Routers

Routers work on recipient email addresses, either by rewriting them or by assigning them to a transport and sending them on their way. A particular router can have multiple instances, each with different options.

You specify a sequence of routers. A message starts with the first router and progresses through the list until the message is either accepted or rejected. The accepting router typically hands the message to a transport driver. Routers handle both incoming and outgoing messages. They feel a bit like subroutines in a programming language.

A router can return any of the following dispositions for a message:

- accept – the router accepts the address and hands it to a transport driver
- pass – this router can't handle the address; go on to the next router
- decline – router chooses not to handle the address; next router, please!
- fail – the address is invalid; router queues it for a bounce message
- defer – leaves the message in the queue for later
- error – there is an error in the router specification; message is deferred

If a message receives a pass or decline from all the routers in the sequence, it is unroutable. Exim bounces or rejects such messages, depending on the context.

If a message meets the preconditions for a router and the router ends with a no_more statement, then that message will not be presented to any additional routers, regardless of its disposition by the current router. For example, if your remote SMTP router has the precondition domains = !+local_domains and has no_more set, then only messages to local users (that is, those that would fail the domains precondition) will continue to the next router in the sequence.

Routers have many possible options; some common examples are preconditions, acceptance or failure conditions, error messages to return, and transport drivers to use.

The next few sections detail the routers called accept, dnslookup, manualroute, and redirect. The example configuration snippets assume that **exim** is running on a local machine in the example.com domain. They're all pretty straightforward; refer to the documentation if you want to use some of the fancier routers.

The accept router

The accept router labels an address as OK and passes the associated message to a transport driver. Below are examples of accept router instances called localusers for delivering local mail and save_to_file for appending to an archive.

```
localusers:
    driver = accept
    domains = example.com
    check_local_user
    transport = my_local_delivery
```