Computer Engineering Degree *Computer Engineering* Year 2017/18	Practical classes **Lab5**	**CSDA** **Unit III** INTERNET

Integration of global services in enterprise environments II:

The INTERNET

Deployment of a secure mail server

*Computer systems and protocols for **E-Mail service** management and **Webmail** access*

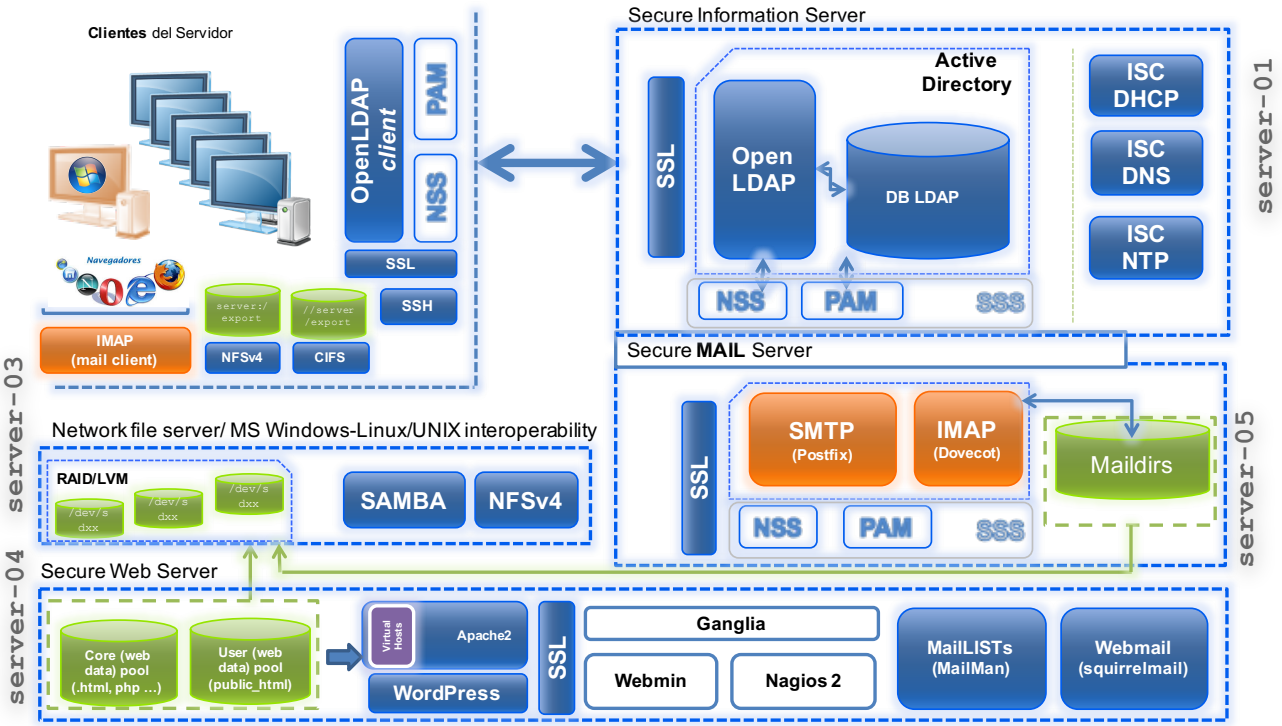


Table of contents

Main goals

- To learn about processes for adapting basic servers to certain needs. In this case:
  - Installation and configuration of a **MAIL service (SMTP/IMAP)** using the "open source" Apache software Postfix/Dovecot implementation.
  - Adapting the web service under certain **organizational** and **security** premises**:**
    - Use of *Maildirs*
    - Integrating LDAP (`server-01`) with IMAP protocol to validate LDAP users in the use of mail.
    - Security in client-server comunications for IMAP protocol: TLS/SSL
  - Installation, configuration and deployment of Web mail access, using:
    - RoundCube
- Adaptation, integration and configuration of the client side for these services.
- To become familiar with and handle different techniques and tools for administration and testing of said services.

Getting started: Creating the clone for lab5

1. Create a new clone from the initial system "**core**".
    a. Select this option: "**Restart MAC address**"
    b. Type: **full** (*)
    c. Select **all** the branches from the snapshot "tree".
2. Create an initial snapshot for that clone before starting the lab class.
    a. Remember to keep the VM off
    b. Call it **snapshot_P5**
3. For **client_LINUX** clone, create a new initial snapshot to complete this lab class.
    a. Remember to keep the VM off
    b. Call it **snapshot_P5**

Assignment 1: The Setting

Updating and initial configuration for `server-05`

1. First, update the system from debian repositories.
2. Then, you will have to adapt your clone_P5 to turn it into a **secure mail server**. So, carry out the tasks required as follows:
    a. Hostname: `server-05`.
    b. Local name resolution:
        1. Hostname (FQDN): `server-05.localdomain`
        2. Alias: `server-05`
    c. Networking:
        1. Make sure that both of the `clone_P5` network interfaces are connected to "type NAT" network *network_1.*
        2. Required data:
            - *IP*: (example)
                - (**eth0**): 192.168.0.**15**
            - *Network mask:* 255.255.255.0
            - *Network*: 192.168.0.0
            - *Broadcast*: 192.168.0.255
            - *Gateway*: 192.168.0.1
    d. DNS servers:
        1. *DNS1*: 193.144.193.11
        2. *DNS2*: 193.144.193.22
        3. *Search domain*: *localdomain*
    e. Disable all those service that you are not going to use. At your own discretion.
    f. Upgrade the server to last available software versions.

3. You have to configure `server-05` as client of service supplied by `server-01` (lab 1):
    a. **NTP Client.** Time (date) of our file server should be automatically synchronized by the NTP `server-01`. Use the `ntpdate-debian` app in a "client-server" model and decide the sync interval.
    b. **DNS client**. Add `server-01` as secondary DNS for your server.
    Make sure that the DNS Server (BIND9) deployed in Lab1 has a **MX** record for the mail server `server-05.localdomain`, according to the authoritative zone **localdomain**
    c. **LDAP client**. Our new server will be able to use the LDAP directory in a safe way (ssl) to **identify**[1] users who are managed by LDAP on `server-01/server-02`.

[1] It is not necessary for users to be able to connect to `server-05` by SSH (PAM)

4. **Configure** `server-05` as a **NFS client** of `server-03` (as **static** mounting):
    a. `server-05` should mount /`export/home` from `server-03` on the local directory `/remote/home`. Take all necessary actions to make it permanent.
    b. Rename initial `/home` and make a symbolic link, called `/home`, to `/remote/home`.

5. [OPTIONAL] **Configure the VirtualBox environment** to access the *web* service (webmail) on `server-05` from the host using the host web browser:
    a. Add a new rule in the custom NAT network "network_1":
        i. Host (PC or Laptop):
            1. IP: 127.0.0.1
            2. Port 8015
        ii. Guest (VM → `server-05`)
            3. IP 192.168.0.15
            4. Port: 80

Assignment 2: The Core (I)

Installation and configuration of a send mail service (SMTP): Postfix

Now our goal is to deploy a *secure* MAIL service that enables managing the e-**mail** of users in an enterprise environment. Users managed by a LDAP active directory service on `server-01`.

More details in [2]

First, let's start with SMTO, which will allow us to manage the sending of e-mail from a MUA (Mail User Agent) to a MTA (Mail Transfer Agent).

1. *Installation* of the SMTP server using **POSTFIX** implementation:
    a. Install the **Postfix** mail service on `server-05` and tools to manage **Maildir** mailboxes.
    b. During installation, *debconf* will ask you about the initial configuration for Postfix:

```
General type of mail configuration: <- Internet Site
mail name: <- server-05.localdomain
```

2. *Initial configuration*:
    a. Check the main configuration files of the Postfix **service** and make sure that they contain the following essential items:

    1.- Disable SMTPs (*smtp over ssl*)

    2.- Server hostname (FQDN): `server-05.localdomain`

    3.- Network interfaces to listen to SMTP requests: `all`

    4.- SMTP port (default): 25

    5.- Authorized remote destinations (relay domains) to forward mail from strangers (clients outside authorized networks) to authorized remote destinations only:

        1. `server-05.localdomain`
        2. `server-05`
        3. `localhost.localdomain`
        4. `localhost`
        5. `localdomain`

    6.- File to store the mail *alias*.

    Aliases are non-real accounts, that is, mail user accounts that do not exist in the system, but that can be associated with one or more real mail accounts.

        1. `systems` → `root`
        2. `support` → `user1`

Create a file with these aliases. When sender sends a mail to *support@localdomain*, the mail is actually received by user1. This file must be a 'Berkeley database' using the command `newaliases`. The result is a file called `aliases.db` that will be used by Postfix.

```
alias_maps = hash:/etc/aliases
alias_database = hash:/etc/aliases
```

7.- List of "trusted" remote SMTP clients that have more privileges than "strangers". To specify the list of trusted networks by hand, specify network blocks in CIDR (network/mask) notation:

```
mynetworks = 192.168.0.0/16, 127.0.0.0/8
[::ffff:127.0.0.0]/104 [::1]/128
```

It is important to remember here what is "*open relay*". Open relays allow anyone to send email from your mail server. The mail server does not check that it is authorized to send mail from the mail address on the third-party email. What this means is that anyone can send email via your cloud server IP address from any mail address. This is one reason why your server IP address can end up on blacklists. Also, your legitimate email is not being received by the people you are sending it to. So it's important to define `mynetworks`.

Also you can add specific client IP addresses that are not allowed to send email to our MTA, using the `/etc/postfix/access` file. This file need be indexed through the `postmap` command. If you decide to use it, you also need to add the following configuration:

```
smtpd_recipient_restrictions = permit_mynetworks
check_relay_domains
smtpd_sender_restrictions = hash:/etc/postfix/access
reject_unknown_sender_domain
```

b. Check the configuration files of Postfix **daemon** and make sure that they contain the following essential items:

1.- Daemon runs only the unsecure service instance (no ssl) `smtpd`. (for the moment)

3. *SMTP checking*:
   a. Looking at logs, make sure that the Postfix service runs without warnings or errors.
   b. Using the `telnet` command, check the SMTP connection:

```
$ telnet server-05 25
EHLO test
quit
```

c. Using the `mail` command, try to send an email to *user1* and check the logs:
    1. Mail data:
        a. `Subject: testing mail`
        b. `Body: eMail for user1, sent from server-05, user 'root'`
        c. `Cc: <empty>`
    2. Check logs again.
    3. Verify where the mail files are stored on server.
    4. As *user1*, read its mail using the `mail` command.
d. Try this again, but this time sending a mail to a foreign user. You can use your personal mail account.

Assignment 3: The Core (II)

Installation and configuration of a secure delivery/receipt mail service (IMAP) [**MDA**]: Dovecot

Continuing with the development of our secure email service on `server-05`, it's now mandatory to develop the mail **delivery** services through the IMAP protocol, which will allow us to transfer mail from the MTA to the client (MUA). This mail system component is called MDA (Mail Delivery Agent).

More details in [3] [8] [9] [10]

1. *Installation* of the IMAP server using **Dovecot implementation:**
    a. Install the **Dovecot** mail service on `server-05`

    For mail delivery from `server-05` to clients (MUAs), we will use the well-known IMAP protocol. Using IMAP4, mail clients such as Webmail, Outlook, Thunderbird or Evolution could access user mail through mailboxes and deliver it. Also, that protocol can work over SSL, so that the communications can be *encrypted* between the MUA and MTA.
    We will use the IMAP's *Dovecot* implementation. There are others like Courier, etc.

2. *Initial configuration:*

    The service's configuration relies on several files in `/etc/dovecot/conf.d`. One of the most important ones for the service is `/etc/dovecot/10-master.conf`, which establishes important aspects concerning the service run instances.

    a. Disable temporally the secure instance for IMAP (*imaps*), which is enabled by default.
    b. Configure Dovecot and Postfix to manage the user mail through *Maildir* directories, instead of *Mailbox* files. Maildir is a *spool* format to store and manage the user's mail using different files with unique names. The key is to have a file for each user. Inside the Maildir structure, there are 3 directories; `tmp`, `new`, and `cur`, each one with a different role. That directory (Maildir) will be automatically created when users receive the first mail.

        Note: Postfix uses `procmail` for mail delivered through the MDA, from the source MTA to the user's Mailbox/Maildir in the recipient MTA.

3. *IMAP checking***:**
    a. Make sure that the mail service is working properly, without errors or warnings. Take a look at the log files and the active network sockets.

b. Use the simple client mail `mutt`[2] (command line) to read user1's mail.

1. As *root*, send to user1 a test mail using `mail` command.
2. Now, login as *user1* on `server-05 (su -)`
3. In `$HOME`, create a file called `.muttrc` and add the following lines:

```
set mbox_type=Maildir
# LOCAL
set folder="~/Maildir"
set spoolfile="~/Maildir"
set mbox="Maildir/"
set mask="!^\\.[^.]"
```

4. After that, run `mutt`
5. You can use the `tmux` utility. It will make it easier.

4. *Security setting* for IMAP Dovecot:

a. **LDAP user authentication**: Now, we want delegate user authentication to the LDAP active directory on `server-01` instead of `/etc/passwd`. This way, only LDAP users will be able to access their mail.

Required data:

- Authentication method: **`ldap`**
- URIs: `ldaps://server-01.localdomain`
- Dn: `cn=admin,dc=localdomain`
- Dnpass: `ldap`
- tls_ca_cert_file: `/etc/ssl/certs/CA_server-01.localdomain.cert`
- tls_require_cert: `demand`
- ldap_version: `3`
- base: `ou=people,dc=localdomain`
- user_attrs: `=mail=maildir:$HOME/Maildir`
- pass_attrs: `uid=user,userPassword=password`
- default_pass_scheme: `SSHA`

Verifications:

```
$ telnet server-05 143 (Ctr^D)
$ netstat …
```

b. **SSL communications**: Now, we want to secure communications between the MTA and the MUA, using SSL/TLS certificates. Specifically, the goal is to encrypt all mail

[2] **Mutt** is a small but very powerful text-based program for reading and sending electronic mail under unix operating systems, including support for color terminals, MIME, OpenPGP, and a threaded sorting mode.

delivery communications (IMAP). First, we must create the corresponding IMAP service SSL (TLS) certificate, as we did for other services:

1. Use the CA (*self-signed*) certificate already created in Lab1 (1)
2. Generate the *IMAP service certificate* that you will sign using the CA certificate (*private key*):
    - Generate a *IMAP certificate private key*:
        a. File name: `imaps_server-05.localdomain.key (2).`
        b. Key generated by default.
    - Generate the IMAP service certificate and sign it using the CA certificate and its *private key.* Save it as `imaps_server-05.localdomain.cert (1).`
        a. *Sign mode:* **signed by a CA**
        b. Profile*:*
            - This certificate will be used to encrypt data
            - This certificate will be used for a TLS server
        c. Other data[3]:
            - *Certificate type*: X.509 (default)
            - *Expiration days*: 365 days
            - *Country of the subject:* ES
            - *State:* Cantabria
            - *Locality: Santander*
            - *Organization:* UC
            - *Unit:* CSDA
            - *"Common name"*: **server-05.localdomain** (3)
            - *e-mail:* sistemas@localdomain

(1) PATH `/etc/ssl/certs`

(2) PATH `/etc/ssl/private`

→ Make sure that the ldap service (slapd) user is the owner (UNIX permissions) of the *LDAP certificate private key* file.

(3) **It is very important** to use the FQDN and not its IP or another value.

After that, configure **Dovecot** to enable secure IMAP instance (*imaps*).

Verifications:

```
$ telnet server-05 993 (Ctr^D)
$ netstat …
$ openssl s_client -connect server-05.localdomain:imaps
```

[3] You can use a template as we used in Lab1: `imaps_server-05.localdomain.info`

Assignment 4: The Client

Configuration and checking of a client mail service (SMTP/IMAP) [**MUA**]: POSTFIX/Dovecot client side

The time has come for us to focus on the client side, using the `client` VM as a MUA. Also, we will use `mutt` to connect to MTA and use the IMAPS protocol to download the user mail.

You will have completed some of the following tasks, so you only have to check them.

1. Configure `client VM` as **client** of **the secure information server** (`server-01`) and the **network file server** (`server-03`) implemented in Lab1, Lab2 and Lab3.
   a. **NTP Client.** Time (date) of our file server should be automatically synchronized by the NTP `server-01`. Use the `ntpdate-debian` app in a "client-server" model and decide the sync interval.
   b. **DNS client**. Add `server-01` as secondary DNS for your server and verify that `/etc/hosts` is fully loaded.
   c. **LDAP client**. Our new server will be able to use the LDAP directory in a safe way (ssl) to **identify and authenticate** users who are managed by LDAP on `server-01/server-02`.
      1. Configure `client` as LDAP client host so that we can access the LDAP directory using the LDAP Client Command utility (`ldapserach, ldapadd, ldapmodify` …). Remember that the connection must be secure, over SSL and that we have a replicated LDAP system.
      2. Add LDAP services on `server-01/server-02` as user/host *identification* method.
      3. Reconfigure the client system to enable LDAP *authentication* for the SSH service.
   d. **NFS client:** Using the NFS service on `server-03`, `client` should mount on demand, using *autofs*, the user's home directories, located in `/export/home` (`server-03`) on the local directory `/remote/home/`.

2. Configure `client VM` as **client** of **the secure MAIL server** (`server-05`)
   a. Be sure that LDAP users are able to connect to `client` using SSH connections.
   b. Then, for *user1* only, properly configure the `mutt` mail client using `.muttrc` file. The objective is for user1 to be able to access its mail account in a secure way.

```
set mbox_type=Maildir
# IMAP
set folder="imaps://$LOGNAME@server-05.localdomain/~/Maildir/"
set spoolfile="imaps://$LOGNAME@server-05.localdomain/"
```

```
set imap_user = $LOGNAME
set imap_authenticators="login”
# SMTP
set smtp_url = "smtp://server-05.localdomain:25"
```

3. *Checking*:
    a. Test 1:
        i. From `server-05` (*root*), send an email to every LDAP user:
            1. `$ echo “Hello, I’m root” |mail -s $HOSTNAME user1@localdomain`
            2. `$ echo “Hello, I’m root” |mail -s $HOSTNAME user2@localdomain`
        ii. Check mail for both users:
            1. Open a SSH session on `client` as *user{1,2}*
            2. Check mail using `mutt`
    b. Test 2:
        i. From `client`, (*user1*), send an email to `user2@localdomain` using `mutt`
        ii. Check *user2’s* mail.
    c. Test 3:
        i. From `client`, (*user2*), send an email to `support@localdomain` using `mutt`
        ii. Check *user1’s* mail.

[OPTIONAL] Assignment 5: Webmail

Installation and configuration of a *web* mail client service on `server-04`:
[Web **MUA**]: RoundCube

A **Webmail** service is a way to access to the user's mail account on mail servers. It's implemented as a web application running on a web server. Users mainly access it using the web browser from their client hosts. This tool allows us to access our mail account using our credentials (username/password) and SSL connections (HTTPS). From web server, webmail establishes IMAP/POP3 (IMAPS/POP3S) connections to the mail server, performing as a MUA, and accessing the MTA/MDA.
Using Webmail, it's not necessary to install any mail client on the client host. We only need an internet connection and a web browser.

The last (optional) task in this laboratory class will be to deploy a webmail service on **`server-04`** based on **RoundCube** implementation.

1. *Roundcube installation*. Install a webmail service on `server-04` (`web`), using the web service **Apache2** that is already installed and operative.
    a. Use debian repositories to install Roundcube webmail and its software dependencies:
        i. `mysql-server` (dependency)
            1. Password: "root"
        ii. `roundcube`
            1. Select "`mysql`" as DB manager.
            2. Password: "root" (Database and webmail)
2. *Apache service configuration*. It's necessary to adjust the apache2 service to publish the webmail service:
    a. Roundcube installation automatically generates a configuration file to integrate itself with apache2. Find it and use it.
    b. Check webmail using your host web browser:
        i. **http**://127.0.0.1:8014/webmail/ (*)

    (*) Be sure that PAT (port forwarding) has been configured in Virtualbox.

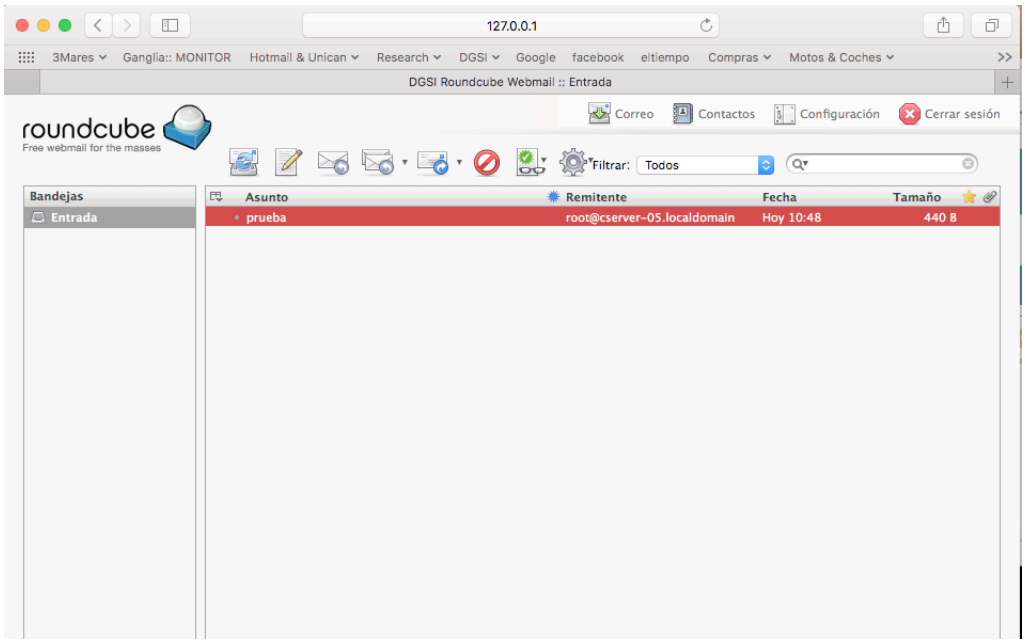
3. *Roundcube configuration*. Configure webmail according to these features:
    a. **Mail transfer service**:
        i. **SMTP**
        ii. Server name: `server-05.localdoamin`
        iii. Port: `25`
        iv. SMTP username format: `%u`
        v. SMTP password format: `%p`

vi. No SSL

b. **Mail delivery service:**
   i. **IMAP**
   ii. Server name: `server-05.localdomain`
   iii. Port: `993`
   iv. Authentication method: LOGIN
   v. SSL

c. More:
   i. *Helo* server: `server-05.localdomain`
   ii. Mail domain: `localdomain`

All these configuration features rely on `/etc/roundcube/main.inc.php`



Resources and references

1. man
2. Google
3. **Slides:**

   → https://gitlab.com/herreroja/G679
4. More:

   Postfix

   [1]http://slav0nic.org.ua/static/books/other/O'Reilly%20-%20Postfix%20The%20Definitive%20Guide.pdf

   [2] https://wiki.debian.org/Postfix

   [3] http://wiki.kartbuilding.net/index.php/Procmail_-_setup_with_postfix

   [4] http://www.redes-linux.com/manuales/Servidor_correo/guia_rapida_postfix.pdf

   Dovecot IMAP

   [4] http://wiki2.dovecot.org

   [5] https://help.ubuntu.com/community/Dovecot

   Others:

   [6] http://wiki2.dovecot.org/Authentication/Kerberos

   [7] http://mindref.blogspot.com.es/2011/02/dovecot-kerberos.html

   [8] https://wiki.debian.org/Mutt

   [9] https://wiki.archlinux.org/index.php/mutt#imap_user

   [10] http://mutt.sourceforge.net/imap/