

Section 43 - Message processing

43. Message processing

Exim performs various transformations on the sender and recipient addresses of all messages that it handles, and also on the messages' header lines. Some of these are optional and configurable, while others always take place. All of this processing, except rewriting as a result of routing, and the addition or removal of header lines while delivering, happens when a message is received, before it is placed on Exim's queue.

Some of the automatic processing takes place by default only for "locally-originated" messages. This adjective is used to describe messages that are not received over TCP/IP, but instead are passed to an Exim process on its standard input. This includes the interactive "local SMTP" case that is set up by the `-bs` command line option.

Note: Messages received over TCP/IP on the loopback interface (127.0.0.1 or ::1) are not considered to be locally-originated. Exim does not treat the loopback interface specially in any way.

If you want the loopback interface to be treated specially, you must ensure that there are appropriate entries in your ACLs. 43.1 Submission mode for non-local messages

Processing that happens automatically for locally-originated messages (unless `suppress_local_fixups` is set) can also be requested for messages that are received over TCP/IP. The term "submission mode" is used to describe this state. Submission mode is set by the modifier `control = submission`

in a MAIL, RCPT, or pre-data ACL for an incoming message (see sections 39.17 and 39.18). This makes Exim treat the message as a local submission, and is normally used when the source of the message is known to be an MUA running on a client host (as opposed to an MTA). For example, to set submission mode for messages originating on the IPv4 loopback interface, you could include the following in the MAIL ACL: `warn hosts = 127.0.0.1 control = submission`

There are some options that can be used when setting submission mode. A slash is used to separate options. For example: `control = submission/sender_retain`

Specifying `sender_retain` has the effect of setting `local_sender_retain` true and `local_from_check` false for the current incoming message. The first of these allows an existing Sender: header in the message to remain, and the second suppresses the check to ensure that From: matches the authenticated sender. With this setting, Exim still fixes up messages by adding Date: and Message-ID: header lines if they are missing, but makes no attempt to check sender authenticity in header lines.

When `sender_retain` is not set, a submission mode setting may specify a domain to be used when generating a From: or Sender: header line. For example: `control = submission/domain=some.domain`

The domain may be empty. How this value is used is described in sections 43.11 and 43.16. There is also a name option that allows you to specify the user's full name for inclusion in a created Sender: or From: header line. For example: `accept authenticated = *`

```
control = submission/domain=wonderland.example/
          name=${lookup {$authenticated_id} \
          lsearch {/etc/exim/namelist}}
```

Because the name may contain any characters, including slashes, the name option must be given last. The remainder of the string is used as the name. For the example above, if `/etc/exim/namelist` contains: `bigegg: Humpty Dumpty`

then when the sender has authenticated as `bigegg`, the generated Sender: line would be: `Sender: Humpty Dumpty <bigegg@wonderland.example>`

By default, submission mode forces the return path to the same address as is used to create the Sender: header. However, if `sender_retain` is specified, the return path is also left unchanged.

Note: The changes caused by submission mode take effect after the predata ACL. This means that any sender checks performed before the fix-ups use the untrusted sender address specified by the user, not the trusted sender address specified by submission mode. Although this might be slightly unexpected, it does mean that you can configure ACL checks to spot that a user is trying to spoof another's address. 43.2 Line endings

RFC 2821 specifies that CRLF (two characters: carriage-return, followed by linefeed) is the line ending for messages transmitted over the Internet using SMTP over TCP/IP. However, within individual operating systems, different conventions are used. For example, Unix-like systems use just LF, but others use CRLF or just CR.

Exim was designed for Unix-like systems, and internally, it stores messages using the system's convention of a single LF as a line terminator. When receiving a message, all line endings are translated to this standard format. Originally, it was thought that programs that passed messages directly to an MTA within an operating system would use that system's convention. Experience has shown that this is not the case; for example, there are Unix applications that use CRLF in this circumstance. For this reason, and for compatibility with other MTAs, the way Exim handles line endings for all messages is now as follows:

-
- LF not preceded by CR is treated as a line ending.
-
- CR is treated as a line ending; if it is immediately followed by LF, the LF is ignored.
-

The sequence “CR, dot, CR” does not terminate an incoming SMTP message, nor a local message in the state where a line containing only a dot is a terminator.

-

If a bare CR is encountered within a header line, an extra space is added after the line terminator so as not to end the header line. The reasoning behind this is that bare CRs in header lines are most likely either to be mistakes, or people trying to play silly games.

-

If the first header line received in a message ends with CRLF, a subsequent bare LF in a header line is treated in the same way as a bare CR in a header line. 43.3 Unqualified addresses

By default, Exim expects every envelope address it receives from an external host to be fully qualified. Unqualified addresses cause negative responses to SMTP commands. However, because SMTP is used as a means of transporting messages from MUAs running on personal workstations, there is sometimes a requirement to accept unqualified addresses from specific hosts or IP networks.

Exim has two options that separately control which hosts may send unqualified sender or recipient addresses in SMTP commands, namely `sender_unqualified_hosts` and `recipient_unqualified_hosts`. In both cases, if an unqualified address is accepted, it is qualified by adding the value of `qualify_domain` or `qualify_recipient`, as appropriate.

Unqualified addresses in header lines are automatically qualified for messages that are locally originated, unless the `-bnq` option is given on the command line. For messages received over SMTP, unqualified addresses in header lines are qualified only if unqualified addresses are permitted in SMTP commands. In other words, such qualification is also controlled by `sender_unqualified_hosts` and `recipient_unqualified_hosts`, 43.4 The UUCP From line

Messages that have come from UUCP (and some other applications) often begin with a line containing the envelope sender and a timestamp, following the word “From”. Examples of two common formats are: From a.oakley@berlin.mus Fri Jan 5 12:35 GMT 1996
From f.butler@berlin.mus Fri, 7 Jan 97 14:00:00 GMT

This line precedes the RFC 2822 header lines. For compatibility with Sendmail, Exim recognizes such lines at the start of messages that are submitted to it via the command line (that is, on the standard input). It does not recognize such lines in incoming SMTP messages, unless the sending host matches `ignore_fromline_hosts` or the `-bs` option was used for a local message and `ignore_fromline_local` is set. The recognition is controlled by a regular expression that is defined by the `uucp_from_pattern` option, whose default value matches the two common cases shown above and puts the address that follows “From” into `$1`.

When the caller of Exim for a non-SMTP message that contains a “From” line is a trusted user, the message's sender address is constructed by expanding the contents of `uucp_sender_address`, whose default

value is “\$1”. This is then parsed as an RFC 2822 address. If there is no domain, the local part is qualified with qualify_domain unless it is the empty string. However, if the command line -f option is used, it overrides the “From” line.

If the caller of Exim is not trusted, the “From” line is recognized, but the sender address is not changed. This is also the case for incoming SMTP messages that are permitted to contain “From” lines.

Only one “From” line is recognized. If there is more than one, the second is treated as a data line that starts the body of the message, as it is not valid as a header line. This also happens if a “From” line is present in an incoming SMTP message from a source that is not permitted to send them. 43.5 Resent- header lines

RFC 2822 makes provision for sets of header lines starting with the string Resent- to be added to a message when it is resent by the original recipient to somebody else. These headers are Resent-Date:, Resent-From:, Resent-Sender:, Resent-To:, Resent-Cc:, Resent-Bcc: and Resent-Message-ID:. The RFC says:

Resent fields are strictly informational. They MUST NOT be used in the normal processing of replies or other such automatic actions on messages.

This leaves things a bit vague as far as other processing actions such as address rewriting are concerned. Exim treats Resent- header lines as follows:

-

A Resent-From: line that just contains the login id of the submitting user is automatically rewritten in the same way as From: (see below).

-

If there's a rewriting rule for a particular header line, it is also applied to Resent- header lines of the same type. For example, a rule that rewrites From: also rewrites Resent-From:.

-

For local messages, if Sender: is removed on input, Resent-Sender: is also removed.

-

For a locally-submitted message, if there are any Resent- header lines but no Resent-Date:, Resent-From:, or Resent-Message-Id:, they are added as necessary. It is the contents of Resent-Message-Id: (rather than Message-Id:) which are included in log lines in this case.

-

The logic for adding Sender: is duplicated for Resent-Sender: when any Resent- header lines are present. 43.6 The Auto-Submitted: header line

Whenever Exim generates an autoreply, a bounce, or a delay warning message, it includes the header line: Auto-Submitted: auto-replied

43.7 The Bcc: header line

If Exim is called with the -t option, to take recipient addresses from a message's header, it removes any Bcc: header line that may exist (after extracting its addresses). If -t is not present on the command line, any existing Bcc: is not removed. 43.8 The Date: header line

If a locally-generated or submission-mode message has no Date: header line, Exim adds one, using the current date and time, unless the suppress_local_fixups control has been specified. 43.9 The Delivery-date: header line

Delivery-date: header lines are not part of the standard RFC 2822 header set. Exim can be configured to add them to the final delivery of messages. (See the generic delivery_date_add transport option.) They should not be present in messages in transit. If the delivery_date_remove configuration option is set (the default), Exim removes Delivery-date: header lines from incoming messages. 43.10 The Envelope-to: header line

Envelope-to: header lines are not part of the standard RFC 2822 header set. Exim can be configured to add them to the final delivery of messages. (See the generic envelope_to_add transport option.) They should not be present in messages in transit. If the envelope_to_remove configuration option is set (the default), Exim removes Envelope-to: header lines from incoming messages. 43.11 The From: header line

If a submission-mode message does not contain a From: header line, Exim adds one if either of the following conditions is true:

-

The envelope sender address is not empty (that is, this is not a bounce message). The added header line copies the envelope sender address.

-

The SMTP session is authenticated and `$authenticated_id` is not empty.

-

If no domain is specified by the submission control, the local part is `$authenticated_id` and the domain is `$qualify_domain`.

-

If a non-empty domain is specified by the submission control, the local part is `$authenticated_id`, and the the domain is the specified domain.

-

If an empty domain is specified by the submission control, `$authenticated_id` is assumed to be the complete address.

A non-empty envelope sender takes precedence.

If a locally-generated incoming message does not contain a From: header line, and the `suppress_local_fixups` control is not set, Exim adds one containing the sender's address. The calling user's login name and full name are used to construct the address, as described in section 43.18. They are obtained from the password data by calling `getpwuid()` (but see the `unknown_login` configuration option). The address is qualified with `qualify_domain`.

For compatibility with Sendmail, if an incoming, non-SMTP message has a From: header line containing just the unqualified login name of the calling user, this is replaced by an address containing the user's login name and full name as described in section 43.18. 43.12 The Message-ID: header line

If a locally-generated or submission-mode incoming message does not contain a Message-ID: or Resent-Message-ID: header line, and the `suppress_local_fixups` control is not set, Exim adds a suitable header line to the message. If there are any Resent-: headers in the message, it creates Resent-Message-ID:. The id is constructed from Exim's internal message id, preceded by the letter E to ensure it starts with a letter, and followed by @ and the primary host name. Additional information can be included in this header line by setting the `message_id_header_text` and/or `message_id_header_domain` options. 43.13 The Received: header line

A Received: header line is added at the start of every message. The contents are defined by the `received_header_text` configuration option, and Exim automatically adds a semicolon and a timestamp to the configured string.

The Received: header is generated as soon as the message's header lines have been received. At this stage, the timestamp in the Received: header line is the time that the message started to be received. This is the value that is seen by the DATA ACL and by the `local_scan()` function.

Once a message is accepted, the timestamp in the Received: header line is changed to the time of acceptance, which is (apart from a small delay while the -H spool file is written) the earliest time at which delivery could start. 43.14 The References: header line

Messages created by the autoreply transport include a References: header line. This is constructed according to the rules that are described in section 3.64 of RFC 2822 (which states that replies should contain such a header line), and section 3.14 of RFC 3834 (which states that automatic responses are not different in this respect). However, because some mail processing software does not cope well with very long header lines, no more than 12 message IDs are copied from the References: header line in the incoming message. If there are more than 12, the first one and then the final 11 are copied, before adding the message ID of the incoming message. 43.15 The Return-path: header line

Return-path: header lines are defined as something an MTA may insert when it does the final delivery of messages. (See the `generic_return_path_add` transport option.) Therefore, they should not be present in messages in transit. If the `return_path_remove` configuration option is set (the default), Exim removes Return-path: header lines from incoming messages. 43.16 The Sender: header line

For a locally-originated message from an untrusted user, Exim may remove an existing Sender: header line, and it may add a new one. You can modify these actions by setting the `local_sender_retain` option true, the `local_from_check` option false, or by using the `suppress_local_fixups` control setting.

When a local message is received from an untrusted user and `local_from_check` is true (the default), and the `suppress_local_fixups` control has not been set, a check is made to see if the address given in the From: header line is

the correct (local) sender of the message. The address that is expected has the login name as the local part and the value of `qualify_domain` as the domain. Prefixes and suffixes for the local part can be permitted by setting `local_from_prefix` and `local_from_suffix` appropriately. If `From:` does not contain the correct sender, a `Sender:` line is added to the message.

If you set `local_from_check` false, this checking does not occur. However, the removal of an existing `Sender:` line still happens, unless you also set `local_sender_retain` to be true. It is not possible to set both of these options true at the same time.

By default, no processing of `Sender:` header lines is done for messages received over TCP/IP or for messages submitted by trusted users. However, when a message is received over TCP/IP in submission mode, and `sender_retain` is not specified on the submission control, the following processing takes place:

First, any existing `Sender:` lines are removed. Then, if the SMTP session is authenticated, and `$authenticated_id` is not empty, a sender address is created as follows:

-

If no domain is specified by the submission control, the local part is `$authenticated_id` and the domain is `$qualify_domain`.

-

If a non-empty domain is specified by the submission control, the local part is `$authenticated_id`, and the the domain is the specified domain.

-

If an empty domain is specified by the submission control, `$authenticated_id` is assumed to be the complete address.

This address is compared with the address in the `From:` header line. If they are different, a `Sender:` header line containing the created address is added. Prefixes and suffixes for the local part in `From:` can be permitted by setting `local_from_prefix` and `local_from_suffix` appropriately.

Note: Whenever a `Sender:` header line is created, the return path for the message (the envelope sender address) is changed to be the same address, except in the case of submission mode when `sender_retain` is specified.

43.17 Adding and removing header lines in routers and transports

When a message is delivered, the addition and removal of header lines can be specified in a system filter, or on any of the routers and transports that process the message. Section 42.6 contains details about modifying headers in a system filter. Header lines can also be added in an ACL as a message is received (see section 39.19).

In contrast to what happens in a system filter, header modifications that are specified on routers and transports apply only to the particular recipient addresses that are being processed by those routers and transports. These changes do not actually take place until a copy of the message is being transported. Therefore, they do not affect the basic set of header lines, and they do not affect the values of the variables that refer to header lines.

Note: In particular, this means that any expansions in the configuration of the transport cannot refer to the modified header lines, because such expansions all occur before the message is actually transported.

For both routers and transports, the result of expanding a `headers_add` option must be in the form of one or more RFC 2822 header lines, separated by newlines (coded as `“\n”`). For example: `headers_add = X-added-header: added by $primary_hostname\n`

`X-added-second: another added header line`

Exim does not check the syntax of these added header lines.

The result of expanding `headers_remove` must consist of a colon-separated list of header names. This is confusing, because header names themselves are often terminated by colons. In this case, the colons are the list separators, not part of the names. For example: `headers_remove = return-receipt-to:acknowledge-to`

When `headers_add` or `headers_remove` is specified on a router, its value is expanded at routing time, and then associated with all addresses that are accepted by that router, and also with any new addresses that it generates. If an address passes through several routers as a result of aliasing or forwarding, the changes are cumulative.

However, this does not apply to multiple routers that result from the use of the `unseen` option. Any header modifications

that were specified by the “unseen” router or its predecessors apply only to the “unseen” delivery.

Addresses that end up with different `headers_add` or `headers_remove` settings cannot be delivered together in a batch, so a transport is always dealing with a set of addresses that have the same header-processing requirements.

The transport starts by writing the original set of header lines that arrived with the message, possibly modified by the system filter. As it writes out these lines, it consults the list of header names that were attached to the recipient address(es) by `headers_remove` options in routers, and it also consults the transport's own `headers_remove` option. Header lines whose names are on either of these lists are not written out. If there are multiple instances of any listed header, they are all skipped.

After the remaining original header lines have been written, new header lines that were specified by routers's `headers_add` options are written, in the order in which they were attached to the address. These are followed by any header lines specified by the transport's `headers_add` option.

This way of handling header line modifications in routers and transports has the following consequences:

-

The original set of header lines, possibly modified by the system filter, remains “visible”, in the sense that the `$header_xxx` variables refer to it, at all times.

-

Header lines that are added by a router's `headers_add` option are not accessible by means of the `$header_xxx` expansion syntax in subsequent routers or the transport.

-

Conversely, header lines that are specified for removal by `headers_remove` in a router remain visible to subsequent routers and the transport.

-

Headers added to an address by `headers_add` in a router cannot be removed by a later router or by a transport.

-

An added header can refer to the contents of an original header that is to be removed, even it has the same name as the added header. For example: `headers_remove = subject`
`headers_add = Subject: new subject (was: $h_subject:)`

Warning: The `headers_add` and `headers_remove` options cannot be used for a redirect router that has the `one_time` option set. 43.18 Constructed addresses

When Exim constructs a sender address for a locally-generated message, it uses the form
`<user name> <login@qualify_domain>`

For example: Zaphod Beeblebrox `<zaphod@end.univ.example>`

The user name is obtained from the `-F` command line option if set, or otherwise by looking up the calling user by `getpwuid()` and extracting the “gecos” field from the password entry. If the “gecos” field contains an ampersand character, this is replaced by the login name with the first letter upper cased, as is conventional in a number of operating systems. See the `gecos_name` option for a way to tailor the handling of the “gecos” field. The `unknown_username` option can be used to specify user names in cases when there is no password file entry.

In all cases, the user name is made to conform to RFC 2822 by quoting all or parts of it if necessary. In addition, if it contains any non-printing characters, it is encoded as described in RFC 2047, which defines a way of including non-ASCII characters in header lines. The value of the `headers_charset` option specifies the name of the encoding that is used (the characters are assumed to be in this encoding). The setting of `print_topbitchars` controls whether characters with the top bit set (that is, with codes greater than 127) count as printing characters or not. 43.19 Case of local parts

RFC 2822 states that the case of letters in the local parts of addresses cannot be assumed to be non-significant. Exim preserves the case of local parts of addresses, but by default it uses a lower-cased form when it is routing, because on most Unix systems, usernames are in lower case and case-insensitive routing is required. However, any particular router

can be made to use the original case for local parts by setting the `caseful_local_part` generic router option.

If you must have mixed-case user names on your system, the best way to proceed, assuming you want case-independent handling of incoming email, is to set up your first router to convert incoming local parts in your domains to the correct case by means of a file lookup. For example: `correct_case`:

```
driver = redirect
domains = +local_domains
data = ${lookup{$local_part}cdb\
    {/etc/usercased.cdb}{$value}fail}\
    @$domain
```

For this router, the local part is forced to lower case by the default action (`caseful_local_part` is not set). The lower-cased local part is used to look up a new local part in the correct case. If you then set `caseful_local_part` on any subsequent routers which process your domains, they will operate on local parts with the correct case in a case-sensitive manner.

RFC 2822 forbids empty components in local parts. That is, an unquoted local part may not begin or end with a dot, nor have two consecutive dots in the middle. However, it seems that many MTAs do not enforce this, so Exim permits empty components for compatibility. 43.21 Rewriting addresses

Rewriting of sender and recipient addresses, and addresses in headers, can happen automatically, or as the result of configuration options, as described in chapter 31. The headers that may be affected by this are `Bcc:`, `Cc:`, `From:`, `Reply-To:`, `Sender:`, and `To:`.

Automatic rewriting includes qualification, as mentioned above. The other case in which it can happen is when an incomplete non-local domain is given. The routing process may cause this to be expanded into the full domain name. For example, a header such as `To: hare@teaparty`

might get rewritten as `To: hare@teaparty.wonderland.fict.example`

Rewriting as a result of routing is the one kind of message processing that does not happen at input time, as it cannot be done until the address has been routed. Strictly, one should not do any deliveries of a message until all its addresses have been routed, in case any of the headers get changed as a result of routing. However, doing this in practice would hold up many deliveries for unreasonable amounts of time, just because one address could not immediately be routed. Exim therefore does not delay other deliveries when routing of one or more addresses is deferred.