

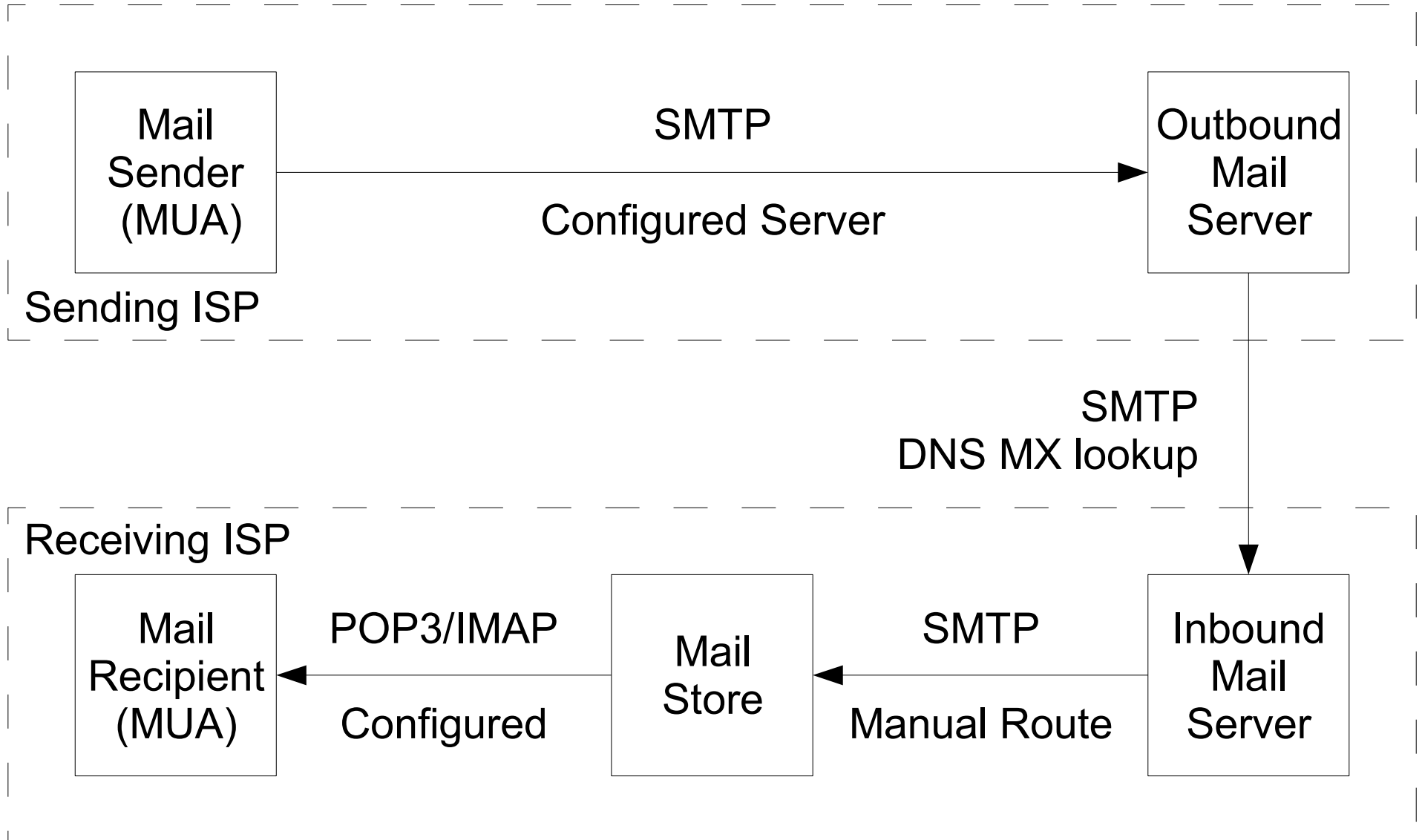
# Exim and Internet Mail

---

Chris Wilson  
Aptivate Ltd, UK  
AfNOG 2009

# How Email Works

---



# What is Exim

---

- ◆ Listens on port 25 (smtp)
- ◆ Accepts mail
- ◆ Delivers it somewhere

# Who uses Exim

---

- ♦ University of Cambridge, UK
- ♦ Energis Squared (formerly Planet Online), UK
- ♦ Shore.Net (large regional ISP in the Northeastern US)
- ♦ Esat Net (longest serving ISP in Ireland)
- ♦ Aptivate

# Why use Exim

---

- ♦ Flexible (lots of features)
- ♦ Reasonably secure
- ♦ Reasonably scalable
- ♦ Good debugging options
- ♦ Sane 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

# Installing Exim

---

- ◆ We will use “sudo” wherever root is required
- ◆ You do not need to be root for this tutorial
- ◆ Compile Exim from the ports tree:
  - ◆ *cd /usr/ports/mail/exim*
  - ◆ *sudo make |*  
*WITH\_MYSQL=yes |*  
*WITH\_CONTENT\_SCAN=yes |*  
*WITH\_AUTH\_RADIUS=yes |*  
*WITH\_RADIUS\_TYPE=RADLIB |*  
*EXTRALIBS\_EXIM=/usr/lib/libradius.so |*  
*WITH\_SASLAUTHD=yes |*  
*WITH\_SPF=yes |*  
*install clean*

# Replacing Sendmail with Exim

---

- Stop and disable Sendmail:
  - ***sudo /etc/rc.d/sendmail stop***
  - ***vi /etc/rc.conf***
    - ***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***



# Starting Exim

---

- Try the following commands:
  - ***sudo /usr/local/etc/rc.d/exim start***
  - ***sudo /usr/local/etc/rc.d/exim status***  
**exim is running as pid XXX**
  - ***sudo /usr/local/etc/rc.d/exim restart***
- Create */etc/periodic.conf* and add these lines:
  - ***daily\_status\_include\_submit\_mailq="NO"***
  - ***daily\_clean\_hoststat\_enable="NO"***



# 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?

# 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)

# 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 default configuration

---

- ◆ 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

# 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
- ♦ 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 `pcXX.sse.ws.afnog.org` ESMTD 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 `pcXX.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*

# 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
  - **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
- ♦ Try 1.2.3.19 and 1.2.5.4, what happens?

# 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

# 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)

# Routers

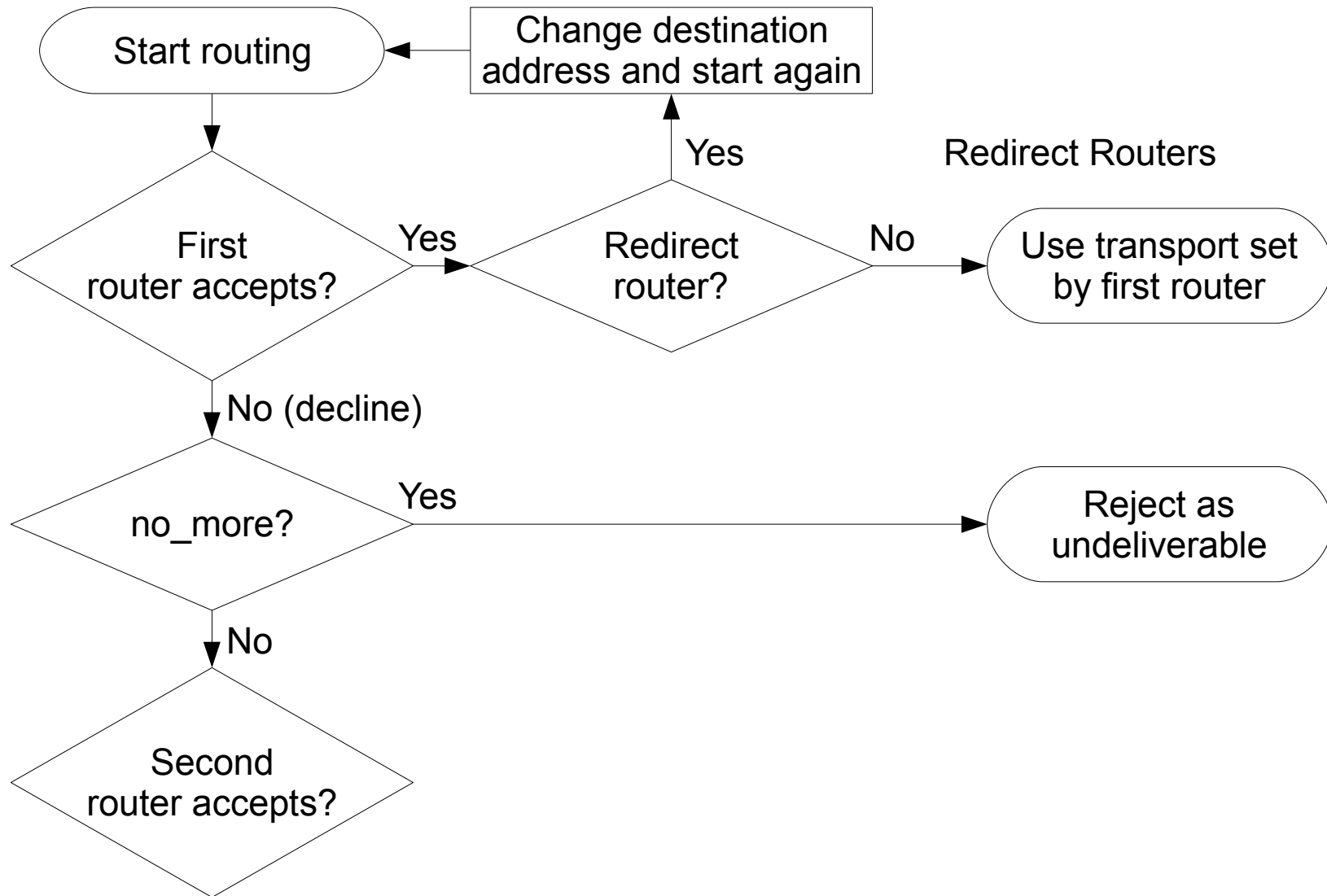
---

- ♦ 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
- ♦ Routers should not be used to reject mail except for undeliverable mail



# Routing Overview

---



# Anatomy of a Router

---

- ◆ Options control how the driver works
  - ◆ domains = +local\_domains
  - ◆ user = mail
- ◆ Specified driver is run
  - ◆ driver = redirect
- ◆ If the driver accepts, assign specified transport
  - ◆ transport = trotro (or matatu)

# 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**
  - ♦ only if destination domain is not in *local\_domains*
- ♦ **driver = dnslookup**
  - ♦ check that the destination domain has MX or A
- ♦ **ignore\_target\_hosts = 0.0.0.0 : 127.0.0.0/8**
- ♦ **no\_more**
  - ♦ if conditions match but router declines then bounce
- ♦ **transport = remote\_smtp**
  - ♦ if router accepts, then use *remote\_smtp* to deliver

# The *system\_aliases* Router

---

- ♦ **driver = redirect**
- ♦ allow\_fail
- ♦ allow\_defer
- ♦ **data =  $\{\text{lookup}\{\text{\$local\_part}\}\text{search}\{\text{/etc/aliases}\}\}$**
- ♦ user = mailnull
- ♦ group = mail
- ♦ file\_transport = address\_file
- ♦ pipe\_transport = address\_pipe

# The *userforward* Router

---

- ♦ `userforward:`  
    `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



# Testing 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***
    - ***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***
  - 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*
  - *manual\_router:*  
*driver = manualroute*  
*domains = manual.example.com*  
*route\_data = noc.sse.ws.afnog.org*  
*transport = remote\_smtp*
- *exim -bt foo@manual.example.com*  
foo@manual.example.com  
router = manual\_router, transport = remote\_smtp  
host noc.sse.ws.afnog.org [196.200.219.200]

# 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.sse.ws.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.sse.ws.afnog.org [196.200.219.200]***



# 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)

# 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
- ◆ 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 pkg\_add -r procmail*
- *vi /home/afnog/.procmailrc:*
  - *:0f*  
*| sed -e 's/foo/bar/'*
- *echo food | mail afnog*
- *tail -2 /var/mail/afnog*  
food
- *sudo vi /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
    - 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)

# 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

# Testing Callouts

---

- *sudo vi /usr/local/etc/exim/configure*
  - domainlist relay\_to\_domains = *rl.example.com*
  - require *message = Sender verify failed*  
verify = sender/*callout=120s*
  - require *message = Recipient verify failed*  
verify = recipient/*callout=120s*
- *exim -bh 1.2.3.4*
  - *mail from:<nonexist@pcXX.sse.ws.afnog.org>*
  - *rcpt to:<afnog@pcXX.sse.ws.afnog.org>*  
550 Sender verify failed
- *exim -bhc 1.2.5.4*
  - *mail from:<afnog@pcXX.sse.ws.afnog.org>*
  - *rcpt to:<nonexist@rl.example.com>*  
550 Recipient verify failed

# Blocking a specific sender/recipient

- 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:
  - ◆ *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](http://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.afnogws.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:
  - **deny spf = fail**  
**message = SPF check failed: \$spf\_smtplib\_comment**  
**log\_message = SPF check failed: \$spf\_result**



# 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

---

- ♦ 

```
drop ! hosts = :  
    ! hosts = 80.248.178.170  
    condition = ${if eq \  
        {$smtp_command_argument} \  
        {80.248.178.170}}  
    message = You are SO lying
```
- ♦ Catches people who say HELO 80.248.178.170 (my own IP address) but are not me!

# Bad Juju

---

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

# Assassinating Spam(mers)

---

- *sudo pkg\_add -r p5-Mail-SpamAssassin*  
Do you wish to run sa-update to fetch new rules [N]? *n*
- *cd /usr/local/etc/mail/spamassassin*
- *sudo cp local.cf.sample local.cf*
- *sudo vi local.cf*
  - *use\_pyzor 0*
  - *use\_razor2 0*
  - *skip\_rbl\_checks 1*
  - *use\_bayes 0*
- *sudo vi /etc/rc.conf*
  - *spamd\_enable="YES"*
- *sudo /usr/local/etc/rc.d/sa-spamd start*
- *spamc -R*
  - *subject: penis enlargement*
- press Ctrl+D to end message

# Filtering Mail through SpamAssassin

- ◆ Uncomment the following lines in the configuration:

- ◆ **deny** spam = nobody  
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 *exim -bs*:

- ◆ **exim -bs**  
**mail from:<>**  
**rcpt to:<afnog@pcXX.sse.ws.afnog.org>**  
**data**  
**message-id: abcd**  
**subject: BUY VIAGRA HERE!!!**  
  
**<html><p>Dear friend</p>**  
**<p>VIAGRA \$10.99</p>**  
**<p>RISK FREE</P></HTML>**  
**.**  
**quit**

# Installing Clam Antivirus

---

- *sudo pkg\_add -r clamav*
- *sudo freshclam*
- *sudo pw usermod clamav -G mail*
- *sudo vi /etc/rc.conf*
  - *clamav\_clamd\_enable="YES"*
  - *clamav\_freshclam\_enable="YES"*
- *sudo /usr/local/etc/rc.d/clamav-clamd start*
- *fetch http://noc.sse.ws.afnog.org/sse/exim/eicar*
- *clamscan eicar*  
/usr/home/afnog/eicar: Eicar-Test-Signature FOUND  
  
Infected files: 1

# Filtering Mail through ClamAV

---

- *sudo vi /usr/local/etc/exim/configure*
  - *av\_scanner = clamd:/var/run/clamav/clamd.sock*
  - *acl\_check\_data:*
    - *deny malware = \**
      - message = This message contains a virus \*  
*(\$malware\_name).*
- *cat eicar*
- *exim -bs*  
*mail from:<afnog@noc.sse.ws.afnog.org>*  
*rcpt to:<afnog@noc.sse.ws.afnog.org>*  
*data*  
*subject: test*  
*X50!P%@AP[4\PZX54(P^)7CC)7}\$EICAR-STANDARD-*  
*ANTIVIRUS-TEST-FILE!\$H+H\**  
*.*  
*550 This message contains a virus...*

# 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 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 sse*  
0: OK "Success."
  - *sudo testsaslauthd -u afnog -p wrong*  
0: NO "authentication failed"

# Enabling SMTP Authentication

---

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

◆ begin authenticators

➤ LOGIN:

```
driver = plaintext
server_prompts = <| Username: | Password:
server_condition = ${if saslauthd{{$auth1} \
{$auth2}}{smtp}}
server_set_id = $1
# server_advertise_condition = ...
```

➤ *exim -bs*

```
220 noc.sse.ws.afnog.org ESMTP Exim 4.69 ...
```

➤ *ehlo 0*

```
250-noc.sse.ws.afnog.org Hello afnog at 0
250-SIZE 52428800
250-PIPELINING
250-AUTH LOGIN
250 HELP
```

# Testing SMTP Authentication

---

- *sudo pkg\_add -r base64*
- *echo -n afnog | base64*  
YWZub2c=
- *echo -n sse | base64*  
c3Nl
- *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=*  
334 UGFzc3dvcmQ6
- *c3Nl*  
235 Authentication succeeded

# Using RADIUS for Authentication

---

- *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 the *openssl s\_client* command to make an encrypted SMTP connection to Exim:
  - *openssl s\_client -connect localhost:25 | -starttls smtp*  
250 HELP
  - *ehlo 0*  
250-AUTH LOGIN  
250 HELP
  - *auth login*  
334 VXNlcm5hbWU6
- ♦ Also test the SMTPS service on port 465:
  - *openssl s\_client -connect localhost:465*



# 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*
    - ◆ LOGIN:

```
server_advertise_condition = \  
    ${if def:tls_cipher}
```
  - *exim -bs*

```
220 noc.sse.ws.afnog.org ESMTP Exim 4.69 ...
```
  - *ehlo 0*

```
250-noc.sse.ws.afnog.org Hello afnog at 0  
250-SIZE 52428800  
250-PIPELINING  
250-STARTTLS  
250 HELP
```

# 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

---

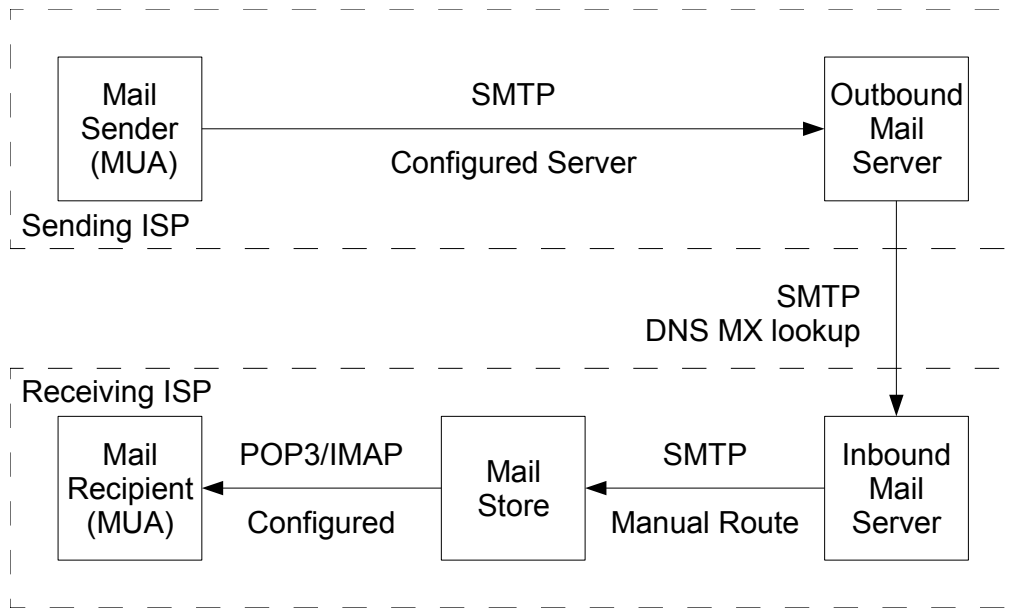
- ♦ The Exim Book
  - ♦ You should get a free copy this week
- ♦ The Exim Manual
  - ♦ <http://www.exim.org/docs.html>
- ♦ AfNOG Mailing List
  - ♦ <http://www.afnog.org/maillinglist.html>
  - ♦ Please subscribe to this list!
- ♦ Exim Users Mailing List
  - ♦ <http://lists.exim.org/mailman/listinfo/exim-users>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

# Exim and Internet Mail

---

Chris Wilson  
Aptivate Ltd, UK  
AfNOG 2009

# How Email Works





# What is Exim

---

- Listens on port 25 (smtp)
- Accepts mail
- Delivers it somewhere

## Who uses Exim

---

- University of Cambridge, UK
- Energis Squared (formerly Planet Online), UK
- Shore.Net (large regional ISP in the Northeastern US)
- Esat Net (longest serving ISP in Ireland)
- Aptivate

# Why use Exim

---

- Flexible (lots of features)
- Reasonably secure
- Reasonably scalable
- Good debugging options
- Sane 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

# Installing Exim

---

- We will use “sudo” wherever root is required
- You do not need to be root for this tutorial
- Compile Exim from the ports tree:
  - *cd /usr/ports/mail/exim*
  - *sudo make \*  
*WITH\_MYSQL=yes \*  
*WITH\_CONTENT\_SCAN=yes \*  
*WITH\_AUTH\_RADIUS=yes \*  
*WITH\_RADIUS\_TYPE=RADLIB \*  
*EXTRALIBS\_EXIM=/usr/lib/libradius.so \*  
*WITH\_SASLAUTHD=yes \*  
*WITH\_SPF=yes \*  
*install clean*

# Replacing Sendmail with Exim

---

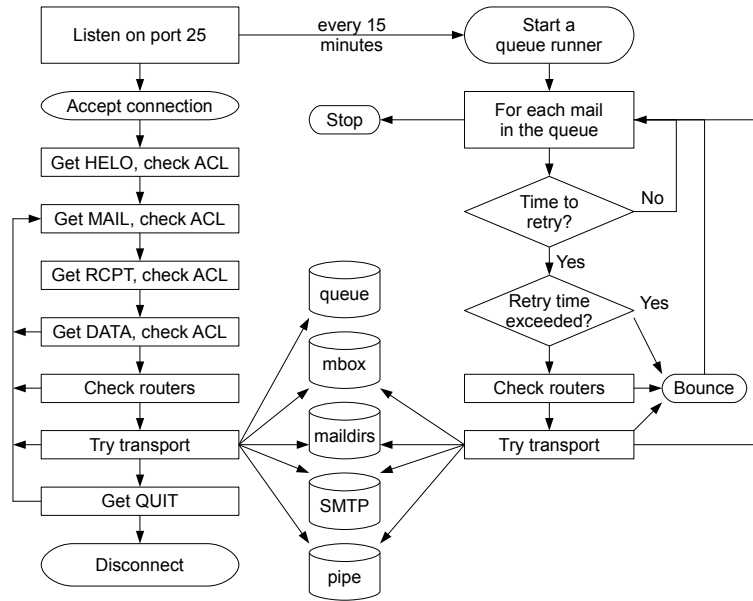
- Stop and disable Sendmail:
  - **`sudo /etc/rc.d/sendmail stop`**
  - **`vi /etc/rc.conf`**
    - **`sendmail_enable="NONE"`**
    - **`sendmail_submit_enable=NO`**
    - **`exim_enable=YES`**
- Edit `/etc/mail/mailer.conf` and change these lines:
  - **`sendmail /usr/local/sbin/exim`**
  - **`send-mail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

# Starting Exim

---

- Try the following commands:
  - ***sudo /usr/local/etc/rc.d/exim start***
  - ***sudo /usr/local/etc/rc.d/exim status***  
**exim is running as pid XXX**
  - ***sudo /usr/local/etc/rc.d/exim restart***
- Create */etc/periodic.conf* and add these lines:
  - ***daily\_status\_include\_submit\_mailq="NO"***
  - ***daily\_clean\_hoststat\_enable="NO"***

# Exim Overview





# 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?

# 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)

# 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 default configuration

---

- ♦ 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 ESMTTP 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

# 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
- 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 `pcXX.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 `pcXX.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*

# 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
  - **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
- ♦ Try 1.2.3.19 and 1.2.5.4, what happens?

# 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

## 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)

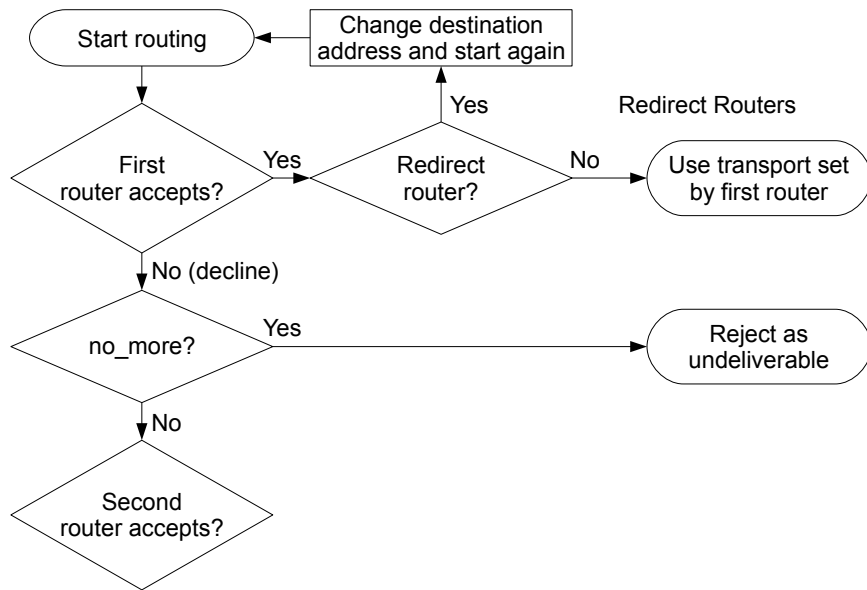
# Routers

---

- ♦ 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
- ♦ Routers should not be used to reject mail except for undeliverable mail

# Routing Overview

---



# Anatomy of a Router

---

- Options control how the driver works
  - domains = +local\_domains
  - user = mail
- Specified driver is run
  - driver = redirect
- If the driver accepts, assign specified transport
  - transport = trotro (or matatu)



## 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**
  - ♦ only if destination domain is not in *local\_domains*
- ♦ **driver = dnslookup**
  - ♦ check that the destination domain has MX or A
- ♦ **ignore\_target\_hosts = 0.0.0.0 : 127.0.0.0/8**
- ♦ **no\_more**
  - ♦ if conditions match but router declines then bounce
- ♦ **transport = remote\_smtp**
  - ♦ if router accepts, then use *remote\_smtp* to deliver

## The *system\_aliases* Router

---

- **driver = redirect**
- allow\_fail
- allow\_defer
- **data = `{lookup}{local_part}search{/etc/aliases}`**
- user = mailnull
- group = mail
- file\_transport = address\_file
- pipe\_transport = address\_pipe

# The *userforward* Router

---

- `userforward:`
  - `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

## Testing 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`
    - `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***
  - 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***
  - ***manual\_router:***
    - driver = manualroute***
    - domains = manual.example.com***
    - route\_data = noc.sse.ws.afnog.org***
    - transport = remote\_smtp***
- ***exim -bt foo@manual.example.com***  
foo@manual.example.com  
router = manual\_router, transport = remote\_smtp  
host noc.sse.ws.afnog.org [196.200.219.200]

# 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.sse.ws.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.sse.ws.afnog.org [196.200.219.200]**

## 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)

## 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
- 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 pkg_add -r procmail`
- `vi /home/afnog/.procmailrc:`
  - `:0f`
    - | `sed -e 's/foo/bar/'`
- `echo food | mail afnog`
- `tail -2 /var/mail/afnog`  
food
- `sudo vi /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
    - 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)



# 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 53

## 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

# Testing Callouts

---

- ***sudo vi /usr/local/etc/exim/configure***
  - domainlist relay\_to\_domains = ***rl.example.com***
  - require ***message = Sender verify failed***  
verify = sender/***callout=120s***
  - require ***message = Recipient verify failed***  
verify = recipient/***callout=120s***
- ***exim -bh 1.2.3.4***
  - ***mail from:<nonexist@pcXX.sse.ws.afnog.org>***
  - ***rcpt to:<afnog@pcXX.sse.ws.afnog.org>***  
550 Sender verify failed
- ***exim -bhc 1.2.5.4***
  - ***mail from:<afnog@pcXX.sse.ws.afnog.org>***
  - ***rcpt to:<nonexist@rl.example.com>***  
550 Recipient verify failed



## Blocking a specific sender/recipient

- 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 = bfsubmit@bfsubmit.com  
message = I hope you catch bird flu and die
- deny senders = \N^.\*mission2007.\*@dgroups.org\$\  
recipients = info@aidworld.org  
message = Please remove me from your list.

# Hate your neighbour?

---

- Add to your RCPT ACL:
  - *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](http://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.afnogws.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:
  - *deny spf = fail*  
*message = SPF check failed: \$spf\_smtp\_comment*  
*log\_message = SPF check failed: \$spf\_result*

# Blackmail

---

- deny      ! hosts = +relay\_from\_hosts  
            ! authenticated = \*  
            dnslists = zen.spamhaus.org  
            message = \$sender\_host\_address \  
                      blacklisted by Spamhaus\  
                      (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

---

- drop ! hosts = :  
! hosts = 80.248.178.170  
condition = \${if eq \  
  \${smtp\_command\_argument} \  
  {80.248.178.170}}  
message = You are SO lying
- Catches people who say HELO 80.248.178.170 (my own IP address) but are not me!

# Bad Juju

---

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



# Assassinating Spam(mers)

---

- > ***sudo pkg\_add -r p5-Mail-SpamAssassin***  
Do you wish to run sa-update to fetch new rules [N]? ***n***
- > ***cd /usr/local/etc/mail/spamassassin***
- > ***sudo cp local.cf.sample local.cf***
- > ***sudo vi local.cf***
  - > ***use\_pyzor 0***
  - > ***use\_razor2 0***
  - > ***skip\_rbl\_checks 1***
  - > ***use\_bayes 0***
- > ***sudo vi /etc/rc.conf***
  - > ***spamd\_enable="YES"***
- > ***sudo /usr/local/etc/rc.d/sa-spamd start***
- > ***spamc -R***
  - > ***subject: penis enlargement***
- > press Ctrl+D to end message

# Filtering Mail through SpamAssassin

- ◆ Uncomment the following lines in the configuration:

- **deny** spam = nobody  
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 *exim -bs*:

- **exim -bs**  
**mail from:<>**  
**rcpt to:<afnog@pcXX.sse.ws.afnog.org>**  
**data**  
**message-id: abcd**  
**subject: BUY VIAGRA HERE!!!**  
  
**<html><p>Dear friend</p>**  
**<p>VIAGRA \$10.99</p>**  
**<p>RISK FREE</P></HTML>**  
**.**  
**quit**

# Installing Clam Antivirus

---

- *sudo pkg\_add -r clamav*
- *sudo freshclam*
- *sudo pw usermod clamav -G mail*
- *sudo vi /etc/rc.conf*
  - *clamav\_clamd\_enable="YES"*
  - *clamav\_freshclam\_enable="YES"*
- *sudo /usr/local/etc/rc.d/clamav-clamd start*
- *fetch http://noc.sse.ws.afnog.org/sse/exim/eicar*
- *clamscan eicar*  
/usr/home/afnog/eicar: Eicar-Test-Signature FOUND  
  
Infected files: 1

# Filtering Mail through ClamAV

---

- *sudo vi /usr/local/etc/exim/configure*
  - *av\_scanner = clamd:/var/run/clamav/clamd.sock*
  - *acl\_check\_data:*
  - *deny malware = \**
    - message = This message contains a virus \*
    - (\$malware\_name).*
- *cat eicar*
- *exim -bs*
  - mail from:<afnog@noc.sse.ws.afnog.org>*
  - rcpt to:<afnog@noc.sse.ws.afnog.org>*
  - data*
  - subject: test*
  - X50!P%@AP[4\PZX54(P^)7CC)7}\$EICAR-STANDARD-*
  - ANTIVIRUS-TEST-FILE!\$H+H\**
  - .*
  - 550 This message contains a virus...*

## 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 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 sse***  
0: OK "Success."
  - ***sudo testsaslauthd -u afnog -p wrong***  
0: NO "authentication failed"

# Enabling SMTP Authentication

---

```
> sudo vi /usr/local/etc/exim/configure
  ♦ begin authenticators
  > LOGIN:
    driver = plaintext
    server_prompts = <| Username: | Password:
    server_condition = ${if saslauthd{${auth1} \
    ${auth2}{smtp}}}}
    server_set_id = $1
    # server_advertise_condition = ...

> exim -bs
220 noc.sse.ws.afnog.org ESMTP Exim 4.69 ...

> ehlo 0
250-noc.sse.ws.afnog.org Hello afnog at 0
250-SIZE 52428800
250-PIPELINING
250-AUTH LOGIN
250 HELP
```



# Testing SMTP Authentication

---

- › *sudo pkg\_add -r base64*
- › *echo -n afnog | base64*  
YWZub2c=
- › *echo -n sse | base64*  
c3Nl
- › *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=*  
334 UGFzc3dvcmQ6
- › *c3Nl*  
235 Authentication succeeded

# Using RADIUS for Authentication

---

- *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 VXN\cm5hbWU6
- *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 the *openssl s\_client* command to make an encrypted SMTP connection to Exim:
  - *openssl s\_client -connect localhost:25 \*  
*-starttls smtp*  
250 HELP
  - *ehlo 0*  
250-AUTH LOGIN  
250 HELP
  - *auth login*  
334 VXNlcm5hbWU6
- Also test the SMTPS service on port 465:
  - *openssl s\_client -connect localhost:465*

# 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***
    - LOGIN:

```
server_advertise_condition = \  
  ${if def:tls_cipher}
```
  - ***exim -bs***

```
220 noc.sse.ws.afnog.org ESMTP Exim 4.69 ...
```
  - ***ehlo 0***

```
250-noc.sse.ws.afnog.org Hello afnog at 0  
250-SIZE 52428800  
250-PIPELINING  
250-STARTTLS  
250 HELP
```

## 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/maillinglist.html>
  - Please subscribe to this list!
- Exim Users Mailing List
  - <http://lists.exim.org/mailman/listinfo/exim-users>
- The Aptivate Team!