PostfixConfiguration < Webmin < TWiki

Postfix is an efficient and feature-rich mail server that was designed by Wietse Venema at the IBM T.J. Watson Research Center (Figure 10.1, Postfix). It was intended to be a replacement for the popular Sendmail. While it still represents only a small percentage of mail server installations worldwide, its popularity is growing rapidly, due to its simple configuration, secure implementation, and high performance architecture. Also, because Postfix is designed to behave outwardly like Sendmail, it is a mostly drop-in replacement for the older, larger, and slower mail server. It does lack some of the obscure features of Sendmail, but the features it lacks are rarely used by the vast majority of users, so they are not often missed.

Figure 10.1 Postfix

The Postfix project, originally named VMailer (fortunately for everyone, the name was changed before release due to legal entanglements of the VMailer name), is designed as a group of related but separate executable components, providing security through segmentation. Smaller parts are easier to debug, as well. The Internet home of Postfix is www.postfix.org. Postfix is an ideal mail server choice for new mail administrators, and even experienced Sendmail administrators might find its simplicity appealing. Because it provides a quite compatible Sendmail-ish exterior, and provides programs of the same names (such as `sendmail` for sending mail, `mailq` for managing the queue, etc.), and can utilize the same type of aliases and forwarding files that Sendmail uses, it is possible to replace Sendmail without reconfiguring existing mail-related tools, or rewriting local scripts. After such a switch, local users may not even notice the difference.

Note

The previous statements should not be viewed as an endorsement of Postfix as being a *better* mail transport agent than Sendmail. The two projects have different emphasis, and have had very different development models. Sendmail has been in use all over the world for over 20 years in one form or another, and thus has an extremely large head start on Postfix with regard to maturity, available documentation, number of experienced administrators, and support tools. Postfix is only a few years old and has much more limited supporting documentation and tools to enhance it. The decision for which mail transfer agent is appropriate for your network will be dictated by the requirements and the availability of local expertise.

General Options

The **General Options** page configures a number of options regarding the general behavior of Postfix. Specifically, most of the configuration options that impact all users and all messages are configured here. Postfix, keeping with its philosophy of simplicity, usually requires only a few configuration file changes to get a mail server running efficiently and securely.

The **General Options** page is divided into two parts. The upper section is labeled **Most Useful**

General Options and the lower section **Other General Options**. In many standard installations, it may be possible to start up a Postfix installation with just configuration of one or more of the three directives in the upper section. Unless otherwise stated, all of the options on this page correspond to directives in the `main.cf` file in the Postfix configuration directory.

Most Useful General Options

The three options in this section are, in some installations, the only options that need to be altered to get Postfix running for both sending and receiving email (Figure 10.2, Most Useful General Options).

Figure 10.2 Most Useful General Options

What domain to use in outbound mail

Here you may specify the domain or host name to use to identify the source on outgoing mail. Postfix defaults to using the host name of the server, but you most likely will want it to identify mail as coming from your domain name instead. If your mail server will be accepting mail for a large number of users under a single domain name, you will most likely configure domain name here, and create a domain-wide alias database to map user names to their respective local mail servers. This option correlates to the `myorigin` Postfix directive.

What domain to receive mail for

This option accepts a list of domains and addresses to receive mail as its final destination. In other words, when mail reaches the server destined for addresses in this field, it will deliver the mail to a local user, rather than forward it to another mail server. By default, this is all configured addresses on the machine as well as `localhost` within the local domain. You may specify any number of domains or host names separated by commas, or you may provide a full path to a file containing similar entries. The variables `$myhostname` and `$mydomain` may be used to represent those concepts to Postfix automatically. The ability of Postfix to use such variables throughout its configuration files makes it easier to maintain a number of Postfix servers with very similar configurations. This option correlates to the `mydestination` directive.

What trouble to report to the postmaster

Postfix provides the ability to select what types of error messages will be mailed to the designated *postmaster* of the mail server. Assuming you have setup a `postmaster` alias that directs mail to a real person, Postfix will send reports of all of the types of trouble designated here. The available classes are:

`bounce`

When this option is selected, whenever a message is undeliverable, a bounce message (called a *single bounce message* will be sent to the sender of the message and the local postmaster. For the sake of privacy only the headers will be sent in the message to the postmaster. If the first bounce to the sender is returned as undeliverable, a *double bounce message* will be sent to the postmaster with the entire contents of the first single bounce message.

`2bounce`

Causes double bounce messages to be sent to the postmaster.

`delay`

If the delivery of a message is delayed, the postmaster will receive a notice, along with the headers of the delayed message.

`policy`

Notifies the postmaster of messages that were rejected due to a unsolicited commercial email policy restriction. The complete transcript of the SMTP session is sent.

`protocol`

Notifies the postmaster of protocol errors, or client requests that contained unimplemented commands. The complete transcript of the SMTP session is included in the message.

`resource`

Informs the postmaster of undelivered mail due to resource problems, such as a queue file write error.

`software`

Notifies the postmaster of mail not delivered due to software failures.

This option correlates to the `notify_classes` directive, and defaults to reporting only

problems that usually indicate a misconfiguration or serious problem (specifically `resource` and `software`). In some high load environments, altering this to include bounce notifications could lead to a large number of notices.

Other General Options

The lower section of this page is devoted to global options which are less likely to need to be altered (Figure 10.3, Other General Options. In many installations these options will remain at their defaults.

Figure 10.3 Other General Options

Send outgoing mail via

This option configures whether outgoing mail should be delivered directly to the recipients mail server, or if a parent mail gateway should be used as an intermediary. If the server is behind a firewall, behind a network address translating router/gateway, or similar, it may be necessary to use an intermediary server to achieve reliable service. Many mail servers on the Internet will not accept mail from a server that does not have a working DNS entry and a routable IP address, in order to help prevent spam from forged addresses. Also, local network use policy may require the use of an intermediary for logging, virus scanning, or other purposes that require aggregation of outgoing mail traffic onto a central server. This option corresponds to the `relayhost` directive and defaults to sending mail directly.

Address that receives bcc of each message

With this option, an optional email address may be specified that will receive a copy of every message that enters the Postfix system, excluding locally generated bounce messages. This can represent a breach of privacy in many circumstances, and may be illegal in some countries. It is advisable to be especially cautious about utilizing this option. It can be useful in some environments, however, where central archival of email is valuable for legal or technical reasons.

This option correlates to the `always_bcc` directive and defaults to none.

Timeout on handling requests

This option determines how long a Postfix daemon will wait on a request to complete before it assumes the daemon has locked up, at which time the daemon will be killed. This option corresponds to the `daemon_timeout` and defaults to 18000 seconds.

Default database type

This option determines the type of database to use in the `postalias` and `postmap`

commands. This option corresponds to the `default_database_type` directive and the default depends on the OS and installed system libraries at the time of building Postfix. Ordinarily on UNIX systems this will be `hash` or `dbm`.

Default message delivery transport

The term *delivery transport* refers to the protocol, or language, used to deliver the message from one mail server to another. The transport on modern systems is nearly always `smtp`, and this is the default in Postfix, but there are still a few legacy `uucp` systems in use. This option is merely the default choice, when no transport is explicitly selected for the destination in the optional transport table. This option corresponds to the `default_transport` directive.

Sender address for bounce mail

In the event a message double-bounces, or first bounces from the recipient and then bounces from the sender when the first bounce notice is sent, the message will be sent to this address. All messages to this address will be silently discarded. In this way bounce-loops can be avoided. This option correlates to the `double_bounce_sender` and defaults to `double-bounce`. The name may be any arbitrary name, but must be unique.

Number of subdir levels below the queue dir

This option configures the number of subdirectory levels below the configured queue directories will be used by Postfix for mail storage. Because of the design of the traditional UNIX filesystem, which includes UFS used by all modern BSD systems and the Linux ext2 and ext3 filesystems, performance becomes measurably slower when an extremely large number of files are stored in a single directory. Thus, programs that generate a large number of files often provide the ability to split files out to a number of subdirectories to keep lookups fast. This option correlates to the `hash_queue_depth` directive and defaults to 2, which is suitable for most moderate and even relatively large installations. Because the number of directories in use increases the search time for object seeks, using a too high value here can be harmful to performance.

Name of queue dirs split across subdirs

Postfix uses a number of queues to organize messages with varying states and destinations. Each of these queues can be configured to use hashed subdirectories or not. If a queue is selected here, it will be stored in a hashed subdirectory. In some cases, a queue mus not be listed here as performance will be severely impacted, specifically the world-writable mail drop directory. The defer log file directory, on the other hand must be stored in hashed directories or

performance will suffer. This option corresponds to the `hash_queue_names` directive and defaults to `incoming,active,deferred,bounce,defer,flush` and it is rarely necessary or beneficial to alter this configuration.

Max number of `Received:` headers

A message that contains more `Received:` headers than this will bounce. An extremely large number of this header may indicate a mail loop or a misconfigured mail server somewhere in the path of this message. This option correlates to the `hopcount_limit` directive and defaults to 50. This value rarely needs to be altered from its default.

Time in hours before sending a warning for no delivery

If a message cannot be delivered immediately, it will be queued for later delivery. If after this number of hours, the message still cannot be delivered, a warning will be sent to the sender notifying them that the server has been unable to send the message for a specified time. This correlates to the `delay_warning_time` directive and defaults to not sending a warning.

Network interfaces for receiving mail

This option configures the network addresses on which Postfic will accept mail deliveries. By default Postfix will accept mail on every active interface. Here, Postfix will accept the variables discussed earlier. This option configures the `inet_interfaces` directive.

Idle time after internal IPC client disconnects

This option sets the time in seconds after which an internal IPC client disconnects. This allows servers to terminate voluntarily. This feature is used by the address resolution and rewriting clients. This option correlates to the `idle_time` directive and defaults to 100s. This option should probably never need to be altered under normal circumstances.

Timeout for I/O on internal comm channels

This option determines the amount of time in seconds the server will wait for I/O on internal communication channels before breaking. If the timeout is exceeded, the server aborts with a fatal error. This directive corresponds to the `ipc_timeout` directive and defaults to 3600 seconds, or 60 minutes.

Mail system name

This option identifies the mail server system in use to connecting users. It will be used in the `smtpd_banner` which is sent in `Received:` headers, the SMTP greeting banner, and in bounced mail. Some security experts, who promote security through obscurity, suggest anonymizing all server software to prevent potential crackers from being able to identify the software in use on the server. It is probably not the best use of an administrators time or effort in most environments, however, and many other security tactics are more effective, without negatively impacting the ability to track software problems. This option correlates to the `mail_name` directive and defaults to `Postfix`.

Mail owner

This option specifies the owner of the Postfix mail queue, and most of the Postfix daemon processes. This user should be unique on the system, and share no groups with other accounts or own any other files or processes on the system. After binding to the SMTP port (25), postfix can then drop root privileges and become the user specified here for all new daemon processes. Because of this, if the Postfix daemon is ever compromised the exploiter will only have access to mail and a few other files. Obviously it is good to avoid this as well, but it is certainly better than a root exploit which would allow the exploiter to access and alter anything on the system. This option correlates to the `mail_owner` directive and defaults to `postfix`.

Official mail system version

This paremeter configures the version number that will be reported by Postfix in the SMTP greeting banner, among other things. This correlates to the `mail_version` directive and defaults to the version of Postfix that is installed. Once again, security by obscurity promoters may encourage obfuscation of this value.

Time to wait for next service request

A Postfix daemon process will exit after the time specified here, if it does not receive a new request for service during that time. This option corresponds to the `max_idle` directive and defaults to `100s`. This directive does not impact the queue manager daemon process.

Max service requests handled before exiting

This option configures the maximum number of requests that a single Postfix daemon process will answer before exiting. This option configures the `max_use` directive and defaults to

`100`.

Internet hostname of this mail system

This option specifies the Internet host name of the mail server. By default this value will be set to the fully qualified host name of the server, as determined by a call to `gethostname()`. This option sets the `$myhostname` variable which is used in the defaults to many other options.

This option correlates to the `myhostname` directive.

Local Internet domain name

This option corresponds to the `mydomain` directive and defaults to the contents of the `$myhostname` variable minus the first component. This option defines the `$mydomain` variable and is used in a number of other configuration option defaults.

Local networks

Postfix provides a flexible set of options to help prevent UCE, or other unauthorized uses of the mail server. This option defines what networks will be considered to be local by Postfix. The value is used to determine whether a client is a local client or a remote client. Policies can be more relaxed for local clients. This option configures the `mynetworks` directive and defaults to a list of all networks attached to the server. For example, if the server has an IP of 192.168.1.48, and a netmask of 255.255.255.0, all of the 192.168.1.0 network will be considered local. If you would like stricter control, or the ability to treat other network blocks as local clients, you can specify them here in the form of network/mask pairs (i.e., `172.16.0.0/16`. Network/mask pairs may be inserted from a separate file, if preferred, by specifying the absolute path to the file here.

Send postmaster notice on bounce to...

This option configures the user name or email address to whom bounce notices will be sent. This option correlates to the `bounce_notice_recipient` and is set to `postmaster` by default.

Send postmaster notice on 2bounce to...

This option configures the user name or email address to whom second bounce messages will

be sent. This allows an administrator to watch for second bounces warnings more closely than first bounce messages, because first bounces are far more common and less likely to indicate serious problems. The option configures the `2bounce_notice_recipient` directive and defaults to `postmaster`.

Send postmaster notice on delay to...

This option configures where delay warnings will be sent. This option correlates to the `delay_notice_recipient` directive and defaults to `postmaster`.

Send postmaster notice on error to...

Specifies where error warnings will be sent. This option correlates to the `error_notice_recipient` directive and defaults to `postmaster`.

Mail queue directory

This specifies the directory where Postfix will store queued mail. This will also be the root directory for Postfix daemons that run in a chroot environment. The queue is where messages that are awaiting delivery are stored, thus enough space to accommodate your user mail load should be provided in this directory. This option correlates to the `queue_directory` directive and usually defaults to a sensible location for your OS. Many Linux systems will have the mail queue in `/var/spool/mail` or `/var/spool/postfix`.

Lock file dir, relative to queue dir

This option configures the location of the Postfix lock directory. It should be specified relative to the queue directory, and generally will simply be a subdirectory of the queue directory. This option configures the `process_id_directory` directive and defaults to `pid`.

Separator between user names and address extensions

This option specifies the separator character between user names and address extensions. This option correlates to the `recipient_delimiter` directive and defaults to using no delimiter. This option impacts **Canonical Mapping**, **Relocated Mapping** and **Virtual Domains**.

Postfix support programs and daemons dir

This option specifies the directory where Postfix will look for its various support programs and daemons. The directory should be owned by `root`. This option correlates to the `program_directory` directive and defaults vary depending on installation method and OS variant. On many Linux systems this will be `/usr/libexec/postfix`.

Relocated mapping lookup tables

Postfix can provide a relocation notice in response to messages sent to users who no longer receive mail from this server. If enabled, this option specifies the location of the file containing a table of contact information for users who no longer exist on this system. By default this feature is disabled. This option correlates to the `relocated_maps` directive. If enabled a reasonable choice for this option might be `/etc/postfix/relocated`.

Disable kernel file lock on mailboxes

On Sun workstations, kernel file locks can cause problems, because the `mailtool` program holds an exclusive lock whenever its window is open. Users of other OS variants, or Sun systems where no Sun mail software is in use, may ignore this option. This option correlates to the `sun_mailtool_compatibility` directive and defaults to `No`.

Max time to send a trigger to a daemon

This option specifies the maximum amount of time allowed to send a trigger to a Postfix daemon. This limit helps prevent programs from getting hung when the mail system is under extremely heavy load. This option correlates to the `opts_trigger_timeout` directive and defaults to `10s`.

Address Rewriting and Masquerading

Postfix offers a relatively easy to use, and flexible, address rewriting system, allowing it to act as a mail gateway for a large network, or as a gateway between legacy mail systems and the Internet at large (Figure 10.4, Address Rewriting and Masquerading).

Figure 10.4 Address Rewriting and Masquerading

Note

The options on this page are also discussed on the Postfix Configuration - Address Manipulation page at the Postfix homepage. It is worth reading if advanced address rewriting is required in your mail system.

Rewrite "user%domain" to "user@domain"

This option is useful for some legacy systems that used strange address trickery such as, *user%domain@otherdomain*. It is not generally useful in modern environments, but it is not harmful so usually defaults to `Yes`. This option correlates to the `allow_percent_hack` directive.

Rewrite "user" to "user@$mydomain"

This option configures how Postfix will handle an address that has no domain name in the destination. If enabled, it will append the value of `$mydomain` to the address. This option correlates to the `append_at_myorigin` directive and defaults to `Yes`. Because most Postfix components expect addresses to be of the form `user@domain` it is probably never appropriate to disable this feature.

Rewrite "user@host" to "user@host.$mydomain"

This option configures whether simple host addresses will have the value of `$mydomain` appended to them. This option correlates to the `append_dot_mydomain` directive and defaults to `Yes`. Some administrators may find that this explicit rewrite has unexpected consequences, but it is very rarely a problem.

Rewrite "site!user" to "user@site"

Legacy UUCP networks use a different addressing format than modern SMTP systems. This option enables Postfix to convert the old-style address to a modern address for delivery via the standard SMTP protocol. This option configures the `swap_bangpath` directive and defaults to `Yes`.

Send mail with empty recipient to...

The specifies the destination of mail that is undeliverable. Typically, this will be bounce notifications and other error messages. This option correlates to the `empty_address_recipient` directive and defaults to `MAILER-DAEMON`, which by default is simply an alias to `postmaster`.

Address masquerading

Address masquerading is a method whereby hosts behind the gateway mail server may be hidden, and all mail will appear to have originated from the gateway server. If enabled, the host and/or subdomain portion of an address will be stripped off and only the domain specified here will be included in the address. For example, if `$mydomain` is specified here, an outgoing mail from `joe@joesmachine.swelltech.com` would become simply `joe@swelltech.com`, assuming the $mydomain variable contains `swelltech.com`.

This option correlates to the `masquerade_domains` directive and it is disabled by default.

Masquerade exceptions

It is possible to skip over the masquerade rules define above for some user names. The names to be excepted from those rules can be entered here. This option corresponds to the `masquerade_exceptions` directive and by default no exceptions are made.

Mail Aliases

Mail aliases provide a means to redirect mail to local recipients. Specifically, it allows mail destined for a number of different addresses to be delivered to a single mailbox. A common use for this is to direct mail for users like `postmaster` to a real person. This page is divided into two sections. The upper section labeled **Aliases Options** contains the location and format of the alias files that Postfix should use to construct its alias databases and specifies the type of database to use. The lower section provides a list of all configured aliases on the system, and what the alias maps to.

Aliases Options

Alias databases used by the local delivery agent

This option sets the filenames that Postfix will use for local delivery alias translation. The filename

will have a suffix appended to it based on the file type. This option correlates to the `alias_maps` directive and the default is system dependent. Some common defaults include `hash:/etc/aliases` or `hash:/etc/postfix/aliases`. The first part of the entry, preceding the colon, is the type of database to use, which will be one of `hash` for systems with a modern Berkeley DB implementation, `dbm` for older style systems that only have `dbm` available, or `nis` for systems that run NIS. The after-colon portion of the entry is the path to the filename from which the database name is derived. The databases will be built from the contents of the flat files by Postfix on startup, or when running the `newaliases` command.

Alias databases built by Postfix

This option, closely related to the above, specifies the alias database file(s) that are built when the `newaliases` or `sendmail -bi` commands are run. These commands generate the alias database from the flat file in the above option, in order to speed alias lookups performed by Postfix. Because there may be thousands of aliases on a large mail server, importing them into a database is necessary to maintain efficiency. This option correlates to the `alias_database` directive. Defaults are system dependent, but will commonly be the same as the above option, with the appropriate database file suffix appended.

Aliases

This section of the page provides a list of all configured aliases. To edit an alias, click on the name of the alias. To create an alias, click on the **Create a new alias** button and fill in the alias `Name`, and `Alias to...` fields. Whenever the aliases files have been modified, it is necessary to recreate the aliases database files as well in order for the changes to take effect. When using Webmin this step is performed automatically, and no additional steps are required.

Note

If adding aliases from the command line, it is possible to regenerate the aliases database using the command **postalias**. The man page for this command is a useful resource for understanding how aliases databases are handled in Postfix.

Canonical Mapping

Canonical mapping in Postfix is used for modifying mail in the incoming queue, and it alters both the message headers and the message envelope information for local or remote mail. This mapping can be useful to replace login names with *Firstname.Lastname* style addresses, or to clean up odd addresses produced by legacy mail systems.

Canonical Mapping Tables

If you use any canonical mapping tables, they must be specified in the first section of the **Canonical Mapping** module. After defining them, you can edit them from the second section of the module.

Address mapping lookup tables

This option specifies the location of the optional canonical address mapping table file. This mapping is applied to both sender and recipient addresses, in both envelopes and headers. This option configures the `canonical_maps` directive and is disabled by default. Much like the aliases files discussed in the last section, canonical mapping files are specified by a database type and a filename. The accepted database types depend on your operating system, and installed components. Usually `hash` and `dbm` are used as the database type. A common choice for this value, then, might be `hash:/etc/postfix/canonical`.

Tables for RECIPIENT addresses

This parameter configures address mapping only on recipient addresses, and not sender addresses. Mapping is performed on both envelopes and headers. These lookups are performed before the above configured **Address mapping lookup tables**. This option correlates to the `recipient_canonical_maps` directive and is disabled by default.

Tables for SENDER addresses

Similar to the previous option, this configures mapping for sender addresses only, and not recipient addresses. Both envelope and header information is modified. This option correlates to the `sender_canonical_maps` directive and by default is disabled.

Editing Canonical Mappings

Once a filename is selected for any of the canonical mapping tables, it may be edited by clicking the appropriate **Edit...** buttons. A new page will open, listing any existing mappings and allowing creation of new mappings. The format of mappings in all files is the same.

Canonical mappings may seem, on the surface, to be similar to aliases or virtual domains. However, they are quite distinct and are useful for other purposes. While aliases merely make a decision about which user will receive an email, and virtual domains only impact the envelope address, the canonical mapping alters both the envelope address and the SMTP header address. This change can be used to make mail appear to come from a different user or domain, or direct mail to a different user or domain by changing the address on the message.

For example, if I have a number of local subdomains, but would like all mail to appear to originate from a single domain, it is possible to create a canonical mapping to make the translations. In the **Edit a Map** page, the `Name` will be a subdomain that is to be mapped to the domain, such as `@lab.swelltech.com`. The `Mapts to...` value will simply be the domain I'd like this subdomain converted to, `@swelltech.com`. After saving the mapping and applying changes, all outgoing mail from `lab.swelltech.com` will appear to originate from `swelltech.com`.

Virtual Domains

Virtual domains functionality in Postfix provides a means to redirect messages to different locations by altering the message envelope address. The header address is not altered by a virtual domain mapping. While some functionality of virtual domains overlaps with features available in aliases, virtual domains can be used for local or non-local addresses, while aliases can only be used for local address.

Domain mapping lookup tables

Much like aliases tables and canonical mapping tables discussed in the previous sections, this is simply the path to a file containing the mapping tables for virtual domains. This is usually something along the lines of `hash:/etc/postfix/virtual`, and must be converted to a database format for use in Postfix. Webmin will perform the database generation step for you.

Transport Mapping

The term transport refers to the mechanism used to deliver a piece of email. Specifically, SMTP and UUCP are mail transports that are supported by Postfix. Transport mapping can be used for a number of purposes, including SMTP to UUCP gatewaying, operating Postfix on a firewall with forwarding to an internal mail server, etc.

Transport mapping lookup tables

This option configures the path to a file containing one or more transport mappings. These tables are much like the mapping tables discussed already, and are converted to a database and used

by Postfix in the same way. This option correlates to the `transport_maps` directive. This feature is disabled by default. A common value for this option is `/etc/postfix/transport`.

To create a new mapping, first define the mapping file. Then click **Add a mapping**. If your goal is to redirect mail to an protected internal host from Postfix running on a firewall, for example, you could enter the outside domain name into the **Name** field, `swelltech.com` and then enter into the **Maps to...** field the address of the internal machine, `smtp:privatehost.swelltech.com`. To further improve upon this, local delivery on this machine could be disabled, and increased controls over where and to whom mail should be accepted. There are more examples of such a configuration in the tutorial section of this chapter.

Relocated Mapping

Using this option it is possible to notify senders if a local user has moved to another address. For example, if a user leaves an organization but still receives occasional mail at her local address, it may be convenient to notify anyone sending mail to the user of the move and new contact information for that user. Usage is just like the previous types of mappings and so won't be documented specifically here, though and example of a relocated mapping will be given to display the types of information that can be provided by this feature.

As an example, let's say I move from my current company to the far more relaxed atmosphere of the Oval Office. To make sure all of my friends and clients can keep in touch with me, I could provide a relocated mapping with a **Name** of `joe@swelltech.com` with a **Maps to...** of `president@whitehouse.gov`. While this won't redirect mail to me at my new home, it will notify the people trying to contact me that I've changed email addresses. Hopefully they will all update their address books and resend their mail to my new address.

Local delivery

Local delivery is what Postfix does when it reaches the end of all of its list of mappings and access controls, and still finds that the message is allowed and destined for a user on the local machine (i.e., a mapping could potentially send the message elsewhere for final delivery, so all mappings as well as various access checks are performed before reaching this stage). This page configures a number of options relating to how Postfix handles the delivery of mail for local users (Figure 10.5, Local Delivery).

Figure 10.5 Local Delivery

Name of the transport for local deliveries

This configures the name of the transport that will be used for delivery to destination that match

the $mydestination or $inet_interfaces variables. This can be a simple mailbox drop handled by the Postfix local delivery agent, or any appropriate delivery command. This option correlates to the `local_transport` directive and defaults to the defined transport type named `local`.

Shell to use for delivery to external command

If a command shell is required to communicate properly with your chosen local delivery transport, this option selects the shell that will be used. By default no shell is used, and the transport command will be executed directly. However, if the command contains shell meta-characters or shell built-in commands they will be passed to `/bin/sh` or whatever shell you configure here.

A popular choice for this is `smrsh`, or *Sendmail's Restricted Shell*, which is included in recent Sendmail distributions. `smrsh` allows for more precise control over what commands users can execute from their `.forward` files. This option corresponds to the `local_command_shell` and defaults to `/bin/sh`.

Search list for forward

This is a comma-separated list of possible locations for user forward files. Postfix will try each entry in the list until a forward file is found, or until all have been checked and no match is found. The forward file allows users to configure delivery options for themselves, including delivery-time processing by a program like `procmail` as well as forwarding of messages to a different server. A number of variable expansions are performed on the entries. The expansions are currently:

Forward search path variable expansions

`$user`

The user name of the recipient.

`$shell`

The shell of the recipient.

`$home`

Recipient's home directory.

`$recipient`

The full recipient address.

`$extensions`

Recipient address extensions. This is a separate part of the email address, separated by the **Separator between user names and address extensions** defined on the **General Options** page.

`$domain`

The recipient's domain name.

`$local`

The entire local part of the recipient address.

`$recipient_delimiter`

The separation delimiter for the recipient.

Valid mail delivery to external commands

This parameter restricts mail delivery to only those commands specified here. The default is to disallow delivery to commands specified in `:include:` files, and allow execution of commands in `alias` and `forward` files. This option correlates to the `allow_mail_to_command` directive.

Valid mail delivery to external files

This option restricts mail delivery to external files. The default is to disallow delivery to files

specified in `:include:` but to allow delivery to files specified in `aliases` and `forward` files. This option correlates to the `allow_mail_to_files` directive.

Default rights of the local delivery agent

This option configures the privileges that the delivery agent will have for delivery to a file or a command. This option should never be a privileged user or the postfix owner. This option corresponds to the `default_privs` directive and defaults to `nobody`.

Pathname of user mailbox file

When delivering mail locally, Postfix will drop mail in the directory configured here, or in its default mail spool directory. If you wish to use the *maildir* format for mail storage, this value can be appended with a trailing slash. For example, to store mail in the users home directory in the `Maildir` subdirectory, the value would be `Maildir/`. This option correlates to the `home_mailbox` directive and usually defaults to some location under `/var/spool/mail` or `/var/spool/postfix`.

Destination address for unknown recipients

If a message is received for a recipient that does not exist, the message is normally bounced. However, it is possible to instead have the message delivered to an alternate address. This option corresponds to the `luser_relay` directive. Variable expansions matching those discussed for the **Search list for forward** are also valid for this directive.

Spool directory

This option specifies the directory where UNIX-style mailboxes are stored. Defaults vary depending on OS variant and version, but a common choice is `/var/spool/mail`. This option correlates to the `mail_spool_directory` option.

External command to use instead of mailbox delivery

This option defines a command to use for delivery instead of delivering straight to the users mailbox. The command will be run as the recipient of the message with appropriate `HOME`,

`SHELL` and `LOGNAME` environment variables set. This option is commonly used to set up system-wide usage of `procmail`. Beware that if you use a command to deliver mail to all users, you *must* configure an alias for `root`, as the command will be executed with the permissions of the `$default_user`. This option correlates to the `mailbox_command` directive and is disabled by default.

Optional actual transport to use

This option configures the message transport to use for all local users, whether they are in the UNIX passwd database or not. If provided, the value will override all other forms of local delivery, including **Destination address for unknown recipients**. This option corresponds to the `mailbox_transport` directive and is disabled by default. This option may be useful in some environments, for example, to delegate all delivery to an agent like the **cyrus** IMAPD.

Optional transport for unknown recipients

If a user cannot be found in the UNIX passwd database, and no alias matches the name, the message will ordinarily be bounced, or handled via the **Destination address for unknown recipients** option. However, if you would like unknown users to be handled by a separate transport method. This option overrides the **Destination address for unknown recipients** option above. This option correlates to the `fallback_transport` directive and is disabled by default.

Max number of parallel deliveries to the same local recipient

This option limits the number of simultaneous deliveries to a single local recipient. If `.forward` files are allowed for users, a user may run a time-consuming command or shell script, leading to overload caused by several such processes being started up at once. This option correlates to the `local_destination_concurrency_limit` directive and the default is `2`. A low value is recommended for this option, unless it is certain that no complex `.forward` files will be in use.

Max number of recipients per local message delivery

This option configures the maximum number of recipients per local message delivery. This option

correlates to the `local_destination_recipient_limit` and is set to the value of `Max number of recipients per message delivery` by default.

Prepend a Delivered-To: when...

This parameter determines when Postfix should insert a `Delivered-to:` message header. By default Postfix inserts this header when forwarding mail and when delivering to a file. The defaults are recommended, and it is generally preferable not to disable insertion into forwarded mail. This option corresponds to the `prepend_delivered_header` directive.

General resource control

This page provides access to the various memory and process limits for the Postfix processes (Figure 10.6, General resource control. It is rarely necessary to alter the values on this page, except for highly loaded servers or very low resource machines.

Figure 10.6 General resource control

Max size of bounced message

This option limits the amount of the original message content in bytes that will be sent in a bounce notification. This option correlates to `bounce_size_limit` and defaults to 50000.

Max time for delivery to external commands

When delivering mail to an external command (rather than via direct mailbox delivery), Postfix will wait this amount of time for the delivery to complete. If this value is to be set to a high limit (3600s or more) the value of **Timeout for I/O on internal comm channels** in **General Options** must also be increased. This option correlates to the `command_time_limit` directive and defaults to `1000s`.

Max number of Postfix child processes

This option limits the number of child processes that Postfix will spawn. On high load servers the default may be too low, and may need to be raised to as much as `500` or more. More likely, for most environments, 50 is more than adequate and may even be overkill. For example on dial-up, or consumer broadband serving one to ten users, a more appropriate limit might be `10`. If

in doubt, leave it at its default unless it causes problems. This option correlates to the `default_process_limit` directive and defaults to `50`.

Max number of addresses remembered by the duplicate filter

While expanding aliases and `.forward` files Postfix will remember addresses that are being delivered to and attempt to prevent duplicate deliveries to the same address. This option limits the number of recipient addresses that will be remembered. It corresponds to the `duplicate_filter_limit` directive and defaults to `1000`. There is probably no compelling reason to increase this value.

Max attempts to acquire file lock

This option limits the number of attempts Postfix will make when attempting to obtain an exclusive lock on a mailbox or other file requiring exclusive access. It corresponds to the `deliver_lock_attempts` directive and defaults to `20`.

Time in seconds between file lock attempts

Postfix will wait a specified time between attempts to lock a given file, after a failed lock attempt. This option configures the `deliver_lock_delay` directive and defaults to `1s`.

Max attempts to fork a process

If Postfix attempts to fork a new process and fails, due to errors or a lack of available resources, it will try again a specified number of times. This option correlates to the `fork_attempts` directive and defaults to `5`.

Time in seconds between fork attempts

Postfix will try to spawn a new process a specified time after a failed attempt. This option correlates to the `fork_delay` directive and defaults to `1s`.

Max memory used for processing headers

This option limits the amount of memory in bytes that Postfix will use to process message headers. If a message header is too large to fit into the memory specified, the headers that do not fit into memory will be treated as part of the message body. This option correlates to the

`header_size_limit` directive and defaults to `102400`.

Max memory used for handling input lines

This option limits the amount of memory in bytes that Postfix will use to handle input lines. And input line is any line read from an `:include:` or `.forward` file. In order to prevent the mail server from using excessive amounts of memory, it will break up these files into chunks of this length. This option correlates to the `line_length_limit` directive and defaults to `2048`.

Max size of a message

This option limits the size in bytes of a message that will be delivered, including the message envelope information. This limit should be set high enough to support any email messages your users will need to be able to send or receive. This option correlates to the `message_size_limit` directive and defaults to `10240000`.

Max number of messages in the active queue

This option limits the number of messages that can exist in the message queue at any given time. It correlates to the `qmgr_message_active_limit` directive and defaults to `10000`.

Max number of in-memory recipients

This parameter limits the number of in-memory recipient data structures. This memory contains the short-term *dead list*, which indicates a destination was unavailable when last contacted, among other things. This option correlates to the `qmgr_message_recipient_limit` directive and defaults to `1000`.

Min free space in the queue file system

Postfix will refuse mail if the filesystem on which the queue is located has less available space in bytes than the value set in this option. This option correlates to the `queue_minfree` directive and defaults to `0`.

Max time after which stale lock is released

This option configures how old an external lock file may be before it is forcibly removed. This option correlates to the `stale_lock_time` and defaults to `500s`.

Time in seconds between attempts to contact a broken MDT

This option configures the time in seconds between the queue manager attempts to contact an unresponsive mail delivery transport. This option correlates to the `transport_retry_time` and defaults to `60s`.

SMTP server options

This page configures the majority of the options that directly effect the behavior of the SMTP server portion of Postfix, specifically the portions of Postfix that impact how the server behaves towards an SMTP client that connects to the server.

SMTP greeting banner

When a client connects to an SMTP server a *greeting banner* will be sent to the client (note the term *client* in this context is not the end user, but rather the email software program that is being used to make the connection). This option configures the text that will follow the status code in the banner. It is possible to use a number of variable expansions, for example, to display the specific version of the server software, though Postfix does not include the version by default. If configuring this option to be other than the default, you must include `$myhostname` at the start of this line, as it allows Postfix to report and respond to a mailer loop rather than overloading the system with many multiple deliveries. This option correlates to the `smtpd_banner` directive and contains `$myhostname ESMTP $mail_name` by default.

Note

A proposed federal law in the US would make it illegal to send unsolicited commercial email through a mail server if the server included in its SMTP greeting the words `NO UCE`. Since spammers are generally of a criminal mindset anyway, it is unlikely that many of them will respect the new law if it is ever passed. Nonetheless, it is worth mentioning in hopes that sometime soon, all Americans will have legal protection against the stolen resources and time that UCE represents.

Max number of recipients accepted for delivery

This option limits the number of recipients that may be specified in a single message header. It is usually rare for legitimate messages to have an extremely large number of recipients specified in a single message header, but it is often done in UCE messages. The legitimate exception is messages to a mailing list (possibly sent by mailing list software like `majordomo` or `mailman`. This option correlates to the `smtpd_recipient_limit` and defaults to `1000`.

Disable SMTP VRFY command

Normally, the SMTP VRFY command is used to verify the existence of a particular user. However, it is also illegitimately used by spammers to harvest live email addresses. Thus it is sometimes useful to disable this command. This option correlates to `disable_vrfy_command` and defaults to `No`.

Timeout in seconds for SMTP transactions

This option sets the timeout in seconds for a client to respond to the SMTP servers response with an SMTP request. The connection process involves the client opening a connection to the server, the server replies with a greeting, and then the client makes its request. If the client request does not come within the time specified here, the connection will be closed. This option correlates to the `opts_smtpd_timeout` directive and defaults to `300s`.

Timeout before sending 4xx/5xx error response

When sending an error response to a client, the server will sleep a specified time. The purpose of this feature is to prevent certain buggy clients from hitting the server with repeated requests in rapid succession. This option correlates to the `smtpd_error_sleep_time` directive and defaults to `5s`.

Error count for temporarily ignore a client

This option configures the number of errors that a client may generate before Postfix will stop responding to requests for a specified time. Some buggy mail clients may send a large number of requests, while ignoring or responding incorrectly to, the error messages that result. Postfix attempts to minimize the impact of these buggy clients on normal service. This option correlates to the `smtpd_soft_error_limit` and defaults to `10`.

Error count for closing connection

If the number exceeds this limit the connection will be closed. This option correlates to the `smtpd_hard_error_limit` and defaults to `100`.

HELO is required

Enabling this option causes Postfix to require clients to introduce themselves with a `HELO` header at the beginning of an SMTP session. This may prevent some UCE software packages from connecting, though it may also impact other legitimate clients from connecting. This option correlates to the `smtpd_helo_required` and defaults to `No`.

Allow untrusted routing

This option configures whether Postfix will forward messages with *sender-specified routing* from untrusted clients to destinations within the accepted relay domains. This feature closes a sneaky potential loophole in access controls that would normally prevent the server from being an open relay for spammers. If this behavior is allowed, a malicious user could possibly exploit a backup MX mail host into forwarding junk mail to a primary MX server which believes the mail has originated from a local address, and thus delivers it as the spammer intended. This option correlates to the `allow_untrusted_routing` and is disabled by default. Enabling this option should only be done with extreme caution and care to prevent turning your Postfix installation into an open relay.

Restrict ETRN command upon...

The SMTP `ETRN` command is a rather clumsy means for a client that is not always connected to the Internet to retrieve mail from the server. The usage of this command is rather outdated, and rarely used, as POP3 and IMAP are better suited to solve this problem in the general case. This option correlates to the `smtpd_etrn_restrictions` directive and the default is to allow `ETRN` from any host. This option accepts the following directives:

`check_etrn_access maptype:mapname`, `permit_naked_ip_address`, `reject_invalid_hostname`, `check_helo_access maptype:mapname`, `reject_maps_rbl`, `reject_unknown_client`, `permit_mynetworks`,

`check_client_access`, `permit`, `reject`, `warn_if_reject`, and `reject_unauth_pipelining`.

This option, as well as the following three **Restrictions...** options accept one or all of the following values in the text field. Each is described only once here and the specific entry will include the list of accepted directives for the option. The impact of some of these choices depends on configuration performed elsewhere, and could potentially open security holes if not configured carefully.

`permit_mynetworks`

Permit the message if the relevant address (sender or recipient depending on the restriction) is within the local network.

reject_unknown_client

The request will be refused is the client IP has no PTR record in the DNS. This means that a client with an IP address that cannot be resolved to a host name cannot send mail to this host.

`check_client_access maptype:mapname`

This option requires the inclusion of an already configured map, as discussed earlier. This will restrict based on the contents of the map, allowing only clients that are allowed by the map. The map may contain networks, parent domains, or client addresses, and Postfix will strip off unnecessary information to match the client to the level of specificity needed.

`check_sender_access maptype:mapname`

This will restrict based on the contents of the map, allowing only senders that are allowed by the map. The map may contain networks, parent domains, or `localpart@`.

reject_maps_rbl

An RBL is a relay domain black hole list. By testing a reverse domain lookup against a name server that receives a domain black hole list transfer, the server can know if the mail was sent through a known open mail relay. There are a number of free and for-fee services providing black hole data. The largest and longest lasting is the service operated by MAPS, while two new similar services are operated by the Open Relay Database and by Distributed Sender Boycott List. All operated on the principle of allowing administrators to choose to refuse mail sent from open mail relays. If this option is listed, the client will be checked against the available RBL domains, and if any match the mail will be refused.

Note

If using any of the free RBL services on the network, consider donating money, time, or resources to the project maintainers. The projects are generally run by volunteer labor, and using network resources that have been paid for by the maintainers.

`reject_invalid_hostname`

If the client host name is invalid, due to bad syntax, the request will be rejected.

`permit_naked_ip_address`

If the client `HELO` or `EHLO` command contains a naked IP address without the enclosing `[]` brackets as require by the mail RFC, the message will be rejected.

Beware that some popular mail clients send a `HELO` greeting that is broken this way.

`reject_unknown_hostname`

Reject the request if the host name in the client `HELO` command has no A or MX record in the DNS.

`reject_non_fqdn_hostname`

If the client host name is not in the form of a fully-qualified domain name, as required by the RFC, the message will be rejected.

`check_helo_access maptype:mapname`

The server will search the named access database map for the `HELO` host name or parent domains. If the result from the database search is `REJECT` or a `4xx` `text`

or `5xx` `text` error code the message will be refused, while a response of `OK` or `RELAY` or an all numerical response the message will be permitted.

`permit`

This simply permits anything. Generally this will be at the end of a set of restrictions in order to allow anything that has not been explicitly prohibited.

`reject`

Rejects everything. This can be used at the end of a chain of restrictions to prohibit anything that has not be explicitly permitted.

`warn_if_reject`

This is a special option that changes the meaning of the following restriction, so that a message that would have been rejected will be logged but still accepted. This can be used for testing new rules on production mail servers without risk of denying mail due to a problem with the rules.

`reject_unauth_pipelining`

If the client sends commands ahead of time without first confirming that the server support SMTP command pipelining, the message will be rejected. This will prevent mail from some poorly written bulk email software that improperly uses pipelining to speed up mass deliveries.

Restrictions on client hostnames/addresses

This restriction applies to the client host name and/or address. By default, Postfix will allow connections from any host, but you may add additional restrictions using the following:

`reject_unknown_client` , `permit_mynetworks` , `check_client_access` `maptype:mapname` , `reject_maps_rbl` , `maps_rbl_reject_code` , `permit` , `reject` , `warn_if_reject` , `reject_unauth_pipelining` .

Restrictions on sends in HELO commands

This option specifies additional restrictions on what information can be sent by client in the

`HELO` and `EHLO` commands. This option correlates to the `smtpd_helo_restrictions` directive. By default Postfix accepts anything, and the following restrictions may be added: `reject_invalid_hostname`, `permit_naked_ip_address`, `reject_unknown_hostname`, `reject_non_fqdn_hostname`, `check_helo_access` `maptype:mapname`, `reject_maps_rbl`, `reject_unknown_client`, `check_client_access` `maptype:mapname`, `permit`, `reject`, `warn_if_reject`, `reject_unauth_pipelining`.

Restrictions on sender addresses

This option restricts what can be contained in the `MAIL FROM` command in a message. It may be used to prevent specific email addresses from sending mail, reject clients without a resolvable host name, etc. This option correlates to the `smtpd_sender_restrictions` directive and may contain any of the following restrictions: `permit_mynetworks:`, `reject_unknown_client`, `reject_maps_rbl`, `reject_invalid_hostname`, `reject_unknown_hostname`, `reject_unknown_sender_domain`, `check_sender_access` `maptype:mapname`, `check_client_access` `maptype:mapname`, `check_helo_access` `maptype:mapname`, `reject_non_fqdn_hostname`, `reject_non_fqdn_sender`, `reject`, `permit`.

Restrictions on recipient addresses

This parameter places restrictions on the recipients that can be contained in the `RCPT TO` command of a sent message. It can be used to dictate where email may be sent. This option correlates to the `smtpd_recipient_restrictions`, and may contain any of the following restrictions: `permit_mynetworks`, `reject_unknown_client`, `reject_maps_rbl`, `reject_invalid_hostname`, `reject_unknown_hostname`, `reject_unknown_sender_domain`, `check_relay_domains`, `permit_auth_destination`, `reject_unauth_pipelining`, `permit_mx_backup`, `reject_unknown_recipient`, `check_recipient_access`, `check_client_access`, `check_helo_access`, `check_sender_access`, `reject_non_fqdn_hostname`, `reject_non_fqdn_sender`, `reject_non_fqdn_recipient`, `reject`, `permit`.

DNS domains for blacklist lookups

This option configures the optional blacklist DNS servers that will be used for all RBL checks that have been specified in all access restrictions. It may contain any number of servers in a whitespace separated list. These services can be used to help prevent spam, as discussed earlier in this section, with the **Restrict ETRN command upon...** parameter. This option configures the `maps_rbl_domains` directive and is empty, by default.

Restrict mail relaying

This option specifies from which hosts, networks, domains, etc. Postfix will relay email for. This option correlates to the `relay_domains` directive, and defaults to `$mydestination`.

SMTP server response on access map violation, **SMTP server response on RBL domains violation**, **SMTP server response on forbidden relaying**, **SMTP server response on unknown client reject**, **SMTP server response on invalid hostname reject**, **SMTP server response on unknown domain reject**, **SMTP server response on unknown hostname reject**

These options configure the error result code that will be sent to the client when any of the specified restrictions are being applied. These errors have sensible default values and generally should not need to be changed. Consult with RFC 822 if you wish to understand more about the SMTP error codes, or have a reason to change any of these values.

SMTP Client Options

The SMTP client options configures how Postfix will behave when dealing with other mail servers as a client, i.e., when sending mail on behalf of a user. This portion of the configuration primarily dictates how the server will respond to certain error conditions..

Action when listed as best MX server

As discussed in the BIND chapter, a mail server performs a name server query to find the MX, or mail server, record for the destination domain. If this record indicates that the local server *is* the server to which mail should be sent, it can respond in a couple of ways. The default is to bounce the message with an error indicating a mail loop. If the field is selected and `local` is entered, the mail will be directed to the local delivery agent instead of bouncing the mail. This option correlates to the `best_mx_transport` directive.

Hosts/domains to hand off mail to on invalid destination

By default, a mail that cannot be delivered because the destination is invalid will be bounced with an appropriate error message. However, it is possible to configure `postfix` to hand off email to another server instead. This option correlates to the `fallback_relay` directive.

Ignore MX lookup error

If a name server query fails to provide an MX record, the server defaults to deferring the mail and trying again later. If `Yes` is selected instead, an A record query will be done and an attempt will be made to deliver to the resulting address. This option correlates to the `ignore_mx_lookup_error` directive.

Skip 4xx greeting

If a remote server responds to a connection with a 4XX status code, `postfix` will, by default, select the next available mail exchanger specified by the MX records. If set to `No`, mail delivery will be deferred after the first mail delivery attempt and another attempt will be made later. This option correlates to the `smtp_skip_4xx_greeting` directive.

Skip wait for the QUIT command

This option configures whether Postfix will wait for the receiving mail server to respond to the `QUIT` command. This option correlates to the `smtp_skip_quit_response` directive and defaults to no.

Max number of parallel deliveries to the same destination

This option specifies the maximum number of deliveries that Postfix will perform to the same destination simultaneously. This option correlates to the `smtp_destination_concurrency_limit` directive and defaults to the system-wide limit for parellel deliveries configured in the **Delivery Rates** page documented in the next section.

Max number of recipients per delivery

Limits the number of recipients per delivery. This option correlates to the `smtp_destination_recipient_limit` directive and defaults to the system-wide limit for recipients per delivery.

Timeout for completing TCP connections

Specifies the time in seconds that the Postfix delivery agent will wait before timing out a TCP connection. This option correlates to the `smtp_connect_timeout` directive and defaults to 0, which disables connection timeouts.

Timeout on waiting for the greeting banner

Limits how long Postfix will wait for a greeting banner to be received from the destination server. This option corresponds to the `smtp_helo_timeout` directive and defaults to 300 seconds.

Timeout on waiting for answer to MAIL FROM

Sets the timeout in seconds for sending the `SMTP MAIL FROM` command and for receiving the destination servers response. This option correlates to the `smtp_mail_timeout`, and defaults to 300 seconds.

Timeout on waiting for answer to RCPT TO

Sets the timeout in seconds for sending the `SMTP RCPT TO` command and for receiving the destination servers response. This option correlates to the `smtp_rcpt_timeout` directive and defaults to 300 seconds.

Timeout on waiting for answer to DATA

Sets the timeout in seconds sending the `SMTP DATA` command and for receiving the destination servers response. This option correlates to the `smtp_data_init_timeout` and defaults to 120 seconds.

Timeout on waiting for answer to transmit of message content

Specifies the SMTP client timeout in seconds for sending the contents of the message. If the connection stalls for longer than this timeout, the delivery agent will terminate to transfer. This option corresponds to the `smtp_data_xfer_timeout` directive and defaults to 180 seconds.

Timeout on waiting for answer to ending "."

Specifies the SMTP client timeout in seconds for sending the closing SMTP "." and receiving the destination servers reply. This option correlates to the `smtp_data_done_timeout` directive and defaults to 600 seconds.

Timeout on waiting for answer to QUIT

Sets the timeout in seconds sending the `SMTP QUIT` command and for receiving the destination servers response. This option correlates to the `smtp_quit_timeout` and defaults to 300 seconds

Delivery Rates

This page contains the options for setting the default rate and concurrency limits for all Postfix components. These rates can usually be overridden within their respective configuration sections.

Max number of parallel deliveries to the same destination

This option specifies the maximum number of deliveries that Postfix will perform to the same destination simultaneously. This option correlates to the `default_destination_concurrency_limit` directive and defaults to `10`.

Max number of recipients per message delivery

Limits the number of recipients per delivery. This option correlates to the `default_destination_recipient_limit` directive and defaults to `50`.

Initial concurrency level for delivery to the same destination

Specifies the initial number of simultaneous deliveries to the same destination. This limit applies to all SMTP, local, and pipe mailer deliveries. A concurrency of less than two could lead to a single problem email backing up delivery of other mail to the same destination. This option configures the `initial_destination_concurrency` directive and defaults to `5`.

Max time (days) in queue before message is undeliverable

Defines the number of days a message will remain queued for delivery in the event of delivery problems before the message is sent back to the sender as undeliverable. This option configures the `maximal_queue_lifetime` directive and defaults to 5 days.

Min time (secs) between attempts to deliver a deferred message

In the event of a delivery deferral, Postfix will wait the specified amount of time before reattempting delivery. This value also specifies the time an unreachable destination will remain in the destination status cache. This option correlates to the `minimal_backoff_time` directive and defaults to 1000 seconds.

Max time (secs) between attempts to deliver a deferred message

Specifies the maximum amount of time between delivery attempts in the event of a deferred delivery. This option configures the `maximal_backoff_time` directive and defaults to 4000 seconds.

Time (secs) between scanning the deferred queue

Specifies the time in seconds between queue scans by the queue management task. This option correlates to the `queue_run_delay` and defaults to 1000 seconds.

Transports that should not be delivered

This field specifies which delivery transports, if any, of the transports defined in the **Transport Mapping** section will not have their messages sent automatically. Instead the messages for these transports will be queued, and can be delivered manually using the `sendmail -q` command. This option correlates to the `defer_transports` directive, and contains nothing by default.

Debugging features

Postfix has two levels of logging. The first level is the normal `maillog`, which reports on all normal mail activities such as received and sent mail, server errors, shutdowns and startups. The second level is more verbose, and can be tuned to log activity relating to specific SMTP clients, host names, or addresses. This page contains the configuration for the second level of logging.

List of domain/network patterns for which verbose log is enabled

This is a list of patterns or addresses that match the clients, hosts, or addresses whose activity you would like to have more verbose logging for. Values here could be an IP address like `192.168.1.1` or a domain name like `swelltech.com`. This option correlates to the `debug_peer_list` directive and is empty by default.

Verbose logging level when matching the above list

Specifies the level of verbosity of the logging for the activity that matches the above patterns. This option correlates to the `debug_peer_level` directive and defaults to 2. The above field must have at least one value for this debug level to have any impact.

Postfix, Unsolicited Commercial Email and Access Controls

Postfix offers an extremely flexible set of access controls, primarily targeted at preventing unsolicited commercial email from being delivered through the server. In order to construct a suitable set of controls it is necessary to understand the order in which rules are checked, and how they interact. By default Postfix will accept mail for delivery from or to any client on your local network and any domains

that are hosted by Postfix. So, by default, Postfix is not an open relay. This is a good beginning, and may be all that is needed in many environments. However, because UCE is such a nuisance for users and network administrators, it may be worthwhile to implement more advanced filtering. This section will address the basics of the Postfix UCE control features.

Access Control List Order

Every message that enters the `smtpd` delivery daemon will be processed by a number of access control lists and checked against a number of rules to insure that the message is one that the administrator actually wants delivered. The goal for most administrators is to prevent unsolicited commercial email from passing through these rules, yet allow every legitimate email to be delivered. This is a lofty goal, and a delicate balance. No perfect solution exists, as long as people are willing to steal the resources of others for their own commercial gain and go to great lengths to overcome the protections in place to prevent such abuse. However, in most environments it is possible to develop a reasonable set of rules that prevents most spam and allows most or all legitimate mail through unharmed.

It is important to understand the order of processing if complex sets or rules are to be used, as attempting to use a particular rule too early in the chain can lead to subtle errors, or strange mail client behavior. Because not all clients react exactly correctly to some types of refusals, and not all clients create correctly formed SMTP requests, it is not unlikely that a misplaced rule will lock out some or all of your clients from sending legitimate mail. It could also just as easily lead to opening a hole in your spam protections early in the rule set, which would allow illicit mail to pass.

The Postfix UCE controls begin with a couple of simple yes or no checks, called `smtpd_helo_required` and `strict_rfc821_envelopes`, both configured in the **SMTP Server Options** page. The first, if enabled, requires a connecting mail client to introduce itself fully by sending a HELO command. This can stop some poorly designed bulk email programs. The second option requires for the envelope to fit the SMTP specification precisely, thus enforcing complete headers. Though the envelope and HELO can be forged by a bulk mailer, it may stop the more hastily implemented variants (well, how many *good* programmers do you know that write tools to help spammers?).

The next stage is the four SMTP restrictions also found on the **SMTP Server Options** page. These further limit from where and to where mail will be delivered. The order of traversal for these four lists of rules is:

1. **Restrictions on client hostnames/addresses** or `smtpd_client_restrictions`
2. **Restrictions on sends in HELO commands** or `smtpd_helo_restrictions`
3. **Restrictions on sender addresses** or `smtpd_sender_restrictions`

4. **Restrictions on recipient addresses** or `smtpd_recipient_restrictions`

Each of these checks can return `REJECT`, `OK`, or `DUNNO`. If `REJECT`, the message will be refused, and no further rules will be checked. If `OK`, no further rules in the given restriction will be checked, and the next restriction list will be checked. If `DUNNO`, the list will continue to process the current restriction until it gets another result (`OK` or `REJECT`) or until the list end is reached, which is an implicit `OK`. If all lists return `OK`, the message will be passed to the regular expressions checks, otherwise it will be rejected.

Next come the regular expression-based `header_checks` and `body_checks`. These options, if enabled, provide a means to test the actual contents of the headers and the body of the email, respectively. Both operate in the same way, though they should be used somewhat differently. Header checks can be used to prevent well-known spamming domains from sending you email, or for stopping some well-known bulk-mailer software. By entering some signature of the offender, like the domain name, or the X-mailer field identifying the software, the mail can be rejected before the body is even sent. Body checks, though the use the same regular expressions and file format as header checks, should be used more sparingly, as the mail must be accepted before it can be checked. Thus bandwidth is wasted on receipt of the mail, and worse, the server will be occupied for a potentially long while with processing the entire contents of every email. In short, use header checks whenever is convenient, and use body checks only when an effective header check cannot be devised. Only `REJECT` or `OK` are permitted for the returned values.

Note

Webmin, as of this writing (version 1.020), does not provide access to the regular expressions based checks, `header_checks` and `body_checks`. It is likely that a near future version will support these features, however.

Tutorial: Setting up a basic Postfix mail server

As with most of the server software documented here, Postfix has an intimidatingly large number of options and features. But, as we've already seen with BIND and Apache, even complex software can be easy and quick to setup if you know just what to do to get started. Postfix is no different. At the end

of this short section you'll have a fully functioning mail server, capable of sending and receiving mail on behalf of one or more domains.

In most environments, only three configuration details are needed to begin providing mail service with Postfix. First, browse the the **General Options** page of the module. The top two options, **What domain to use in outbound mail** and **What domains to receive mail for**, need to be configured to suit your environment.

For the first option, you will likely want to select `Use domainname` in order to select the domain name of your server as the source of email sent from it. For example, if my mail server is named `mail.swelltech.com` and I selected this option, mail will appear to originate from `swelltech.com`.

The second option specifies the domains for which you will receive email. The default is probably too restrictive in that it will only permit receipt of mail to `$mydomainname` and `localhost.$mydomain`, or the server itself. While this depends on your environment and needs, it is likely that you will want to at least add the `$mydomain` variable to the list of accepted domains.

The last step to making Postfix fully functional for sending and receiving mail is to insure the **Local networks** parameter is set appropriately. If you only have one network block, this will already be set appropriately, as the default is to accept mail for delivery from all attached networks (i.e., all configured and active network addresses). However, if you have a public and private network interface, you'll likely want to remove to the public interface to prevent other clients of your ISP from being able to relay mail through your server.

Click the **Save and Apply** button to make your changes take effect. It is, of course, a good idea to test your changes to make sure things are working as intended. First, assuming an appropriate DNS MX record has already been configured as discussed in the BIND tutorials, you can send yourself an email at the new domain. Watch the `maillog` in the **System Logs** module for errors and to see if the message is delivered as expected. Next configure your mail client to send through your new mail server, to insure it is working for sending mail, as well. The `maillog` will likely give clues about what is wrong in the event of problems.

Tutorial: Virtual Hosting email with Postfix

At this point, if you've performed the configuration in the previous tutorial, you'll be able to accept mail for any number of domains. However, this is not the same as providing independent virtual hosting

support with Postfix, because you can only have one user of a given name and mail sent to that user name at any of the domains for which you accept mail will be delivered to that user. So, for example, if you hosted `swelltech.com`, `penguinfeet.org`, and `nostarch.com` on the same server, and mail was sent to user `joe` at each of those domains, all three mails would end up in the same mailbox. Therefore, you have to introduce another layer to solve this problem.

Postfix has two commonly used methods for solving this problem. The first is the native Postfix method, using a virtual table to direct mail to the correct destination. The second method is modeled after the way Sendmail handles the problem, and is therefore a lot more complex. Because simplicity is better than complexity, you'll learn the native Postfix mechanism exclusively. The Postfix `virtual` man page covers both methods in moderate detail. If you have an older Sendmail installation that is being converted to Postfix you may wish to use the second method and maintain your current virtual mail configuration. If you will be running an extremely large number of virtual domains, it is likely preferable to use the second method, as well.

The first step for setting up virtual domain delivery is for you to create a virtual map table using the **Virtual Domains** page (Figure 10.7, The Virtual Domains Table. Enter the map type (`hash`, `dbm`, etc.), followed by the file name of the flat file that will contain the table information. For example, you could use `/etc/postfix/virtual` for this purpose. This is a pretty common location for this file.

Figure 10.7 The Virtual Domains Table

Save and apply the change, and return to the **Virtual Domains** page. Now, you can click the **New mapping** button. You first have to create a generic map for the new domain. So, for the **Name** field, enter your virtual domain name. In the **Maps to...** field, you can technically enter anything you like (as long as we enter something). The custom seems to be to enter `virtual` in this field, as that is its purpose. Click **Save mapping** to add it to the virtual table.

Next, you'll want to add a `postmaster` alias, as all mail servers must have a functioning `postmaster` address to be compliant with the relevant RFC. So, click **New mapping** again. This time enter `postmaster@virtual.domain` into the **Name** field, where `virtual.domain` is the name of your domain. Then enter `postmaster` into the **Maps to...** field so that mail to this

address will be mapper to the local `postmaster` address for normal delivery.

Finally, you're ready to start adding your virtual domain users to the table. Once again, create a new mapping. Fill in your new virtual domain mail address in the **Name** field. For example, you might fill in `joe@virtual.domain`. In the **Maps to...** section, enter the name of a local user that you would like to receive mail for this address. In this case, you would use `virtual-joe` or perhaps `virtual.domain.joe`. This new local user must exist for mail to be delivered, therefore you'll need to add the new user to the system.

Now, **Save and Apply** your changes, and test it out! The virtual maps can be handled by various database types, or exported to an LDAP database. There is no reasonable limit to the number of virtual users and domains you can have.

This topic: Webmin > PostfixConfiguration
Topic revision: r3 - 05 Oct 2007 - 00:29:55 - MattAlbright