.com, c=US
objectClass: inetLocalMailRecipient
mailLocalAddress: tom@example.com
mailRoutingAddress: thomas@mailhost.example.com
```

This would deliver mail for tom@example.com to thomas@mailhost.example.com.

```
dn: uid=dick, o=example.com, c=US
objectClass: inetLocalMailRecipient
mailLocalAddress: dick@example.com
mailHost: eng.example.com
```

This would relay mail for dick@example.com to the same address but redirect the mail to MX records listed for the host eng.example.com (unless the mailertable overrides).

```
dn: uid=harry, o=example.com, c=US
objectClass: inetLocalMailRecipient
mailLocalAddress: harry@example.com
mailHost: mktmail.example.com
mailRoutingAddress: harry@mkt.example.com
```

This would relay mail for harry@example.com to the MX records listed for the host mktmail.example.com using the new address harry@mkt.example.com when talking to that host.

```
dn: uid=virtual.example.com, o=example.com, c=US
objectClass: inetLocalMailRecipient
mailLocalAddress: @virtual.example.com
mailHost: server.example.com
mailRoutingAddress: virtual@example.com
```

This would send all mail destined for any username @virtual.example.com to the machine server.example.com's MX servers and deliver to the address virtual@example.com on that relay machine.

```
+-----+
| ANTI-SPAM CONFIGURATION CONTROL |
+-----+
```

The primary anti-spam features available in sendmail are:

- * Relaying is denied by default.
- * Better checking on sender information.
- * Access database.
- * Header checks.

Relaying (transmission of messages from a site outside your host (class {w}) to another site except yours) is denied by default. Note that this changed in sendmail 8.9; previous versions allowed relaying by default. If you really want to revert to the old behaviour, you will need to use FEATURE('promiscuous_relay'). You can allow certain domains to relay through your server by adding their domain name or IP address to class {R} using RELAY_DOMAIN() and RELAY_DOMAIN_FILE() or via the access database (described below). Note that IPv6 addresses must be prefaced with "IPv6:". The file consists (like any other file based class) of entries listed on separate lines, e.g.,

```
sendmail.org
128.32
IPv6:2002:c0a8:02c7
IPv6:2002:c0a8:51d2::23f4
host.mydomain.com
[UNIX:localhost]
```

Notice: the last entry allows relaying for connections via a UNIX socket to the MTA/MSP. This might be necessary if your configuration doesn't allow relaying by other means in that case, e.g., by having localhost.\$m in class {R} (make sure \$m is not just a top level domain).

If you use

```
FEATURE('relay_entire_domain')
```

then any host in any of your local domains (that is, class {m}) will be relayed (that is, you will accept mail either to or from any host in your domain).

You can also allow relaying based on the MX records of the host portion of an incoming recipient address by using

```
FEATURE('relay_based_on_MX')
```

For example, if your server receives a recipient of user@domain.com and domain.com lists your server in its MX records, the mail will be accepted for relay to domain.com. This feature may cause problems if MX lookups for the recipient domain are slow or time out. In that case, mail will be temporarily rejected. It is usually better to maintain a list of hosts/domains for which the server acts as relay. Note also that this feature will stop spammers from using your host to relay spam but it will not stop outsiders from using your server as a relay for their site (that is, they set up an MX record pointing to your mail server, and you will relay mail addressed to them without any prior arrangement). Along the same lines,

```
FEATURE('relay_local_from')
```

will allow relaying if the sender specifies a return path (i.e. MAIL FROM: <user@domain>) domain which is a local domain. This is a dangerous feature as it will allow spammers to spam using your mail server by simply specifying a return address of user@your.domain.com. It should not be used unless absolutely necessary. A slightly better solution is

```
FEATURE('relay_mail_from')
```

which allows relaying if the mail sender is listed as RELAY in the access map. If an optional argument 'domain' is given, the domain portion of the mail sender is also checked to allowing relaying. This option only works together with the tag From: for the LHS of the access map entries (see below: Finer control...). This feature allows spammers to abuse your mail server by specifying a return address that you enabled in your access file. This may be harder to figure out for spammers, but it should not be used unless necessary. Instead use SMTP AUTH or STARTTLS to allow relaying for roaming users.

If source routing is used in the recipient address (e.g., RCPT TO: <user%site.com@othersite.com>), sendmail will check user@site.com for relaying if othersite.com is an allowed relay host in either class {R}, class {m} if FEATURE('relay_entire_domain') is used, or the access database if FEATURE('access_db') is used. To prevent the address from being stripped down, use:

```
FEATURE('loose_relay_check')
```

If you think you need to use this feature, you probably do not. This should only be used for sites which have no control over the addresses

that they provide a gateway for. Use this FEATURE with caution as it can allow spammers to relay through your server if not setup properly.

NOTICE: It is possible to relay mail through a system which the anti-relay rules do not prevent: the case of a system that does use FEATURE('nouucp', 'nospecial') (system A) and relays local messages to a mail hub (e.g., via LOCAL_RELAY or LUSER_RELAY) (system B). If system B doesn't use FEATURE('nouucp') at all, addresses of the form <example.net!user@local.host> would be relayed to <user@example.net>. System A doesn't recognize '!' as an address separator and therefore forwards it to the mail hub which in turns relays it because it came from a trusted local host. So if a mailserver allows UUCP (bang-format) addresses, all systems from which it allows relaying should do the same or reject those addresses.

As of 8.9, sendmail will refuse mail if the MAIL FROM: parameter has an unresolvable domain (i.e., one that DNS, your local name service, or special case rules in ruleset 3 cannot locate). This also applies to addresses that use domain literals, e.g., <user@[1.2.3.4]>, if the IP address can't be mapped to a host name. If you want to continue to accept such domains, e.g., because you are inside a firewall that has only a limited view of the Internet host name space (note that you will not be able to return mail to them unless you have some "smart host" forwarder), use

```
FEATURE('accept_unresolvable_domains')
```

Alternatively, you can allow specific addresses by adding them to the access map, e.g.,

```
From:unresolvable.domain      OK
From:[1.2.3.4]                OK
From:[1.2.4]                  OK
```

Notice: domains which are temporarily unresolvable are (temporarily) rejected with a 451 reply code. If those domains should be accepted (which is discouraged) then you can use

```
LOCAL_CONFIG
C{ResOk}TEMP
```

sendmail will also refuse mail if the MAIL FROM: parameter is not fully qualified (i.e., contains a domain as well as a user). If you want to continue to accept such senders, use

```
FEATURE('accept_unqualified_senders')
```

Setting the DaemonPortOptions modifier 'u' overrides the default behavior, i.e., unqualified addresses are accepted even without this FEATURE. If this FEATURE is not used, the DaemonPortOptions modifier 'f' can be used to enforce fully qualified domain names.

An ''access'' database can be created to accept or reject mail from selected domains. For example, you may choose to reject all mail originating from known spammers. To enable such a database, use

```
FEATURE('access_db')
```

Notice: the access database is applied to the envelope addresses and the connection information, not to the header.

The FEATURE macro can accept as second parameter the key file definition for the database; for example

```
FEATURE('access_db', 'hash -T<TMPF> /etc/mail/access_map')
```

Notice: If a second argument is specified it must contain the option '-T<TMPF>' as shown above. The optional third and fourth parameters may be 'skip' or 'lookupdotdomain'. The former enables SKIP as value part (see below), the latter is another way to enable the feature of the same name (see above).

Remember, since /etc/mail/access is a database, after creating the text file as described below, you must use makemap to create the database map. For example:

```
makemap hash /etc/mail/access < /etc/mail/access
```

The table itself uses e-mail addresses, domain names, and network numbers as keys. Note that IPv6 addresses must be prefaced with "IPv6:". For example,

spammer@aol.com	REJECT
cyberspammer.com	REJECT
192.168.212	REJECT
IPv6:2002:c0a8:02c7	RELAY
IPv6:2002:c0a8:51d2::23f4	REJECT

would refuse mail from spammer@aol.com, any user from cyberspammer.com (or any host within the cyberspammer.com domain), any host on the 192.168.212.* network, and the IPv6 address 2002:c0a8:51d2::23f4. It would allow relay for the IPv6 network 2002:c0a8:02c7::/48.

The value part of the map can contain:

OK	Accept mail even if other rules in the running ruleset would reject it, for example, if the domain name is unresolvable. "Accept" does not mean "relay", but at most acceptance for local recipients. That is, OK allows less than RELAY.
RELAY	Accept mail addressed to the indicated domain or received from the indicated domain for relaying through your SMTP server. RELAY also serves as an implicit OK for the other checks.
REJECT	Reject the sender or recipient with a general purpose message.
DISCARD	Discard the message completely using the \$#discard mailer. If it is used in check_compat, it affects only the designated recipient, not the whole message as it does in all other cases. This should only be used if really necessary.
SKIP	This can only be used for host/domain names and IP addresses/nets. It will abort the current search for this entry without accepting or rejecting it but causing the default action.
### any text	where ### is an RFC 821 compliant error code and "any text" is a message to return for the command. The string should be quoted to avoid surprises, e.g., sendmail may remove spaces otherwise. This type is deprecated, use one the two

```
ERROR: entries below instead.
ERROR:### any text
as above, but useful to mark error messages as such.
ERROR:D.S.N:### any text
where D.S.N is an RFC 1893 compliant error code
and the rest as above.
```

For example:

```
cyberspammer.com      ERROR:550 "We don't accept mail from spammers"
okay.cyberspammer.com OK
sendmail.org          RELAY
128.32                 RELAY
IPv6:1:2:3:4:5:6:7    RELAY
[127.0.0.3]           OK
[IPv6:1:2:3:4:5:6:7:8] OK
```

would accept mail from okay.cyberspammer.com, but would reject mail from all other hosts at cyberspammer.com with the indicated message. It would allow relaying mail from and to any hosts in the sendmail.org domain, and allow relaying from the 128.32.*.* network and the IPv6 1:2:3:4:5:6:7:* network. The latter two entries are for checks against `${client_name}` if the IP address doesn't resolve to a hostname (or is considered as "may be forged"). That is, using square brackets means these are host names, not network numbers.

Warning: if you change the RFC 821 compliant error code from the default value of 550, then you should probably also change the RFC 1893 compliant error code to match it. For example, if you use

```
user@example.com      ERROR:450 mailbox full
```

the error returned would be "450 5.0.0 mailbox full" which is wrong. Use "ERROR:4.2.2:450 mailbox full" instead.

Note, UUCP users may need to add hostname.UUCP to the access database or class {R}.

If you also use:

```
FEATURE('relay_hosts_only')
```

then the above example will allow relaying for sendmail.org, but not hosts within the sendmail.org domain. Note that this will also require hosts listed in class {R} to be fully qualified host names.

You can also use the access database to block sender addresses based on the username portion of the address. For example:

```
FREE.STEALTH.MAILER@  ERROR:550 Spam not accepted
```

Note that you must include the @ after the username to signify that this database entry is for checking only the username portion of the sender address.

If you use:

```
FEATURE('blacklist_recipients')
```

then you can add entries to the map for local users, hosts in your

domains, or addresses in your domain which should not receive mail:

```
badlocaluser@          ERROR:550 Mailbox disabled for this username
host.mydomain.com      ERROR:550 That host does not accept mail
user@otherhost.mydomain.com ERROR:550 Mailbox disabled for this recipient
```

This would prevent a recipient of badlocaluser@mydomain.com, any user at host.mydomain.com, and the single address user@otherhost.mydomain.com from receiving mail. Please note: a local username must be now tagged with an @ (this is consistent with the check of the sender address, and hence it is possible to distinguish between hostnames and usernames). Enabling this feature will keep you from sending mails to all addresses that have an error message or REJECT as value part in the access map. Taking the example from above:

```
spammer@aol.com        REJECT
cyberspammer.com       REJECT
```

Mail can't be sent to spammer@aol.com or anyone at cyberspammer.com.

There are several DNS based blacklists, the first of which was the RBL ('Realtime Blackhole List') run by the MAPS project, see <http://mail-abuse.org/>. These are databases of spammers maintained in DNS. To use such a database, specify

```
FEATURE('dnsbl')
```

This will cause sendmail to reject mail from any site in the original Realtime Blackhole List database. This default DNS blacklist, blackholes.mail-abuse.org, is a service offered by the Mail Abuse Prevention System (MAPS). As of July 31, 2001, MAPS is a subscription service, so using that network address won't work if you haven't subscribed. Contact MAPS to subscribe (<http://mail-abuse.org/>).

You can specify an alternative RBL server to check by specifying an argument to the FEATURE. The default error message is

```
Mail from IP-ADDRESS refused by blackhole site SERVER
```

where IP-ADDRESS and SERVER are replaced by the appropriate information. A second argument can be used to specify a different text. By default, temporary lookup failures are ignored and hence cause the connection not to be rejected by the DNS based rejection list. This behavior can be changed by specifying a third argument, which must be either 't' or a full error message. For example:

```
FEATURE('dnsbl', 'dnsbl.example.com', '',
  '"451 Temporary lookup failure for " ${client_addr} " in dnsbl.example.com"')
```

If 't' is used, the error message is:

```
451 Temporary lookup failure of IP-ADDRESS at SERVER
```

where IP-ADDRESS and SERVER are replaced by the appropriate information.

This FEATURE can be included several times to query different DNS based rejection lists, e.g., the dial-up user list (see <http://mail-abuse.org/dul/>).

Notice: to avoid checking your own local domains against those blacklists, use the access_db feature and add:

```
Connect:10.1          OK
Connect:127.0.0.1     RELAY
```

to the access map, where 10.1 is your local network. You may want to use "RELAY" instead of "OK" to allow also relaying instead of just disabling the DNS lookups in the backlists.

The features described above make use of the check_relay, check_mail, and check_rcpt rulesets. If you wish to include your own checks, you can put your checks in the rulesets Local_check_relay, Local_check_mail, and Local_check_rcpt. For example if you wanted to block senders with all numeric usernames (i.e. 2312343@bigisp.com), you would use Local_check_mail and the regex map:

```
LOCAL_CONFIG
Kallnumbers regex -a@MATCH ^[0-9]+$

LOCAL_RULESETS
SLocal_check_mail
# check address against various regex checks
R$*          $: $>Parse0 $>3 $1
R$+ < @ bigisp.com. > $*    $: $(allnumbers $1 $)
R@MATCH      $#error $: 553 Header Error
```

These rules are called with the original arguments of the corresponding check_* ruleset. If the local ruleset returns \$#OK, no further checking is done by the features described above and the mail is accepted. If the local ruleset resolves to a mailer (such as \$#error or \$#discard), the appropriate action is taken. Otherwise, the results of the local rewriting are ignored.

Finer control by using tags for the LHS of the access map

Read this section only if the options listed so far are not sufficient for your purposes. There is now the option to tag entries in the access map according to their type. Three tags are available:

```
Connect:      connection information (${client_addr}, ${client_name})
From:         envelope sender
To:           envelope recipient
```

If the required item is looked up in a map, it will be tried first with the corresponding tag in front, then (as fallback to enable backward compatibility) without any tag, unless the specific feature requires a tag. For example,

```
From:spammer@some.dom    REJECT
To:friend.domain         RELAY
Connect:friend.domain    OK
Connect:from.domain      RELAY
From:good@another.dom    OK
From:another.dom         REJECT
```

This would deny mails from spammer@some.dom but you could still

send mail to that address even if FEATURE('blacklist_recipients') is enabled. Your system will allow relaying to friend.domain, but not from it (unless enabled by other means). Connections from that domain will be allowed even if it ends up in one of the DNS based rejection lists. Relaying is enabled from from.domain but not to it (since relaying is based on the connection information for outgoing relaying, the tag Connect: must be used; for incoming relaying, which is based on the recipient address, To: must be used). The last two entries allow mails from good@another.dom but reject mail from all other addresses with another.dom as domain part.

Delay all checks

By using FEATURE('delay_checks') the rulesets check_mail and check_relay will not be called when a client connects or issues a MAIL command, respectively. Instead, those rulesets will be called by the check_rcpt ruleset; they will be skipped if a sender has been authenticated using a "trusted" mechanism, i.e., one that is defined via TRUST_AUTH_MECH(). If check_mail returns an error then the RCPT TO command will be rejected with that error. If it returns some other result starting with \$# then check_relay will be skipped. If the sender address (or a part of it) is listed in the access map and it has a RHS of OK or RELAY, then check_relay will be skipped. This has an interesting side effect: if your domain is my.domain and you have

```
my.domain      RELAY
```

in the access map, then all e-mail with a sender address of <user@my.domain> gets through, even if check_relay would reject it (e.g., based on the hostname or IP address). This allows spammers to get around DNS based blacklist by faking the sender address. To avoid this problem you have to use tagged entries:

```
To:my.domain      RELAY
Connect:my.domain  RELAY
```

if you need those entries at all (class {R} may take care of them).

FEATURE('delay_checks') can take an optional argument:

```
FEATURE('delay_checks', 'friend')
    enables spamfriend test
FEATURE('delay_checks', 'hater')
    enables spamhater test
```

If such an argument is given, the recipient will be looked up in the access map (using the tag Spam:). If the argument is 'friend', then the other rulesets will be skipped if the recipient address is found and has RHS friend. If the argument is 'hater', then the other rulesets will be applied if the recipient address is found and has RHS hater.

This allows for simple exceptions from the tests, e.g., by activating the friend option and having

```
Spam:abuse@      FRIEND
```

in the access map, mail to abuse@localdomain will get through. It is also possible to specify a full address or an address with +detail:

```
Spam:abuse@my.domain      FRIEND
Spam:me+abuse@            FRIEND
Spam:spam.domain         FRIEND
```

Note: The required tag has been changed in 8.12 from To: to Spam:. This change is incompatible to previous versions. However, you can (for now) simply add the new entries to the access map, the old ones will be ignored. As soon as you removed the old entries from the access map, specify a third parameter ('n') to this feature and the backward compatibility rules will not be in the generated .cf file.

Header Checks

You can also reject mail on the basis of the contents of headers. This is done by adding a ruleset call to the 'H' header definition command in sendmail.cf. For example, this can be used to check the validity of a Message-ID: header:

```
LOCAL_RULESETS
HMessage-Id: $>CheckMessageId

SCheckMessageId
R< $+ @ $+ >          $@ OK
R$*                  $#error $: 553 Header Error
```

The alternative format:

```
HSubject: $>+CheckSubject
```

that is, \$>+ instead of \$>, gives the full Subject: header including comments to the ruleset (comments in parentheses () are stripped by default).

A default ruleset for headers which don't have a specific ruleset defined for them can be given by:

```
H*: $>CheckHdr
```

Notice:

1. All rules act on tokens as explained in doc/op/op.{me,ps,txt}. That may cause problems with simple header checks due to the tokenization. It might be simpler to use a regex map and apply it to \${currHeader}.
2. There are no default rulesets coming with this distribution of sendmail. You can either write your own or you can search the WWW for examples, e.g., http://www.digitalanswers.org/check_local/

After all of the headers are read, the check_eoh ruleset will be called for any final header-related checks. The ruleset is called with the number of headers and the size of all of the headers in bytes separated by \$|. One example usage is to reject messages which do not have a Message-Id: header. However, the Message-Id: header is *NOT* a required header and is not a guaranteed spam indicator. This ruleset is an example and should probably not be used in production.

```
LOCAL_CONFIG
Kstorage macro
```

```

LOCAL_RULESETS
HMessage-Id: $>CheckMessageId

SCheckMessageId
# Record the presence of the header
R$*                               $: $(storage {MessageIdCheck} $@ OK $) $1
R< $+ @ $+ >                     $@ OK
R$*                               $#error $: 553 Header Error

Scheck_eoh
# Check the macro
R$*                               $: < ${MessageIdCheck} >
# Clear the macro for the next message
R$*                               $: $(storage {MessageIdCheck} $) $1
# Has a Message-Id: header
R< $+ >                           $@ OK
# Allow missing Message-Id: from local mail
R$*                               $: < ${client_name} >
R< >                             $@ OK
R< $=w >                         $@ OK
# Otherwise, reject the mail
R$*                               $#error $: 553 Header Error

```

```

+-----+
| STARTTLS |
+-----+

```

In this text, cert will be used as an abbreviation for X.509 certificate, DN (CN) is the distinguished (common) name of a cert, and CA is a certification authority, which signs (issues) certs.

For STARTTLS to be offered by sendmail you need to set at least this variables (the file names and paths are just examples):

```

define('confCACERT_PATH', '/etc/mail/certs/')
define('confCACERT', '/etc/mail/certs/CA.cert.pem')
define('confSERVER_CERT', '/etc/mail/certs/my.cert.pem')
define('confSERVER_KEY', '/etc/mail/certs/my.key.pem')

```

On systems which do not have the compile flag HASURANDOM set (see sendmail/README) you also must set confRAND_FILE.

See doc/op/op.{me,ps,txt} for more information about these options, especially the sections ``Certificates for STARTTLS`` and ``PRNG for STARTTLS``.

Macros related to STARTTLS are:

```

${cert_issuer} holds the DN of the CA (the cert issuer).
${cert_subject} holds the DN of the cert (called the cert subject).
${cn_issuer} holds the CN of the CA (the cert issuer).
${cn_subject} holds the CN of the cert (called the cert subject).
${tls_version} the TLS/SSL version used for the connection, e.g., TLSv1,
                TLSv1/SSLv3, SSLv3, SSLv2.
${cipher} the cipher used for the connection, e.g., EDH-DSS-DES-CBC3-SHA,
                EDH-RSA-DES-CBC3-SHA, DES-CBC-MD5, DES-CBC3-SHA.
${cipher_bits} the keylength (in bits) of the symmetric encryption algorithm
                used for the connection.
${verify} holds the result of the verification of the presented cert.

```

Possible values are:

OK	verification succeeded.
NO	no cert presented.
NOT	no cert requested.
FAIL	cert presented but could not be verified, e.g., the cert of the signing CA is missing.
NONE	STARTTLS has not been performed.
TEMP	temporary error occurred.
PROTOCOL	protocol error occurred (SMTP level).
SOFTWARE	STARTTLS handshake failed.

`${server_name}` the name of the server of the current outgoing SMTP connection.

`${server_addr}` the address of the server of the current outgoing SMTP connection.

Relaying

SMTP STARTTLS can allow relaying for senders who have successfully authenticated themselves. This is done in the ruleset `RelayAuth`. If the verification of the cert failed (`${verify} != OK`), relaying is subject to the usual rules. Otherwise the DN of the issuer is looked up in the access map using the tag `CERTISSUER`. If the resulting value is `RELAY`, relaying is allowed. If it is `SUBJECT`, the DN of the cert subject is looked up next in the access map using the tag `CERTSUBJECT`. If the value is `RELAY`, relaying is allowed.

To make things a bit more flexible (or complicated), the values for `${cert_issuer}` and `${cert_subject}` can be optionally modified by regular expressions defined in the m4 variables `_CERT_REGEX_ISSUER_` and `_CERT_REGEX_SUBJECT_`, respectively. To avoid problems with those macros in rulesets and map lookups, they are modified as follows: each non-printable character and the characters `'<'`, `'>'`, `'('`, `')'`, `'\"'`, `'+'` are replaced by their HEX value with a leading `'+'`. For example:

```
/C=US/ST=California/O=endmail.org/OU=private/CN=Darth Mail (Cert)/Email=
darth+cert@endmail.org
```

is encoded as:

```
/C=US/ST=California/O=endmail.org/OU=private/CN=
Darth+20Mail+20+28Cert+29/Email=darth+2Bcert@endmail.org
```

(line breaks have been inserted for readability).

Examples:

To allow relaying for everyone who can present a cert signed by

```
/C=US/ST=California/O=endmail.org/OU=private/CN=
Darth+20Mail+20+28Cert+29/Email=darth+2Bcert@endmail.org
```

simply use:

```
CERTIssuer:/C=US/ST=California/O=endmail.org/OU=private/CN=
Darth+20Mail+20+28Cert+29/Email=darth+2Bcert@endmail.org      RELAY
```

To allow relaying only for a subset of machines that have a cert signed by

```
/C=US/ST=California/O=endmail.org/OU=private/CN=
```

Darth+20Mail+20+28Cert+29/Email=darth+2Bcert@endmail.org

use:

```
CERTIssuer:/C=US/ST=California/O=endmail.org/OU=private/CN=
Darth+20Mail+20+28Cert+29/Email=darth+2Bcert@endmail.org      SUBJECT
CERTSubject:/C=US/ST=California/O=endmail.org/OU=private/CN=
DeathStar/Email=deathstar@endmail.org                        RELAY
```

Note: line breaks have been inserted after "CN=" for readability,
each tagged entry must be one (long) line in the access map.

Of course it is also possible to write a simple ruleset that allows
relaying for everyone who can present a cert that can be verified, e.g.,

```
LOCAL_RULESETS
SLocal_check_rcpt
R$*      $: ${verify}
ROK      $# OK
```

Allowing Connections

The rulesets `tls_server`, `tls_client`, and `tls_rcpt` are used to decide whether
an SMTP connection is accepted (or should continue).

`tls_server` is called when sendmail acts as client after a `STARTTLS` command
(should) have been issued. The parameter is the value of `${verify}`.

`tls_client` is called when sendmail acts as server, after a `STARTTLS` command
has been issued, and from `check_mail`. The parameter is the value of
`${verify}` and `STARTTLS` or `MAIL`, respectively.

Both rulesets behave the same. If no access map is in use, the connection
will be accepted unless `${verify}` is `SOFTWARE`, in which case the connection
is always aborted. For `tls_server/tls_client`, `${client_name}/${server_name}`
is looked up in the access map using the tag `TLS_Srv/TLS_Clt`, which is done
with the ruleset `LookUpDomain`. If no entry is found, `${client_addr}`
(`${server_addr}`) is looked up in the access map (same tag, ruleset
`LookUpAddr`). If this doesn't result in an entry either, just the tag is
looked up in the access map (included the trailing colon). Notice:
requiring that e-mail is sent to a server only encrypted, e.g., via

```
TLS_Srv:secure.domain  ENCR:112
```

doesn't necessarily mean that e-mail sent to that domain is encrypted.
If the domain has multiple MX servers, e.g.,

```
secure.domain.  IN MX 10      mail.secure.domain.
secure.domain.  IN MX 50      mail.other.domain.
```

then mail to `user@secure.domain` may go unencrypted to `mail.other.domain`.
`tls_rcpt` can be used to address this problem.

`tls_rcpt` is called before a `RCPT TO:` command is sent. The parameter is the
current recipient. This ruleset is only defined if `FEATURE('access_db')`
is selected. A recipient address `user@domain` is looked up in the access
map in four formats: `TLS_Rcpt:user@domain`, `TLS_Rcpt:user@`, `TLS_Rcpt:domain`,
and `TLS_Rcpt;`; the first match is taken.

The result of the lookups is then used to call the ruleset `TLS_connection`, which checks the requirement specified by the RHS in the access map against the actual parameters of the current TLS connection, esp. `${verify}` and `${cipher_bits}`. Legal RHSs in the access map are:

```
VERIFY          verification must have succeeded
VERIFY:bits     verification must have succeeded and ${cipher_bits} must
                be greater than or equal bits.
ENCR:bits       ${cipher_bits} must be greater than or equal bits.
```

The RHS can optionally be prefixed by `TEMP+` or `PERM+` to select a temporary or permanent error. The default is a temporary error code (403 4.7.0) unless the macro `TLS_PERM_ERR` is set during generation of the `.cf` file.

If a certain level of encryption is required, then it might also be possible that this level is provided by the security layer from a SASL algorithm, e.g., `DIGEST-MD5`.

Furthermore, there can be a list of extensions added. Such a list starts with `'+'` and the items are separated by `'++'`. Allowed extensions are:

```
CN:name         name must match ${cn_subject}
CN              ${server_name} must match ${cn_subject}
CS:name         name must match ${cert_subject}
CI:name         name must match ${cert_issuer}
```

Example: e-mail sent to `secure.example.com` should only use an encrypted connection. E-mail received from hosts within the `laptop.example.com` domain should only be accepted if they have been authenticated. The host which receives e-mail for `darth@endmail.org` must present a cert that uses the CN `smtp.endmail.org`.

```
TLS_Srv:secure.example.com    ENCR:112
TLS_Clt:laptop.example.com    PERM+VERIFY:112
TLS_Rcpt:darth@endmail.org    ENCR:112+CN:smtp.endmail.org
```

Disabling STARTTLS And Setting SMTP Server Features

By default `STARTTLS` is used whenever possible. However, there are some broken MTAs that don't properly implement `STARTTLS`. To be able to send to (or receive from) those MTAs, the ruleset `try_tls` (`srv_features`) can be used that work together with the access map. Entries for the access map must be tagged with `Try_TLS` (`Srv_Features`) and refer to the hostname or IP address of the connecting system. A default case can be specified by using just the tag. For example, the following entries in the access map:

```
Try_TLS:broken.server    NO
Srv_Features:my.domain   v
Srv_Features:             V
```

will turn off `STARTTLS` when sending to `broken.server` (or any host in that domain), and request a client certificate during the TLS handshake only for hosts in `my.domain`. The valid entries on the RHS for `Srv_Features` are listed in the Sendmail Installation and Operations Guide.

Received: Header

The Received: header reveals whether STARTTLS has been used. It contains an extra line:

```
(version=${tls_version} cipher=${cipher} bits=${cipher_bits} verify=${verify})
```

```
+-----+  
| SMTP AUTHENTICATION |  
+-----+
```

The macros `${auth_authen}`, `${auth_author}`, and `${auth_type}` can be used in anti-relay rulesets to allow relaying for those users that authenticated themselves. A very simple example is:

```
SLocal_check_rcpt  
R$*      $: ${auth_type}  
R$+      $# OK
```

which checks whether a user has successfully authenticated using any available mechanism. Depending on the setup of the CYRUS SASL library, more sophisticated rulesets might be required, e.g.,

```
SLocal_check_rcpt  
R$*      $: ${auth_type} $| ${auth_authen}  
RDIGEST-MD5 $| $+@$=w    $# OK
```

to allow relaying for users that authenticated using DIGEST-MD5 and have an identity in the local domains.

The ruleset `trust_auth` is used to determine whether a given `AUTH=` parameter (that is passed to this ruleset) should be trusted. This ruleset may make use of the other `${auth_*}` macros. Only if the ruleset resolves to the error mailer, the `AUTH=` parameter is not trusted. A user supplied ruleset `Local_trust_auth` can be written to modify the default behavior, which only trust the `AUTH=` parameter if it is identical to the authenticated user.

Per default, relaying is allowed for any user who authenticated via a "trusted" mechanism, i.e., one that is defined via `TRUST_AUTH_MECH('list of mechanisms')`
For example:
`TRUST_AUTH_MECH('KERBEROS_V4 DIGEST-MD5')`

If the selected mechanism provides a security layer the number of bits used for the key of the symmetric cipher is stored in the macro `${auth_ssf}`.

If sendmail acts as client, it needs some information how to authenticate against another MTA. This information can be provided by the ruleset `authinfo` or by the option `DefaultAuthInfo`. The `authinfo` ruleset looks up `{server_name}` using the tag `AuthInfo:` in the access map. If no entry is found, `{server_addr}` is looked up in the same way and finally just the tag `AuthInfo:` to provide default values.

Notice: the default configuration file causes the option `DefaultAuthInfo`

to fail since the ruleset authinfo is in the .cf file. If you really want to use DefaultAuthInfo (it is deprecated) then you have to remove the ruleset.

The RHS for an AuthInfo: entry in the access map should consists of a list of tokens, each of which has the form: "TDstring" (including the quotes). T is a tag which describes the item, D is a delimiter, either ':' for simple text or '=' for a base64 encoded string.

Valid values for the tag are:

U	user (authorization) id
I	authentication id
P	password
R	realm
M	list of mechanisms delimited by spaces

Example entries are:

```
AuthInfo:other.dom "U:user" "I:user" "P:secret" "R:other.dom" "M:DIGEST-MD5"
AuthInfo:more.dom "U:user" "P=c2VjcmV0"
```

User or authentication id must exist as well as the password. All other entries have default values. If one of user or authentication id is missing, the existing value is used for the missing item. If "R:" is not specified, realm defaults to \$j. The list of mechanisms defaults to those specified by AuthMechanisms.

Since this map contains sensitive information, either the access map must be unreadable by everyone but root (or the trusted user) or FEATURE('authinfo') must be used which provides a separate map. Notice: It is not checked whether the map is actually group/world-unreadable, this is left to the user.

```
+-----+
| ADDING NEW MAILERS OR RULESETS |
+-----+
```

Sometimes you may need to add entirely new mailers or rulesets. They should be introduced with the constructs MAILER_DEFINITIONS and LOCAL_RULESETS respectively. For example:

```
MAILER_DEFINITIONS
Mnymailer, ...
...

LOCAL_RULESETS
Smyruleset
...
```

Local additions for the rulesets srv_features, try_tls, tls_rcpt, tls_client, and tls_server can be made using LOCAL_SRV_FEATURES, LOCAL_TRY_TLS, LOCAL_TLS_RCPT, LOCAL_TLS_CLIENT, and LOCAL_TLS_SERVER, respectively. For example, to add a local ruleset that decides whether to try STARTTLS in a sendmail client, use:

```
LOCAL_TRY_TLS
R...
```

Note: you don't need to add a name for the ruleset, it is implicitly defined by using the appropriate macro.

```
+-----+
| ADDING NEW MAIL FILTERS |
+-----+
```

Sendmail supports mail filters to filter incoming SMTP messages according to the "Sendmail Mail Filter API" documentation. These filters can be configured in your mc file using the two commands:

```
MAIL_FILTER('name', 'equates')
INPUT_MAIL_FILTER('name', 'equates')
```

The first command, `MAIL_FILTER()`, simply defines a filter with the given name and equates. For example:

```
MAIL_FILTER('archive', 'S=local:/var/run/archivesock, F=R')
```

This creates the equivalent sendmail.cf entry:

```
Xarchive, S=local:/var/run/archivesock, F=R
```

The `INPUT_MAIL_FILTER()` command performs the same actions as `MAIL_FILTER` but also populates the m4 variable `'confINPUT_MAIL_FILTERS'` with the name of the filter such that the filter will actually be called by sendmail.

For example, the two commands:

```
INPUT_MAIL_FILTER('archive', 'S=local:/var/run/archivesock, F=R')
INPUT_MAIL_FILTER('spamcheck', 'S=inet:2525@localhost, F=T')
```

are equivalent to the three commands:

```
MAIL_FILTER('archive', 'S=local:/var/run/archivesock, F=R')
MAIL_FILTER('spamcheck', 'S=inet:2525@localhost, F=T')
define('confINPUT_MAIL_FILTERS', 'archive, spamcheck')
```

In general, `INPUT_MAIL_FILTER()` should be used unless you need to define more filters than you want to use for `'confINPUT_MAIL_FILTERS'`.

Note that setting `'confINPUT_MAIL_FILTERS'` after any `INPUT_MAIL_FILTER()` commands will clear the list created by the prior `INPUT_MAIL_FILTER()` commands.

```
+-----+
| QUEUE GROUP DEFINITIONS |
+-----+
```

In addition to the queue directory (which is the default queue group called "mqueue"), sendmail can deal with multiple queue groups, which are collections of queue directories with the same behaviour. Queue groups can be defined using the command:

```
QUEUE_GROUP('name', 'equates')
```

For details about queue groups, please see `doc/op/op.{me,ps,txt}`.

```
+-----+
| NON-SMTP BASED CONFIGURATIONS |
+-----+
```

+-----+

These configuration files are designed primarily for use by SMTP-based sites. They may not be well tuned for UUCP-only or UUCP-primarily nodes (the latter is defined as a small local net connected to the rest of the world via UUCP). However, there is one hook to handle some special cases.

You can define a ``smart host'' that understands a richer address syntax using:

```
define('SMART_HOST', 'mailer:hostname')
```

In this case, the ``mailer:'' defaults to "relay". Any messages that can't be handled using the usual UUCP rules are passed to this host.

If you are on a local SMTP-based net that connects to the outside world via UUCP, you can use LOCAL_NET_CONFIG to add appropriate rules. For example:

```
define('SMART_HOST', 'uucp-new:uunet')
LOCAL_NET_CONFIG
R$* < @ $* . $m. > $*      $smtp $@ $2. $m. $: $1 < @ $2. $m. > $3
```

This will cause all names that end in your domain name (\$m) via SMTP; anything else will be sent via uucp-new (smart UUCP) to uunet. If you have FEATURE('nocanonify'), you may need to omit the dots after the \$m. If you are running a local DNS inside your domain which is not otherwise connected to the outside world, you probably want to use:

```
define('SMART_HOST', 'smtp:fire.wall.com')
LOCAL_NET_CONFIG
R$* < @ $* . > $*      $smtp $@ $2. $: $1 < @ $2. > $3
```

That is, send directly only to things you found in your DNS lookup; anything else goes through SMART_HOST.

You may need to turn off the anti-spam rules in order to accept UUCP mail with FEATURE('promiscuous_relay') and FEATURE('accept_unresolvable_domains').

+-----+
| WHO AM I? |
+-----+

Normally, the \$j macro is automatically defined to be your fully qualified domain name (FQDN). Sendmail does this by getting your host name using gethostname and then calling gethostbyname on the result. For example, in some environments gethostname returns only the root of the host name (such as "foo"); gethostbyname is supposed to return the FQDN ("foo.bar.com"). In some (fairly rare) cases, gethostbyname may fail to return the FQDN. In this case you MUST define confDOMAIN_NAME to be your fully qualified domain name. This is usually done using:

```
Dmbar.com
define('confDOMAIN_NAME', '$w.$m')dnl
```

```
+-----+
| ACCEPTING MAIL FOR MULTIPLE NAMES |
+-----+
```

If your host is known by several different names, you need to augment class {w}. This is a list of names by which your host is known, and anything sent to an address using a host name in this list will be treated as local mail. You can do this in two ways: either create the file /etc/mail/local-host-names containing a list of your aliases (one per line), and use `'FEATURE('use_cw_file')'` in the .mc file, or add `'LOCAL_DOMAIN('alias.host.name')'`. Be sure you use the fully-qualified name of the host, rather than a short name.

If you want to have different address in different domains, take a look at the virtusertable feature, which is also explained at <http://www.sendmail.org/virtual-hosting.html>

```
+-----+
| USING MAILERTABLES |
+-----+
```

To use FEATURE('mailertable'), you will have to create an external database containing the routing information for various domains. For example, a mailertable file in text format might be:

.my.domain	xnet:%1.my.domain
uuhost1.my.domain	uucp-new:uuhost1
.bitnet	smtp:relay.bit.net

This should normally be stored in /etc/mail/mailertable. The actual database version of the mailertable is built using:

```
makemap hash /etc/mail/mailertable < /etc/mail/mailertable
```

The semantics are simple. Any LHS entry that does not begin with a dot matches the full host name indicated. LHS entries beginning with a dot match anything ending with that domain name (including the leading dot) -- that is, they can be thought of as having a leading `".+"` regular expression pattern for a non-empty sequence of characters. Matching is done in order of most-to-least qualified -- for example, even though `".my.domain"` is listed first in the above example, an entry of `"uuhost1.my.domain"` will match the second entry since it is more explicit. Note: e-mail to `"user@my.domain"` does not match any entry in the above table. You need to have something like:

my.domain	esmtphost:my.domain
-----------	---------------------

The RHS should always be a "mailer:host" pair. The mailer is the configuration name of a mailer (that is, an M line in the sendmail.cf file). The "host" will be the hostname passed to that mailer. In domain-based matches (that is, those with leading dots) the `"%1"` may be used to interpolate the wildcarded part of the host name. For example, the first line above sends everything addressed to `"anything.my.domain"` to that same host name, but using the (presumably experimental) xnet mailer.

In some cases you may want to temporarily turn off MX records,

particularly on gateways. For example, you may want to MX everything in a domain to one machine that then forwards it directly. To do this, you might use the DNS configuration:

```
*.domain.      IN      MX      0      relay.machine
```

and on relay.machine use the mailertable:

```
.domain      smtp:[gateway.domain]
```

The [square brackets] turn off MX records for this host only. If you didn't do this, the mailertable would use the MX record again, which would give you an MX loop.

```
+-----+
| USING USERDB TO MAP FULL NAMES |
+-----+
```

The user database was not originally intended for mapping full names to login names (e.g., Eric.Allman => eric), but some people are using it that way. (it is recommended that you set up aliases for this purpose instead -- since you can specify multiple alias files, this is fairly easy.) The intent was to locate the default maildrop at a site, but allow you to override this by sending to a specific host.

If you decide to set up the user database in this fashion, it is imperative that you not use FEATURE('stickyhost') -- otherwise, e-mail sent to Full.Name@local.host.name will be rejected.

To build the internal form of the user database, use:

```
makemap btree /etc/mail/userdb < /etc/mail/userdb.txt
```

As a general rule, it is an extremely bad idea to using full names as e-mail addresses, since they are not in any sense unique. For example, the UNIX software-development community has at least two well-known Peter Deutsches, and at one time Bell Labs had two Stephen R. Bournes with offices along the same hallway. Which one will be forced to suffer the indignity of being Stephen_R_Bourne_2? The less famous of the two, or the one that was hired later?

Finger should handle full names (and be fuzzy). Mail should use handles, and not be fuzzy.

```
+-----+
| MISCELLANEOUS SPECIAL FEATURES |
+-----+
```

Plussed users

Sometimes it is convenient to merge configuration on a centralized mail machine, for example, to forward all root mail to a mail server. In this case it might be useful to be able to treat the root addresses as a class of addresses with subtle differences. You can do this using plussed users. For example, a client might include the alias:

```
root: root+client1@server
```

On the server, this will match an alias for "root+client1".
If that is not found, the alias "root+*" will be tried,
then "root".

```
+-----+  
| SECURITY NOTES |  
+-----+
```

A lot of sendmail security comes down to you. Sendmail 8 is much more careful about checking for security problems than previous versions, but there are some things that you still need to watch for. In particular:

- * Make sure the aliases file isn't writable except by trusted system personnel. This includes both the text and database version.
- * Make sure that other files that sendmail reads, such as the mailertable, are only writable by trusted system personnel.
- * The queue directory should not be world writable PARTICULARLY if your system allows "file giveaways" (that is, if a non-root user can chown any file they own to any other user).
- * If your system allows file giveaways, DO NOT create a publically writable directory for forward files. This will allow anyone to steal anyone else's e-mail. Instead, create a script that copies the .forward file from users' home directories once a night (if you want the non-NFS-mounted forward directory).
- * If your system allows file giveaways, you'll find that sendmail is much less trusting of :include: files -- in particular, you'll have to have /SENDMAIL/ANY/SHELL/ in /etc/shells before they will be trusted (that is, before files and programs listed in them will be honored).

In general, file giveaways are a mistake -- if you can turn them off, do so.

```
+-----+  
| TWEAKING CONFIGURATION OPTIONS |  
+-----+
```

There are a large number of configuration options that don't normally need to be changed. However, if you feel you need to tweak them, you can define the following M4 variables. This list is shown in four columns: the name you define, the default value for that definition, the option or macro that is affected (either Ox for an option or Dx for a macro), and a brief description. Greater detail of the semantics can be found in the Installation and Operations Guide.

Some options are likely to be deprecated in future versions -- that is, the option is only included to provide back-compatibility. These are marked with "*".

Remember that these options are M4 variables, and hence may need to be quoted. In particular, arguments with commas will usually have to

be ``double quoted, like this phrase`` to avoid having the comma confuse things. This is common for alias file definitions and for the read timeout.

M4 Variable Name =====	Configuration =====	Description & [Default] =====
confMAILER_NAME	\$n macro	[MAILER-DAEMON] The sender name used for internally generated outgoing messages.
confDOMAIN_NAME	\$j macro	If defined, sets \$j. This should only be done if your system cannot determine your local domain name, and then it should be set to \$w.Foo.COM, where Foo.COM is your domain name.
confCF_VERSION	\$Z macro	If defined, this is appended to the configuration version name.
confLDAP_CLUSTER	`\${sendmailMTACluster}` macro	If defined, this is the LDAP cluster to use for LDAP searches as described above in ``USING LDAP FOR ALIASES, MAPS, AND CLASSES``.
confFROM_HEADER	From:	[\$?x\$x <\$g>\$ \$g\$.] The format of an internally generated From: address.
confRECEIVED_HEADER	Received: [`\${sfrom \$s \$.\${?}_(\${s\$ from \$.\${?}_) \$.\${?}{auth_type}(authenticated) \$.by \$j (\$v/\$Z)\${?r with \$r\$. id \$i\${?u for \$u; \$; \$.b}]	The format of the Received: header in messages passed through this host. It is unwise to try to change this.
confCW_FILE	Fw class	[/etc/mail/local-host-names] Name of file used to get the local additions to class {w} (local host names).
confCT_FILE	Ft class	[/etc/mail/trusted-users] Name of file used to get the local additions to class {t} (trusted users).
confCR_FILE	FR class	[/etc/mail/relay-domains] Name of file used to get the local additions to class {R} (hosts allowed to relay).
confTRUSTED_USERS	Ct class	[no default] Names of users to add to the list of trusted users. This list always includes root, uucp, and daemon. See also FEATURE('use_ct_file').
confTRUSTED_USER	TrustedUser	[no default] Trusted user for file ownership and starting the daemon. Not to be confused with confTRUSTED_USERS (see above).
confSMTP_MAILER	-	[esmtplib] The mailer name used when SMTP connectivity is required. One of "smtp", "smtp8", "esmtplib", or "dsmtplib".
confUUCP_MAILER	-	[uucp-old] The mailer to be used by default for bang-format recipient addresses. See also discussion of class {U}, class {Y}, and class {Z} in the MAILER('uucp') section.

confLOCAL_MAILER	-	[local] The mailer name used when local connectivity is required. Almost always "local".
confRELAY_MAILER	-	[relay] The default mailer name used for relaying any mail (e.g., to a BITNET_RELAY, a SMART_HOST, or whatever). This can reasonably be "uucp-new" if you are on a UUCP-connected site.
confSEVEN_BIT_INPUT	SevenBitInput	[False] Force input to seven bits?
confEIGHT_BIT_HANDLING	EightBitMode	[pass8] 8-bit data handling
confALIAS_WAIT	AliasWait	[10m] Time to wait for alias file rebuild until you get bored and decide that the apparently pending rebuild failed.
confMIN_FREE_BLOCKS	MinFreeBlocks	[100] Minimum number of free blocks on queue filesystem to accept SMTP mail. (Prior to 8.7 this was minfree/maxsize, where minfree was the number of free blocks and maxsize was the maximum message size. Use confMAX_MESSAGE_SIZE for the second value now.)
confMAX_MESSAGE_SIZE	MaxMessageSize	[infinite] The maximum size of messages that will be accepted (in bytes).
confBLANK_SUB	BlankSub	[.] Blank (space) substitution character.
confCON_EXPENSIVE	HoldExpensive	[False] Avoid connecting immediately to mailers marked expensive.
confCHECKPOINT_INTERVAL	CheckpointInterval	[10] Checkpoint queue files every N recipients.
confDELIVERY_MODE	DeliveryMode	[background] Default delivery mode.
confERROR_MODE	ErrorMode	[print] Error message mode.
confERROR_MESSAGE	ErrorHeader	[undefined] Error message header/file.
confSAVE_FROM_LINES	SaveFromLine	Save extra leading From_ lines.
confTEMP_FILE_MODE	TempFileMode	[0600] Temporary file mode.
confMATCH_GECOS	MatchGECOS	[False] Match GECOS field.
confMAX_HOP	MaxHopCount	[25] Maximum hop count.
confIGNORE_DOTS*	IgnoreDots	[False; always False in -bs or -bd mode] Ignore dot as terminator for incoming messages?
confBIND_OPTS	ResolverOptions	[undefined] Default options for DNS resolver.
confMIME_FORMAT_ERRORS*	SendMimeErrors	[True] Send error messages as MIME-encapsulated messages per RFC 1344.
confFORWARD_PATH	ForwardPath	[\$z/.forward.\$w:\$z/.forward] The colon-separated list of places to search for .forward files. N.B.: see the Security Notes section.
confMCI_CACHE_SIZE	ConnectionCacheSize	[2] Size of open connection cache.
confMCI_CACHE_TIMEOUT	ConnectionCacheTimeout	[5m] Open connection cache timeout.
confHOST_STATUS_DIRECTORY	HostStatusDirectory	[undefined] If set, host status is kept on disk between sendmail runs in the named directory tree. This need not be a full pathname, in which case it is interpreted relative to the queue directory.

confSINGLE_THREAD_DELIVERY	SingleThreadDelivery	[False] If this option and the HostStatusDirectory option are both set, single thread deliveries to other hosts. That is, don't allow any two sendmails on this host to connect simultaneously to any other single host. This can slow down delivery in some cases, in particular since a cached but otherwise idle connection to a host will prevent other sendmails from connecting to the other host.
confUSE_ERRORS_TO*	UseErrorsTo	[False] Use the Errors-To: header to deliver error messages. This should not be necessary because of general acceptance of the envelope/header distinction.
confLOG_LEVEL	LogLevel	[9] Log level.
confME_TOO	MeToo	[True] Include sender in group expansions. This option is deprecated and will be removed from a future version.
confCHECK_ALIASES	CheckAliases	[False] Check RHS of aliases when running newaliases. Since this does DNS lookups on every address, it can slow down the alias rebuild process considerably on large alias files.
confOLD_STYLE_HEADERS*	OldStyleHeaders	[True] Assume that headers without special chars are old style.
confPRIVACY_FLAGS	PrivacyOptions	[authwarnings] Privacy flags.
confCOPY_ERRORS_TO	PostmasterCopy	[undefined] Address for additional copies of all error messages.
confQUEUE_FACTOR	QueueFactor	[600000] Slope of queue-only function.
confQUEUE_FILE_MODE	QueueFileMode	[undefined] Default permissions for queue files (octal). If not set, sendmail uses 0600 unless its real and effective uid are different in which case it uses 0644.
confDONT_PRUNE_ROUTES	DontPruneRoutes	[False] Don't prune down route-addr syntax addresses to the minimum possible.
confSAFE_QUEUE*	SuperSafe	[True] Commit all messages to disk before forking.
confTO_INITIAL	Timeout.initial	[5m] The timeout waiting for a response on the initial connect.
confTO_CONNECT	Timeout.connect	[0] The timeout waiting for an initial connect() to complete. This can only shorten connection timeouts; the kernel silently enforces an absolute maximum (which varies depending on the system).
confTO_ICONNECT	Timeout.icontect	[undefined] Like Timeout.connect, but applies only to the very first attempt to connect to a host in a message. This allows a single very fast pass followed by more careful delivery attempts in the future.
confTO_ACONNECT	Timeout.aconnect	[0] The overall timeout waiting for all connection for a single delivery

		attempt to succeed. If 0, no overall limit is applied.
confTO_HELO	Timeout.helo	[5m] The timeout waiting for a response to a HELO or EHLO command.
confTO_MAIL	Timeout.mail	[10m] The timeout waiting for a response to the MAIL command.
confTO_RCPT	Timeout.rcpt	[1h] The timeout waiting for a response to the RCPT command.
confTO_DATAINIT	Timeout.datainit	[5m] The timeout waiting for a 354 response from the DATA command.
confTO_DATABLOCK	Timeout.datablock	[1h] The timeout waiting for a block during DATA phase.
confTO_DATAFINAL	Timeout.datafinal	[1h] The timeout waiting for a response to the final "." that terminates a message.
confTO_RSET	Timeout.rset	[5m] The timeout waiting for a response to the RSET command.
confTO_QUIT	Timeout.quit	[2m] The timeout waiting for a response to the QUIT command.
confTO_MISC	Timeout.misc	[2m] The timeout waiting for a response to other SMTP commands.
confTO_COMMAND	Timeout.command	[1h] In server SMTP, the timeout waiting for a command to be issued.
confTO_IDENT	Timeout.ident	[5s] The timeout waiting for a response to an IDENT query.
confTO_FILEOPEN	Timeout.fileopen	[60s] The timeout waiting for a file (e.g., :include: file) to be opened.
confTO_LHLO	Timeout.lhlo	[2m] The timeout waiting for a response to an LMTP LHLO command.
confTO_AUTH	Timeout.auth	[10m] The timeout waiting for a response in an AUTH dialogue.
confTO_STARTTLS	Timeout.starttls	[1h] The timeout waiting for a response to an SMTP STARTTLS command.
confTO_CONTROL	Timeout.control	[2m] The timeout for a complete control socket transaction to complete.
confTO_QUEUERETURN	Timeout.queuereturn	[5d] The timeout before a message is returned as undeliverable.
confTO_QUEUERETURN_NORMAL	Timeout.queuereturn.normal	[undefined] As above, for normal priority messages.
confTO_QUEUERETURN_URGENT	Timeout.queuereturn.urgent	[undefined] As above, for urgent priority messages.
confTO_QUEUERETURN_NONURGENT	Timeout.queuereturn.non-urgent	[undefined] As above, for non-urgent (low) priority messages.
confTO_QUEUEWARN	Timeout.queuewarn	[4h] The timeout before a warning message is sent to the sender telling them that the message has been

```

                                deferred.
confTO_QUEUEWARN_NORMAL Timeout.queuewarn.normal
                                [undefined] As above, for normal
                                priority messages.
confTO_QUEUEWARN_URGENT Timeout.queuewarn.urgent
                                [undefined] As above, for urgent
                                priority messages.
confTO_QUEUEWARN_NONURGENT
                                Timeout.queuewarn.non-urgent
                                [undefined] As above, for non-urgent
                                (low) priority messages.
confTO_HOSTSTATUS           Timeout.hoststatus
                                [30m] How long information about host
                                statuses will be maintained before it
                                is considered stale and the host should
                                be retried. This applies both within
                                a single queue run and to persistent
                                information (see below).
confTO_RESOLVER_RETRANS Timeout.resolver.retrans
                                [varies] Sets the resolver's
                                retransmission time interval (in
                                seconds). Sets both
                                Timeout.resolver.retrans.first and
                                Timeout.resolver.retrans.normal.
confTO_RESOLVER_RETRANS_FIRST Timeout.resolver.retrans.first
                                [varies] Sets the resolver's
                                retransmission time interval (in
                                seconds) for the first attempt to
                                deliver a message.
confTO_RESOLVER_RETRANS_NORMAL Timeout.resolver.retrans.normal
                                [varies] Sets the resolver's
                                retransmission time interval (in
                                seconds) for all resolver lookups
                                except the first delivery attempt.
confTO_RESOLVER_RETRY      Timeout.resolver.retry
                                [varies] Sets the number of times
                                to retransmit a resolver query.
                                Sets both
                                Timeout.resolver.retry.first and
                                Timeout.resolver.retry.normal.
confTO_RESOLVER_RETRY_FIRST Timeout.resolver.retry.first
                                [varies] Sets the number of times
                                to retransmit a resolver query for
                                the first attempt to deliver a
                                message.
confTO_RESOLVER_RETRY_NORMAL Timeout.resolver.retry.normal
                                [varies] Sets the number of times
                                to retransmit a resolver query for
                                all resolver lookups except the
                                first delivery attempt.
confTIME_ZONE              TimeZoneSpec [USE_SYSTEM] Time zone info -- can be
                                USE_SYSTEM to use the system's idea,
                                USE_TZ to use the user's TZ envvariable,
                                or something else to force that value.
confDEF_USER_ID            DefaultUser  [1:1] Default user id.
confUSERDB_SPEC            UserDatabaseSpec
                                [undefined] User database
                                specification.
confFALLBACK_MX            FallbackMXhost [undefined] Fallback MX host.
confTRY_NULL_MX_LIST       TryNullMXList [False] If this host is the best MX

```

		for a host and other arrangements haven't been made, try connecting to the host directly; normally this would be a config error.
confQUEUE_LA	QueueLA	[varies] Load average at which queue-only function kicks in. Default values is (8 * numproc) where numproc is the number of processors online (if that can be determined).
confREFUSE_LA	RefuseLA	[varies] Load average at which incoming SMTP connections are refused. Default values is (12 * numproc) where numproc is the number of processors online (if that can be determined).
confDELAY_LA	DelayLA	[0] Load average at which sendmail will sleep for one second on most SMTP commands and before accepting connections. 0 means no limit.
confMAX_ALIAS_RECURSION	MaxAliasRecursion	[10] Maximum depth of alias recursion.
confMAX_DAEMON_CHILDREN	MaxDaemonChildren	[undefined] The maximum number of children the daemon will permit. After this number, connections will be rejected. If not set or <= 0, there is no limit.
confMAX_HEADERS_LENGTH	MaxHeadersLength	[32768] Maximum length of the sum of all headers.
confMAX_MIME_HEADER_LENGTH	MaxMimeHeaderLength	[undefined] Maximum length of certain MIME header field values.
confCONNECTION_RATE_THROTTLE	ConnectionRateThrottle	[undefined] The maximum number of connections permitted per second per daemon. After this many connections are accepted, further connections will be delayed. If not set or <= 0, there is no limit.
confWORK_RECIPIENT_FACTOR	RecipientFactor	[30000] Cost of each recipient.
confSEPARATE_PROC	ForkEachJob	[False] Run all deliveries in a separate process.
confWORK_CLASS_FACTOR	ClassFactor	[1800] Priority multiplier for class.
confWORK_TIME_FACTOR	RetryFactor	[90000] Cost of each delivery attempt.
confQUEUE_SORT_ORDER	QueueSortOrder	[Priority] Queue sort algorithm: Priority, Host, Filename, Random, Modification, or Time.
confMIN_QUEUE_AGE	MinQueueAge	[0] The minimum amount of time a job must sit in the queue between queue runs. This allows you to set the queue run interval low for better responsiveness without trying all jobs in each run.
confDEF_CHAR_SET	DefaultCharSet	[unknown-8bit] When converting unlabeled 8 bit input to MIME, the character set to use by default.
confSERVICE_SWITCH_FILE	ServiceSwitchFile	

		[/etc/mail/service.switch] The file to use for the service switch on systems that do not have a system-defined switch.
confHOSTS_FILE	HostsFile	[/etc/hosts] The file to use when doing "file" type access of hosts names.
confDIAL_DELAY	DialDelay	[0s] If a connection fails, wait this long and try again. Zero means "don't retry". This is to allow "dial on demand" connections to have enough time to complete a connection.
confNO_RCPT_ACTION	NoRecipientAction	[none] What to do if there are no legal recipient fields (To:, Cc: or Bcc:) in the message. Legal values can be "none" to just leave the nonconforming message as is, "add-to" to add a To: header with all the known recipients (which may expose blind recipients), "add-apparently-to" to do the same but use Apparently-To: instead of To: (strongly discouraged in accordance with IETF standards), "add-bcc" to add an empty Bcc: header, or "add-to-undisclosed" to add the header `To: undisclosed-recipients:;`.
confSAFE_FILE_ENV	SafeFileEnvironment	[undefined] If set, sendmail will do a chroot() into this directory before writing files.
confCOLON_OK_IN_ADDR	ColonOkInAddr	[True unless Configuration Level > 6] If set, colons are treated as a regular character in addresses. If not set, they are treated as the introducer to the RFC 822 "group" syntax. Colons are handled properly in route-addrs. This option defaults on for V5 and lower configuration files.
confMAX_QUEUE_RUN_SIZE	MaxQueueRunSize	[0] If set, limit the maximum size of any given queue run to this number of entries. Essentially, this will stop reading each queue directory after this number of entries are reached; it does <u>not</u> pick the highest priority jobs, so this should be as large as your system can tolerate. If not set, there is no limit.
confMAX_QUEUE_CHILDREN	MaxQueueChildren	[undefined] Limits the maximum number of concurrent queue runners active. This is to keep system resources used within a reasonable limit. Relates to Queue Groups and ForkAllJobs.
confMAX_RUNNERS_PER_QUEUE	MaxRunnersPerQueue	[1] Only active when MaxQueueChildren defined. Controls the maximum number of queue runners (aka queue children) active at the same time in a work group. See also MaxQueueChildren.

confDONT_EXPAND_CNAMES	DontExpandCnames	[False] If set, \$[... \$] lookups that do DNS based lookups do not expand CNAME records. This currently violates the published standards, but the IETF seems to be moving toward legalizing this. For example, if "FTP.Foo.ORG" is a CNAME for "Cruft.Foo.ORG", then with this option set a lookup of "FTP" will return "FTP.Foo.ORG"; if clear it returns "Cruft.FOO.ORG". N.B. you may not see any effect until your downstream neighbors stop doing CNAME lookups as well.
confFROM_LINE	UnixFromLine	[From \$g \$d] The From_ line used when sending to files or programs.
confSINGLE_LINE_FROM_HEADER	SingleLineFromHeader	[False] From: lines that have embedded newlines are unwrapped onto one line.
confALLOW_BOGUS_HELO	AllowBogusHELO	[False] Allow HELO SMTP command that does not include a host name.
confMUST_QUOTE_CHARS	MustQuoteChars	[.] Characters to be quoted in a full name phrase (@,;:\()[] are automatic).
confOPERATORS	OperatorChars	[.:%#@!^/[]+] Address operator characters.
confSMTP_LOGIN_MSG	SmtgGreetingMessage	[\$j Sendmail \$v/\$Z; \$b] The initial (spontaneous) SMTP greeting message. The word "ESMTP" will be inserted between the first and second words to convince other sendmails to try to speak ESMTP.
confDONT_INIT_GROUPS	DontInitGroups	[False] If set, the initgroups(3) routine will never be invoked. You might want to do this if you are running NIS and you have a large group map, since this call does a sequential scan of the map; in a large site this can cause your ypserv to run essentially full time. If you set this, agents run on behalf of users will only have their primary (/etc/passwd) group permissions.
confUNSAFE_GROUP_WRITES	UnsafeGroupWrites	[False] If set, group-writable :include: and .forward files are considered "unsafe", that is, programs and files cannot be directly referenced from such files. World-writable files are always considered unsafe.
confCONNECT_ONLY_TO	ConnectOnlyTo	[undefined] override connection address (for testing).
confCONTROL_SOCKET_NAME	ControlSocketName	[undefined] Control socket for daemon management.
confDOUBLE_BOUNCE_ADDRESS	DoubleBounceAddress	[postmaster] If an error occurs when sending an error message, send that "double bounce" error message to this

confDEAD_LETTER_DROP	DeadLetterDrop	address. If it expands to an empty string, double bounces are dropped. [undefined] Filename to save bounce messages which could not be returned to the user or sent to postmaster. If not set, the queue file will be renamed.
confRRT_IMPLIES_DSN	RrtImpliesDsn	[False] Return-Receipt-To: header implies DSN request.
confRUN_AS_USER	RunAsUser	[undefined] If set, become this user when reading and delivering mail. Causes all file reads (e.g., .forward and :include: files) to be done as this user. Also, all programs will be run as this user, and all output files will be written as this user. Intended for use only on firewalls where users do not have accounts.
confMAX_RCPTS_PER_MESSAGE	MaxRecipientsPerMessage	[infinite] If set, allow no more than the specified number of recipients in an SMTP envelope. Further recipients receive a 452 error code (i.e., they are deferred for the next delivery attempt).
confBAD_RCPT_THROTTLE	BadRcptThrottle	[infinite] If set and more than the specified number of recipients in an envelope are rejected, sleep for one second after each rejected RCPT command.
confDONT_PROBE_INTERFACES	DontProbeInterfaces	[False] If set, sendmail will _not_ insert the names and addresses of any local interfaces into class {w} (list of known "equivalent" addresses). If you set this, you must also include some support for these addresses (e.g., in a mailertable entry) -- otherwise, mail to addresses in this list will bounce with a configuration error. If set to "loopback" (without quotes), sendmail will skip loopback interfaces (e.g., "lo0").
confPID_FILE	PidFile	[system dependent] Location of pid file.
confPROCESS_TITLE_PREFIX	ProcessTitlePrefix	[undefined] Prefix string for the process title shown on 'ps' listings.
confDONT_BLAME_SENDMAIL	DontBlameSendmail	[safe] Override sendmail's file safety checks. This will definitely compromise system security and should not be used unless absolutely necessary.
confREJECT_MSG	-	[550 Access denied] The message given if the access database contains REJECT in the value portion.
confRELAY_MSG	-	[550 Relaying denied] The message given if an unauthorized relaying attempt is rejected.

confDF_BUFFER_SIZE	DataFileBufferSize	[4096] The maximum size of a memory-buffered data (df) file before a disk-based file is used.
confXF_BUFFER_SIZE	XScriptFileBufferSize	[4096] The maximum size of a memory-buffered transcript (xf) file before a disk-based file is used.
confAUTH_MECHANISMS	AuthMechanisms	[GSSAPI KERBEROS_V4 DIGEST-MD5 CRAM-MD5] List of authentication mechanisms for AUTH (separated by spaces). The advertised list of authentication mechanisms will be the intersection of this list and the list of available mechanisms as determined by the CYRUS SASL library.
confDEF_AUTH_INFO	DefaultAuthInfo	[undefined] Name of file that contains authentication information for outgoing connections. This file must contain the user id, the authorization id, the password (plain text), the realm to use, and the list of mechanisms to try, each on a separate line and must be readable by root (or the trusted user) only. If no realm is specified, \$j is used. If no mechanisms are given in the file, AuthMechanisms is used. Notice: this option is deprecated and will be removed in future versions; it doesn't work for the MSP since it can't read the file. Use the authinfo ruleset instead. See also the section SMTP AUTHENTICATION.
confAUTH_OPTIONS	AuthOptions	[undefined] If this option is 'A' then the AUTH= parameter for the MAIL FROM command is only issued when authentication succeeded. Other values (which should be listed one after the other without any intervening characters except for space or comma) are a, c, d, f, p, and y. See doc/op/op.me for details.
confAUTH_MAX_BITS	AuthMaxBits	[INT_MAX] Limit the maximum encryption strength for the security layer in SMTP AUTH (SASL). Default is essentially unlimited.
confTLS_SRV_OPTIONS	TLSSrvOptions	If this option is 'V' no client verification is performed, i.e., the server doesn't ask for a certificate.
confLDAP_DEFAULT_SPEC	LDAPDefaultSpec	[undefined] Default map specification for LDAP maps. The value should only contain LDAP specific settings such as "-h host -p port -d bindDN", etc. The settings will be used for all LDAP maps unless they are specified in

		the individual map specification ('K' command).
confCACERT_PATH	CACERTPath	[undefined] Path to directory with certs of CAs.
confCACERT	CACERTFile	[undefined] File containing one CA cert.
confSERVER_CERT	ServerCertFile	[undefined] File containing the cert of the server, i.e., this cert is used when sendmail acts as server.
confSERVER_KEY	ServerKeyFile	[undefined] File containing the private key belonging to the server cert.
confCLIENT_CERT	ClientCertFile	[undefined] File containing the cert of the client, i.e., this cert is used when sendmail acts as client.
confCLIENT_KEY	ClientKeyFile	[undefined] File containing the private key belonging to the client cert.
confDH_PARAMETERS	DHParameters	[undefined] File containing the DH parameters.
confRAND_FILE	RandFile	[undefined] File containing random data (use prefix <code>file:</code>) or the name of the UNIX socket if EGD is used (use prefix <code>egd:</code>). STARTTLS requires this option if the compile flag HASURANDOM is not set (see sendmail/README).
confNICE_QUEUE_RUN	NiceQueueRun	[undefined] If set, the priority of queue runners is set the given value (<code>nice(3)</code>).
confDIRECT_SUBMISSION_MODIFIERS	DirectSubmissionModifiers	[undefined] Defines {daemon_flags} for direct submissions.
confUSE_MSP	UseMSP	[false] Use as mail submission program, see sendmail/SECURITY.
confDELIVER_BY_MIN	DeliverByMin	[0] Minimum time for Deliver By SMTP Service Extension (RFC 2852).
confSHARED_MEMORY_KEY	SharedMemoryKey	[0] Key for shared memory.
confFAST_SPLIT	FastSplit	[1] If set to a value greater than zero, the initial MX lookups on addresses is suppressed when they are sorted which may result in faster envelope splitting. If the mail is submitted directly from the command line, then the value also limits the number of processes to deliver the envelopes.
confMAILBOX_DATABASE	MailboxDatabase	[pw] Type of lookup to find information about local mailboxes.
confDEQUOTE_OPTS	-	[empty] Additional options for the dequote map.
confINPUT_MAIL_FILTERS	InputMailFilters	A comma separated list of filters which determines which filters and the invocation sequence are contacted for incoming SMTP messages. If none are set, no filters will be contacted.

confMILTER_LOG_LEVEL	Milter.LogLevel [9] Log level for input mail filter actions, defaults to LogLevel.
confMILTER_MACROS_CONNECT	Milter.macros.connect [empty] Macros to transmit to milters when a session connection starts.
confMILTER_MACROS_HELO	Milter.macros.helo [empty] Macros to transmit to milters after HELO command.
confMILTER_MACROS_ENVFROM	Milter.macros.envfrom [empty] Macros to transmit to milters after MAIL FROM command.
confMILTER_MACROS_ENVRCPT	Milter.macros.envrcpt [empty] Macros to transmit to milters after RCPT TO command.

See also the description of OSTYPE for some parameters that can be tweaked (generally pathnames to mailers).

ClientPortOptions and DaemonPortOptions are special cases since multiple clients/daemons can be defined. This can be done via

```
CLIENT_OPTIONS('field1=value1,field2=value2,...')
DAEMON_OPTIONS('field1=value1,field2=value2,...')
```

Note that multiple CLIENT_OPTIONS() commands (and therefore multiple ClientPortOptions settings) are allowed in order to give settings for each protocol family (e.g., one for Family=inet and one for Family=inet6). A restriction placed on one family only affects outgoing connections on that particular family.

If DAEMON_OPTIONS is not used, then the default is

```
DAEMON_OPTIONS('Port=smtp, Name=MTA')
DAEMON_OPTIONS('Port=587, Name=MSA, M=E')
```

If you use one DAEMON_OPTIONS macro, it will alter the parameters of the first of these. The second will still be defaulted; it represents a "Message Submission Agent" (MSA) as defined by RFC 2476 (see below). To turn off the default definition for the MSA, use FEATURE('no_default_msa') (see also FEATURES). If you use additional DAEMON_OPTIONS macros, they will add additional daemons.

Example 1: To change the port for the SMTP listener, while still using the MSA default, use

```
DAEMON_OPTIONS('Port=925, Name=MTA')
```

Example 2: To change the port for the MSA daemon, while still using the default SMTP port, use

```
FEATURE('no_default_msa')
DAEMON_OPTIONS('Name=MTA')
DAEMON_OPTIONS('Port=987, Name=MSA, M=E')
```

Note that if the first of those DAEMON_OPTIONS lines were omitted, then there would be no listener on the standard SMTP port.

Example 3: To listen on both IPv4 and IPv6 interfaces, use

```
DAEMON_OPTIONS('Name=MTA-v4, Family=inet')
DAEMON_OPTIONS('Name=MTA-v6, Family=inet6')
```

A "Message Submission Agent" still uses all of the same rulesets for processing the message (and therefore still allows message rejection via the check_* rulesets). In accordance with the RFC, the MSA will ensure that all domains in the envelope are fully qualified if the message is relayed to another MTA. It will also enforce the normal address syntax rules and log error messages. Additionally, by using the M=a modifier you can require authentication before messages are accepted by the MSA. Notice: Do NOT use the 'a' modifier on a public accessible MTA! Finally, the M=E modifier shown above disables ETRN as required by RFC 2476.

Mail filters can be defined using the INPUT_MAIL_FILTER() and MAIL_FILTER() commands:

```
INPUT_MAIL_FILTER('sample', 'S=local:/var/run/fl.sock')
MAIL_FILTER('myfilter', 'S=inet:3333@localhost')
```

The INPUT_MAIL_FILTER() command causes the filter(s) to be called in the same order they were specified by also setting confINPUT_MAIL_FILTERS. A filter can be defined without adding it to the input filter list by using MAIL_FILTER() instead of INPUT_MAIL_FILTER() in your .mc file. Alternatively, you can reset the list of filters and their order by setting confINPUT_MAIL_FILTERS option after all INPUT_MAIL_FILTER() commands in your .mc file.

```
+-----+
| MESSAGE SUBMISSION PROGRAM |
+-----+
```

The purpose of the message submission program (MSP) is explained in sendmail/SECURITY. This section contains a list of caveats and a few hints how for those who want to tweak the default configuration for it (which is installed as submit.cf).

Notice: do not add options/features to submit.mc unless you are absolutely sure you need them. Options you may want to change include:

- confTIME_ZONE on OS that don't use the default, e.g., Irix.
- confDELIVERY_MODE is set to interactive in msp.m4 instead of the default background mode.

Some things are not intended to work with the MSP. These include features that influence the delivery process (e.g., mailertable, aliases), or those that are only important for a SMTP server (e.g., virtusertable, DaemonPortOptions, multiple queues). Moreover, relaxing certain restrictions (RestrictQueueRun, permissions on queue directory) or adding features (e.g., enabling prog/file mailer) can cause security problems.

Other things don't work well with the MSP and require tweaking or workarounds. For example, to allow for client authentication it is not just sufficient to provide a client certificate and the corresponding key, but it is also necessary to make the key group (smmmsp) readable and tell sendmail not to complain about that, i.e.,

```
define('confDONT_BLAAME_SENDMAIL', 'GroupReadableKeyFile')
```

If the MSP should actually use AUTH then the necessary data should be placed in a map as explained in SMTP AUTHENTICATION:

```
FEATURE('authinfo', 'DATABASE_MAP_TYPE /etc/mail/msp-authinfo')
```

/etc/mail/msp-authinfo should contain an entry like:

```
AuthInfo:127.0.0.1      "U:smmmsp" "P:secret" "M:DIGEST-MD5"
```

The file and the map created by makemap should be owned by smmsp, its group should be smmsp, and it should have mode 640. The database used by the MTA for AUTH must have a corresponding entry. Additionally the MTA must trust this authentication data so the AUTH= part will be relayed on to the next hop. This can be achieved by adding the following to your sendmail.mc [file](#):

```
LOCAL_RULESETS
SLocal_trust_auth
R$*      $: ${auth_authen}
Rsmmsp   $# OK
```

feature/msp.m4 defines almost all settings for the MSP. Most of those should not be changed at all. Some of the features and options can be overridden if really necessary. It is a bit tricky to do this, because it depends on the actual way the option is defined in feature/msp.m4. If it is directly defined (i.e., define()) then the modified value must be defined after

```
FEATURE('msp')
```

If it is conditionally defined (i.e., ifdef()) then the desired value must be defined before the FEATURE line in the .mc file. To see how the options are defined read feature/msp.m4.

```
+-----+
| FORMAT OF FILES AND MAPS |
+-----+
```

Files that define classes, i.e., F{classname}, consist of lines each of which contains a single element of the class. For example, /etc/mail/local-host-names may have the following content:

```
my.domain
another.domain
```

Maps must be created using makemap(8) , e.g.,

```
makemap hash MAP < MAP
```

In general, a text file from which a map is created contains lines of the form

```
key      value
```

where 'key' and 'value' are also called LHS and RHS, respectively. By default, the delimiter between LHS and RHS is a non-empty sequence of white space characters.

```
+-----+
| DIRECTORY LAYOUT |
+-----+
```

Within this directory are several subdirectories, to wit:

m4	General support routines. These are typically very important and should not be changed without very careful consideration.
cf	The configuration files themselves. They have ".mc" suffixes, and must be run through m4 to become complete. The resulting output should have a ".cf" suffix.
ostype	Definitions describing a particular operating system type. These should always be referenced using the OSTYPE macro in the .mc file. Examples include "bsd4.3", "bsd4.4", "sunos3.5", and "sunos4.1".
domain	Definitions describing a particular domain, referenced using the DOMAIN macro in the .mc file. These are site dependent; for example, "CS.Berkeley.EDU.m4" describes hosts in the CS.Berkeley.EDU subdomain.
mailer	Descriptions of mailers. These are referenced using the MAILER macro in the .mc file.
sh	Shell files used when building the .cf file from the .mc file in the cf subdirectory.
feature	These hold special orthogonal features that you might want to include. They should be referenced using the FEATURE macro.
hack	Local hacks. These can be referenced using the HACK macro. They shouldn't be of more than voyeuristic interest outside the .Berkeley.EDU domain, but who knows?
siteconfig	Site configuration -- e.g., tables of locally connected UUCP sites.

```
+-----+
| ADMINISTRATIVE DETAILS |
+-----+
```

The following sections detail usage of certain internal parts of the sendmail.cf file. Read them carefully if you are trying to modify the current model. If you find the above descriptions adequate, these should be {boring, confusing, tedious, ridiculous} (pick one or more).

RULESETS (* means built in to sendmail)

- 0 * Parsing
- 1 * Sender rewriting
- 2 * Recipient rewriting
- 3 * Canonicalization
- 4 * Post cleanup

5 * Local address rewrite (after aliasing)
 1x mailer rules (sender qualification)
 2x mailer rules (recipient qualification)
 3x mailer rules (sender header qualification)
 4x mailer rules (recipient header qualification)
 5x mailer subroutines (general)
 6x mailer subroutines (general)
 7x mailer subroutines (general)
 8x reserved
 90 Mailertable host stripping
 96 Bottom half of Ruleset 3 (ruleset 6 in old sendmail)
 97 Hook for recursive ruleset 0 call (ruleset 7 in old sendmail)
 98 Local part of ruleset 0 (ruleset 8 in old sendmail)

MAILERS

0	local, prog	local and program mailers
1	[e]smtp, relay	SMTP channel
2	uucp-*	UNIX-to-UNIX Copy Program
3	netnews	Network News delivery
4	fax	Sam Leffler's HylaFAX software
5	mail11	DECnet mailer

MACROS

A	
B	Bitnet Relay
C	DECnet Relay
D	The local domain -- usually not needed
E	reserved for X.400 Relay
F	FAX Relay
G	
H	mail Hub (for mail clusters)
I	
J	
K	
L	Luser Relay
M	Masquerade (who you claim to be)
N	
O	
P	
Q	
R	Relay (for unqualified names)
S	Smart Host
T	
U	my UUCP name (if you have a UUCP connection)
V	UUCP Relay (class {V} hosts)
W	UUCP Relay (class {W} hosts)
X	UUCP Relay (class {X} hosts)
Y	UUCP Relay (all other hosts)
Z	Version number

CLASSES

A	
B	domains that are candidates for bestmx lookup
C	

D
 E addresses that should not seem to come from \$M
 F hosts this system forward for
 G domains that should be looked up in genericstable
 H
 I
 J
 K
 L addresses that should not be forwarded to \$R
 M domains that should be mapped to \$M
 N host/domains that should not be mapped to \$M
 O operators that indicate network operations (cannot be in local names)
 P top level pseudo-domains: BITNET, DECNET, FAX, UUCP, etc.
 Q
 R domains this system is willing to relay (pass anti-spam filters)
 S
 T
 U locally connected UUCP hosts
 V UUCP hosts connected to relay \$V
 W UUCP hosts connected to relay \$W
 X UUCP hosts connected to relay \$X
 Y locally connected smart UUCP hosts
 Z locally connected domain-ized UUCP hosts
 . the class containing only a dot
 [the class containing only a left bracket

M4 DIVERSIONS

1 Local host detection and resolution
 2 Local Ruleset 3 additions
 3 Local Ruleset 0 additions
 4 UUCP Ruleset 0 additions
 5 locally interpreted names (overrides \$R)
 6 local configuration (at top of file)
 7 mailer definitions
 8 DNS based blacklists
 9 special local rulesets (1 and 2)

\$Revision: 8.600 \$, Last updated \$Date: 2002/01/10 17:43:41 \$

Generated by unregistered txt2pdf v.5.6 © SANFACE Software 2002
 Available at <http://www.sanface.com/txt2pdf.html>