

EMAIL SYSTEM

ADMINISTRATOR MANUAL FOR EMAIL-NG

Postfix, Dovecot, Amavisd, Spamassassin, ClamAV, openLDAP

THIS MANUAL IS STILL IN DEVELOPMENT

By Andreas Kasenides

ank@cs.ucy.ac.cy

Revision History

0.9

- Redefined the way mailing lists are created and managed to take Mailman into consideration.
- CentOS 7 considerations in several places

0.8 – Mailman and Webmail description. Many other description enhancements.

0.7 – EmailNG officially launched with mail.cs.ucy.ac.cy moved to theano 18/06/2014

0.6 – New generation set-up with relay, virtual hosts etc. started

0.5 – Greatly expanded and restructured

0.4 – Dual server appendix start, added more material to almost every section

0.3 – General system description,

0.2 – Minimal explanation of the basic concepts

0.1 – Started from the user manual

Last revision: 16. Feb. 2015

A Complete and Secure E-Mail Infrastructure system

Table of Contents

Introduction.....	5
The Target Environment.....	5
Objectives.....	5
General System Description.....	7
Server Groups Arrangement.....	8
Server functionality.....	10
Operating System Software Setup.....	10
General OS Setup and Software.....	11
Prerequisites.....	11
Setup on All Servers.....	11
Logging.....	12
OS Security.....	12
DNS Configuration.....	12
A records.....	12
PTR records.....	13
MX records.....	13
SPF records.....	13
TLS/SSL certificates.....	13
Mail Services Description.....	14
Types of Mail Servers.....	14
Front end systems.....	14
Back end systems.....	14
Client systems.....	14
Functionality Summary Set up.....	14
Access Control Model.....	16
Authentication, Names, Aliases, Groups, Lists.....	17
General Description.....	17
Authentication.....	18
Names and Aliases.....	18
Groups and Lists.....	19
LDAP schema.....	22
Mail Storage.....	22
Quota with Dovecot LDA.....	23
E-Mail Software Setup.....	23
Setting up Postfix.....	24
Front-End Relay (theano, elysso).....	24
Back-End (gorgo, mirto).....	26
Setting up Dovecot.....	26
Front End (IMAP PROXY or Dovecot Director) Configuration (theano, elysso).....	27
Back-end Configuration (gorgo, mirto).....	27
Summary table.....	27
Quota.....	28
Sieve Filtering and ManageSieve.....	28
The Dovecot Configuration (dovecot -n).....	29
Virus and SPAM Control Setup.....	30
Strategy.....	30
AMAVISD.....	31
Install and configure.....	31
Postfix Integration.....	32
SpamAssassin.....	32
ClamAV.....	32
Message Quarantine.....	32

Releasing Messages.....	32
Managing/cleaning the quarantine folder.....	32
Security.....	32
Server OS security.....	32
Network Protection.....	33
Server protection.....	33
Fail2ban.....	33
Session security.....	33
Mail server protection.....	34
Back Scatter E-Mail.....	34
Mail System Clients.....	34
Pine.....	34
WebMail Servers.....	34
Web mail client and server configuration.....	34
Mailing List Server.....	35
Administration.....	35
General.....	35
Critical OS checks.....	35
OS upgrades.....	36
Postfix.....	36
Dovecot.....	36
Performance.....	37
Fault tolerance.....	37
Backup.....	37
Monitoring the Mail system.....	37
Logging.....	37
Mail messages.....	38
Nagios Monitoring.....	38
Statistics.....	39
Message cleanup.....	39
Synchronizing a Dual server configuration.....	39
Upgrades.....	40
Troubleshooting.....	40
Testing SMTP.....	40
Testing IMAP.....	40
Testing POP3.....	40
Appendix A: Compiling and Installing from Sources.....	42
Prerequisites.....	42
Postfix – Compile and Install.....	42
Add Postfix to the reboot start-up procedures.....	43
Dovecot – Compile, Install.....	44
Prerequisites.....	44
Pigeonhole Compile and Install.....	45
Appendix B: Future Work.....	46
Future Storage.....	46
Appendix C: Notes and Pending Issues.....	48
Appendix D: Helper Scripts.....	50
Temporary user management scripts.....	50
Mail clean up script.....	50
Appendix E: Configuration Listings.....	52
Fail2ban.....	52
SSHD (/etc/sshd_config).....	52
Appendix F: References.....	53
Relevant RFC reference.....	53
Appendix G: Random Notes.....	54
Troubleshooting Errors.....	54

Introduction

THIS MANUAL CONTAINS ESSENTIAL INFORMATION FOR ADMINISTRATORS TO GUIDE IN THE INSTALLATION, OPERATION AND TROUBLESHOOTING OF THE MAIL SYSTEM AT THE DEPARTMENT OF COMPUTER SCIENCE.

THIS IS NOT A COMPLETE MANUAL OF EITHER MAIL SYSTEMS OR THE SOFTWARE USED. SEE THE BIBLIOGRAPHY FOR LINKS ON MORE INFORMATION ABOUT THE SYSTEMS AND SOFTWARE USED.

Email systems in the academic computing environment are unique. For one they are in a constant state of flux. Students come and go at a relatively steady rate all year round (with peaks around semester starts). These users are not all at the same level of expertise.

This manual shows how primarily Postfix, Dovecot and a group of popular Open Source Software (OSS) can be used to set up a fast and secure mail system. It should be noted that this is a relatively complex system and there are countless ways to install such an environment. The description here concerns the Computer Science department mail system only.

The reader is expected to extract enough information from this document to understand in sufficient detail the set-up in order to be able to operate, troubleshoot and possibly customize it. The reader is expected to be at an advanced stage of Unix knowledge.

Postfix and Dovecot make up the the basic nucleus of software and Amavisd, Spamassassin, ClamAV, Sieve, the Horde/IMP web mail suite and the RoundCube Webmail application provide important facilities for a modern mail system:

The Target Environment

The target environment is a complicated multi-user environment. It is complicated because users are at various expertise levels, their primary location is not well defined, moving in and out of their primary location, often based in remote locations such as home offices and on trips in far away lands for many days. They are, sometimes, stranded in strange lands with nothing more than a slow Internet connection and mostly a browser to work with.

Users use their own favorite e-mail devices and clients (not just web mail) and use them in the strangest ways possible and on the strangest machines and OS combination available.

Welcome to the academic environment.

Objectives

The objectives of the e-mail infrastructure system are:

- Create an email environment with advanced capabilities fit to support a complex and mobile user environment geared towards the academic environment
- Robustness, fault tolerance and availability are crucial for an e-mail system. **Ability to respond to failures in less than a few hours and ability to service systems without loss of functionality.**

- Expand-ability: ability to add more servers or to split functionality among servers (mail delivery functionality, users' access, SPAM filtering, virus filtering etc)
- Security at every level (SMTP operations, users' access, internal communications will be encrypted)
- Speed is of great importance due to the present day large mailboxes generated by multimedia messages and the necessity for clients to query servers at very small and regular intervals.
- Particular attention is paid to robustness, availability and control. No email should be lost. Ability to track server actions as soon as remote clients or servers contact it.
- Support the email functions of regular and advanced users
- Support a diverse community of entry level students and expert faculty and staff
- Integrate with it virus and spam control measures
- Give the ability to the user to control some email functionality like SPAM control, filtering, password changes, vacation auto-replies, mail sorting using filters, email redirecting, folders etc
- Interface independence. Support a variety of clients in both POP and IMAP configurations. Support for the largest possible number of E-Mail clients including IMAP, POP and Web mail in their plain and secure versions (must be tested with Thunderbird, Outlook/Live, Pine, Horde/IMP, Roundcube).
- Low cost: Based on Commodity off-the shelf Hardware/Virtual Servers and 100% Open Source Software.
- Eliminate the use of custom scripts (like /Mail/common/Mail/scripts we used in the past) and replace them with native software (Postfix, Dovecot, Sieve, LDAP) operations.
- Vendor independence. OSS provide the ability to be free from custom applications. No lock-in with any vendor.

General System Description

The general hardware setup of the previous mail system is presented in figure 1. This system made use of storage in external and internal devices mostly via simple network file system (NFS) mounts. It consisted of Postfix, Dovecot, MailScanner, ClamAV and Spamassassin software. It also made use of external SPAM databases.

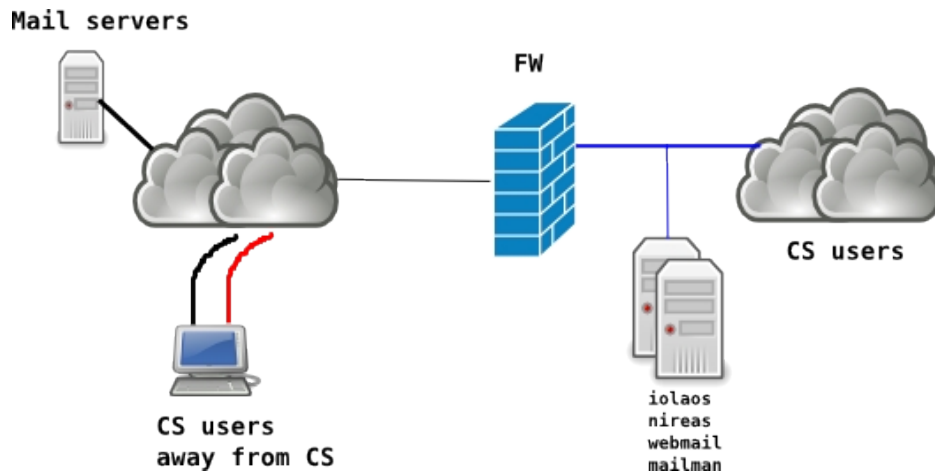


Fig. 1: The previous email system

The new generation email system “hardware” set up is presented in Fig. 2. A functional presentation of the same system is shown in Fig. 3. The server machines shown are fully virtualized hosts operating from two different virtual/hardware hosts.

The system design is based on a split front-end, back-end design. In this design there are front-end relay/proxy/filter servers and back-end mail storage management servers. All servers and services are active at all times.

The hardware is based on dual server system (total of four servers) with every function found in two places. Both system couples are active at the same time. The objective is to create a dual-server system where failure or scheduled unavailability of any of the servers will not affect the operation of the entire system.

The major characteristics of the system is its enormous flexibility, its attention to security and fault tolerance.

Figure 3 presents the system from an email protocol functionality perspective. It shows how each protocol traffic (SMTP, IMAP etc) are routed through the email network to reach its destination.

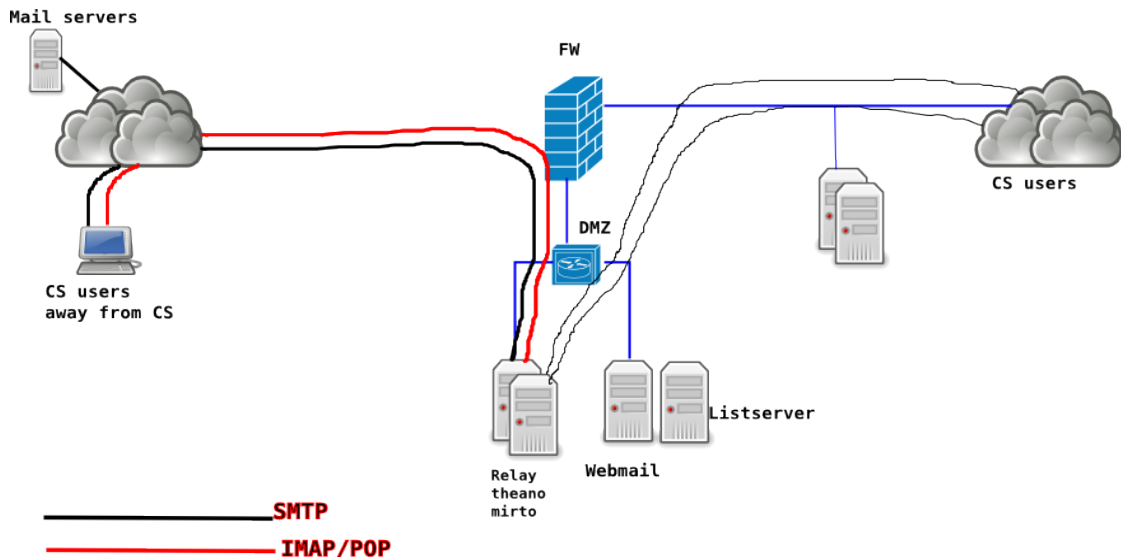


Fig. 2: General View – Physical and Functional Description

The system software continues the previous strategy of being interface independent. The choice of interface is left to the user. The system just provides the necessary interfaces for the user to make use of it. The only interface provided is the web mail applications which are installed on independent servers. This is just a matter of user convenience than a system choice. Other web interfaces can be used.

Fig. 3 show the functional view of the system. Two distinct perspectives are shown on how the system works: user and server clients. The user perspective shows what happens when an email is sent or retrieved by a user including the web mail application. The server perspective shows how the system works when a message is received or sent by the server components. For both views hardware and software involved are shown including the processes and the message storage systems are accessed.

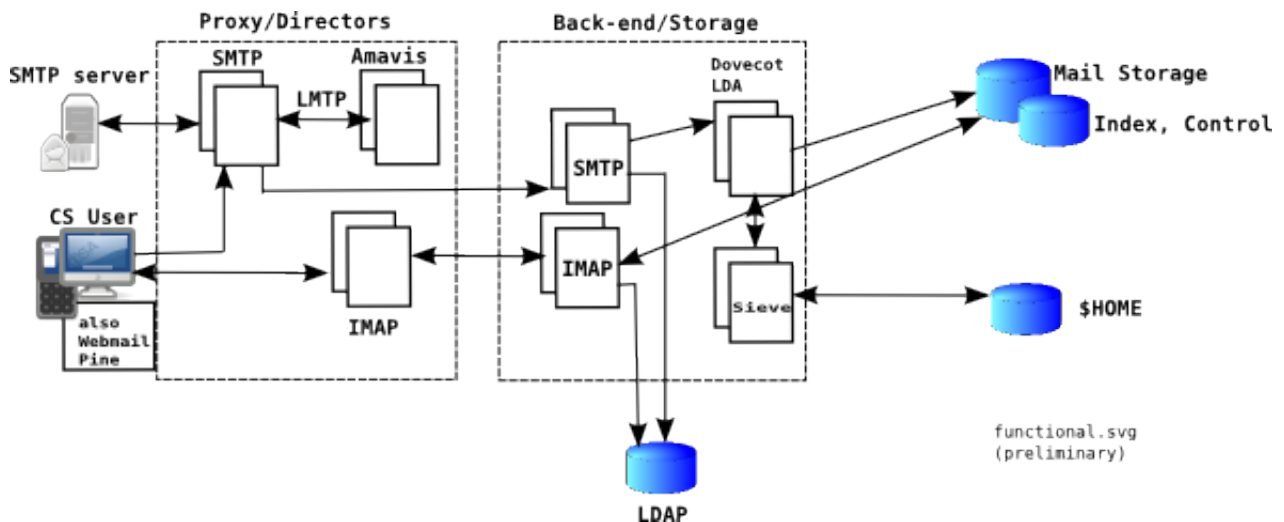


Fig. 3: Functional

Server Groups Arrangement

The system is split into two groups of services/servers: the front end servers and the back end servers.

Table 1: Server groups general configuration

	Functionality	Server name	VHOST	Notes
1	SMTP relay #1 Dovecot front end director #1	theano	iraklis	194.42.17.130 (DMZ#2)
2	SMTP relay #2 Dovecot front end director #2	elysso	danae	194.42.17.141 (DMZ#2)
3	SMTP back end #1 Dovecot back end #1	gorgo	iraklis	10.16.1.113
4	SMTP back end #2 Dovecot back end #2	mirto	danae	10.16.1.114
5	WebMail Mailing List-server (Dovecot and SMTP client)	thalia	danae	194.42.17.136 (DMZ#2)

Server functionality

	Theano Relay #1	Elysso Relay #2	Gorgo Back end #1	Mirto Back end #2	Thalia Webmail	Ariadne Listserver (obsolete)
SMTPD (receive) (Postfix MX - port 25)	✓	✓			✓ (from relays only)	✓ (from relays only)
SMTPD (receive submission) (Postfix - port 587)	✓	✓				
SMTPD user verification <i>reject_unverified_recipient</i>	✓	✓				
SMTP Relay (send) (front end - LMTP)	✓	✓			✓ (SMTP relay to nireas)	✓ (SMTP relay to nireas)
IMAP front end (Dovecot director/proxy) – port 143 Managesieve proxy – port 4190	✓	✓				
SMTP Back end Receiver (LMTP) Sieve (Global Sieve) (Users Sieve) using Dovecot LDA			✓	✓		
SMTP out going (postfix)	✓	✓				
IMAP back end Dovecot - LDA - dsync - Managesieve back end – port 4190			✓	✓		
AmavisD - SPAM - virus	✓	✓				
WebMail client					✓	
Mailing List Manager (mailman) client					✓	✓

Table 2: Server services summary

Operating System Software Setup

The OS described here is the RedHat/CentOS line of distributions. The instructions in this guide are based largely on the CentOS 6 system. The Linux Installation guide (CS-wiki) is followed for initial OS setup.

The email system assumes that there is a working Linux system available with a proper connection to the

network (via the individual virtual hosts) and proper access to Internet. Each virtual machine is placed in its proper network as indicated in figure 3 and detailed in Table 1.

Of particular importance for any email server are the following:

- Proper network access provided to the server especially if behind a firewall. See the Security section for the required TCP/UDP ports that need to be open on the firewall and their functionality for proper communication between servers.
- The server has a proper DNS entry both forward (A record) and reverse (PTR record) especially for outward facing systems
- Debug utilities (dig, nslookup, telnet etc) are available

General OS Setup and Software

Prerequisites

It is assumed that the latest CentOS is available. Follow the wiki instructions for setting it up.

Below are specific software that need to be addressed:

Setup on All Servers

- The general set-up is based on the CentOS minimal install. Follow the instructions in https://wiki.cs.ucy.ac.cy/index.php/OTY_Internal:Linux_Installation_from_MinimalInstall and https://wiki.cs.ucy.ac.cy/index.php/OTY_Internal:Linux_Installation in order to install and setup the software.

- File systems should be changed to

- /var – 10GB
- /sys-data – 10GB

- **General Software to install and configure on all systems:**

Services should be started using `chkconfig <service> on`

- yum repositories set-up
 - Note: need to check this every time a yum upgrade is done and remove/move CentOS-Base.repo.
- tcsh with link from /usr/bin/tcsh to /bin/tcsh
- development software to compile software (postfix, dovecot etc)
 - gcc, make
- ntp
 - See Administration/NTP for setup
- cronie
 - NOTE: PROBLEM: cronie requires sendmail, procmail, hesiod (!!!) make sure you disable sendmail when you install postfix. See the POSTFIX install below.
- sysstat, tcpdump, traceroute – troubleshooting software
- wget, vim, man, rsync
- logwatch
- syslog (rsyslog) initial setup
 - syslog should be enabled at this really stage to get messages that help in debugging. See the Administration section for more elaborate configuration.
- logrotate
 - See Administration/Logging for setup
- openssh-clients

- fail2ban, jwhois (do restorecon /sbin/iptables* to fix an selinux problem)
 - See Administration/fail2ban for setup
- nsswitch configurations
 - aliases: files
 - automount: files
 - comment out publickey setting
- crypto-utils (if generating keys locally) (??? - openssl is better and does not require additional packages)
- openldap-clients (ldapsearch) for debugging purposes (only on back-ends)
- Arrange for OS back ups (see also administration)
- create a local user <x>system as described in Linux installation.
 - useradd --home /home/<x>system <x>system
 - *Note that further user authentication is required and described in the*
- Debugging and testing software
 - telnet
 - bind-utils (dig etc)

Logging

Producing proper logs is of critical importance in setting up a complex system. Logs will discover mundane errors like mistyping and save your day. Enable logging at this early stage. See also the Administration section.

Make sure that you have enabled the syslog (rsyslog service) facilities since it is being used extensively. Make sure that logrotate package is installed and optionally logwatch is installed.

Pay particular attention to the logging configuration for each application installed since each one has different configuration parameters. In the initial phases of the installation enable debug logging to produce maximum information. This can be disabled when operations are satisfactorily stable.

OS Security

As soon as the OS is working enable security for the OS. This is particularly important for outward facing systems exposed to all kinds of threats. Visit Server OS security. Do not overlook or postpone this.

DNS Configuration

DNS plays a crucial role in mail systems. Without proper naming the email system will not work properly.

By its nature mail servers make extensive use of the DNS services.

[Consider a DNS caching server on each mail server]

A records

- theano
- elysso

- mail.cs.ucy.ac.cy 194.42.16.130 (theano) [eventually this will resolve to two]
 - mail1.cs.ucy.ac.cy 194.42.16.130
 - mail2.cs.ucy.ac.cy 194.42.16.141 (mirto)
 - mail3.cs.ucy.ac.cy 194.42.16.130
- 194.42.16.141
- *this is for testing purposes only – users should not make use of it*
 - gorgo.in
 - mirto.in
 - *these are only used internally – no user usage*

PTR records

A must for outgoing servers per RFC

MX records

MX records tell SMTP delivery agents which machines receive mail for our domain.

- Load balance MX records by giving the same weight
 - @ IN MX 100 mail1.cs.ucy.ac.cy.
 - @ IN MX 100 mail2.cs.ucy.ac.cy.

SPF records

1. Add the following SPF records to the cs.ucy.ac.cy domain

@	IN	TXT	"v=spf1 mx ip4:194.42.17.130 -all"
@	IN	SPF	"v=spf1 mx ip4:194.42.17.130 -all"

2. In every other domain we are authorized (hosting) for ex. in.cs.ucy.ac.cy

@	IN	TXT	"v=spf1 -all"
@	IN	SPF	"v=spf1 -all"

TLS/SSL certificates

TLS/SSL certificates are used by both Postfix and Dovecot to enable the TLS/SSL software to encrypt sessions. Encryption is used throughout the communications sessions. It provides security for both the user and the transfer of data especially when there is a need to transfer user names and passwords over the network.

The same certificates are used by Postfix and Dovecot.

All certificates/keys are placed in /etc/pki/tls/{certs, private} respectively. Related files are kept under /root/certs.

Also under /root/certs or /root/mail-certs we keep the original certificates for reference. Key are kept as above for protection.

Certificates and keys are created and signed by the Terena CA via the Cynet procedures.

See the [Setting up Postfix](#) and [Setting up Dovecot](#) sections on how to install them for each one.

Mail Services Description

Generally speaking, the hardware plays only a minor role in the management of the email infrastructure. Unless we are trying to set up an extremely high volume mail system, there is no need for exotic and ultra fast hardware. Of course, the faster the systems employed the better. Often memory availability plays a much more crucial role than the CPU speed. Also, email systems are I/O intensive and therefore good storage subsystems help greatly. Having said that, it is obvious that inadequate hardware systems for the job will suffer under load and produce unacceptable delays.

The software employed in the email infrastructure is characterized by spawning multiple processes to achieve their tasks. Examples of these are Postfix, Dovecot, Amavisd, ClamAV, Spamassassin. Especially during high loads the system should be capable to spawn thousands of processes and service them in an efficient manner. Therefore memory and I/O are quite important.

Types of Mail Servers

There are two types of servers. They are characterized by their location but also by their functionality. Front end systems are exposed to the outside network while back end systems are internal to the network.

Front end systems

- SMTP
- SMTP relays
- IMAP proxies
- Message Filters (SPAM, Virus etc)

Back end systems

- SMTP storage servers
- IMAP storage servers

Client systems

Webmail, Mailman

Functionality Summary Set up

For our set up we use four virtual machines for mail services and an additional host for the Webmail and Mailman applications which also functions as an active mail server. See Table 1 and Appendix E.

Server Functionality

This is a comprehensive high level detail record of how each system is set-up.

	Function	TCPIP port	Front-end (theano, elysso)	Back-end (gorgo, mirto)	Webmail, MMailman	Comment
1	SMTP/Server	25	√			
2	SMTP/Submission	587	√			
3	SMTP:Over quota check		√			
4	IMAP	143				
5	SASL auth: 10001			√		
6	Amavisd	10024	√			
7	12340					
8	12341					
9	12342					
10	12343					
11	12344					
12	12345					

Table 3: Server Function Listing

SMTP Relay (front end) theano, elysso:

- sit in the DMZ
- receive mail on port 25 (MX) from other SMTPs
- receives mail on port 587 (submissions) from OUR users
- Email Security
 - before queue filtering
 - reject mail from unknown (no reverse IP) hosts
 - reject mail destined to unknown local users (*verify*)
 - after queue filtering
 - SPAM and VIRUS detection
 - Blacklisting, Greylisting, rate limiting
 - Postfix PolicyD
- Check for local user over quota and reject receipt
- (If everything is OK) Relay to destination back end SMTP servers for delivery

Internal SMTP (back end) gorgo, mirto:

- receives incoming messages from SMTP relays (theano, elysso)
- answers user verification requests from front end SMTPs
- uses Dovecot-LDA to deliver local messages to the message store
 - Dovecot-LDA uses Sieve to implement server filtering
 - Dovecot-LDA indexes messages at delivery
- stores and manages messages in the message store(s)

- **synchronize message store to alternative location (dsync) (pending)**
- uses the SMTP front end as a smart host relay (when needed) to send messages to external destinations (for example in case of failure to deliver messages ie message storage problems)

IMAP Director (front end) theano, elysso:

- sits in the DMZ as a widely accessible system
- IMAP proxy
 - receives IMAP connection requests from inside or outside
 - proxies IMAP (forwards) requests to the inside IMAP back-end servers
 - implements IMAP load balancing (Dovecot Director)
- Managesieve proxy
 - receives Managesieve protocol connection requests (port 4190) from inside or outside
 - proxies (forwards) requests to the inside IMAP back-end servers

Internal IMAP (back end) gorgo, mirto:

- sits in the internal network
- services requests from the IMAP proxies
- manages the message store(s) for users
- manages the Managesieve protocol for user requests to set-up Sieve filters

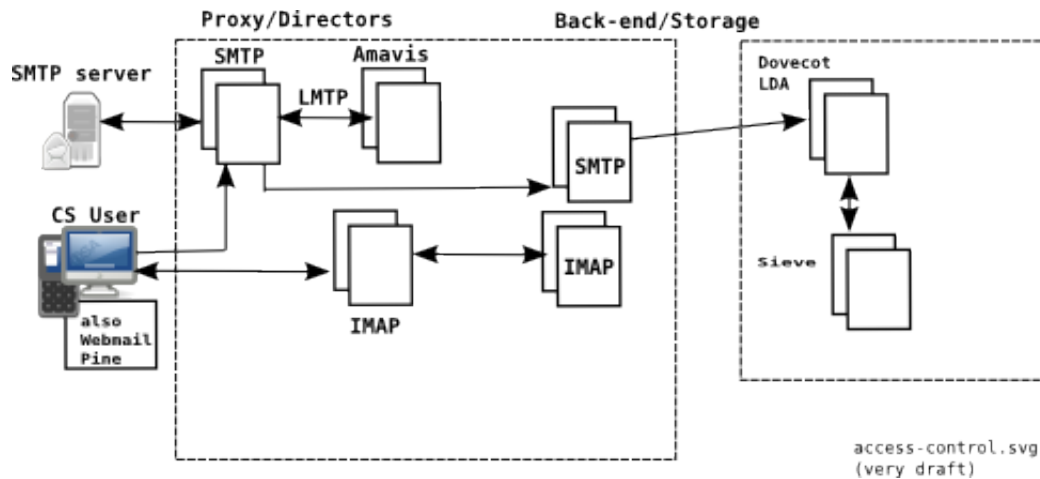
WebMail and Mailman list server

- receives message from the front end relays ONLY
- relays messages (SMTP client) to outgoing servers ONLY
- uses Mailman to manage mailing lists

Access Control Model

The access control model specifies the high level concepts of how the entire system allows access to its functions. It specifies how messages are either accepted for further processing and delivery or rejected *This is very preliminary.*

- *postfix/access*
- *postfix user verification (reject_unverified_recipient)*
- *postfix/overquota*
-



Authentication, Names, Aliases, Groups, Lists

Email delivery cannot exist without user names, which form (together with the domain) the final delivery address destination. User names have a dual role:

- to authenticate the user and provide proper access to the system,
- to form the final address which serves as the destination of messages.

General Description

Relay SMTP hosts (theano, elysso) need to have access either to the list of valid users or be able to verify the existence of all the names and aliases (for single or multiple users ie groups) we support. This is essential in order to reject incoming mail at the very early stage ie entry point such that messages will not need to be rejected at a later stage (on back-end machines) with a non-receipt reply producing back-scatter email. **Any address not verified as local and not belonging to a different domain is rejected as an invalid user.**

Since theano, elysso are exposed to the outside world the decision has been made NOT to give them direct access to the list of names (LDAP) but to allow them to verify the existence of the names via the Postfix VERIFY facility. Front end systems do not care about resolving the aliases just the fact that it is a valid name for CS. Our user name DB and the preparation of aliases/groups are internal (LDAP and custom scripts). There are two ways to allow access from relay systems:

- the legacy way: the names and aliases are prepared on internal systems (nireas) and forwarded via a system of cron scripts so that we do not expose our LDAP DB. *This system is not described in any detail since it is being phased out*
- a new verification system: names and aliases are verified by the relay systems via the **VERIFY Postfix facility essentially using the LDAP remotely and indirectly**
 - `smtpd_recipient_restrictions =`

```
.....
reject_unverified_recipient
.....
```

These two systems will work simultaneously until the legacy system becomes obsolete. For now the legacy system takes precedence over the LDAP system but this will eventually change.

Back-end SMTP hosts have complete access to both the legacy lists and the LDAP DB and authentication services. Therefore they know where each alias points to ex. andreas.kasenides --> ank or cspg --> all pg students. This is possible through both the lists maintained manually or from the LDAP DB to which they have direct access. Back-end systems provide verification services to the relay servers as described above.

Authentication

Authentication and authorization gives users privileges that enable them to make use of the system in a controlled manner. It also allows administrative users to access the OS. *Note that authentication exists only on back-end systems. Front-end systems VERIFY existence of users or proxy authentication to back end systems.*

It is (currently) based on

- SSSD which is configured to make use of the central LDAP (these are the so called system users)
- Home directories are available for these users
- Setup on SMTP back-end and IMAP/POP back-end (gorgo, mirto)
 - Install sssd
 - configure
 - /etc/sss/sss.conf
 - /etc/pam.d/system-auth
 - /etc/pam.d/password-auth
 - LDAP Authentication system: make sure that authentication works before proceeding (see the Wiki pages)
 - our authentication (for now) is based on SSSD therefore a correct configuration of sssd/pam/ldap is a must (install sssd).
 - With this setup users are considered “system users” as far as Dovecot is concerned which means that there are home directories on the local scale (ie NFS mounts)
 - Mail Storage - Mount the mail storage area (/Mail)
 - \$HOME storage – Mount the Home Directories storage (all)
 - Install keys/certificates. The certificates will be used by both Postfix (for TLS SMTP submission of SASL authentication) and Dovecot (for encrypted IMAP sessions). See the section below. (Do we use the same keyw for DKIM??)
 - Disable authconfig application to avoid destroying setup by accident
 - `chmod -x /usr/sbin/authconfig (??)`
 - test authentication
 - `getent passwd <user>`

It should be noted that the system does not make use of virtual users ie users that exist only for the purposes of email. All CS users are defined in the LDAP repository. Also all groups will eventually be defined in the LDAP repository.

Names and Aliases

Aliases are extensively used in mail systems. This is an easy way to receive messages on more

humanly readable addresses ex. george.andreou@cs.ucy.ac.cy rather than gandre01@cs.ucy.ac.cy. Aliases have a one-to-one correspondence ie an alias points to one user name.

For the CS department, each user:

- has a default user name that is identical to his/her log in name (ex. gandre01). This cannot be changed and it corresponds to his/her log in name given by the University registration system. The user can receive messages at gandre01@cs.ucy.ac.cy.
- automatically gets an alias as defined at the central University systems ie andreou.george@cs.ucy.ac.cy. *Note that the user also has a corresponding alias andreou.george@ucy.ac.cy which can receive messages through the central email systems.*
- At the CS dept email system ONLY the user can have one additional alias with a minimum of 10 characters (this is done to ensure that there is no conflict with central system names) which is only valid on the CS local systems and is user defined provided it does not conflict with already existing aliases or user names. For example george.andreou@cs.ucy.ac.cy.

Aliases used to be in legacy text files but are now being transferred to LDAP repository as explained above.

Groups and Lists

For the mail system, groups and mailing lists are almost identical since an alias (a different name) is established that is used to redirect messages to a group of people rather than a single one.

There are three ways to configure mailing groups/lists:

- the legacy way: this method used plain text files (/etc/aliases) and scripts to retrieve list memberships from our user DB (either system defined or from LDAP). This method is becoming obsolete. See below for a minimal description.
- LDAP based definitions. Groups/Lists are defined in LDAP and the Postfix server retrieves the definitions when needed.
- Mailman Lists. This method is described below.
- For our purposes there are system groups which are defined in the LDAP system and the corresponding information is obtained from there. Example csstaff.

The Legacy method

The following configuration exists on back-end machines which resolve the automatically generated groups for Postfix. This is the legacy configuration that is slowly being migrated to a more dynamic set-up with the LDAP repository as explained above.

The respective /etc/mailaliases/aliases files are generated on nireas by /etc/cron.daily/autoaliases and transferred.

```
#####  
### Temporary aliases and groups  
### eventually these will migrate to a better  
### set up like LDAP  
### support: :include:/etc/groups-aliases/support  
#####
```

```

##helpdesk:      support
representatives: :include:/etc/mailaliases/representatives
taassists: :include:/etc/mailaliases/taassists
healthware: :include:/etc/mailaliases/lists/healthware
labsupervisor: :include:/etc/mailaliases/lists/labsupervisor
cyta-announce: :include:/etc/mailaliases/lists/cyta-announce
freshmen: :include:/etc/mailaliases/aliases/freshmen
sophomores: :include:/etc/mailaliases/aliases/sophomores
juniors: :include:/etc/mailaliases/aliases/juniors
seniors: :include:/etc/mailaliases/aliases/seniors
undergrad: :include:/etc/mailaliases/aliases/undergrad
cspg: :include:/etc/mailaliases/aliases/cspg
raadmin: :include:/etc/mailaliases/aliases/raadmin
csassist: :include:/etc/mailaliases/aliases/csassist
csresearch: :include:/etc/mailaliases/aliases/csresearch
csfaculty: :include:/etc/mailaliases/aliases/csfaculty
csvisiting: :include:/etc/mailaliases/aliases/csvisiting
csmcsc: :include:/etc/mailaliases/aliases/csmcsc
cspgd: :include:/etc/mailaliases/aliases/cspgd
csstaff: :include:/etc/mailaliases/aliases/csstaff
support: :include:/etc/mailaliases/aliases/support
cssupport: :include:/etc/mailaliases/aliases/support
cstspecial: :include:/etc/mailaliases/aliases/cstspecial
csall: :include:/etc/mailaliases/aliases/csall
csstudents: :include:/etc/mailaliases/aliases/undergrad,:include:/etc/mailaliases/aliases/cspg
cssecretaries: :include:/etc/mailaliases/lists/secretaries
evoting: :include:/etc/mailaliases/lists/evoting
egee-users: :include:/etc/mailaliases/lists/egee-users
etaps10_satellite_events: :include:/etc/mailaliases/lists/etaps10_satellite_events
graduates: :include:/etc/mailaliases/lists/graduates
scrat-group: :include:/etc/mailaliases/lists/scrat-group
smartp2p: :include:/etc/mailaliases/lists/smartp2p
smartlib: :include:/etc/mailaliases/lists/smartlib
socialelectricity: :include:/etc/mailaliases/lists/socialelectricity
smartlab: :include:/etc/mailaliases/lists/smartlab
personaweb: :include:/etc/mailaliases/lists/personaweb
commonsense2013: :include:/etc/mailaliases/lists/commonsense2013
reconlife: :include:/etc/mailaliases/lists/reconlife

```

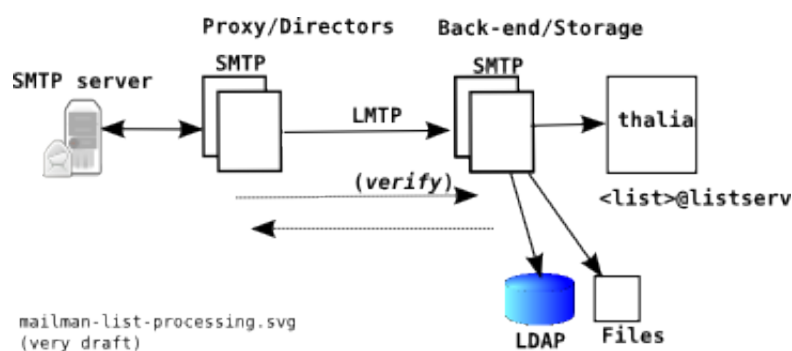
LDAP Groups

With the LDAP system lists are configured as groups in the repository and are given the ability to receive email (*mail* LDAP attribute). Manually configured LDAP lists are useful when the number of members for the list is rather small, for example less than 10, OR the list is set-up temporarily for testing purposes. These lists are defined similarly to virtual users in file CS-mailing-lists file.

Note: /etc/aliases is no longer used by the mail system to resolve user or group aliases. It is only used for system specific local aliases as define by the OS. For example mysql -> root ie the mysql user on each server is forwarded to the root user. /etc/aliases plays an important role in system monitoring. See the Linux Installation Guide.

Mailman Lists

- Mailman lists are created using the methods of the Mailman software.
- The necessary aliases information for these lists, generated by Mailman, are placed in file CS-mailman-lists.
- List memberships are taken from LDAP groups (*uniqueMember*). It is also possible to create entirely independent, from LDAP, lists. See Mailman in Wiki.
- List message forwarding and redirection is taken from LDAP (*mailForwardingAddress*)



NOTE: See the section in mailman mailing lists on how the mailing lists are created.

File used	Purpose	Front end (theano, elysso)	Back-end (gorgo, mirto)
/etc/aliases	used only for OS users, so it is static and does not need to change	✓	✓
/usr/local/etc/postfix/access	Universal access permissions/control	✓	
UCY-virtual-alias	UCY aliases as forwarded to us from YPS – valid for both @ucy and @cs	✓	
CS-virtual-alias	CS aliases as defined by CS – valid only for <alias>@cs.ucy.ac.cy	✓	
CS-mailing-lists	CS custom mailing lists	✓	✓ (?)
CS-mailman-lists	CS mailing lists (mailman)	✓	✓ (?)

Table 4: Summary of user groups/lists and alias definition files

Temporary arrangements:

For legacy system:

- aliases, groups and lists are prepared on nireas under /etc/postfix and are transferred to other servers via cron. See files /root/sendfiles, /root/sendmailaliases.
- Processing on the receivers is also done by cron. See crontab -l.
- This will be changed eventually when all names, groups, aliases, lists are cleaned up and the functionality is transferred to the LDAP repository.

Note1: Some users/groups are already ONLY on LDAP (13/10/2014)

Note2: A root@in.cs.ucy.ac.cy -> root@cs.ucy.ac.cy is required to catch all servers @in sending messages since the real servers do not receive email on @in.

Protecting Internal Mail lists

(NOTE2: Policy on lists and groups. Who can send to which list (ex csall, eplxix etc?)

The preferred method of protecting mailing lists is by using the Mailman to define them. With Mailman a list can be configured such that only members of the list can send to it. Additional senders are also possible

(ex. Administrative staff).

Internal mailing lists can also be protected from abuse by not allowing clients beyond our network to post to them. This is enforced in `/usr/local/etc/postfix/access` and based on the `$my_network` Postfix configuration setting. *This method prevents CS members from sending to the lists when away from the department.*

LDAP schema

Currently we use:

- `/etc/aliases`
 - on the incoming servers to resolve aliases
 - this file contains a variety of things including aliases, groups (lists of people actually) and lists (again a list of people)
 - which makes use of the legacy `/etc/mailaliases` directory
 - both of these are copied from `/Mail/common`
- `/usr/local/etc/postfix/virtual-users`
 - that contains a full list of users on `cs.uc.ac.cy` and `ucy.ac.cyc` as before
 - note that this also has the ability (and it does) to resolve aliases
- Mailman mailing lists which are handled by a different server

for creating aliases. Eventually this will migrate to LDAP in the following way.

LDAP Attributes used:

mail – <original user name>@cs.ucy.ac.cy

mailalternateaddress – <default alias>@cs.ucy.ac.cy

mailalternateaddress - <additional alias>@cs.ucy.ac.cy (user option)

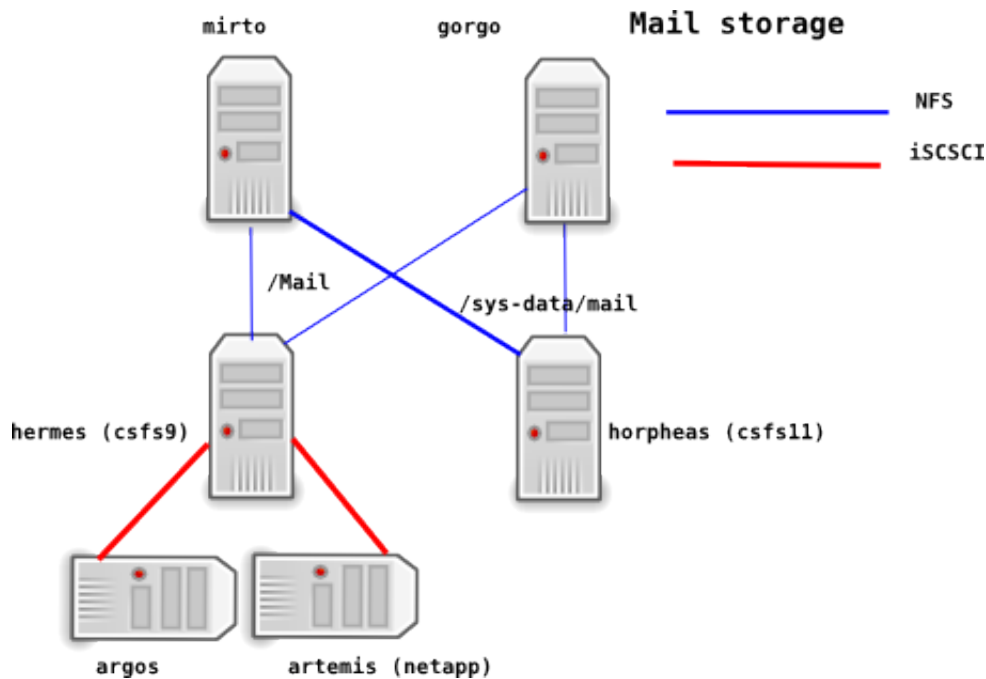
mailforwardingaddress – forward of email

mailquota – quota of user

mailuserstatus – active or inactive

Mail Storage

The Mail servers use a reliable, scale-able, consolidated storage based on iSCSI technology already in place.



The message store is accessed by the SMTP/IMAP servers by using NFS. Future options to change to more involved network storage infrastructure (GFS or Gluster) is possible.

Note: /Mail perms 757 for dovecot-lda to autocreate directories

Note2: HERMES export (rw, sync, insecure, no_root_squash) for mirto, gorgo.

Quota with Dovecot LDA

Quota is implemented using the Dovecot-LDA.

See the Postfix and Dovecot configuration.

Temp: -> overquota should be implemented on front-end so that backscatter is not produced, This requires maildir++ and the epolicy service.

-> temporarily this is done with the overquota file on both front-end and back-end: reason: aliases will pass through.

Migrating from maildir to mbox

Replication

you need to specify

- Domain = cs.ucy.ac.cy in /etc/imapd.conf

E-Mail Software Setup

A large number of software modules is required to implement a properly secure and robust email system. Most of these subsystems are optional. They are not necessary to have a working email system. They are however necessary if we are to build a system that will be secure, fault tolerant and user friendly. Each of these modules has a large number of configuration parameters that if combined together will produce an extremely large combination list. The system depicted in Figure 2 provides a basis for a concrete target system. The system in figure 2 will be presented

below.

Details on what is needed and how to properly configure software modules to create the system in Fig. 2 are presented below. In order for a system to become operational is is necessary to install all its software.

The complete e-mail system is based on the following suite of applications and utilities software:

- Postfix MTA
- Dovecot
 - IMAP server
 - POP3 server
 - LMTP service
 - LDA service
 - Pigeonhole Sieve and Managesieve plug-in
- Amavisd-new
 - ClamAV
 - SpamAssassin
- SASL for submission
- OpenSSL (openssl software)
- TLS
- LDAP
- Webmail clients
 - Horde
 - Roundcube

Setting up Postfix

Postfix is compiled from sources for this set-up.

1. To compile and install Postfix. See Postfix – Compile and InstallPostfix – Compile and Install in Appendix A for install and configuration details.
2. Enable logging as soon as installation is successful on each server. Watch your logs for problems. See the Administration section.

The SMTP servers are divided into two groups: front-end (relay), back-end (storage).

Front-End Relay (theano, elysso)

Front-end (or Relay) SMTP servers are outward facing. They receive AND send messages to the bad Internet. They are the entry point of all incoming messages and the exit point of all outgoing messages. They also are the first line of defense of the mail system against any problems or malicious activity.

Functions:

- receive mail from outside CS and relay it to the back-end servers for final delivery and storage
- receive mail from inside CS that needs to be forwarded to the outside of CS
- implement a variety of protection mechanisms, validity checks and filters
 - reject email from non-privileged networks (see \$mynetworks, that are trusted and can

- relay messages through the system). See below the trusted networks.
- check the validity of a receiving user by requesting a confirmation from the back-end servers
- check the ability to deliver messages to users of incoming messages (ex. overquota)
- uses amavisd to do virus and spam filtering with spamassassin and clamav

Open ports:

- 25 (tcp) from all
- 587 (tcp) from all
- 10001 for SASL (from back-end servers)

Set-up:

1. install amavisd/clamd (see below)
2. edit main.cf
3. edit master.cf
4. edit transport
5. virtual users (???)

Trusted Networks

Trusted networks can relay messages through Postfix. These will eventually be severely limited so that clients cannot relay without authorization.

Current configuration:

```
mynetworks = 194.42.16.0/24,
              194.42.17.0/24,
              194.42.18.0/24,
              194.42.27.0/24,
              !10.16.3.0/24,
              !10.16.5.0/24,
              !10.16.6.0/24,
              !10.16.7.0/24,
              !10.16.8.0/24,
              !10.16.9.0/24,
              !10.16.10.0/24,
              !10.16.20.0/24,
              !10.16.21.0/24,
              10.16.0.0/16,
              194.42.0.0/24,
              194.42.2.0/24,
              194.42.34.0/24
```

SASL configuration

See [Postfix SASL Howto: Dovecot SASL](#)

SASL (Simple Authentication and Security Layer) is an SMTP extension to enable authentication of a client to the SMTP server. This authentication is necessary to allow a remote client to gain higher privileges (ex. to send mail to remote users).

We use the Dovecot SASL implementation over a TCP connection to back-end servers that hold the user data. See also the back-end server Dovecot configuration for this.

```
#####
# SASL configuration
#####
smtpd_sasl_auth_enable = yes
smtpd_sasl_type = dovecot
smtpd_sasl_path = inet:gorgo.in.cs.ucy.ac.cy:12345
```

```
smtpd_sasl_security_options = noanonymous
```

SSL/TLS configuration

See 1) *Postfix TLS Support*, 2) *CentOS HowTo on Postfix/Dovecot TLS/SSL*

```
#####  
# TLS configuration  
#####  
smtpd_tls_security_level = may  
smtpd_tls_key_file = /etc/pki/tls/private/mail.cs.ucy.ac.cy.key  
smtpd_tls_cert_file = /etc/pki/tls/certs/mail.cs.ucy.ac.cy.cert  
smtpd_tls_loglevel = 1  
smtpd_tls_session_cache_timeout = 3600s  
smtpd_tls_session_cache_database = btree:/var/lib/postfix/smtpd_tls_cache  
tls_random_source = dev:/dev/urandom  
smtpd_tls_auth_only = yes
```

Back-End (gorgo, mirto)

Back-end (or Storage) SMTP servers are inside the CS network. They do not interact with outside servers or services. They receive messages from front-end SMTP servers for delivery to user mailboxes. To deliver messages they use a Local Delivery Agent (Dovecot LDA).

Functions:

- keep records and have knowledge of all valid users
- keep records of banned, non delivery users (ex. Over quota)
- use above records to answer ability to deliver requests from front-end servers
- receive mail from front-end SMTP servers (as a destination of a relay action)
- use an LDA to deliver messages to user mailboxes

Software Setup

- General access control (/etc/postfix/access)
- *Authentication is via LDAP*
- Virtual users are enabled (/etc/postfix/virtual_users)
- Virtual users control is enabled (via /etc/postfix/virtual_users_access)
- Over quota users are denied access (/etc/postfix/overquota_users)
- Internal mailing lists and aliases are enabled with (/etc/aliases). This can also be done with virtual users. (Note: there is room of improvement here in order to properly set up aliases: per person, groups, small mail lists that are now created with system accounts via LDAP.

Open ports:

- 25 (tcp) from front-end servers

Set-up:

1. edit main.cf
2. edit master.cf
3. edit transport
4. virtual users (???)

Setting up Dovecot

1. To compile and install Dovecot and Pigeonhole Sieve. See Appendix A.

2. Enable logging as soon as installation is successful (10-logging.conf). Watch your logs.
See the Administration section.

The Dovecot (IMAP, POP3, Sieve, ManageSieve) servers are divided into two groups: front-end (directors), back-end (storage).

Front End (IMAP PROXY or Dovecot Director) Configuration (theano, elysso)

Proxy: <auth-config> - for statis proxying
10-auth.conf
!include auth-static.ext.conf

Back-end Configuration (gorgo, mirto)

TLS/SSL configuration

In 10-ssl.conf
ssl = yes
ssl_cert = </etc/pki/tls/certs/mail.cs.ucy.ac.cy.cert.pem
ssl_key = </etc/pki/tls/private/mail.cs.ucy.ac.cy.key.pem

Summary table

Service	Front End (theano, elysso)	Back end (gorgo, mirto)
IMAP	IMAP	
POP3		
Managesieve	Managesieve	
Sieve		
Director		

Table 5: Dovecot Services Setup

TCP port	Front End (theano, elysso)	Back end (gorgo, mirto)
110 (tcp, udp) from all		
143 (tcp, udp) from all		
4190 (tcp) from all		
9090 (director communication)		
10001 SASL authentication		Mirto only
12345 SASL authentication		Gorgo only
12340 quota status service		Mirto, gorgo

Table 6: Dovecot TCP ports usage

Dovecot Storage

Mail storage comes from /Mail.

INDEX and CONTROL files go in /sys-data/mail/dovecot/{index, control} where these files are not subject to quota controls. /sys-data/mail is an NFS mounted system from csfs11 (orfeas) and exists ONLY on back-end systems.

Quota

Quota is implemented with the Dovecot Quota plug-in. See file 90-quota.conf

Sieve Filtering and ManageSieve

Dovecot's Pigeonhole extension implements the Sieve and Managesieve protocols.

Sieve filtering is installed on the back-end IMAP machines to do the filtering before the Dovecot LDA/LMTP delivers each message to the user mailbox.

The Managesieve protocol is enabled on both the proxy servers and the back-end servers. Proxies simply work as font-end machines that proxy users to the back end services. Very much like IMAP proxying.

Sieve is part of Dovecot. The scripts are called by the Dovecot LDA during its message delivery process. Therefore it runs ONLY on the back end systems (gorgo, mirto) where delivery is happening.

Compile and install dovecot-pigeonhole extension. See Appendix A.

Default script run for all users: /usr/local/var/lib/dovecot/default.sieve.

```
[root@gorgo dovecot]# more default.sieve
require ["fileinto", "envelope", "editheader"];
if header :contains "subject" ["[SPAM]"] {
    fileinto "SPAM";
}
if header :contains "subject" ["[.SPAM.]"] {
    fileinto "SPAM";
}
```

```

if header :contains "subject" ["[..SPAM.."] {
    fileinto "SPAM";
}
addheader "X-CS@UCY-Information" "Please contact support@cs.ucy.ac.cy for help.";
addheader "X-CS@UCY-Information" "Report abuse to abuse@cs.ucy.ac.cy";

```

Compile manually at set up and when changing

```
> sievec default.sieve
```

to avoid giving permissions to users on the directory. Otherwise users will not be able to compile at run time

Sieve directory enabled: sieve_dir = ~/.sieve

Default user script: \$HOME/.dovecot.sieve.

This can be a link to the directory \$HOME/.sieve where multiple scripts can be saved.

The Dovecot Configuration (dovecot -n)

```

[root@gorgo log]# dovecot -n
# 2.2.4: /usr/local/etc/dovecot/dovecot.conf
# OS: Linux 2.6.32-358.14.1.el6.x86_64 x86_64 CentOS release 6.4 (Final) nfs
mail_location = maildir:/newmail/%u:INDEX=/sys-data/dovecot/indexes/%u:CONTROL=/sys-
data/dovecot/control/%u
namespace inbox {
    inbox = yes
    location =
    mailbox Drafts {
        special_use = \Drafts
    }
    mailbox Junk {
        special_use = \Junk
    }
    mailbox Sent {
        special_use = \Sent
    }
    mailbox "Sent Messages" {
        special_use = \Sent
    }
    mailbox Trash {
        special_use = \Trash
    }
    prefix =
}
passdb {
    driver = pam
}
ssl_cert = </etc/pki/dovecot/certs/dovecot.pem
ssl_key = </etc/pki/dovecot/private/dovecot.pem
syslog_facility = local0
userdb {
    driver = passwd
}

[root@iolao logrotate.d]# more dovecot
/var/log/dovecot.log {
    missingok
    notifempty
    delaycompress
    sharedscripts
    postrotate
        /bin/kill -USR1 `cat /var/run/dovecot/master.pid 2>/dev/null` 2> /dev/null || true
    endsript
}

```

```
[root@gorgo pam.d]# more dovecot
```

```

##PAM-1.0
auth        required      pam_nologin.so
auth        include       password-auth
account     include       password-auth
session     include       password-auth

```

```
[root@gorgo logrotate.d]# more dovecot
# dovecot SIGUSR1: Re-opens the log files.
/var/log/dovecot.log
/var/log/dovecot-info.log
{
    missingok
    notifempty
    delaycompress
    sharedscripts
    size 20M
    postrotate
        /bin/kill -USR1 `cat /var/run/dovecot/master.pid 2>/dev/null` 2> /dev/null || true
    endsript
}
```

Advanced Dovecot Features

- dsync

A general overview of the functionality of the system is presented in the figure below.

Installing the software

The method we use in setting up the software is the top down approach. This approach builds work its way from the top software (postfix) and builds around it the functionality of the system required. Another approach might be the bottom up approach where we build the functionality required on top of a working operating system, testing the software at every step to verify its functionality. These methods are summarized in the table below.

The top down approach is described below.

Virus and SPAM Control Setup

AMAVISD-NEW is used as the front end for VIRUS and SPAM control

(<http://www.ijs.si/software/amavisd/>).

Amavis can use a variety of software and facilities to operate. We use amavis with

- spamassassin (<http://spamassassin.apache.org/>)
- clamav (www.clamav.net)

The strategy below is designed to guard against false SPAM positives, giving the chance to the user to scan special folders in case such a positive occurs. This has been found beneficial in the past especially in the case of SPAM since determination of whether a message is SPAM or not is statistical.

Strategy

Overall Description

- Every incoming and every outgoing message is scanned for viruses, spam and banned content
- Scanning is done on the entry point servers (SMTP relays: theano, elysso). Note that these servers are entry points for both incoming mail from outside sources and outgoing mail from inside sources. Inside, here, means our users whether on the local network or not.
- Messages found to contain viruses are **immediately quarantined and NOT forwarded to the user**. The user is notified and can request a release from the quarantine.
- Messages found to be SPAM with a very high degree of certainty (**score of >20 from Spamassassin**) are immediately quarantined and NOT forwarded to the user. The user is

- notified and can request a release from the quarantine.
- Messages found to be SPAM below the score of 20 are NOT discarded but are marked as such with special Subject: header additions, [.SPAM.] or [..SPAM..], and are then forwarded for delivery to the user.
- Messages found to have banned content (ex. executable files) are immediately quarantined. The user is notified and can request a release from the quarantine.
- The delivery software (Dovecot LDA/LMTP) with the help of the Sieve language plug-in scans the subject and delivers SPAM messages into .SPAM special IMAP user folder.
- .SPAM folders are automatically cleaned every night with aging criteria as follows
 - messages are allowed to remain in the .SPAM folder for 14 days
- Users can enforce their own filtering rules if they want to using the Sieve language as explained in the User manual. (how??)

A list of systems and procedures used

- SPF DNS records are both checked and advertised
- DKIM signing is both enforced and checked by Spamassassin

AMAVISD

The overall SPAM software structure and interrelationships are presented in the figure below.

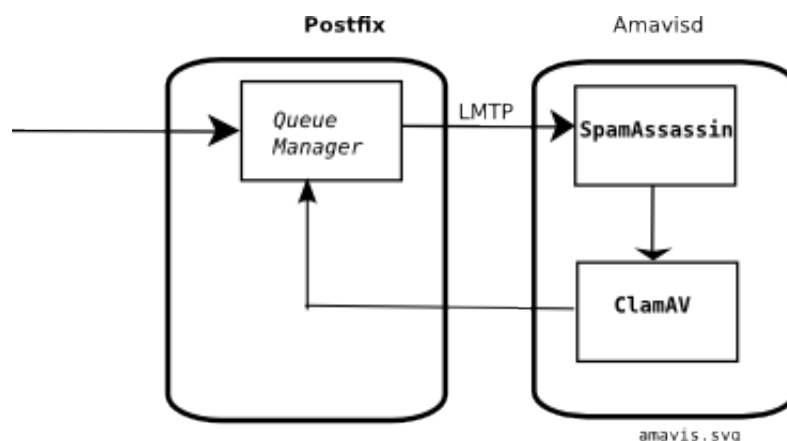


Fig. 4: Conceptual software interconnection in receive mode

Install and configure

Amavisd-new is in the EPEL directory and should be installed from there (note that the software required is in the CENTOS-CSEXTRAS repo so a simple `yum install amavisd-new` should be enough). The configuration here describes the customization of the system over the CentOS default installation

- start amavisd service
 - `chkconfig amavisd on`
 - configure in `/etc/amavisd/amavisd.conf`
- clamav configuration
 - use `clamd.amavisd` service which start with user `amavis` (instead of `clamd`)
 - `chkconfig clamd.amavisd on`
- There is no need to configure Spamassassin in amavisd since it makes use of it by default via library calls. See below for Spamassassin configuration.

Postfix Integration

- Edit master.cf see the config examples

SpamAssassin

Spamassassin is installed as an amavisd prerequisite. There is no need to start spamassassin as a service since AMAVIS uses it by direct library calls to access its functionality.

Here we describe any customization placed after the installation and explanations where things are not obvious

ClamAV

Clamav is installed as an amavisd prerequisite

Here we describe any customization placed after the installation and explanations where things are not obvious

Message Quarantine

Every time the system detects a message that contains objectionable content (Spam, Virus, Bad content or messages that do not conform to standards) it sends a message to the administrator. The administrator in our case is supporthelp@cs.ucy.ac.cy.

All quarantined notifications are stored in the Quarantine folder of user supporthelp.

Releasing Messages

- Check the notification of a line like:
 - The message has been quarantined as: spam-MGRkEnTKdz4w.gz
 - The message has been quarantined as: banned-2IvQ04s5M0Lq
- On the machine reporting the quarantine do (for example theano)
 - amavisd-release spam-MGRkEnTKdz4w.gz

Managing/cleaning the quarantine folder

Security

Outside facing systems (front-ends)

Inside systems (back-ends)

Server OS security

- Harden the OS
- Protect at the border (FW2)
- / (root) protection
- /var protection
- sudo
- ssh permissions
- fail2ban – local Intrusion Detection software
 - enables the ssh-iptables jail
 - custom jail postfix-dns
 - bans SMTP port for clients that have no reverse resolution IP and repeatedly try to send messages – possibly DOS, or mis-configured SPAM servers
- OS files (all files except mounted file systems) clamav scanning

Network Protection

The network is protected from the mail servers by having them in the DMZ. Only absolutely necessary and very directed permissions are given from the DMZ to the internal network (violation!!)

Server protection

Fail2ban

SSH jail

SMTP jail

Dovecot jail (IMAP, POP3)

Session security

Security MUST be looked at every process:

1. IMAP access
2. POP3 access
3. SMTP access
4. HTTP for WebMail

IMAP4 session	User-to-server		SSL protected
POP3 session	user-to-server		SSL protected
SMTP	user-to-server(submissions		SSL protected session
SMTP session	smtp-to-smtp		SSL protected
WebMail	user-to-webmail		HTTPS
Webmail	webmai-client-to-smtp	SSL	
webmail	webmail-to-IMAP		SSL

Setting up HTTPS for encryption of web mail sessions. SASL for authentication with the SMTP AUTH method

Port list that need to be blocked/opened and are required by the software:

Port	Action	Mask	Description
25	enabled	(tcp,udp)	SMTP
109 for POP2	blocked	(tcp,udp)	POP2 not supported
110 for POP3	enabled	(tcp, udp)	Theano,
143 for IMAP	enabled	(tcp, udp)	
587	enabled		Postfix Relays SMTP submission
4190	enabled		From Dovecot front end to Dovecot back end -- Managesieve protocol
1234			SASL authentication from Postfix relays to

			back-end Dovecot
9090	enabled	tcp theano, elysso	Dovecot Director communications
5666	enabled	(tcp, udp) - ALL	NRPE

Mail server protection

Back Scatter E-Mail

Mail System Clients

Some machines which provide mail services are actually clients to the mail system and not part of it. We currently have three client email services: Pine (system ada), Webmail and Mailman hosted on thalia.

Basically the mail system supports all clients that are IMAP, POP3 or SMTP compliant. Even some departure from the standards can be tolerated.

The configuration has been tested to work with;

- Thunderbird
- Microsoft Outlook
- Other IMAP and POP3 clients mainly from mobile devices
- Pine
- Webmail clients – Horde, Roundcube
- Mailman
- A large array of other SMTP/IMAP clients

Pine

- Pine is available from system ada.
- Pine uses IMAP and SMTP (like every other client) to communicate with the email system.

WebMail Servers

There are two software suites providing webmail services: Horde and Roundcube.

Web mail client and server configuration

- Incoming Server configuration
 - Horde – current setup
 - mailer-type = 'sendmail'
 - sendmail_path = '/usr/lib/sendmail'
 - sednmail_args = '-ci'
 - imp
 - imap – mail
- Outgoing server configuration (SMTP)
 - to local server first

- relayed to mail.cs.ucy.ac.cy

Thalia uses a local SMTP server to communicate with the mail system (relay).

Mailing List Server

See also: Authentication, Names, Aliases, Groups, Lists

Mailman is on system thalia (194.42.17.136).

See mailman web site and documentation for details on Mailman.

Administration

Administration guidelines and procedures.

General

Be very careful with what you are changing. It is possible to break the system entirely including losing mail messages if you break things. Follow the instruction carefully below.

See Table 2: Server services summary and Table 2: Server services summary for the functionality and configuration of each server and server group.

While Postfix is a critical part of the entire mail system most of the time troubleshooting concentrates on the Dovecot part is directly accessed by the users.

Critical OS checks

- A) Make sure that all software and services needed are installed and configured as explained in the previous chapters. If in doubt follow the relevant chapter.
- B) Make sure all services will be restarted upon reboot (chkconfig –list). The following list relates to email only but the usual OS service should be started (ex. NTP). There are differences between back-end and front-end servers.
 - Postfix
 - Dovecot
 - amavisd
 - clamd
 - clamd.amavisd
 - fail2ban
 - Spamassassin
- C) Other software that need to be configured
 - ldap (back-end only)
- D) Make sure that your /etc/fstab will mount all the required directories for the mail system to function. This is highly dependent on the configuration (ex. /Mail repository, /sys-data/mail or other temporary file systems set up).

OS upgrades

- stop postfix
- stop dovecot
- upgrade
- reboot

Postfix Administration

Before making any serious configuration changes that may impact the delivery of messages on any Postfix system first do:

```
soft_bounce = yes
```

in the configuration. This will prevent the loss of messages due to possible configuration errors. Messages that are likely to bounce because of mis-configuration will be retried.

Some useful commands:

- display the queue
 - mailq
 - postqueue -p
- view messages in the queue
 - postcat -q <queue id>
- process the queue
 - postqueue -f
 - postfix flush
- delete messages from queue
 - delete all
 - postsuper -d ALL
 - delete all from deferred
 - postsuper -d ALL deferred

Dovecot

Back-end system maintenance.

To move all user sessions to a different back-end. From the front-end Director server:

1. Disable any system that will undo changes you make to back-end server weights, such as poolmon or any crons that make such changes..
2. Set the weight of the back-end server to be worked on to 0:
 - doveadm director add <backend server ip> 0
3. Flush current assignments to disable new connections to this server:
 - doveadm director flush <backend server ip> (???)
4. **From the backend server** close all open sessions:
 - doveadm kick 0.0.0.0/0

Performance

Fault tolerance

Backup

- backup (how an to do an archive for permanent storage?? For how long to expire?)

Monitoring the Mail system

There are several ways to monitor the activity of both the OS and the E-Mail system:

- log files with syslog or direct
- mail messages generated by the OS and the different modules
- nagios
-
- supporthelp account. Nagios is an active method of monitoring while the rest are passive. Generated log messages are generally affected by various configuration parameters on each system. Statistics can also provide valuable information on how the system is functioning.

Logging with syslog and files

Log files are an invaluable resource when things do not work as expected. See the logging section for each application setup to understand in more detail the way logs are kept.

In summary the email application use the following:

- Dovecot Syslog Facility: local0 -> /var/log/dovecot.log
 LDA delivery -> /var/log/dovecot-lda.log
 Information Log -> /var/log/dovecot-info.log*
 Debug -> /var/log/dovecot-debug.log*
 - * these are not always enabled. Enable only when debugging or need to obtain extra information. See `/usr/local/etc/dovecot/conf.d/10-logging.conf` file to determine if these are enabled or not.
 - The verbosity of logging is kept at minimum in regular operational mode. See `10-logging.conf`
- Postfix Syslog Facility: mail -> /var/log/maillog
 - See the `notify_classes` parameter to increase reporting capability
- Amavisd
 - Spamassassin -> /var/log/spamd.log
 - ClamAV -> /var/log/clamav/clamav.log or
 clamd.log or freshclam.log

Log rotation

Rotating the logs and determining that this works well is a MUST operation. If not done properly, log files will overrun the file system, log file will overflow with the result of OS crash and loss of information. Some applications (ex. dovecot) do not automatically provide logrotate scripts during install to facilitate log rotations and must be created manually. Example below:

General logrotate set up:

- daily

- compress
- rotate 21
 - 3 weeks of log files for all system logs
- size 1M
 - this establishes a default that can be overridden if required

Example logrotate configuration for Dovecot (on all systems):

/etc/logrotate.d/dovecot

```
/var/log/dovecot.log
/var/log/dovecot-info.log
/var/log/dovecot-lda.log
{
    missingok
    notifempty
    delaycompress
    sharedscripts
    size 20M
    postrotate
        /bin/kill -USR1 `cat /var/run/dovecot/master.pid 2>/dev/null` 2> /dev/null || true
    endscript
}
```

Mail messages

The operating system and many applications have the ability to send email messages (usually to system user root) when predetermined events are encountered (ex. non delivery of messages will generate a report to the postmaster). These messages should go to the special user supporthelp for CS. Make sure that in the system /etc/aliases:

```
##### support: postmaster
root: supporthelp@cs.ucy.ac.cy
```

Nagios Monitoring

These are basic instructions to Nagios monitoring.

On all servers

- yum install nrpe
 - yum install nagios-plugins
 - yum install nagios-plugins-load
 - custom Nagios plug-in's go into /usr/lib64/nagios/cs-plugins
 - *mkdir /usr/lib64/nagios/cs-plugins*
 - *create the check_postfix_queue*
 - *chkconfig nrpe on*
 - *service nrpe start*
 - *Port 5666 needs to open for the Nagios server NRPE service (10.16.0.1).*
1. *yum install nrpe*
 2. *yum install nagios-plugin-load*
 3. *Get check_postfix_queue from <http://exchange.nagios.org/directory/Plugins/Email-and-Groupware/Postfix/> and install in cs_plugins*

Table 7 gives a summary of Nagios monitoring for the mail server components.

Entity	Nagios plug-in	Type	Notes
CPU load	check_load	Local/nrpe	gorgo/mirto: -w 25,20,15 -c 35,30,25
Postfix (SMTP)	check_smtp	Remote/triton	
Dovecot (IMAP)	check_imap	Remote/triton	
Postfix queue	check_postfix_queue	Local/nrpe	<p>the script was changed so that it responds to 8 characters of queue_id instead of 10. See the script at /usr/lib64/nagios/cs-plugins</p> <ul style="list-style-type: none"> • Checking the queues: • on internal systems (gorgo, mirto) keep values small (-w 50 -c 100) since these systems should deliver messages immediately • on relays (theano, elysso) (-w 300 -c 1000) <p><i>Note: the best approach would be to do a time based check ie. Check the queue record values and check a fixed amount of time later several times, if queues exceed a value, remain or keep increasing issue a warning. Do the same for critical value. This will require change of the check script.</i></p>
amavisd process (only on relays)			
clamd.amavisd (only on relays)			

Table 7: Summary of Nagios monitoring for Mail Servers

Statistics

Message cleanup

Mailcleanup operations (cleanup scripts). See the Appendix D: Helper Scripts .

Script hermes:/mail/common/Mail/scripts/cleanup-Mail cleans up mailboxes of old messages according to the following criteria:

- OLDSPAM = 10 days from .SPAM folders
- OLDVIRUS = 7 days from .Virus folders
- OLDTRASH = 10 days from .Trash folders

The script currently runs on hermes since this machine is the closest to the file system that needs to be cleaned.

Synchronizing a Dual server configuration

In a dual server configuration, many options must be synchronized. The synchronization of many aspects of both the relay and back-end systems is necessary in order to achieve a consistent operation for th

esystems.

Some preliminary things to synchronize:

- 1 main.cf – must have some same options like restrictions (the two main.cf files are not the same).
- 2 access file on relay systems
 - i. protect internal mailing lists, either LDAP or manual
- 3 /etc/aliases
- 4 SpamAssassin
- 5 ClamAV
- 6
- 7 Crontab entries
- 8 LDAP options and configuration
- 9 Nagios monitor configuration (nrpe.conf)

Upgrades

- When scheduled for upgrades or down times note that queues will remain the machine.

Troubleshooting

Testing SMTP

- If upon
 - telnet <smtpserver> 25 or
 - telnet <smtpserver> 587

we get nothing then it means the server did not start correctly. Look in /var/log/maillog for errors in the the config file (ex. main.cf)

```
[root@mirto ]# telnet elysso 587
Trying 194.42.17.141...
Connected to elysso.
Escape character is '^]'.
220 elysso.cs.ucy.ac.cy ESMTP Postfix
EHLO mirto
250-elysso.cs.ucy.ac.cy
250-PIPELINING
250-SIZE 25600000
250-VRFY
250-ETRN
250-STARTTLS
250-ENHANCEDSTATUSCODES
250-8BITMIME
250 DSN
```

Problems with authentication:

- check if ESMTP is spoken between client and server
 - telnet <server> 25 EHLO <client>
- Should receive a reply similar to above.
 - If not then ESMTP is blocked by firewall then. Fixup protocol SMTP On CISCO PIX

Testing IMAP


```

telnet: > telnet imap.example.com imap
telnet: Trying 192.0.2.2...
telnet: Connected to imap.example.com.
telnet: Escape character is '^]'.
server: * OK Dovecot ready.
client: a1 LOGIN MyUsername MyPassword
server: a1 OK Logged in.
client: a2 LIST "" "*"
server: * LIST (\HasNoChildren) "." "INBOX"
server: a2 OK List completed.
client: a3 EXAMINE INBOX
server: * FLAGS (\Answered \Flagged \Deleted \Seen \Draft)
server: * OK [PERMANENTFLAGS ()] Read-only mailbox.
server: * 1 EXISTS
server: * 1 RECENT
server: * OK [UNSEEN 1] First unseen.
server: * OK [UIDVALIDITY 1257842737] UIDs valid
server: * OK [UIDNEXT 2] Predicted next UID
server: a3 OK [READ-ONLY] Select completed.
client: a4 FETCH 1 BODY[]
server: * 1 FETCH (BODY[] {405})
server: Return-Path: sender@example.com
server: Received: from client.example.com ([192.0.2.1])
server:         by mx1.example.com with ESMTP
server:         id <20040120203404.CCCC18555.mx1.example.com@client.example.com>
server:         for <recipient@example.com>; Tue, 20 Jan 2004 22:34:24 +0200
server: From: sender@example.com
server: Subject: Test message
server: To: recipient@example.com
server: Message-Id: <20040120203404.CCCC18555.mx1.example.com@client.example.com>
server:
server: This is a test message.
server: )
server: a4 OK Fetch completed.
client: a5 LOGOUT
server: * BYE Logging out
server: a5 OK Logout completed.

```

Testing POP3

```

telnet: > telnet pop.example.com pop3
telnet: Trying 192.0.2.2...
telnet: Connected to pop.example.com.
telnet: Escape character is '^]'.
server: +OK InterMail POP3 server ready.
client: USER MyUsername
server: +OK please send PASS command
client: PASS MyPassword
server: +OK MyUsername is welcome here
client: LIST
server: +OK 1 messages
server: 1 1801
server: .
client: RETR 1
server: +OK 1801 octets
server: Return-Path: sender@example.com
server: Received: from client.example.com ([192.0.2.1])
server:         by mx1.example.com with ESMTP
server:         id <20040120203404.CCCC18555.mx1.example.com@client.example.com>
server:         for <recipient@example.com>; Tue, 20 Jan 2004 22:34:24 +0200
server: From: sender@example.com
server: Subject: Test message
server: To: recipient@example.com
server: Message-Id: <20040120203404.CCCC18555.mx1.example.com@client.example.com>
server:
server: This is a test message.
server: .
client: DELE 1

```

```
server: +OK  
client: quit  
server: +OK MyUsername InterMail POP3 server signing off.
```

Appendix A: Compiling and Installing from Sources

Compiling all used packages from sources is chosen to, mainly, take advantage of the new features that appear but also to act quickly on possible bugs or security problems. CentOS usually lags behind these features and past experience has shown that there is clear advantage of compiling from sources.

Compilation and installation strategy is:

1. Compilation and most importantly installation is done in a way such that it is compatible with the stock RPM installs. Where necessary (ex. creation of users and groups, start-up scripts etc) the installation follows the original CentOS conventions. This helps in case we later decide to convert to the original RPMs into the OS.

We can do a preliminary install of Dovecot/Postfix to get all the defaults in place. These will need to be revised when the custom compilation is complete and of course the RPMs removed to avoid confusion.

2. Configuration/compilation is done on each machine independently in the /root home directory
3. All new files (including configuration files) are installed in /usr/local to make them independent of RPM files/packages
4. run time files are kept in their default place (ex. queues, lock files etc). These files are described for each application separately since they are application dependent.
5. Installation of the relevant RPMs is disabled in /etc/yum.conf to avoid confusion

This strategy helps in case we decide, for whatever reason, to revert back to the original CentOS packages but also to make each machine independent.

Prerequisites

- Install gcc

Postfix – Compile and Install

NOTE: Compilation of front-end and back-end systems is slightly different.

Instructions based on version 2.10.x.

- install necessary software: db4-devel, openldap-devel (only on back-end systems)
- cd /root; download source (see www.postfix.org); unpack.
- In /root there is a postfix-make-<host>.sh where host is the host being used to build Postfix. Each host has been optimized with its own build. Copy it to the new source directory.
- cd the new source directory
- Run postfix-make-<host>.sh
 - this file customizes the compilation/installation for our purposes.
 - (To start from scratch do a **make clean**)

- Create user accounts and related groups (if not available already). These user setting are the same with CentOS default UID/GID.

```
/etc/passwd:
postfix:x:89:89::/var/spool/postfix:/sbin/nologin

/etc/group:
postfix:x:89:
```

- Create a group "postdrop"

```
/etc/group:
postdrop:x:90:
```

- Install new postfix
 - make install
 - you should take the defaults on the options given since the compilation has already determined the place of install..
 - OR make upgrade if installing an upgrade
- Prevent installation of Postfix RPM in the future. In /etc/yum.conf do
 - **exclude = postfix***
- chown postfix.root /usr/local/var/lib/postfix/
- chmod 700 /usr/local/var/lib/postfix/

Add Postfix to the reboot start-up procedures

Centos 6: Use the chkconfig system

- cp -p /root/postfix.init.d to /etc/init.d/postfix
- chkconfig --add postfix
- chkconfig postfix on
- chkconfig --list ; check it

Centos 7: Use the systemd system

- cp -p /root/systemd.postfix.* to /etc/systemd/system
-
-
-

The procedures above, when successful, allow us to stop and start Postfix with the familiar

- service postfix start (systemd for Centos 7)
- service postfix stop
- etc
- we need to disable sendmail and switch to the new Postfix software
 - USE THE alternatives system – the following command to switch to the new Postfix related software (there is no need to do this during upgrade)

```
/usr/sbin/alternatives --install /usr/sbin/sendmail mta /usr/local/sbin/sendmail 25 \
--slave /usr/bin/mailq mta-mailq /usr/local/bin/mailq \
```

```
--slave /usr/bin/newaliases mta-newaliases /usr/local/bin/newaliases \
--slave /usr/share/man/man1/mailq.1.gz mta-mailqman /usr/local/share/man/man1/mailq.1 \
--slave /usr/share/man/man1/newaliases.1.gz mta-newaliasesman
/usr/local/share/man/man1/newaliases.1 \
--slave /usr/share/man/man8/sendmail.8.gz mta-sendmailman /usr/local/share/man/man1/sendmail.1 \
--slave /usr/share/man/man5/aliases.5.gz mta-aliasesman /usr/local/share/man/man5/aliases.5 \
--initscript postfix
```

Remove sendmail and Postfix RPMs

We need to remove the Postfix RPMs to avoid confusion as to which software is running.

Postfix has several packages dependencies (cronie etc). To remove do

See what is installed

```
rpm -qa | grep sendmail
```

```
rpm -qa | grep postfix
```

Then Remove without touching the dependencies

```
rpm -e --nodeps sendmail
```

```
rpm -e --nodeps postfix
```

Go to the Postfix configuration. Without proper configuration Postfix will usually not start successfully.

Dovecot – Compile, Install

Prerequisites

Install:

- zlib-devel
- openssl-devel
- pam-devel
- quota-devel, quota
- tcp_wrappers-devel
- openldap-devel

Download sources:

- expand into its directory (ex. tar -zxvf dovecot-2.2.4.tar.gz)
- cd into directory
- **./configure --with-pam --with-ldap --with-ssl=openssl --with-zlib --with-libwrap --with-mysql**
 - Install prefix : /usr/local
 - File offsets ... : 64bit
 - I/O polling : epoll
 - I/O notifs : inotify
 - SSL : yes (OpenSSL)
 - GSSAPI : no
 - passdbs : static passwd passwd-file shadow pam checkpassword ldap sql
 - : -bsdauth -sia -sql -vpopmail
 - userdbs : static prefetch passwd passwd-file checkpassword ldap sql nss
 - : -sql -vpopmail
 - SQL drivers :
 - : -mysql -pgsql -sqlite
 - Full text search : squat
 - : -lucene -solr
- **make**
- **make install** (into /usr/local/ by default)
- Create necessary users if required

- dovecot
 - dovecot:x:97:97:Dovecot Local IMAP server:/usr/local/libexec/dovecot:/sbin/nologin
 - dovenull:x:498:498:Dovecot's unauthorized user:/usr/libexec/dovecot:/sbin/nologin
 - groupadd -g 97 dovecot
 - groupadd -g 496 dovenull
 - useradd -g 97 -u 97 -c Dovecot Local IMAP server -d /usr/local/libexec/dovecot -M -N -s /sbin/nologin dovecot
 - useradd -g 496 -u 496 -c Dovecot Local unauthorized user -d /usr/local/libexec/dovecot -M -N -s /sbin/nologin dovenull
 - *default uid and gid change from time to time but are not critical*
- to uninstall: **make uninstall**
- Disable install from CentOS repositories so that we do not end up with two different installs. In /etc/yum.conf
 - **exclude = dovecot***
- Transfer the example configs to /usr/local/etc/dovecot from /usr/local/share/doc/dovecot/example-configs

Add Dovecot to the boot start-up procedures

Centos 6

- there is a /root/dovecot.init.d
 - cp -p dovecot.init.d /etc/init.d/dovecot
 - chkconfig --add dovecot
 - chkconfig on dovecot
 - chkconfig --list dovecot

Centos 7

- cp -p /root/systemd.dovecot.* to /etc/systemd/system
 - /usr/lib/systemd/system/dovecot.service
 - /usr/lib/systemd/system/dovecot.socket
-

Start/Restart procedures

- go to Dovecot configuration
- service dovecot start

Pigeonhole Compile and Install

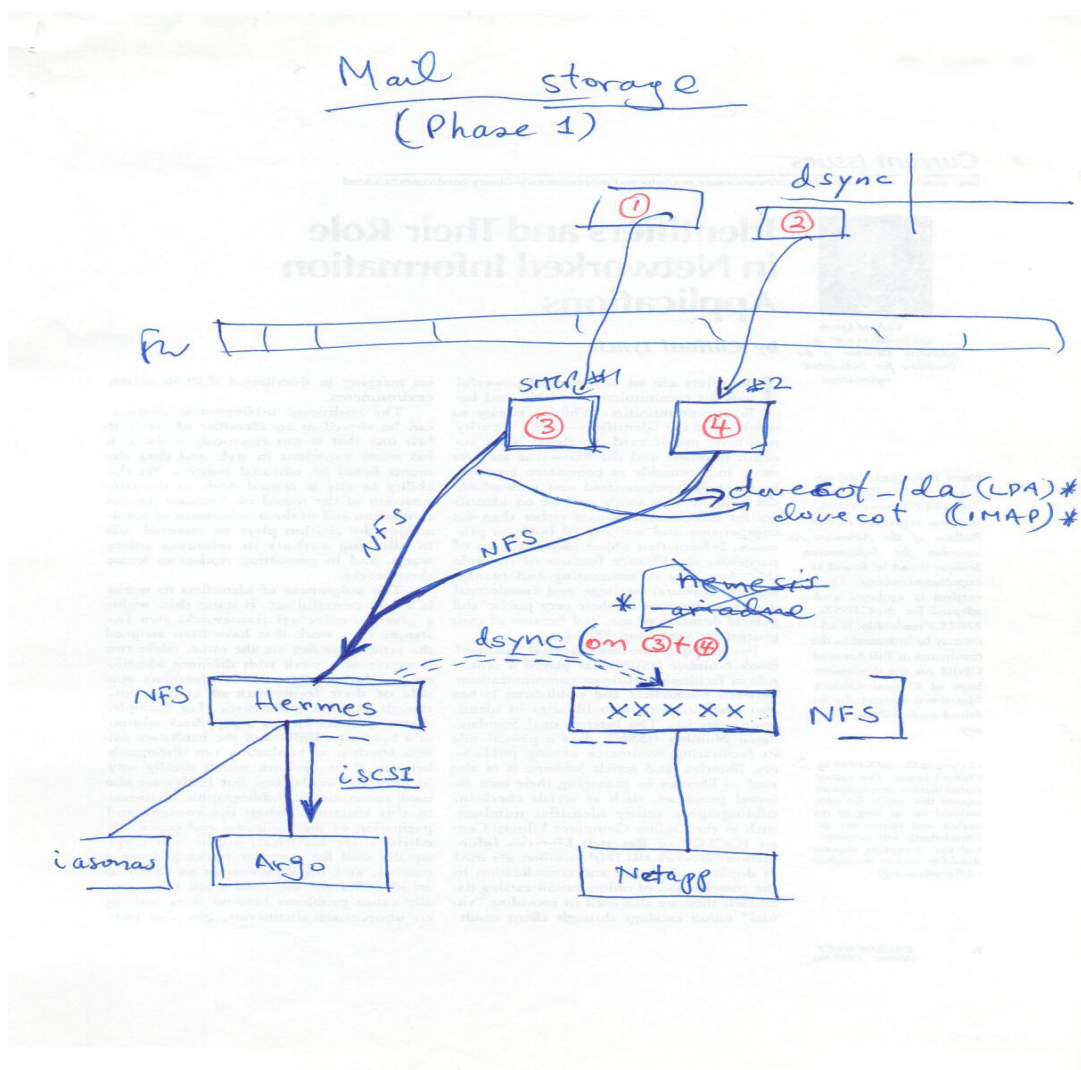
ucompress

```
./configure --with-dovecot=/usr/local/lib/dovecot
make
make test
make install
```

Copy sieve.conf, sieve-extprogs.conf, managesieve.conf from /usr/local/share/docs/dovecot/example-config/conf.d/ /usr/local/etc/Dovecot/conf.d

Use these files to configure Pigeonhole (managesieve and sieve)

Future Storage



part five
dovecot -lda
dovecot -dsync

(replica location)

hermes

Netapp

argo

NFS

NFS

NFS

dual sync option

Email System-Administrator Manual-0.9.odt - Version 0.9.39 - Feb 16, 2015

Appendix C: Notes and Pending Issues

1. Also solve the issue of failed mounts (check Dovecot's alternative location feature)

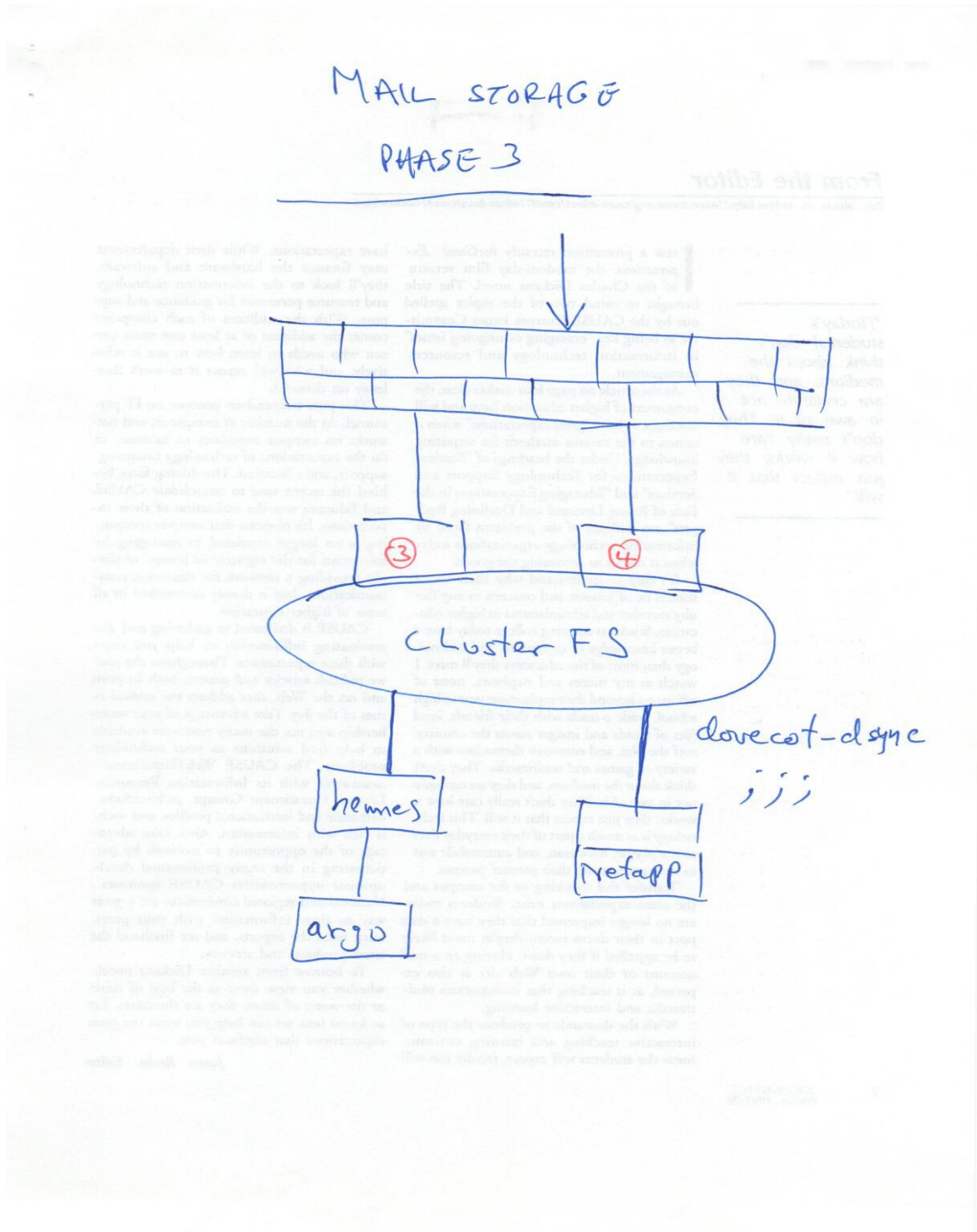


Illustration 6: Storage Phase 3

2. How to solve the problem of giving access to the external relay to the users DB (internal
 1. check the verify (8) Postfix facility

2. The scenario is a classic one:
 1. one or more relay SMTP servers in DMZ
 2. one or more backend SMTP servers on the inside network
 3. There may or may not be separate incoming or outgoing designated SMTP servers.

Now the desired functionality is (of course):

1. relay machines receive messages from outside AND inside
 2. relays check for all the bad things (spam, viruses etc).
 3. for incoming messages relays check for valid local users and reject messages for invalid users
- Such scenario allows all checks to be done at the entry point allowing back-ends to function with the real nice messages and at a much reduced load. But there is a problem. If you are a DMZ admin (or a security hawk) #3 functionality above is not possible without violating the DMZ policy, especially if you are dealing with internal LDAP and DB servers which essentially house personal information.

Have a look at `verify(8)` and the `address_verify_*` variables in `main.cf`

With this you can dynamically verify recipients on your backend SMTP servers.

Appendix D: Helper Scripts

These may change. Look at the locations indicated below for exact configuration.

Temporary user management scripts

- **See crontab -l on theano, elysso**
 - [root@theano postfix]# crontab -l
 - 15 * * * * cp /tmp/overquota.users /usr/local/etc/postfix/overquota.users
 - 17 * * * * cp /tmp/virtual-users /usr/local/etc/postfix/virtual-users
 - ### 18 * * * * cp /tmp/virtual-users-cs /usr/local/etc/postfix/virtual-user-cs
 - 19 * * * * cp /tmp/CS-virtual-alias /usr/local/etc/postfix/CS-virtual-alias
 - 19 * * * * cp /tmp/UCY-virtual-alias /usr/local/etc/postfix/UCY-virtual-alias
 - 20 * * * * cd /usr/local/etc/postfix/; /usr/local/sbin/postmap overquota.users
 - 21 * * * * cd /usr/local/etc/postfix/; /usr/local/sbin/postmap virtual-users
 - ### 22 * * * * cd /usr/local/etc/postfix/; /usr/local/sbin/postmap virtual-users-cs
 - 23 * * * * cd /usr/local/etc/postfix/; /usr/local/sbin/postmap CS-virtual-alias
 - 23 * * * * cd /usr/local/etc/postfix/; /usr/local/sbin/postmap UCY-virtual-alias
- **See crontab -l on nireas**
 - # Backup Directory Server every day at 2am
 - 0 2 * * * /etc/cron.d/backupdirsrv
 -
 - # Send files to theano, elysso
 - 0 */1 * * * /root/sendfile
 -
 - # Send /etc/mailalaises to gorgo, mirto
 - 05 */6 * * * /root/sendmailaliases

Mail clean up script

Currently on hermes:/mail/common/Mail/scripts/cleanup-Mail

```
#!/bin/bash
# The purpose of this file is to maintain the e-mail system and its associated
# storage filesystem. This includes automatic spam learning and the
# clean up of all the .VIRUS and .SPAM and .Trash user folders
# in user directories. It must be run from the /Mail directory to have
# practical effect.
# The clean up happens on messages that are older than 15 days for SPAM
# 7 days for VIRUS and 20 days for Trash. These can be adjusted using the OLDSPAM and OLDVIRUS
# and OLDTRASH variables.
#
#
# This is run daily via cron on the entire /Mail storage area.
#####
LOGFILE=/var/log/mailcleanup.log
LEARNFROM="./ank-test/.SPAM/cur ./antonis/.SPAM/cur ./george/.SPAM/cur ./yanos/.SPAM/cur ./skevos/.SPAM/cur
./savvasn/.SPAM/cur ./kekkos/.SPAM/cur ./ank-test/.SPAM/new ./antonis/.SPAM/new ./george/.SPAM/new ./yanos/.SPAM/new
./skevos/.SPAM/new ./savvasn/.SPAM/new ./kekkos/.SPAM/new"

###OLDSPAM=5
##OLDSPAM=20
OLDSPAM=10
OLDVIRUS=7
```

OLDTRASH=10

```
cd /mail
echo "===== " >> $LOGFILE
echo "This log is produced by /Mail/common/scripts/cleanup-Mail. See this file for details" >> $LOGFILE
echo `date` >> $LOGFILE
#
#echo "Starting Spamassassin learning ....." >> $LOGFILE
#for i in $LEARNFROM; do
#echo "Learning from $i" >> $LOGFILE
#/usr/bin/sa-learn --spam --showdots $i >>$LOGFILE 2>&1
#sleep 3
#done;
#
#
#
echo "_____ " >> $LOGFILE
echo "Starting cleanup of .SPAM and .VIRUS folders ...." >> $LOGFILE
echo "I am deleting the following files..." >> $LOGFILE
for i in *; do find $i/.SPAM/cur -type f -mtime +$OLDSPAM -print -exec rm {} \; >>$LOGFILE 2>&1; done;
for i in *; do find $i/.SPAM/new -type f -mtime +$OLDSPAM -print -exec rm {} \; >>$LOGFILE 2>&1; done;
###
for i in *; do find $i/.VIRUS/cur -type f -mtime +$OLDVIRUS -print -exec rm {} \; >>$LOGFILE 2>&1; done;
for i in *; do find $i/.VIRUS/new -type f -mtime +$OLDVIRUS -print -exec rm {} \; >>$LOGFILE 2>&1; done;
###
echo "Starting cleanup of .Trash folders ....." >> $LOGFILE
for i in *; do find $i/.Trash/cur -type f -mtime +$OLDTRASH -print -exec rm {} \; >>$LOGFILE 2>&1; done;
###
###echo "Starting cleanup of /Mail/tmp folders ....." >> $LOGFILE
###/usr/sbin/tmpwatch --verbose 72 /Mail/tmp /tmp >>$LOGFILE 2>&1
```

disabled 9/1/2015

Appendix E: Configuration Listings

A complete listing of the systems configuration will eventually be found on platon.

Custom differences only:

Fail2ban

Filter dovecot-pop3imap.conf

[Definition]

```
failregex = (?: pop3-login|imap-login): (?:Authentication failure|Aborted login \(auth failed|Aborted
login \(tried to use disabled|Disconnected \(proxy dest auth failed).*rip=(?P<host>\S*),.*
ignoreregex =
```

SSHD (/etc/sshd_config)

```
#####
PermitRootLogin no
#####
# Live for 6 hours
ClientAliveInterval 21600
ClientAliveCountMax 5
#####
```

Appendix F: References

www.redhat.com
www.centos.org
www.postfix.org
www.dovecot.org
www.amavisd.org
www.clamav.net
www.spamassassin.apache.org
www.sieve.org
www.roundcube.org
www.horde.org

Relevant RFC reference

RFC #*	Subject
2822	SMTP
4954 (3463, 2554)	SASL authentication
5230	Sieve vacation extension

*obsolete in parentheses

Appendix G: Random Notes

SSL certificates:

View: `openssl x509 -in <certificate> -text -noout`

To tail and highlight a selections do:

- `tail -f /var/log/maillog | perl -pe 's/keyword/\e[1;31;43m$&\e[0m/g'`
- or `tail -h <file> <keyword>` on any of the mail servers

Certificate Mgmt

- View
 - `openssl x509 -in <cert> -noout -text`

ChangeLog

0.6 - Original EmailNG administration manual – the start

0.7 - Changeover to the EmailNG project by the department. There is still much work to be done. 18/06/2014<http://hcmportal.vi.ucy.ac.cy>

Troubleshooting Errors

imap(ank): Debug: Namespace : /Mail/ank/.Trash.Parking doesn't exist yet, using default permissions

Thunderbird: [CANNOT] Renaming not supported across conflicting directory permissions.

This most probably is related to conflicting permissions like trying to move (or delete) a folder that has different permissions than its destination folder. Dovecot is picky about that even though such