Computer Engineering Degree *Computer Engineering* Year 2017/18	Practical classes **Lab4**	**CSDA** **Unit III** INTERNET

Integration of global services in enterprise environments II:

The INTERNET

Deployment of a secure web server

*Computer systems for **WEB services** and **content management (CMS)***

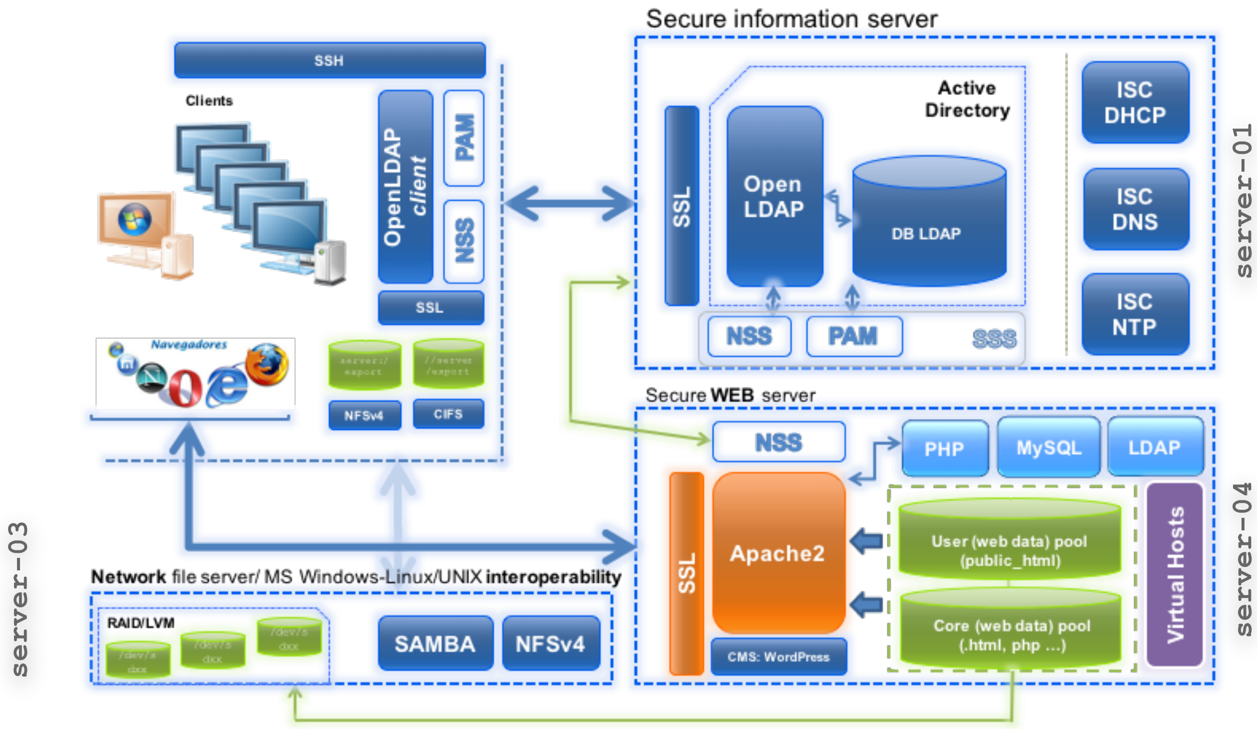


Table of contents

Table of contents 2
Main goals 3
Getting started: Creating the clone for lab4 4
Assignment 1: The Setting Updating and initial configuration for `server-04` 5
Assignment 2: The Core Installation and configuration of the WEB service: APACHE2 7
[OPTIONAL] Assignment 3: The CMS (Addon) Installation and configuration of a Content Management System: Wordpress 11
References and resources 13
Annex 1 14

Main goals

- To learn about processes for adapting basic servers to certain needs. In this case:
  - Installation and configuration of **WEB service (HTTP)** using the “open source” Apache software APACHE2 implementation.
  - Adapting the web service under certain **organizational** and **security** premises**:**
    - Vitualhosts management
    - Access control
    - Security in client-server communications: TLS/SSL
  - Integration with LDAP service to avoid WEB access to resources and contents by unauthorized users.
  - Installation, configuration and deployment of a CMS:
    - Wordpress
- 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 lab4

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_P4**
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_P4**

Assignment 1: The Setting

Updating and initial configuration for `server-04`

1. First, update the system from debian repositories.
2. Then, you will have to adapt your clone_P4 to turn it into a **secure web server**. So, carry out the tasks required as follows:
    a. Hostname: `server-04`.
    b. Local name resolution:
        1. Hostname(FQDN): `server-04`.l`ocaldomain`
        2. Alias: `server-04`
    c. Networking:
        1. Make sure that both of the `clone_P4` network interfaces are connected to "type NAT" network *network_1.*
        2. Required data:
            - *IP*: (example)
                - (**eth0**): 192.168.0.**14**
            - *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 services 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-04` 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.
    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 connect to `server-04` by SSH (PAM)

4. **Build a store backend** for the new web service. We will use the NFS service `server-03`:
    a. Add an additional LVM volume on `server-03` (NFS server) to store the web content for the new web service:
        i. `server-03-web` → 1 GB
        ii. Make an **ext4** file system.
    b. Mount it permanently:
        i. `server-03-web` → **`/export/web`**
        ii. mounting options: → (`noatime,nodiratime,` *`errors=remount-ro`*), neither *dumps* nor file system checks on boot time.
    c. Configure NFS service to **export** that volume (**`/export/web`**) to `server-04` exclusively[2].
        i. Use the same security options as the rest of volumes/file systems exported, except *root_squash*.

5. **Configure** `server-04` as **NFS client** of `server-03`:
    a. **Static** mounting:
        i. `server-04` should mount /`export/web` from `server-03` on the local directory `/var/www`.
    b. **Dynamic** mounting:
        ii. Also, `server-04` should mount on demand, using *autofs*, the user's home directories, located in /`export/home` (`server-03)` on the local directory `/remote/home/`.

6. [OPTIONAL] **Configure the VirtualBox environment** to access the *web* service on `server-04` 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 8014
        ii. Guest (VM → `server-04)`
            3. IP 192.168.0.14
            4. Port: 80.

[2] Restrict access to `server-04` only.

Assignment 2: The Core

Installation and configuration of the WEB service: APACHE2

Our goal now is to deploy a *secure* WEB service that enables publishing and managing web contents.

1. *Installation* of the web server apache2 on `server-04`[3]::
    a. Install the **apache2** package:
        1. Initially, keep the default configuration.

    b. Restart the service without the service stopping.

    c. Quick check:
        1. From `client`, check the service is up using `curl`[4].

2. *Initial configuration*:
    a. Change the default HTTP port from 80 to 8080 and check it using `lynx`[5].
        1. Change it again to operate in port 80 permanently.

    b. Check which Apache2 modules are initially loaded (default).

    c. Disable the following modules that are not going to be used:
        1. `authz_groupfile_module`
        2. `deflate_module`

    d. Identify the website initially (default) active on the server.

    e. Modify the *default* website to show this content:

```
                         CSDA
Student:                              <your FULL name>
Server version:                        Apache/2.2.22 (Debian)
```

3. *Advanced configuration*:
    a. **User web directories**: Enable user web directories in apache2 for LDAP users. They will have their own web space for storing web content (*personal pages*). For this we will need to create the `$HOME/public_html` (NFS) directory for each one and to enable that functionality on apache2. Make the changes only for *user1*.

    b. **Restrict access** to user web directories: Restrict access to these directories in Apache2 using the **.htaccess** method.

    c. Definition of **Virtualhosts**: Configure a new "*virtualhost*" that uses the same server IP but another FQDN:

[3] Use the official debian software repositories

[4] `curl` is a Linux "command line" tool for transferring data from server. It can use the following protocols: DICT, FILE, FTP, FTPS, GOPHER, **HTTP**, HTTPS, IMAP, LDAP, LDAPS, POP3 …

[5] `lynx` is a Linux "command line" web browser

        1. Name of “*virtualhost*” → **csda**
        2. FQDN for the “*virtualhost*” → **www.localdomain**
        3. Web directory to the “*virtualhost*” → `/var/www/csda`
            a. The content will be a single html page that shows “DSGI” in upper case and large size.
        4. Log files for the “*virtualhost*”:
            a. Error logs: `csda_error.log`
            b. Access logs: `csda_access.log`
        5. Disable the directory listing for this “*virtualhost*”
    d. Enable **PHP content** management in apache2 for the new (csda) “*virtualhost*”
        1. Change the web content from html to php by using the attached`.php` file in ANNEX 1.
        2. Check that it works ok.

4. *Security settings* for apache2:
    a. **Restrict IP access** to web content managed by the "VirtualHost" **csda** created in the preceding section so that <u>only</u> the computer with IP 192.168.0.20 (`client`) is able to access it through the Apache server.
        1. Check it using `lynx` command from `server-04`.
    b. **LDAP user authentication**: Now, we want to delegate user authentication to the LDAP active directory on `server-01` instead of `htaccess`. This way, only LDAP users will be able to access web content. Specifically, the content of user web directories (`$HOME/public_html`).
        1. Disable .`htaccess`
        2. Enable LDAP **integration** on Apache2.
        3. Information required for integration:
            a. Message to be shown: "OpenCourseWare Web services"
            b. LDAP bind dn: **`cn=admin,dc=localdomain`**
            c. LDAP admin password: **“ldap”**
            d. LDAP auth URL: **`ldaps://server-01.localdomain:636/ou=people,dc=localdomain?uid`**
            e. Use SSL.
    c. **Secure access to web content using SSL**: Configure a new secure “*virtualhost*” (**SSL**) that uses SSL/TLS certificate to conduct encrypted communications:
        1. Name of “*virtualhost*” → **secure_csda**
        2. FQDN for the “*virtualhost*” → **www.localdomain**[6]
        3. Web directory for the “*virtualhost*”

[6] We will use the same name as **csda** “virtualhost”. The difference will be the protocol to be used: http**s.**

→ `/var/www/secure_csda`

   a. The content will be a single html page that shows the message “HTTPs Secure Access” in upper case and large size.

4. Log files for the “*virtualhost*”:
   a. Error logs: `secure_csda_error.log`
   b. Access logs: `secure_csda_access.log`
5. Disable the directory listing for this “*virtualhost*”

In order to configure SSL access correctly, we will need to create a new TLS certificate for our web service. Use the same procedure as the one we used in Lab1.

1. Use the CA (*self-signed*) certificate already created in Lab1 (1).
2. Generate the *WEB service certificate* that you will sign using the CA certificate (*private key*):
   - Generate a *WEB certificate private key*:
     a. File name: `www_server-04.localdomain.key (2).`
     b. Key generated by default.
   - Generate the WEB service certificate and sign it using the CA certificate and its *private key.* Save it as `www_server-04.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[7]:
        - *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-04.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

[7] You can use a template like the one used in Lab1: `www_server-04.localdomain.info`

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

Now, use the new TLS service certificate and key in Apache in order to enable SSL communications in **secure_csda** “*virtualhost*”

5. *Checking*:
   a. **Basic configuration**:
      1. Use the host (PC/laptop) web browser to access `server-04` web service: → PAT virtualBox (8014 → 80)
      2. From `client`, use the `curl` y/o `lynx` commands to check access to the web service.

   b. **Advanced configuration**:
      1. Check the `.htaccess` mechanism
      2. Access to content of “*virtualhost*” **csda**
         a. Take a look at log files of that virtualhost.
      3. Check that PHP functionality is loaded: → http://www.localdomain/index.php

   b. Security settings:
      1. Restrict access to web contents on `server-04` (“virtualhost” **csda**) to only IP 192.168.0.20.
      2. *LDAP* user authentication
      3. *SSL* communications in “*virtualhost*” **secure_csda**.

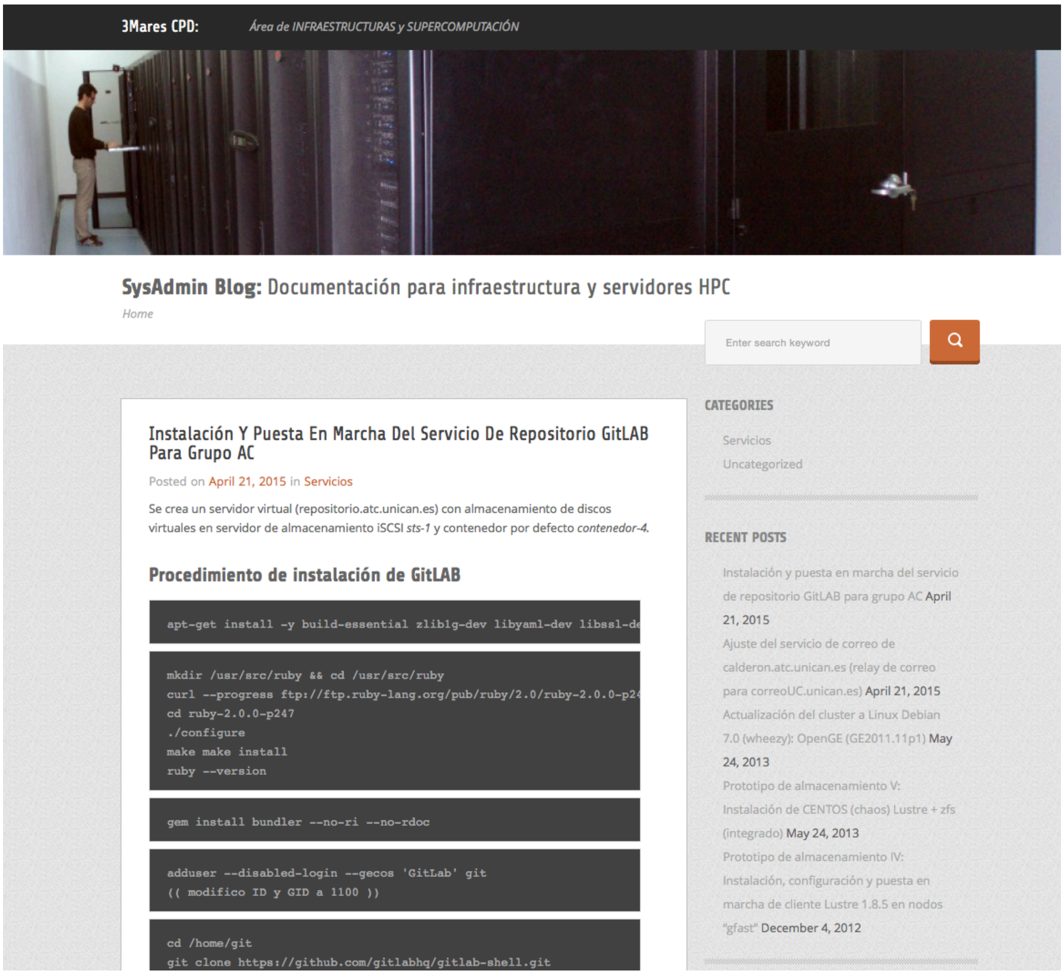
[OPTIONAL] Assignment 3: The CMS (Addon)

Installation and configuration of a Content Management System: Wordpress

Once our HTTP service has been deployed, the new target is to start a web content management service (CMS) using **Wordpress** software.

1. *Installation and configuration*:
   a. Install **wordpress** from *wordpress.org*.
   b. Use a *mysql* and *wordpress* password as you want.
   c. Aspects to be considered:
      - Location (storage) for our web content: `/var/www`
      - *mysql* database name for *wordpress*: **WPcsda**
      - WPcsda admin: '**admin**" (with the password that you want)
      - Host: `server-04.localdomain`.
   d. *Reload* the new integrated configuration for the web service
2. *Checking*:
   a. From `client`, check that the CMS service on `server-04` is operative.
      - You can use `linx`.

3. Develop a **custom "Blog"** for CSDA where we can chronologically manage a list of entries of every procedure for our practical classes. It is a *kind of "bitacora"* for our work.
   a. Adapt your Wordpress services at your own discretion.
   b. Add at least one entry to the "blog".



SysAdmin Blog: Documentación para infraestructura y servidores HPC

Instalación Y Puesta En Marcha Del Servicio De Repositorio GitLAB Para Grupo AC

Posted on April 21, 2015 in Servicios

Se crea un servidor virtual (repositorio.atc.unican.es) con almacenamiento de discos virtuales en servidor de almacenamiento iSCSI *sts-1* y contenedor por defecto *contenedor-4*.

Procedimiento de instalación de GitLAB

```
apt-get install -y build-essential zlib1g-dev libyaml-dev libssl-de
```

```
mkdir /usr/src/ruby && cd /usr/src/ruby
curl --progress ftp://ftp.ruby-lang.org/pub/ruby/2.0/ruby-2.0.0-p24
cd ruby-2.0.0-p247
./configure
make make install
ruby --version
```

```
gem install bundler --no-ri --no-rdoc
```

```
adduser --disabled-login --gecos 'GitLab' git
(( modifico ID y GID a 1100 ))
```

```
cd /home/git
git clone https://github.com/gitlabhq/gitlab-shell.git
```

CATEGORIES

- Servicios
- Uncategorized

RECENT POSTS

- Instalación y puesta en marcha del servicio de repositorio GitLAB para grupo AC April 21, 2015
- Ajuste del servicio de correo de calderon.atc.unican.es (relay de correo para correoUC.unican.es) April 21, 2015
- Actualización del cluster a Linux Debian 7.0 (wheezy): OpenGE (GE2011.11p1) May 24, 2013
- Prototipo de almacenamiento V: Instalación de CENTOS (chaos) Lustre + zfs (integrado) May 24, 2013
- Prototipo de almacenamiento IV: Instalación, configuración y puesta en marcha de cliente Lustre 1.8.5 en nodos "gfast" December 4, 2012

References and resources

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

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

   Apache

   [1] http://httpd.apache.org/docs/current/
   [2] http://www.apache.org
   [3] http://www.bdat.net/documentos/apache/book1.html
   [4] Rich Bowen, Ken Coar (2007) Apache Cookbook, 2nd Edition
   [5] Solutions and Examples for Apache Administration O'Reilly Media

   Security:

   [6]http://www.yolinux.com/TUTORIALS/LinuxTutorialApacheAddingLoginSiteProtection.html#LDAP

   CMS (Wordpress)

   [7] http://www.wordpress.org

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

Annex 1

index.php

```
<?php phpinfo(); ?>
```