The Exim Mail Transfer Agent

Part 6: Advanced and more detailed stuff

Large installations

- Use a local name server with plenty of memory
- Exim is limited by disk I/O
  - Use fast disk hardware; evaluate hardware/OS/filesystem
  - With Reiserfs, disable disk block sharing
  - Put hints on RAM disk; spool and log files on different disks
  - Disable **msglog** files, **rejectlog**; set **split_spool_directory**
  - Use multiple directories for user mailboxes
- Avoid linear password files
- Use maildir format to allow parallel deliveries
- Plan to expand “sideways” with parallel servers
  - This also helps add more disk access capacity
- Separate incoming and outgoing mail
- Keep output queue as short as possible
  - Use fallback hosts and/or **$message_age** for several levels

Multilevel queueing (3 levels)

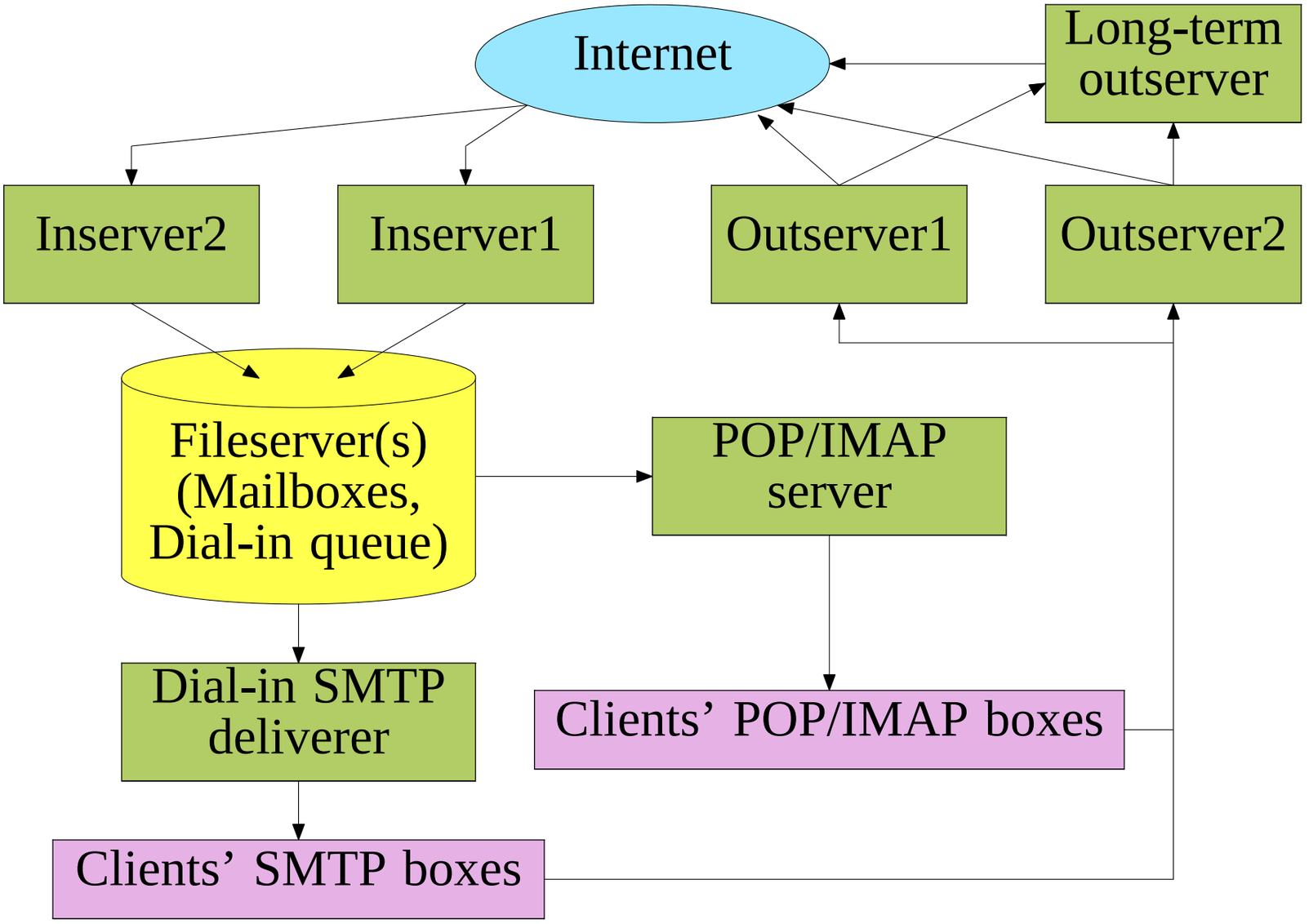
- Frontline host uses

```
remote_smtp:
  driver = smtp
  fallback_hosts = fb11.example:\
                   fb12.example:\
                   ...
  hosts_randomize
```

- First-level fallback hosts have this router first

```
too_old:
  driver = manualroute
  condition = ${if > {$message_age}{21600}\
                {yes}{no}}
  transport = remote_smtp
  route_list = * fb21.example:\
                 fb22.example:\
                 ...
```

Separating mail functions

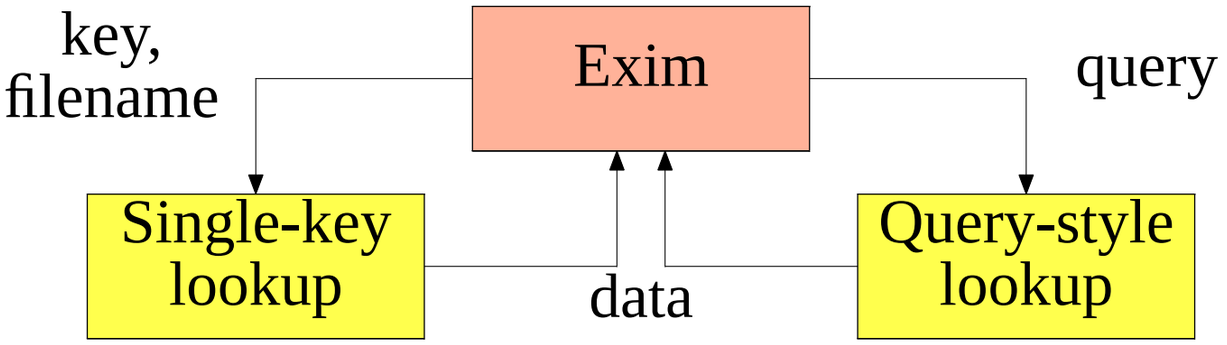


Data lookups (1)

lsearch	linear search	Single-key lookups
dbm	keyed file (choice of library)	
cdb	keyed read-only file	
nis	NIS lookup	
dsearch	directory search	
wildlsearch	linear search with wildcards	Single-key wild lookups
iplsearch	linear search with CIDR IP addresses	
nisplus	NIS+ lookup	Query-style lookups
ldap	LDAP lookup (URL format queries)	
mysql	MySQL lookup	
pgsql	PostgreSQL lookup	
oracle	Oracle lookup	
passwd	Password data lookup	
dnsdb	DNS lookup	
whoson	Whoson lookup	

Data lookups (2)

- Internal API between Exim and the lookup modules



- Lookup is a “black box” to the rest of Exim
- No defaults or partial matches at this level
  - They are implemented for single-key lookups by repeated calls from Exim to the lookup module
- Data from lookups used as indexes in lists is not used
  - `hold_domains = cdb;/etc/hold.cdb`
- Exception: **$domain_data** and **$local_part_data** from **domains** and **local_parts** router options and ACLs

Single-key lookups (1)

- Be aware of the difference between
  ```
  queue_domains = /etc/qdomains
  ```
  (not a lookup)
  ```
  queue_domains = lsearch;/etc/qdomains
  ```
- **lsearch** does not work with wildcard keys such as
  ```
  ^seat\d+\.example\.com
  ```
- **wildlsearch** does support wildcard keys
  - But cannot be converted to another lookup type
- Defaults (e.g. **cdb***) require multiple calls to lookup code
  - First look up the given key; if not found look up "*"
  - In **lsearch** files, put default entries first for efficiency
- Address defaults (e.g. **dbm*@**) use three lookup calls
  - First look up the whole key (e.g. **user@domain**)
  - If not found, replace local part by "*" (e.g. ***@domain**)
  - If not found, look up "*"

Single-key lookups (2)

- Partial lookups are available for domains and host names
  - The lookup type is preceded by “partial-” (e.g. **partial-cdb**)
- Multiple calls to lookup code
  - First look up the whole name (e.g. **abc.example.com**)
  - If not found, add “*” (e.g. ***.abc.example.com**)
  - If not found, remove first component (e.g. ***.example.com**)
  - Continue if more than two components
- The minimum number of fixed components can be set
  - **partial3-lsearch** requires three; **partial0-** continues to “*”
- ACL example

```
deny domains = partial-lsearch;/etc/baddoms
```

- The file might contain lines like this:

```
adomain.example.com
*.bdomain.example.com
```

Query-style lookups

- Lookup code is handed a complete database query

```
accept hosts = nisplus;\
  [hostip=$sender_host_address],relay.org_dir
accept hosts = nisplus;\
  [host=$sender_host_name],relay.org_dir
```

- The use of **$sender_host_name** forces a reverse DNS lookup
  - Avoid this if you can (slows performance; may fail)
- Multiple values are returned as *name=value* pairs
- No defaults or partial matching are implemented in Exim
  - Use the facilities of the query language
- In expansion strings, use nested queries for defaults

```
${lookup mysql{...query1...}{$value}{\
  ${lookup mysql{...query2...}}\
}}
```

LDAP example

- Queries are in URL format (RFC 2255)

```
ldap_aliases:
  driver = redirect
  data = ${lookup ldap {ldap:///cn=\
    $local_part,o=SomeOrg,c=UK?inbox?base?}}
```

- List of servers in **ldap_default_servers**

  Or put host (and port) into the query

  `ldap://ldaphost.example:1234/...`

- Parameters for the connection can be given first

```
data = ${lookup ldap{user="cn=manager,\
  o=UniCamb,c=UK" pass=secret ldap:///o=\
  UniCamb,c=UK?sn?(cn=foo)}}
```

- SIZE sets a limit for the number of entries returned
- TIME sets a timeout for the query

SQL example

- Queries are SQL statements

```
mysql_aliases:
  driver = redirect
  data = ${lookup mysql {select newaddress \
    from aliases where id='$local_part'}}
```

- **mysql_servers** must be set

```
hide mysql_servers = \
       localhost/users/root/secret:\
       otherhost/users/root/private
```

- "Hide" stops **-bP** from showing the data to non-admins
- Backups are for redundancy, not "either/or" lookups
- Similarly for PostgreSQL and Oracle queries

  In MySQL (only) the database name can appear in the query

```
select newaddress from users.aliases ...
```

Quoting in query-style lookups

- External data (e.g. local parts) must always be quoted
- Each query-style lookup has its own quoting rules
- Special expansions are provided

```
${quote_nisplus:xxxx}
${quote_ldap:xxxx}
${quote_mysql:xxxx}
${quote_pgsql:xxxx}
...
```

- Example:

```
data = ${lookup pgsql {select mailbox from \
  accounts where id=\
    '${quote_pgsql:$local_part}'}}
```

Message munging and SPAM checking

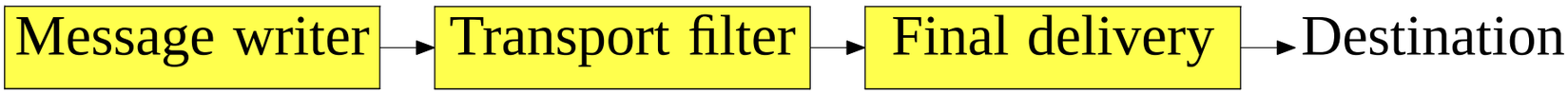
- People want to: Add disclaimers (sometimes very silly ones)
  Strip attachments
  Play around with header lines
  Check for viruses and SPAM
- None of this is really the job of the MTA
  - Best done by auxiliary programs
- Some SPAM checking can be done in filters (system or user)
- Adding and removing header lines can be done
  - In an ACL (adding only)
  - In a system filter (the **headers** command)
  - In routers and transports (**headers_add** and **headers_remove**)
- Several ways of using an external program or script
  - Via Exiscan at ACL time
  - Via **local_scan()** at message reception time
  - Via a transport filter
  - Deliver to a process that re-injects the message

Transport filters

- Scan or modify a message as it is delivered

```
remote_smtp:
  driver = smtp
  transport_filter = /usr/local/outfilter
```

- Must take care not to break RFC 2822 syntax
- Requires two additional processes



- Message writer assembles message
- Final delivery handles transport requirements (CRLF etc.)
- May be a problem with SIZE on **smtp** transports

Passing messages through external programs (1)

- Requirements
  - Must preserve the envelope sender
  - Must preserve the recipients
  - Message must be identifiable afterwards
- If the scanning program runs on a different host
  - SMTP preserves the sender and recipients
  - Processed messages are those that come from that host
- If the scanning program runs on the same host (see next slide)
  - Use batch SMTP (BSMTP) to preserve envelopes
  - Set **batch_max** to ensure many recipients per delivery
  - Use **$received_protocol** to identify processed messages
- Message is returned to Exim by running

```
exim -bS -oMr scanned
```

- The scanning program must run as a trusted user

```
trusted_users = scanuser
```

Passing messages through external programs (2)

- Make this the first router

```
scan:
  driver = accept
  transport = send_to_scanner
  condition = ${if eq {$received_protocol}\
                {scanned}{no}{yes}}
  no_verify
```

- The external program can be run as a transport filter

```
send_to_scanner:
  driver = pipe
  batch_max = 1000
  use_bsmtp
  command = /usr/sbin/exim -oMr scanned -bS
  transport_filter = /usr/bin/spamc -s 5000
  user = scanuser
```

Intermittently connected hosts

- Exim is not really designed for this

- On the server
  - Mixing different kinds of queue is not ideal
  - Workable on a small scale
  - Set large retry times; use **-R** (via ETRN or otherwise)
  - On large systems, it is better to “deliver” to holding storage
    - (example on next slide)

- On the client
  - Use **queue_domains** to delay delivery to specified domains
  - Use **-qf** to force delivery when online

Saving mail for onward delivery

- Avoids contaminating Exim's queue
- Allows time and space control
- Use BSMTP to preserve envelopes

```
bsmtp_router:
  driver = manualroute
  transport = bsmtp_transport
  route_list = *.bsmtp.domains.example

bsmtp_transport:
  driver = appendfile
  batch_max = 1000
  use_bsmtp
  use_crlf                          (optional)
  file = /var/bsmtp/$domain         (or use maildir format)
  user = mail
```

- Note the use of **manualroute** with a local transport

Variable Envelope Return Paths

- VERP encodes the recipient address as part of the sender
  Requires a separate copy for each recipient

```
remote_smtp:
  driver = smtp
  max_rcpt = 1
  return_path = ${if match {$return_path}\
    {\N^(.+?)-request@dom.ex$\N}\
    {${quote_local_part:\
    $1-request=$local_part%$domain}\
    @dom.ex}fail}
```

Original return path:	**alist-request@dom.ex**
Recipient:	**user@dom2.ex**
New return path:	**alist-request=user%dom2.ex@dom.ex**

- Remember to set up a configuration for handling the bounces!

Local part prefixes and suffixes

- Useful for recognizing VERP bounces

```
verp:
  driver = accept
  local_part_suffix = -request=*
  require_files = /etc/lists/$local_part
  transport = verp_bounce
```

- Suffix is accessible in **$local_part_suffix**

If local part is	**alist-request=user%dom2.ex**
$local_part is	**alist**
$local_part_suffix is	**-request=user%dom2.ex**

Multiple user mailboxes

- Another use for local part prefixes and suffixes

```
userforward:
  driver = redirect
  check_local_user
  file = $home/.forward
  allow_filter
  local_part_suffix = -*
  local_part_suffix_optional
```

- Filter could contain

```
if $local_part_suffix is -exim then
  save $home/mail/exim
elsif $local_part_suffix is -pcre then
  save $home/mail/pcre
endif
...
```

Automatic responses (**autoreply** transport)

- Used with no options for replies created by filter files
- Can also create its own messages

```
warning_t:
  driver = autoreply
  user = exim
  file = /usr/local/mail/warning.txt
  file_expand
  from = postmaster@example.com
  to = $sender_address
  subject = Re: Your mail to \
            $local_part@$domain
```

- Could be used like this

```
auto_warning:
  driver = accept
  <some conditions>
  transport = warning_t
  unseen
```

String expansions

- Variable and header line substitutions
- String operations: substrings, hashing, IP address masking, character substitution, regex substitution (like Perl “s”), quoting for regex and lookup queries
- Simple arithmetic (plus, minus, multiply, divide)
- Conditional expansion: string matching, numerical tests, combining conditions with “and” and “or”
- Lookups are another form of conditional
- Password checking using **crypt()**, PAM, Radius, LDAP, Cyrus pwcheck or Cyrus saslauthd
- Calling other programs; reading from files and sockets

Exim is available from

ftp://ftp.csx.cam.ac.uk/pub/software/email/exim/...

.../exim4/exim-4.*xx***.tar.gz** (or **.bz2**) is the latest release

- GNU General Public Licence
- ASCII documentation included
- PostScript, PDF, Texinfo, and HTML are also available
- FAQ in ASCII and HTML with keyword-in-context index
- See also: **http://www.exim.org**
- Discussion list: **exim-users@exim.org**
- Announce list: **exim-announce@exim.org**
- Indexed archive: **http://www.exim-users.org**