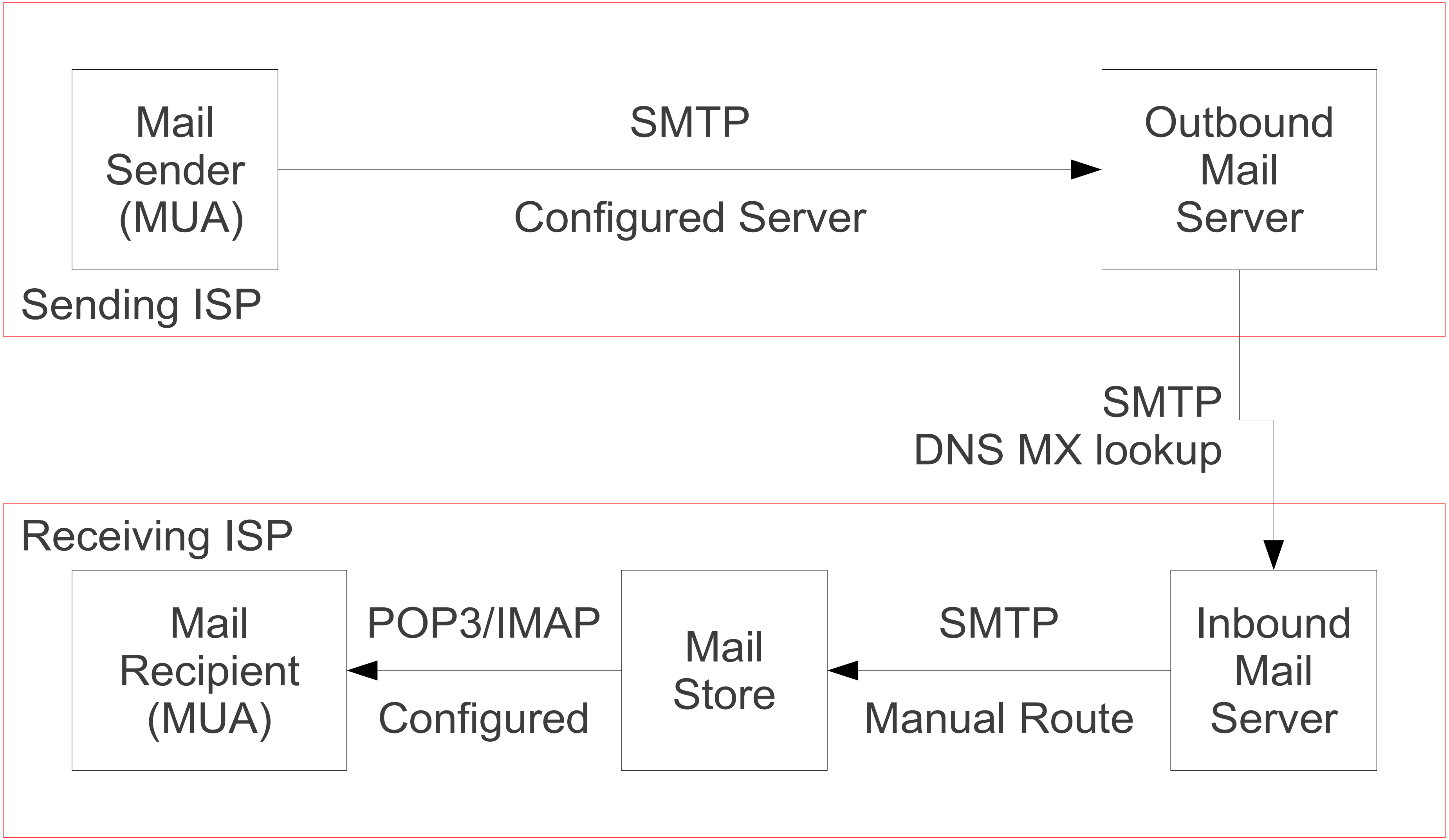
Exim and Internet Mail

Chris Wilson
Aptivate Ltd, UK
AfNOG 2013

Download this presentation at:
http://www.ws.afnog.org/afnog2013/sse/exim

How Internet Email Works



Mail Sender (MUA)

SMTP

Configured Server

Outbound Mail Server

Sending ISP

SMTP

DNS MX lookup

Receiving ISP

Mail Recipient (MUA)

POP3/IMAP

Configured

Mail Store

SMTP

Manual Route

Inbound Mail Server

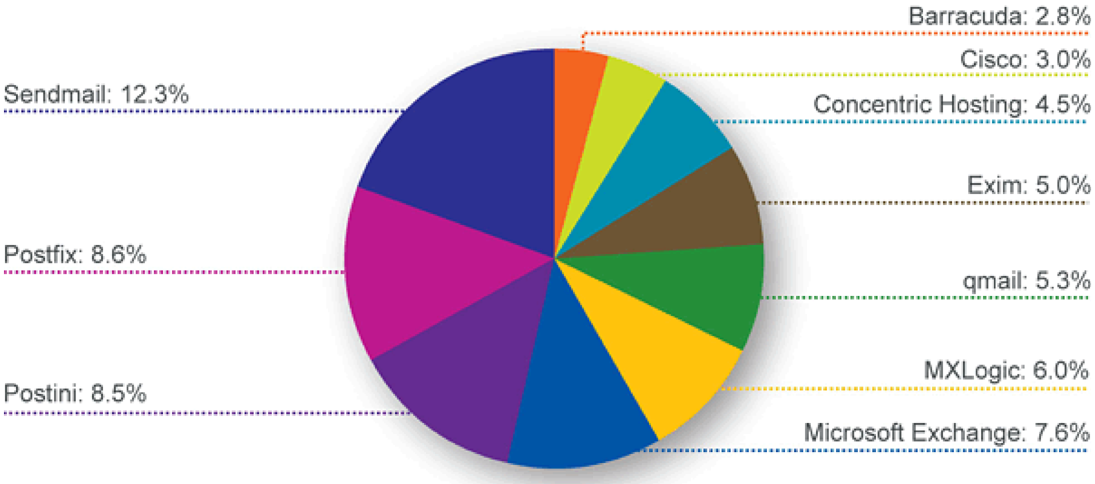
What is Exim

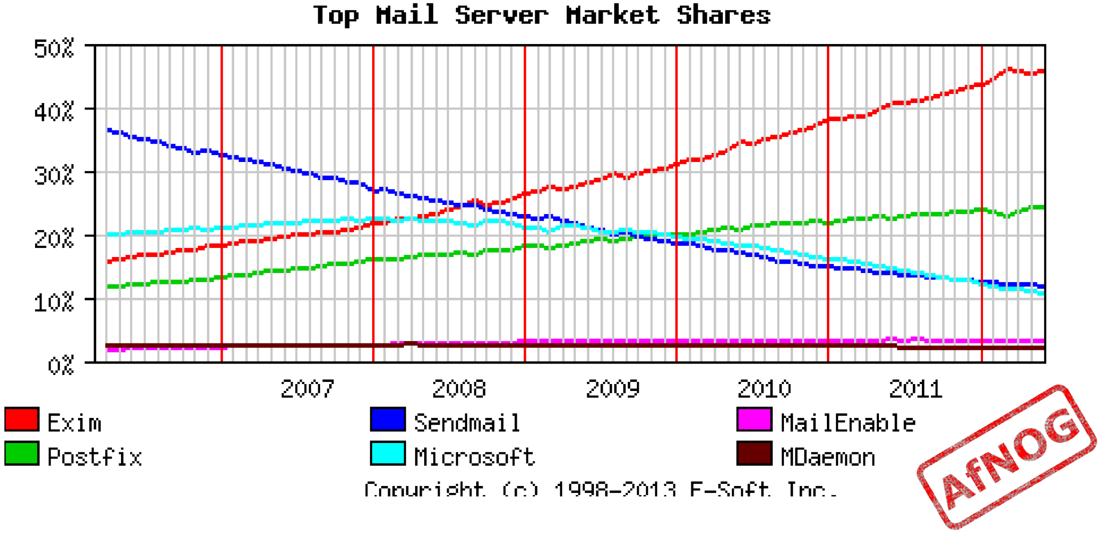
- Listens on port 25 (smtp)
- Accepts mail
- Queues mail
- Delivers it somewhere
  - Using SMTP, LMTP, LDA, mbox or maildir
- No POP, IMAP, calendars, to-do lists, Crackberry!

AfNOG

Who uses Exim

- Most popular public-facing MX in the world!
  - According to one company, results differ!





Copyright (c) 1998-2013 E-Soft, Inc.

AfNOG

Why use Exim

- Flexible (lots of features)
- Reasonably secure
- Reasonably scalable
- Good debugging options
- Sane (but complex) configuration syntax

Why not to use Exim

- Not every problem is a nail
- Simplicity? Use postfix or qmail
- Top security? Use qmail
- Faster delivery? Use postfix or sendmail
- Insane configuration file? Use sendmail
- Note: Exim is not designed for spooling large amounts of mail and not very good at it

Conventions

- File names and technical terms are in *italics*
- Commands to type are shown in monospaced bold italic purple type:

  -
    ```
    cat /etc/monospaced/bold/italic/purple
    ```
- Long command lines are wrapped, but with a single bullet point at the start:

  -
    ```
    cat /usr/local/etc/foo/bar | less | more |
    grep | sed | awk > /usr/local/tmp/foo/bar
    ```
- Text that is output by a program, or should already be in a file, is shown in plain monospaced type:

  -
    ```
    sshd_enable="YES"
    ```

Root and Sudo

- We will use "`sudo`" wherever *root* access is required
- Please work through this tutorial as a normal user, not as *root*
- If you use *root*, some error messages from Exim will be different and this may confuse you

Installing Exim (1)

- Install some dependencies as packages, not ports:
  - `sudo -E pkg_add -r libspf2 cyrus-sasl-saslauthd perl pcre mysql51-client`
- Compile Exim from the ports tree:
  - `cd /usr/ports/mail/exim`
  - `sudo make config`
- Enable the following options:
  - `AUTH_RADIUS`
  - `CONTENT_SCAN`
  - `MYSQL`
  - `SASLAUTHD`
  - `SPF`

Installing Exim (2)

- Now compile Exim:
  - ***sudo make SUBDIR=old WITH_RADIUS_TYPE=RADLIB EXTRALIBS_EXIM=/usr/lib/libradius.so install clean***
  - All on one line!
  - Should take a while compiling, and end with:
  - `===>  Cleaning for exim-4.80.1`

Checking Exim Installation

- ***/usr/local/sbin/exim -bV***
- Exim version 4.80.1 ...
- Support for: crypteq iconv() IPv6 use_setclassresources PAM Perl Expand_dlfunc OpenSSL **Content_Scanning** Old_Demime **Experimental_SPF**
- Lookups: lsearch wildlsearch nwildlsearch iplsearch cdb dbm dbmnz dnsdb dsearch **mysql** nis nis0 passwd
- Authenticators: cram_md5 dovecot plaintext spa
- If you don't have these options:
  - ***cd /usr/ports/mail/exim***
  - ***make deinstall clean***
  - Go back to *Installing Exim (1)*

Replacing Sendmail with Exim

- Stop Sendmail:
  - ***`sudo /etc/rc.d/sendmail stop`***
- Edit */etc/rc.conf* and add these lines:
  - ***`sendmail_enable="NONE"`***
  - ***`sendmail_submit_enable="NO"`***
  - ***`exim_enable="YES"`***
- Edit */etc/mail/mailer.conf* and change these lines:
  - `sendmail` ***`/usr/local/sbin/exim`***
  - `send-mail` ***`/usr/local/sbin/exim`***
  - `mailq` ***`/usr/local/sbin/exim -bp`***
  - `newaliases` ***`/bin/true`***

AfNOG

Starting Exim

- Try the following commands:

  -
    ```
    sudo /usr/local/etc/rc.d/exim start
    Starting exim.
    ```

  -
    ```
    sudo /usr/local/etc/rc.d/exim status
    exim is running as pid XXX
    ```

  -
    ```
    sudo /usr/local/etc/rc.d/exim restart
    Stopping exim.
    Starting exim.
    ```
- Create */etc/periodic.conf.local* and add these lines:

  -
    ```
    daily_status_include_submit_mailq="NO"
    ```

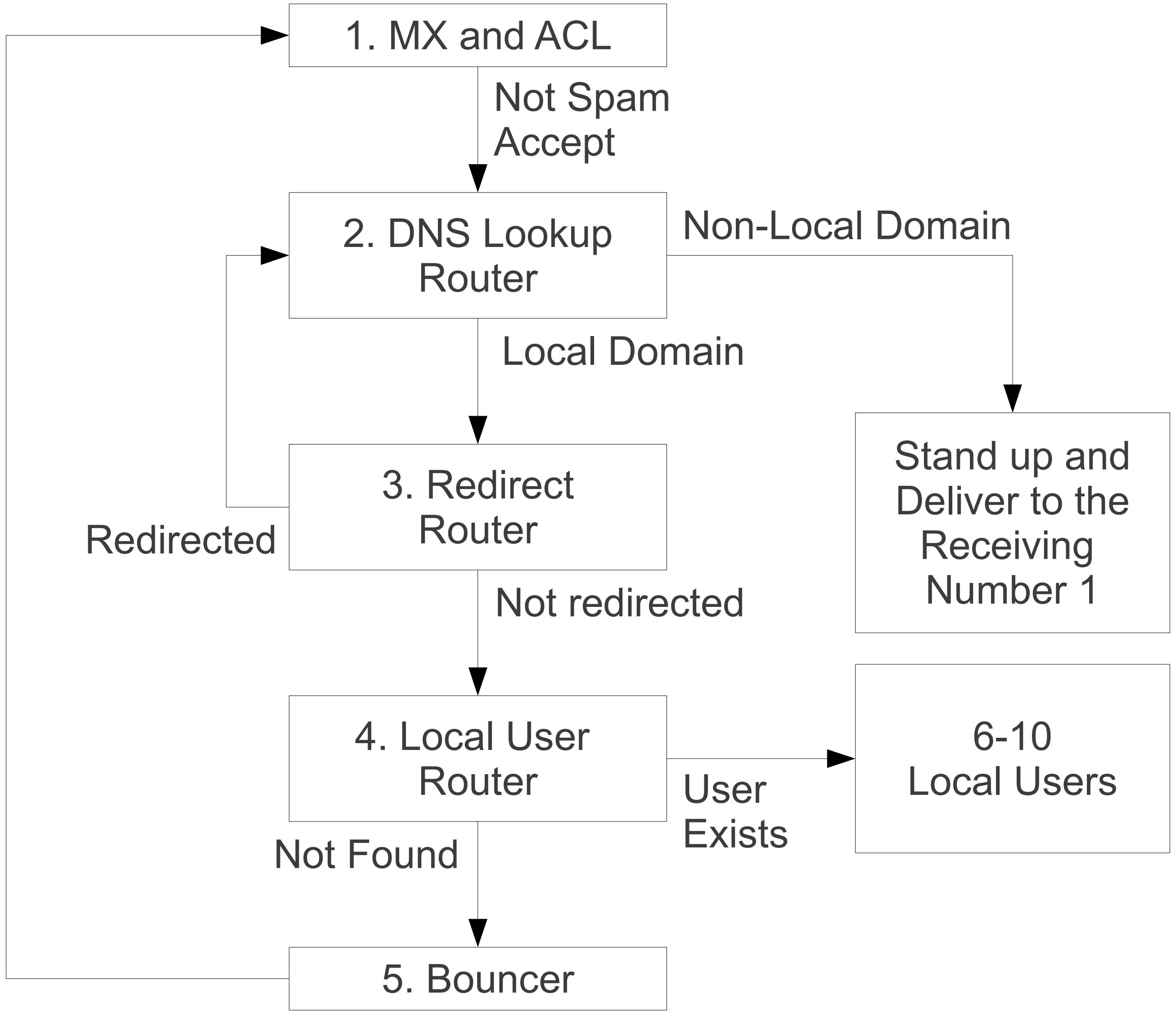
  -
    ```
    daily_clean_hoststat_enable="NO"
    ```

The Exim Game: Preparation

- Divide into groups (e.g. by table)
- Assign a domain name to each group
  - Write these up at the front in the “DNS”
  - Make a public list of valid email addresses
- Each group has at least one person in every role:
  - MX and ACL, DNS Lookup, Redirect, Local Delivery, Bouncer
- Each group creates at least one redirect rule
- Everyone writes an email to someone

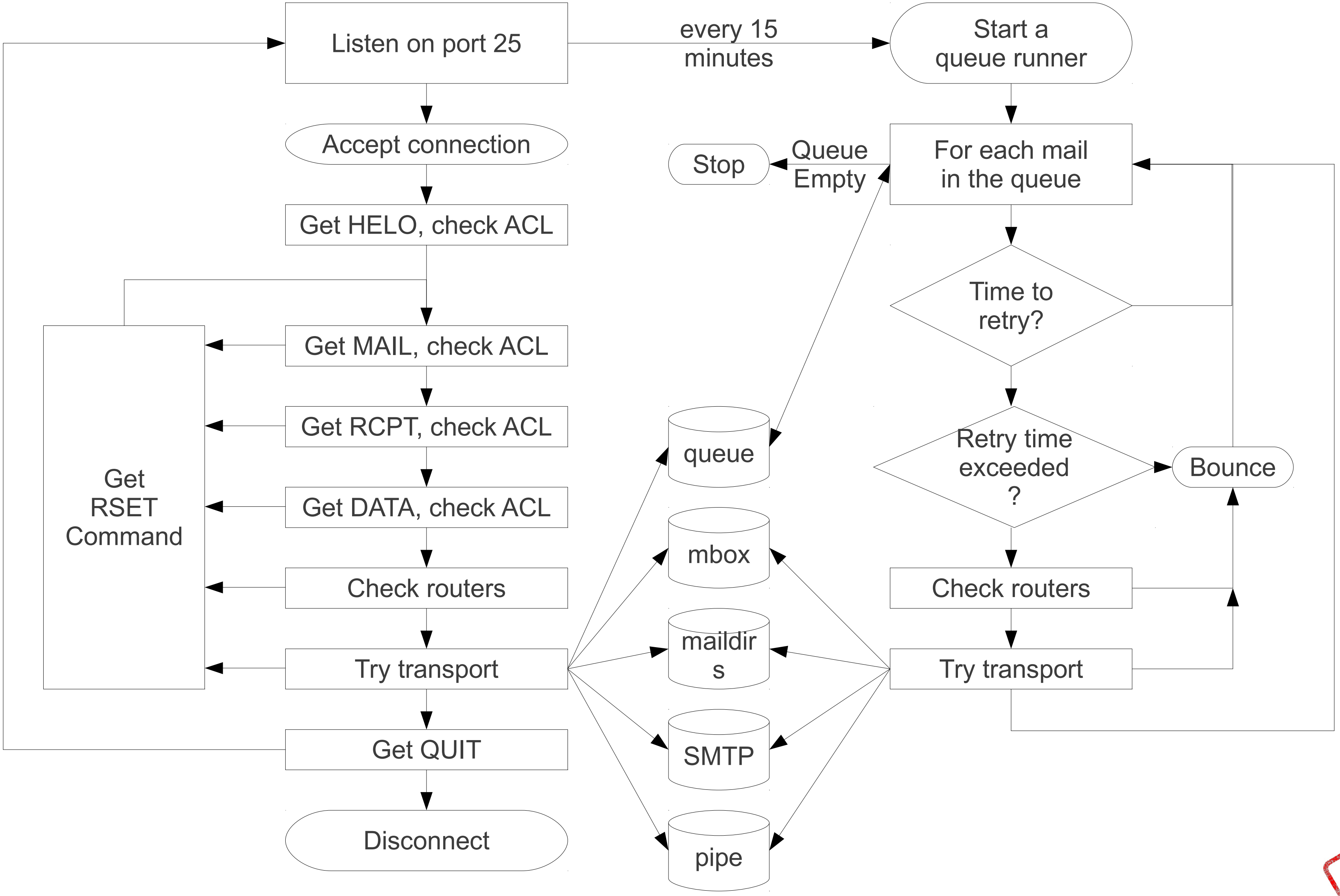
AfNOG

The Exim Game



AfNOG

Exim Overview



AfNOG

Basic Configuration

- Configuration file is */usr/local/etc/exim/configure*
- First section has global options
- Other sections start with the word “begin”
- What are they?

AfNOG

Configuration Sections

- ➢ **Global** (no name)
- • ACL (access control lists, allow or deny mail)
- • Routers (decide what to do with mail)
- • Transports (control how exactly it is delivered)
- ✗ Retry rules (advanced feature)
- ✗ Rewrite (advanced feature)
- • Authenticators (will cover this later)
- ✗ Local Scan (advanced feature)

AfNOG

Global Settings

- The most important default settings:
  - # primary_hostname =
  - domainlist local_domains = @
  - domainlist relay_to_domains =
  - hostlist relay_from_hosts = localhost
  - acl_smtp_rcpt = acl_check_rcpt
  - acl_smtp_data = acl_check_data
  - host_lookup = *
  - rfc1413_hosts = *
  - rfc1413_query_timeout = 5s
  - ignore_bounce_errors_after = 2d
  - timeout_frozen_after = 7d
- See Exim manual, chapter 7 for more details

Testing the defaults

- Send email to afnog@pcXX.sse.ws.afnog.org:

  -
    ```
    telnet localhost 25
    Trying 127.0.0.1...
    Connected to localhost.
    Escape character is '^]'.
    220 pcXX.sse.ws.afnog.org ESMTP Exim 4.69 ...
    ```

  -
    ```
    mail from:<afnog@pcXX.sse.ws.afnog.org>
    250 OK
    ```

  -
    ```
    rcpt to:<afnog@pcXX.sse.ws.afnog.org>
    250 Accepted
    ```

  -
    ```
    data
    354 Enter message, ending with "." on a line by itself
    ```

  -
    ```
    hello world
    .
    250 OK id=1M3RuH-0006WJ-Ia
    ```

  -
    ```
    quit
    221 pcXX.sse.ws.afnog.org closing connection
    ```

AfNOG

Did it work?

- Check your mailbox:

```
cat /var/mail/afnog
From afnog@vm56.sse.ws.afnog.org Mon May 07 11:13:10 2012
Return-path: <afnog@vm56.sse.ws.afnog.org>
Envelope-to: afnog@vm56.sse.ws.afnog.org
Delivery-date: Mon, 07 May 2012 11:13:10 +0000
Received: from localhost ([::1])
by vm56.sse.ws.afnog.org with smtp (Exim 4.77 (FreeBSD))
(envelope-from <afnog@vm56.sse.ws.afnog.org>)
id 1SRLsI-0000FL-Hr
for afnog@vm56.sse.ws.afnog.org; Mon, 07 May 2012 11:13:10
+0000
Message-Id: <E1SRLsI-0000FL-Hr@vm56.sse.ws.afnog.org>
From: afnog@vm56.sse.ws.afnog.org
Date: Mon, 07 May 2012 11:13:10 +0000

hello world
```

Terminology

- In the email address *joe@example.com*:
  - *joe* is the **local part**
  - *example.com* is the **mail domain** (or just **domain**)
- Exim tends to split them apart, so it's easier to treat them separately in the Exim config

Adding another local domain

- Tell Exim to accept mail for *mydomain.example.com*
- Use a domain that doesn’t exist yet (no MX records), otherwise Exim will try to deliver it by SMTP (why?)
- How will we know when we’ve done it?
  - Use an “address test” to see what Exim will do with the mail:

  -
    ```
    exim -bt afnog@mydomain.example.com
    afnog@mydomain.example.com is undeliverable
    ```
  - Let’s make it deliverable!

Adding another local domain

- Add a new entry to the domain list, using the “:” character to separate it from the previous entry:

  -
    ```
    sudo vi /usr/local/etc/exim/configure
    ```

    -
      ```
      domainlist local_domains = @ : mydomain.example.com
      ```
- Now what does the address test say?

  -
    ```
    exim -bt afnog@mydomain.example.com
    afnog@mydomain.example.com
      router = localuser, transport = local_delivery
    ```

Testing the new local domain

- Send email to afnog@mydomain.example.com:

  -
    ```
    exim -bs
    220 vmXX.sse.ws.afnog.org ESMTP Exim 4.69 ...
    ```

  -
    ```
    mail from:<afnog@pcXX.sse.ws.afnog.org>
    250 OK
    ```

  -
    ```
    rcpt to:<afnog@mydomain.example.com>
    250 Accepted
    ```

  -
    ```
    data
    354 Enter message, ending with "." on a line by itself
    ```

  -
    ```
    hello my lovely new domain!
    .
    250 OK id=1M3RuH-0006WJ-Ia
    ```

  -
    ```
    quit
    221 vmXX.sse.ws.afnog.org closing connection
    ```

  -
    ```
    tail /var/mail/afnog
    ...
    hello my lovely new domain!
    ```

Testing Notes

- **`exim -bs`** is “command-line SMTP mode”
  - similar to connecting to port 25
  - can quit with Control+C
  - no need to restart exim in this case
  - useful for testing new configurations
- we did not restart Exim, so the daemon listening on port 25 is still running the old configuration

  -
    ```
    sudo /usr/local/etc/rc.d/exim restart
    Stopping exim.
    Starting exim.
    ```

Relay Testing

- `exim -bs` and `telnet localhost 25` both connect “from” localhost
- localhost has special privileges:
  - `hostlist relay_from_hosts = localhost`
  - `accept hosts = +relay_from_hosts`
- try using `exim -bh` to simulate mail relaying by an untrusted server, with IP address *1.2.3.4*:

  -
    ```
    exim -bh 1.2.3.4
    220 noc.sse.ws.afnog.org ESMTP Exim 4.69 ...
    ```

  -
    ```
    mail from:<afnog@pcXX.sse.ws.afnog.org>
    250 OK
    ```

  -
    ```
    rcpt to:<afnog@anotherdomain.example.com>
    550 relay not permitted
    ```

Allow Relaying

- Change hostlist relay_from_hosts:

  -
    ```
    hostlist relay_from_hosts = localhost : 1.2.3.0/24
    ```
- Try exim -bh again:

  -
    ```
    exim -bh 1.2.3.4
    220 noc.sse.ws.afnog.org ESMTP Exim 4.69 ...
    ```

  -
    ```
    mail from:<afnog@pcXX.sse.ws.afnog.org>
    250 OK
    ```

  -
    ```
    rcpt to:<afnog@anotherdomain.example.com>
    250 Accepted
    ```
- What would you expect to happen with:
  - `exim -bh 1.2.3.19`
  - `exim -bh 1.2.5.4`

Types of Lists

- domainlist
  - `*.mydomain.com : @`
- hostlist
  - `192.168.1.0/24 : hostname.domain.com`
- addresslist
  - `*@example.com : example.com : *.example.com :`
- local parts list (not covered here)
- string list (simple)
- see Exim manual chapter 10 for more details

AfNOG

Next up: Routers

- ✔ Global (no name)
- ➢ **Routers (decide what to do with mail)**
- Transports (control how exactly it is delivered)
- ◆ Access Control (who is allowed to send mail)
- ◆ Authenticators (logging in to relay mail)
- ◆ Troubleshooting (when things go wrong)

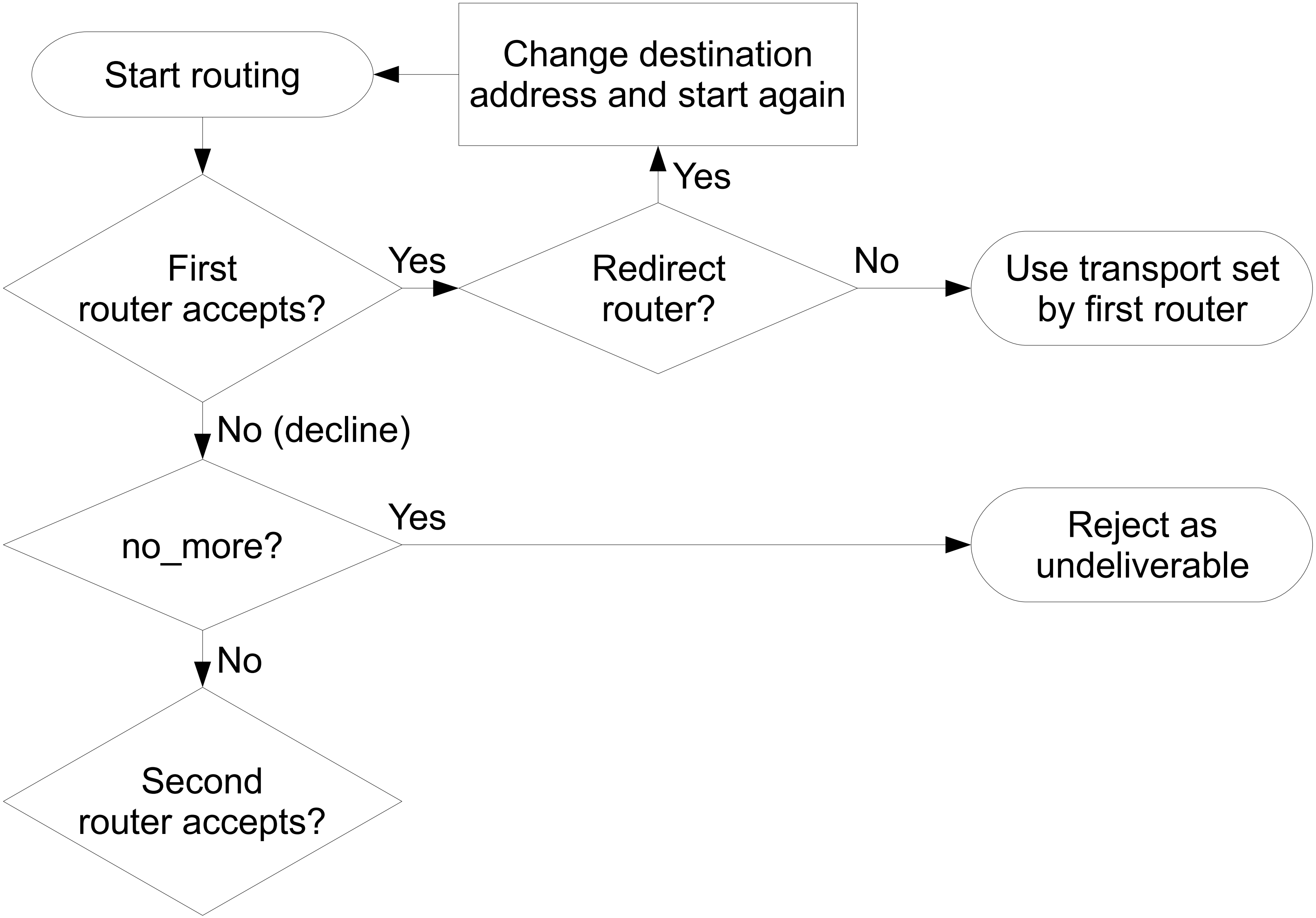
AfNOG

Routers

- Decide where to deliver mail to
  - Run in order until one accepts the mail
  - Accepting router sets the transport for the mail
- Can also redirect mail (change the destination)
- Can check whether mail is deliverable:
  - local recipients exist
  - remote domains are routable
- Reject mail in Access Control instead of Routers if possible
  - Router failure → bounce email → Joe Job spam!

AfNOG

Routing Overview



Anatomy of a Router

- Conditions control whether the driver runs:
  - address_test, check_local_user, condition
  - domains = +local_domains
  - user = mail
  - transport = trotro (or matatu)
- A driver is specified:
  - driver = accept, redirect, manualroute
- Options control what the driver does (if run)
- Specified driver is run
  - Result may be *accept*, *decline* or *fail*

Types of Routers (drivers)

- A router is a configuration item
- A driver is a piece of code within Exim
  - *accept*: enqueue mail to a transport
    - incoming email, to a file or program (*appendfile* or *pipe*)
  - *dnslookup*: enqueue if domain exists
    - outgoing email, delivery to MX (an *smtp* transport)
  - *manualroute*: enqueue if domain in list
    - outgoing email, delivery to a smarthost (an *smtp* transport)
  - *redirect*: change the destination address
    - used for forwarding, aliases, virtual domains

The Default Routers

- dnslookup (for outbound email via SMTP)
- system_aliases (lookup in /etc/aliases, redirect)
- userforward (local user .forward files, redirect)
- localuser (deliver to local mbox or maildir)

The *dnslookup* Router

- **domains = ! +local_domains** ← *condition*
  - only if destination domain is not in *local_domains*
- **driver = dnslookup** ← *driver*
  - check that the destination domain has MX or A
- **ignore_target_hosts** = 0.0.0.0 : 127.0.0.0/8 ← *option*
- **no_more** ← *option*
  - if conditions match but router declines then bounce
- **transport = remote_smtp** ← *option*
  - if router accepts, then use remote_smtp to deliver

The *system_aliases* Router

- **driver = redirect**
- allow_fail
- allow_defer
- **data = ${lookup {$local_part} lsearch {/etc/aliases}}**
- user = mailnull
- group = mail
- file_transport = address_file
- pipe_transport = address_pipe

The *userforward* Router

-
  ```
  driver = redirect
    check_local_user
    file = $home/.forward
    no_verify
    no_expn
    check_ancestor
    file_transport = address_file
    pipe_transport = address_pipe
    reply_transport = address_reply
    condition = ${if exists{$home/.forward} {yes} {no} }
  ```

- The contents of `$home/.forward` is read and used as “data” for the redirect router driver

- The `condition` could be replaced by:
  ```
  require_files = $home/.forward
  ```

The *localuser* Router

-
  ```
  localuser:
    driver = accept
    check_local_user
    transport = local_delivery
    cannot_route_message = Unknown user
  ```
- This is the last router, so if it does not accept, the message is bounced as undeliverable
- This driver always accepts, if the conditions are met
- `check_local_user` means that the local user must exist
- `cannot_route_message` sets the message that will be returned to the SMTP client when this happens

The Redirect Driver

- Tells Exim to call an internal router module called *redirect* to do the routing
- *redirect* is used for aliases files, virtual domains, .forward files... anything that redirects mail
- In the manual this driver is called the “redirect router” (chapter 22)
- Not the same as a router called “redirect”, which could use any driver you like
- I prefer to call it “the redirect driver”
- The `data` option is expanded to the new destination

The system_aliases Router

- Redirect root’s mail to the afnog user
  - ***exim -bt root***

    ```
    root@pcXX.sse.ws.afnog.org
      router = localuser, transport = local_delivery
    ```
  - ***sudo vi /etc/aliases***
    - ***root: afnog***
  - ***exim -bt root***

    ```
    afnog@pcXX.sse.ws.afnog.org
        <-- root@pcXX.sse.ws.afnog.org
      router = localuser, transport = local_delivery
    ```
- Did it work? How do you know?

Simple Redirecting Router

- Redirect a single local part to another local part
  -
    ```
    exim -bt foo@mydomain.example.com
    foo@mydomain.example.com is undeliverable
    ```
  -
    ```
    sudo vi /usr/local/etc/exim/configure
    ```
    -
      ```
      begin routers
      redirect_foo_to_afnog:
         driver = redirect
         domains = mydomain.example.com
         local_parts = foo
         data = afnog
      ```
  -
    ```
    exim -bt foo@mydomain.example.com
    afnog@pcXX.sse.ws.afnog.org
        <-- foo@mydomain.example.com
      router = localuser, transport = local_delivery
    ```
- Did it work? How do you know?

Adding a Virtual Domain

- Tell Exim what to do with the mail domain *virtual.example.com*:

  -
    ```
    exim -bt foo@virtual.example.com
    foo@virtual.example.com is undeliverable
    ```

  -
    ```
    sudo vi /usr/local/etc/exim/configure
    ```

    -
      ```
      begin routers
      ```

    -
      ```
      virtual_domain_router:
      ```

    -
      ```
        driver = redirect
      ```

    -
      ```
        domains = virtual.example.com
      ```

    -
      ```
        data = ${lookup {$local_part} lsearch
      {/usr/local/etc/exim/virtual.example.com}}
      ```

  -
    ```
    exim -bt foo@virtual.example.com
    foo@virtual.example.com cannot be resolved at this time
    ```
- What’s wrong?

Debugging Routers

- ***sudo exim -bt -d-all+route foo@virtual.example.com***
  - `routing foo@virtual.example.com`
  - `--------> virtual_domain_router router <-------- local_part=foo domain=virtual.example.com`
  - `virtual_domain_router router: defer for foo@virtual.example.com`
  - `  message: failed to expand "${lookup {$local_part} lsearch {/usr/local/etc/exim/virtual.example.com}}": failed to open` **`/usr/local/etc/exim/virtual.example.com`** `for linear search:` **`No such file or directory`**
- Exim tried to open *\/usr/local/etc/exim/virtual.example.com*
- The file did not exist
- So the router deferred the message.

Fixing the Problem

- Create the file */usr/local/etc/exim/virtual.example.com:*
  - ➢ ***sudo vi /usr/local/etc/exim/virtual.example.com***
    - ➢ ***foo: afnog***
- Test again:
  - ➢ ***exim -bt foo@virtual.example.com***

    ```
    afnog@pcXX.sse.ws.afnog.org
        <-- foo@virtual.example.com
      router = localuser, transport = local_delivery
    ```
- Note that we did not add *virtual.example.com* to our local_domains list. Why did it work?

Running many Virtual Domains

- ***exim -bt john@toomany.example.com***

  ```
  john@toomany.example.com is undeliverable
  ```

- ***sudo vi /usr/local/etc/exim/configure***
  - `begin routers`

  -
    ```
    virtual_domain_router:
      driver = redirect
      require_files = /usr/local/etc/exim/$domain
      data = ${lookup {$local_part} lsearch \
    {/usr/local/etc/exim/$domain}}
    ```
  - don’t forget to remove the “domains” line!
- ***sudo vi /usr/local/etc/exim/toomany.example.com***
  - ***john: afnog***
- ***exim -bt john@toomany.example.com***

  ```
  afnog@pcXX.sse.ws.afnog.org
      <-- john@toomany.example.com
    router = localuser, transport = local_delivery
  ```

Manual Routing a Domain

- ***exim -bt foo@manual.example.com***

```
foo@manual.example.com is undeliverable
```

- ***sudo vi /usr/local/etc/exim/configure***
  - begin routers
  - ***manual_router:***

```
manual_router:
   driver = manualroute
   domains = manual.example.com
   route_data = noc.mtg.afnog.org
   transport = remote_smtp
```

- ***exim -bt foo@manual.example.com***

```
foo@manual.example.com
  router = manual_router, transport = remote_smtp
  host noc.mtg.afnog.org [196.200.223.10]
```

Manual Routing all Domains

- ***exim -bt foo@example.com***

```
foo@example.com
  router = dnslookup, transport = remote_smtp
  host example.com [208.77.188.166]
```

- ***sudo vi /usr/local/etc/exim/configure***

  -
    ```
    # replace the default dnslookup router
    smarthost:
      driver = manualroute
      route_data = noc.mtg.afnog.org
      domains = ! +local_domains
      transport = remote_smtp
      ignore_target_hosts = 0.0.0.0 : 127.0.0.0/8
      no_more
    ```

- ***exim -bt foo@example.com***

```
foo@example.com
  router = smarthost, transport = remote_smtp
  host noc.mtg.afnog.org [196.200.223.1]
```

Delivering to RADIUS users (1)

- No local account, so *localuser* router won't work
- Edit */usr/local/etc/exim/configure*
- Add the MySQL login details to global section, before `begin acl`:
  - `hide mysql_servers = localhost/radius/radius/radpass`
- Add a new router, before the *localuser* router:
  - `radius:`
  - `  driver = accept`
  - `  local_parts = mysql;SELECT 1 FROM radcheck WHERE username = '${quote_mysql:$local_part}';`
  - `  transport = local_delivery`

Delivering to RADIUS users (2)

- Edit */usr/local/etc/exim/configure*, find the *local_delivery* transport, and comment out this line:
  - `# user = $local_part`
- Test with exim -bt:
  - ***`sudo exim -bt afnog@vmXX.sse.ws.afnog.org`***
    - `afnog@vmXX.sse.ws.afnog.org`
    - `router =` **`localuser`**`, transport = local_delivery`
  - ***`sudo exim -bt fred@vmXX.sse.ws.afnog.org`***
    - `fred@vmXX.sse.ws.afnog.org`
    - `router =` **`radius`**`, transport = local_delivery`
  - ***`sudo exim -bt fredd@vmXX.sse.ws.afnog.org`***
    - `fredd@vmXX.sse.ws.afnog.org is undeliverable:` **`Unknown user`**

Delivering to RADIUS users (3)

- Restart Exim
- Test with SWAKS (thanks Joost!)

  -
    ```
    sudo -E pkg_add -r swaks
    ```

  -
    ```
    swaks -t afnog@vmXX.sse.ws.afnog.org
    <-  250 OK id=1OHduc-0005Qx-CO
    ```

  -
    ```
    grep -A2 "Message-Id.*1OHduc-0005Qx-CO"
    /var/mail/afnog
    This is a test mailing
    ```

  -
    ```
    swaks -t fred@vmXX.sse.ws.afnog.org
    <-  250 OK id=1OHdxG-0005RH-HC
    ```

  -
    ```
    sudo grep -A2 "Message-Id.*1OHdxG-0005RH-HC"
    /var/mail/fred
    This is a test mailing
    ```

  -
    ```
    swaks -t fredd@vmXX.sse.ws.afnog.org
    ```

Aptivate's Routers

- **net4dev** (manualroute)
- dnslookup
- **domain_aliases** (redirect, virtual domains)
- **domain_aliases_suffixed** (ditto)
- **default_aliases** (renamed system_aliases)
- **no_more_aliases** (not local_domains)
- user_forward
- **procmail** (user ~/.procmailrc files)
- **localuser_nosuffix** (renamed localuser)

AfNOG

Local Part Suffixes

- Allows you to send mail to afnog-anything and have it delivered to afnog
- Users can filter mail to different boxes
- Configured in the router:
  - `local_part_suffix = +* : -*`
  - `local_part_suffix_optional`
- If user names contain a suffix character, that part of the username will be removed!
  - Put a router without suffixes before the one with suffixes
- Prefix is possible as well

Next up: Transports

- ✔ Global (no name)
- ✔ Routers (decide what to do with mail)
- ➢ **Transports (control how exactly it is delivered)**
- ◆ Access Control (who is allowed to send mail)
- ◆ Authenticators (logging in to relay mail)
- ◆ Troubleshooting (when things go wrong)

Transports

- Control how messages are delivered
- Only used when referenced from routers
  - Order does not matter
- Always have a *driver* (type of transport)
- Standard transports:
  - remote_smtp
  - local_delivery
  - address_pipe
  - address_file
  - address_reply

The *remote_smtp* Transport

-
  ```
  remote_smtp:
    driver = smtp
  ```
- no options or conditions
- driver specifies a chunk of Exim code
- this time a transport driver (not a router driver)
- the *smtp* driver delivers mail to another server using SMTP
- the remote server is set by the *dnslookup* or *manualroute* driver

The *local_delivery* Transport

-
  ```
  local_delivery:
    driver = appendfile
    file = /var/mail/$local_part
    delivery_date_add
    envelope_to_add
    return_path_add
    group = mail
    user = $local_part
    mode = 0660
    no_mode_fail_narrower
  ```

- Delivers mail to a file in mbox format
  - One large file, bad for scalability

Procmail Router

- `sudo -E pkg_add -r procmail`
- Create or edit */home/afnog/.procmailrc*:

  -
    ```
    :0f
    | sed -e 's/foo/bar/'
    ```
- `echo food | mail afnog`

-
  ```
  tail -2 /var/mail/afnog
  food
  ```
- Sudo edit */usr/local/etc/exim/configure*:
  - `begin routers`

  -
    ```
    procmail_router:
      driver = accept
      check_local_user
      transport = procmail_pipe
      require_files = ${home}/.procmailrc
      no_verify
    ```

Procmail Transport

- ***sudo vi /usr/local/etc/exim/configure***
  - begin transports
  -
    ```
    procmail_pipe:
      driver = pipe
      command = "/usr/local/bin/procmail"
      return_path_add
      delivery_date_add
      envelope_to_add
    ```
- ***sudo /usr/local/etc/rc.d/exim restart***
- ***echo food | mail afnog***
-
  ```
  tail -2 /var/mail/afnog
  bard
  ```
- ***rm ~/.procmailrc***

Switch to Maildirs

- ***sudo vi /usr/local/etc/exim/configure***

  -
    ```
    local_delivery:
      driver = appendfile
      # file = /var/mail/$local_part
      maildir_format
      directory = $home/mail
      delivery_date_add
      envelope_to_add
      return_path_add
      group = mail
      user = $local_part
      mode = 0660
      no_mode_fail_narrower
    ```

- ***sudo /usr/local/etc/rc.d/exim restart***

- ***ls /home/afnog/mail***

- ***echo test | mail afnog***

- ***ls /home/afnog/mail***

Next up: Access Control

- ✓ Global (no name)
- ✓ Routers (decide what to do with mail)
- ✓ Transports (control how exactly it is delivered)
- ➢ **Access Control (who is allowed to send mail)**
- ◆ Authenticators (logging in to relay mail)
- ◆ Troubleshooting (when things go wrong)

AfNOG

Access Control

- Controls who is allowed to send you mail, or not
- Most useful weapon in the war against spam
- Most SMTP commands are subject to an Access Control List (ACL) (see chapter 40 of the manual)
- Most commonly used are RCPT and DATA ACLs
  - Why not MAIL?
- DATA ACL applies at the end of the DATA command, after the message body has been sent
  - Too late to reject individual recipients
  - Too late to save bandwidth

Using Access Control Lists

- ACLs are named followed by a colon : and usually start with *acl*_
  - which ACLs does Exim include by default?
- ACLs can appear in any order in the “acl” section
- ACLs are not used unless:
  - referenced in the global configuration, or
  - called by another ACL
- Look for acl_* statements in the global section
  - which ACLs does Exim use by default?

Anatomy of an ACL

- Every ACL consists of Access Control Entries
- Every entry starts with a **verb**
  - every verb ends the previous entry and starts a new one
- Other lines are **conditions** and **options**
  - Conditions control **whether** the verb is executed
  - Options control **what** the verb does when executed
- Order of entries and lines in an ACL is important
  - Processing of an entry stops as soon as a condition fails
  - Options after a condition that fails are not used
  - Can change the options and then apply more conditions

Access Control Verbs

- **accept:** the command is allowed
- **defer:** command refused, returns a temporary error
- **deny:** command refused, returns a permanent error
- **discard:** returns success but throws away the recipient or message
- **drop:** like deny, but drops the connection too
- **require:** opposite of deny, denies the message if not all conditions are met
- **warn:** writes a warning message to the logs, but allows command to proceed

The *acl_check_rcpt* ACL

```
• accept  hosts = :

• deny    message       = Restricted characters in address
          domains       = +local_domains
          local_parts   = ^[.] : ^.*[@%!/|]

• accept  local_parts   = postmaster
          domains       = +local_domains

• require verify        = sender

• accept  hosts         = +relay_from_hosts
          control       = submission

• accept  authenticated = *
          control       = submission

• require message = relay not permitted
          domains = +local_domains : +relay_to_domains

• require verify = recipient
```

Address Verification

- *verify = sender* or *verify = recipient*
- $sender_verify_failure or $recipient_verify_failure will contain one of the following words:
  - **qualify** (the address was unqualified (no domain), and the message was neither local nor came from an exempted host)
  - **route** (routing failed)
  - **mail** (routing succeeded, and a callout was attempted; rejection occurred at or before the MAIL command)
  - **recipient** (the RCPT command in a callout was rejected)
  - **postmaster** (the postmaster check in a callout was rejected)

Callouts

- Standard address verification only uses the Exim configuration file and the DNS
- Callouts make a pretend SMTP connection
  - Sender callouts connect to the sender domain's MX
  - Recipient callouts connect to the recipient domain's MX
- Callouts can reduce spam by rejecting invalid addresses
- Callouts do block some legitimate email
- Callouts are controversial, some consider them abuse

AfNOG

Configuring Callouts

- Sudo edit */usr/local/etc/exim/configure*:

  -
    ```
    domainlist relay_to_domains = rl.example.com
    ```

  -
    ```
    acl_check_rcpt:
    ```

  -
    ```
    require message = Sender verify failed
            verify = sender/callout=120s
    ```

  -
    ```
    require message = Recipient verify failed
            verify = recipient/callout=120s
    ```

Testing Callouts

- ***exim -bhc 1.2.3.4***
  - ***mail from:<nonexist@pcXX.sse.ws.afnog.org>***
  - ***rcpt to:<afnog@pcXX.sse.ws.afnog.org>***
    ```
    550 Sender verify failed
    ```
  - ***rset***
  - ***mail from:<afnog@pcXX.sse.ws.afnog.org>***
  - ***rcpt to:<nonexist@rl.example.com>***
    ```
    550 Recipient verify failed
    ```

Blocking Senders and Recipients

-
  ```
  deny    senders = naNaijaadmin@list.nanaija.com
  ```

-
  ```
  deny    senders = *@web-performers.com
          message = Get lost, you lying link exchange \
                    spammers
  ```

-
  ```
  deny    hosts   = *.mailserve.net
          message = Get lost, you lying link exchange \
                    spammers
  ```

-
  ```
  deny    senders = bfsummit@bfsummit.com
          message = I hope you catch bird flu and die
  ```

-
  ```
  deny    senders    = \N^.*mission2007.*@dgroups.org$\N
          recipients = info@aidworld.org
          message    = Please remove me from your list.
  ```

Hate your neighbour?

- Add to your RCPT ACL:

  -
    ```
    acl_check_rcpt:
    ```

  -
    ```
    deny hosts = pcYY.sse.ws.afnog.org
         message = I don’t like your socks
    ```

-
  ```
  sudo /usr/local/etc/rc.d/exim restart
  ```
- Ask your neighbour to test it:

  -
    ```
    telnet pcXX.sse.ws.afnog.org 25
    ```

  -
    ```
    mail from:<afnog@pcYY.sse.ws.afnog.org>
    ```

  -
    ```
    rcpt to:<afnog@pcXX.sse.ws.afnog.org>
    550 I don't like your socks
    ```
- How would you block everyone in the classroom?
- What do you see in the logs?

Sender Policy Framework

- Allows you to say which IPs are allowed to send from your domain (prevent spammers from using it)
- Useful when you want to block all mail from a domain, or only participate in SRS mailing lists
- Only works when people reject mails that fail SPF
- Causes problems for mailing lists not using SRS
- Many people complain, but it works for me!

Enable SPF for your domain

- Generate your SPF record for your domain using *www.openspf.org* that only allows your PC to send:
  - e.g. `"v=spf1 a:pcXX.sse.ws.afnog.org -all"`
- Edit the zone file for XXXX.bogus.gh and add:
  - ***`@ IN TXT "v=spf1 a:pcXX.sse.ws.afnog.org ~all"`***
- Reload the zone and query the TXT record using *dig*
- Add an SPF check high up in your RCPT ACL:
  - `acl_check_rcpt:`

  -
    ```
    deny spf = fail
         message = SPF check failed: $spf_smtp_comment
         log_message = SPF check failed: $spf_result
    ```

Testing SPF on your domain

- Decide who will be the recipient
  - SPF protects recipients against forged senders
  - Ideally cooperate with someone else to test each other
- You need to pass sender verification:
  - Either add xxxx.bogus.gh to local_domains on the MX
  - Or remove verify = sender from the recipient's exim
- `exim -bh 1.2.3.4`
  - `mail from:<afnog@bogus.gh>`
  - `rcpt to:<afnog@pcXX.sse.ws.afnog.org>`
  - `550 Sender verify failed`

Blackmail

-
  ```
  deny    ! hosts = +relay_from_hosts
          ! authenticated = *
          dnslists = zen.spamhaus.org
          message = $sender_host_address \
          blacklisted by Spamhaus\n\
          (http://www.spamhaus.org/query/bl?
  ip=$sender_host_address)\n\
          $dnslist_text
  ```

-
  ```
  warn    ! hosts = +relay_from_hosts
          ! authenticated = *
          dnslists = bl.spamcop.net
          message = X-Warning: \
            $sender_host_address blacklisted \
            by $dnslist_domain ($dnslist_text)
  ```

Name Calling

-
  ```
  deny condition = ${if match \
       {${lookup dnsdb \
          {zns=${sender_address_domain}}}} \
       {.*\.ip4dns\.com}}
       message = You look like a spammer to me
  ```
- Searches for nameservers for the sender’s mail domain, and recursively up until it finds some
- Pattern match against `.*\.ip4dns\.com`
  - ns1.ip4dns.com
  - ns2.ip4dns.com

Don’t Pretend to be Me

- Catch people who say HELO (my own IP address):
  - Global section:
    - ***acl_smtp_helo = acl_check_helo***
  - `begin acl` section:
    - `acl_check_helo:`
    - ***drop ! hosts = :***
    - ***`      ! hosts = 80.248.178.170`***
    - ***`      condition = ${if eq \`***
    - ***`        {$smtp_command_argument} \`***
    - ***`        {80.248.178.170}}`***
    - ***`      message = You are SO lying`***
    - `acl_check_rcpt: …`

AfNOG

Ignore people who don't say HELO

- `acl_smtp_helo = acl_check_helo`
- acl_check_helo:

```
acl_check_helo:
drop condition = ${if or { \
     {!match{$smtp_command_argument} \
            {\\.}} \
     { match{$smtp_command_argument} \
            {\\d+[.-]\\d+[.-]\\d+[.-]\\d+}} \
     }}
     message = Please configure your mail \
               server with a real hostname
     log_message = Invalid HELO
accept
```

- acl_check_rcpt:

```
acl_check_rcpt:
deny condition = ${if eq {$sender_helo_name}{}}
     message   = Please say HELO first
```

Assassinating Spam(mers)

- `sudo -E pkg_add -r p5-Mail-SpamAssassin`
- `cd /usr/local/etc/mail/spamassassin`
- `sudo cp local.cf.sample local.cf`
- `sudo sa-update`
- `sudo vi /etc/rc.conf`
  - `spamd_enable="YES"`
- `sudo /usr/local/etc/rc.d/sa-spamd start`

  ```
  Starting spamd.
  ```
- `fetch  http://www.ws.afnog.org/afnog2012/sse/exim/spam.txt`
- `spamc -R < spam.txt`

  ```
  Spam detection software, running on the system
  "vm56.sse.ws.afnog.org", has identified this incoming email
  as possible spam...
  ```

Filtering Mail through SpamAssassin

- Uncomment and modify the following lines:
  - `acl_check_data`

  -
    ```
    deny    spam       = nobody
            message    = Possible spam detected
            add_header = X-Spam_score: $spam_score\n\
                         X-Spam_score_int: $spam_score_int\n\
                         X-Spam_bar: $spam_bar\n\
                         X-Spam_report: $spam_report
    ```
- Test with *swaks*:
  - ***`swaks -t afnog@localhost --body - < spam.txt`***
  - `<** 550 Administrative prohibition`

Installing Clam Antivirus (1)

- Install and enable ClamAV software:

```
➢ sudo -E pkg_add -r clamav
➢ sudo pw usermod clamav -G mail
➢ sudo vi /etc/rc.conf
    ➢ clamav_clamd_enable="YES"
    ➢ clamav_freshclam_enable="YES"
```

- Add the following lines to */usr/local/etc/freshclam.conf*:

```
• HTTPProxyServer 196.200.223.1
• HTTPProxyPort 3128
```

Installing Clam Antivirus (2)

- Download the latest definitions:
  ```
  sudo freshclam
  ```
- This will take a long time, and then output:
  ```
  Database updated (1212167 signatures) from database.clamav.net
  WARNING: Clamd was NOT notified: Can't connect to clamd through /var/run/clamav/clamd.sock
  ```
- Start the ClamAV daemon:
  ```
  sudo /usr/local/etc/rc.d/clamav-clamd start
  Starting clamav_clamd.
  ```

Testing Clam Antivirus

- Test that it works!
  - ***fetch http://www.ws.afnog.org/afnog2012/sse/exim/eicar***
  - ***clamdscan eicar***
  - `/usr/home/afnog/eicar: Eicar-Test-Signature FOUND`
  - `----------- SCAN SUMMARY -----------`
  - `Infected files: 1`

Filtering Mail through ClamAV

- Edit */usr/local/etc/exim/configure*:
  - Uncomment and change these lines:
  - ***`av_scanner = clamd:/var/run/clamav/clamd.sock`***

  -
    ```
    deny malware = *
    ```

  -
    ```
         message = This message contains a virus \
    ($malware_name).
    ```
- Restart Exim
- Test with *swaks*:
  - ***`swaks -t afnog@localhost -d - < eicar`***

  -
    ```
    <** 550 This message contains a virus (Eicar-Test-Signature).
    ```

Next up: Authenticators

- ✓ Global (no name)
- ✓ Routers (decide what to do with mail)
- ✓ Transports (control how exactly it is delivered)
- ✓ Access Control (who is allowed to send mail)
- ➢ **Authenticators (logging in to relay mail)**
- ◆ Troubleshooting (when things go wrong)

Why use SMTP Authentication?

- Your boss wants to send outbound mail from home
- You want to reduce spam from your customers
- You want to use the same server for inbound and outbound mail
- **Warning:** it’s easy to enable SMTP authentication and not use SSL, resulting in plain text passwords being sent over the Internet
- PAM doesn’t work directly from Exim on FreeBSD, so we’ll install *saslauthd* for PAM authentication

Installing *saslauthd*

- Install the binary package (may already be installed):
  - `sudo -E pkg_add -r cyrus-sasl-saslauthd`
- Enable and start it:
  - `sudo vi /etc/rc.conf`
    - `saslauthd_enable="YES"`
  - `sudo /usr/local/etc/rc.d/saslauthd start`
- Test that it authenticates properly:

  -
    ```
    sudo testsaslauthd -u afnog -p afnog
    0: OK "Success."
    ```

  -
    ```
    sudo testsaslauthd -u afnog -p wrong
    0: NO "authentication failed"
    ```

Enabling SMTP Authentication

- ***sudo vi /usr/local/etc/exim/configure***
  - After `begin authenticators`, uncomment and change this:

    ```
    LOGIN:
      driver = plaintext
      server_set_id = $auth1
      server_prompts = <| Username: | Password:
      server_condition = ${if saslauthd{{$auth1} \
    {$auth2} {smtp}}}
    ```

  - But leave this line commented out:

    ```
    # server_advertise_condition = ...
    ```

- ***exim -bs***

  ```
  220 vmXX.sse.ws.afnog.org ESMTP Exim 4.69 …
  ehlo 1.2.3
  250-vmXX.sse.ws.afnog.org Hello afnog at 1.2.3
  250-AUTH LOGIN
  ```

Testing SMTP Authentication

-
  ```
  sudo /usr/local/etc/rc.d/exim restart
  Stopping exim.
  Starting exim.
  ```

-
  ```
  swaks -t afnog@localhost -a LOGIN -au afnog -ap afnog
  <-  235 Authentication succeeded
   -> MAIL FROM:<afnog@freebsd82>
  <-  250 OK
  ```

-
  ```
  swaks -t afnog@localhost -a LOGIN -au afnog -ap wrong
  <** 535 Incorrect authentication data
  *** No authentication type succeeded
   -> QUIT
  ```

AfNOG

Using RADIUS for Authentication

-
  ```
  sudo radtest afnog afnog localhost 0 afnog
  rad_recv: Access-Accept packet ...
  ```

-
  ```
  vi /etc/radius.conf
  ```

  -
    ```
    auth localhost afnog
    ```

-
  ```
  sudo vi /usr/local/etc/exim/configure
  ```

  -
    ```
    LOGIN:
    ```

  -
    ```
    server_condition = ${if radius {$auth1:$auth2}}
    ```

-
  ```
  sudo -u mailnull exim -bh 1.2.4.5
  220 noc.sse.ws.afnog.org ESMTP Exim 4.69 ...
  ```

-
  ```
  ehlo 0
  ... 250-AUTH LOGIN ...
  ```

-
  ```
  auth login
  334 VXNlcm5hbWU6
  ```

-
  ```
  YWZub2c=
  ```

-
  ```
  YWZub2c=
  235 Authentication succeeded
  ```

Testing Authenticated Relaying

-
  ```
  sudo -u mailnull exim -bh 1.2.4.5
  220 noc.sse.ws.afnog.org ESMTP Exim 4.69 ...
  ```

-
  ```
  mail from:<afnog@mydomain.example.com>
  250 OK
  ```

-
  ```
  rcpt to:<example@example.com>
  550 relay not permitted
  ```

-
  ```
  ehlo 0
  ```

-
  ```
  auth login
  ```

-
  ```
  YWZub2c=
  ```

-
  ```
  c3Nl
  235 Authentication succeeded
  ```

-
  ```
  mail from:<afnog@mydomain.example.com>
  ```

-
  ```
  rcpt to:<example@example.com>
  250 Accepted
  ```

Encrypting SMTP Sessions

- Sending password without encryption is a bad idea!
- SSL encryption requires a certificate for the server
- We will re-use the self-signed SSL certificate we generated for Apache earlier
- In production you should use a purchased SSL certificate, to avoid man-in-the-middle attacks
- Encryption on port 25 uses STARTTLS to start encryption
- Port 465 forces encryption without STARTTLS, but conflicts with some Cisco routers

Enabling SSL Encryption

- Copy the certificates from Apache:

  ```
  cd /usr/local/etc/apache22
  sudo cp server.* ../exim
  ```

- Edit the Exim configuration and uncomment:

  ```
  sudo vi /usr/local/etc/exim/configure
  ```

  ```
  tls_advertise_hosts = *
  tls_certificate = /usr/local/etc/exim/server.crt
  tls_privatekey = /usr/local/etc/exim/server.key
  daemon_smtp_ports = 25 : 465 : 587
  tls_on_connect_ports = 465
  ```

- Restart Exim to activate the changes

  ```
  sudo /usr/local/etc/rc.d/exim restart
  ```

Testing SSL Encryption

- Use swaks again to test that TLS encrypted connections work:
  - ***swaks --helo 1.2.3 --to afnog@localhost --auth LOGIN --auth-user afnog --auth-password afnog –tls***

    ```
    …
    <~  235 Authentication succeeded
    <~  250 OK id=1QRlDN-0000LL-0h
    ```
- Also test the SMTPS service on port 465:
  - swaks --helo 1.2.3 --to afnog@localhost --auth LOGIN --auth-user afnog --auth-password afnog –tls***c***

    ```
    …
    <~  235 Authentication succeeded
    <~  250 OK id=1QRlDN-0000LL-0h
    ```

AfNOG

Requiring SSL for Authentication

- Disable advertising the SMTP AUTH command when the session is not encrypted (chapter 33)
  - ***sudo vi /usr/local/etc/exim/configure***
  - Uncomment this line:

    -
      ```
      LOGIN
      …
      server_advertise_condition = \
          ${if def:tls_cipher}
      ```

  -
    ```
    swaks --helo 1.2.3 --to afnog@localhost --auth LOGIN
    --auth-user afnog --auth-password afnog
    *** Host did not advertise authentication
    ```

  -
    ```
    swaks --helo 1.2.3 --to afnog@localhost --auth LOGIN
    --auth-user afnog --auth-password afnog –tls
    <~  235 Authentication succeeded
    <~  250 OK id=1QRlDN-0000LL-0h
    ```

Next up: Troubleshooting

- ✓ Global (no name)
- ✓ Routers (decide what to do with mail)
- ✓ Transports (control how exactly it is delivered)
- ✓ Access Control (who is allowed to send mail)
- ✓ Authenticators (logging in to relay mail)
- ➢ **Troubleshooting (when things go wrong)**

Logs and Debugging

- The main Exim log files are:
  - */var/log/exim/mainlog* (everything)
  - */var/log/exim/rejectlog* (rejected messages only)
  - */var/log/exim/paniclog* (errors about lost messages)
- What do the logs say for a successful mail?
- Use exigrep to find messages matching an address, user or message ID:

```
sudo exigrep john /var/log/exim/mainlog
```

- What does it output? Why is it better than *grep*?

The Mail Queue

- When Exim accepts a message that it cannot deliver immediately, it is placed in the queue
- Stored in */var/spool/exim/input*
- Two files per message: *id*-D and *id*-H
- What do they contain? Have a look:
  - Put a message in the queue:

    -
      ```
      exim -odq afnog@mydomain.example.com
      This is a test
      .
      ```
  - Run `sudo mailq` or `sudo exim -bp` to see the message ID

The Mail Queue

- Viewing messages on the queue:
  - `sudo exim -Mvb <message-id>` (view body only)
  - `sudo exim -Mvh <message-id>` (view headers only)
  - `sudo exim -Mvc <message-id>` (view whole message)
  - `sudo exim -Mvl <message-id>` (view logs)
- Force a queue run, to see why the message is failing:
  - `sudo exim -v -qf <message-id>`

Where to Get Help

- The Exim Book
  - You should get a free copy this week
- The Exim Manual
  - http://www.exim.org/docs.html
- AfNOG Mailing List
  - http://www.afnog.org/mailinglist.html
  - Please subscribe to this list!
- Exim Users Mailing List
  - http://lists.exim.org/mailman/listinfo/exim-users
- The Aptivate Team!

AfNOG