

2016

Ignacio L. B.

SERVIDOR APACHE

1. **La arquitectura Web es un modelo compuesto de tres capas, ¿cuáles son dichas capas, cuál es la función de cada una de esas capas y qué tecnologías se podrían usar en cada capa (HTML5, CSS, JavaScript, PHP, etc.)? Justifica tus respuestas.**

Las 3 capas que forman la arquitectura web serian:

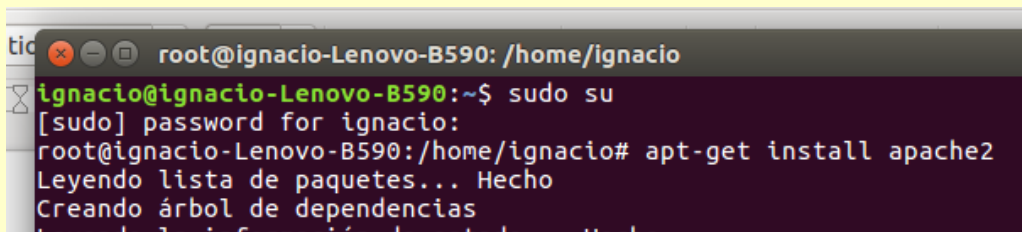
1. La capa de base de datos (capa de datos), donde estarían todos los datos de la aplicación que se pretenden administrar mediante el servicio web. Las tecnologías que se suelen usar en esta capa son MySQL, Microsoft Access, Microsoft SQL Server, PostgreSQL, etc.
2. La capa de servidores de aplicaciones Web (capa de proceso), donde se ejecutan las aplicaciones web que operan con los datos de los servidores de datos y controlan y mandan la presentación de estas web. Tecnologías que trabajan esta capa son Apache, Tomcat, Resin, etc.
3. La capa de clientes del servicio web (capa de presentación), donde se interpretan las peticiones del usuario y presentan los resultados procedentes de los servidores web al usuario. Las tecnologías que tratan con esta capa son los navegadores como FireFox, Chrome, IExplorer etc.

2. **Una plataforma web es el entorno de desarrollo de software empleado para diseñar y ejecutar un sitio web; hay muchos modelos de plataformas web y el sistema operativo suele ser determinante a la hora de implantar una plataforma u otra. ¿Describe una plataforma web que se base en Linux y otra que se base en Windows? Pueden ser las plataformas webs que quieras. Indica que tecnologías se utilizarían para cubrir los aspectos de servicio web (HTTP), contenido dinámico y almacenamiento de datos en cada una de dichas tecnologías.**

- La plataforma más usada basada en Linux es la conocida como LAMP, que trabaja con componentes de software libre. El nombre LAMP viene de las iniciales de los componentes que la integran:
 - **L**inux: sistema operativo.
 - **A**pache: servidor web.
 - **M**ySql: gestor de base de datos.
 - **P**HP: lenguaje interpretado, aunque a veces se sustituye por Perl o Python.
- Una de las plataformas más usadas que se basa en Windows es WISA, sus componentes son de software propietarios ya que son todos de Microsoft, y la componen los siguientes elementos:
 - **W**indows: sistema operativo.

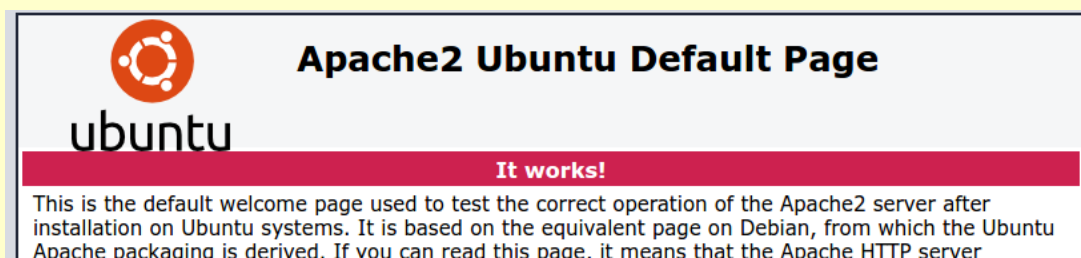
- Internet Information Services: servidor web.
 - SQL Server: gestor de base de datos.
 - ASP o ASP.NET: como lenguaje para scripting del lado del servidor.
- Tecnologías para cubrir los aspectos de servicios web (HTTP) serian los servidores web, como Apache, IIS, Tomcat (diseñado para los servlets), lighttpd (diseñado para ser rápido).
 - Tecnologías para contenido dinámico serian PHP, Perl, Python, ASP, JSP, etc.
 - Y para el almacenamiento de datos tendríamos tecnologías como MySQL, SQL Server, PostgreSQL, SQLite, etc.
3. **Documenta como se realiza la instalación del servidor web Apache y PHP (de forma que trabaje con Apache) desde terminal.**

Primero debemos instalar el servidor web Apache, para ello abrimos la terminal como root haciendo `sudo su` e introduciendo nuestra contraseña, una vez hecho esto introducimos la orden *“apt-get install apache2”*:



```
root@ignacio-Lenovo-B590: /home/ignacio
ignacio@ignacio-Lenovo-B590:~$ sudo su
[sudo] password for ignacio:
root@ignacio-Lenovo-B590:/home/ignacio# apt-get install apache2
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
```

Podemos comprobar que se ha instalado correctamente si desde el navegador de la propia máquina al poner la dirección <http://localhost> se nos abre la siguiente página:



Una vez instalado el servidor apache procedemos a instalar el modulo de PHP en este, recordar que siempre podemos buscar paquetes desde consola con el comando *apt-cache search* conociendo el nombre o parte de él filtrando nombres con el comando *grep* como por ejemplo:

```
root@ignacio-Lenovo-B590:/home/ignacio# apt-cache search php5.5
libapache2-mod-php7.0 - lado servidor, lenguaje de guionización HTML-embebido (módulo Apache 2)
libapache2-mod-php - server-side, HTML-embedded scripting language (Apache 2 module) (default)
libapache2-mod-auth-memcookie - Apache2 authentication and authorization module.
libapache2-mod-auth-tkt - lightweight single-sign-on authentication module for Apache
libapache2-mod-authn-yubikey - Yubikey authentication provider for Apache
libapache2-mod-headers - Process monitoring Apache module
```


Como vemos uno de los paquetes que nos ha salido es “*libapache2-mod-php7.0*” este es el que tenemos que instalar para ello hacemos uso del comando que utilizamos para instalar el paquete de apache “*apt-get install*”

Pondríamos “*apt-get install libapache2-mod-php7.0*” .

Para comprobar que la instalación fue correcta podemos crear un archivo llamado prueba con formato .php (prueba.php) que contenga el siguiente script

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

y guardarlo en la carpeta **/var/www/html**. Si desde el navegador de la propia máquina ponemos <http://localhost/prueba.php> y nos aparece esta página la instalación fue correcta:

PHP Version 7.0.8-0ubuntu0.16.04.3

System	Linux ignacio-Lenovo-B590 4.4.0-53-generic #74-Ubuntu SMP Fri Dec 2 15:59:10 UTC 2016 x86_64
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/7.0/apache2
Loaded Configuration File	/etc/php/7.0/apache2/php.ini
Scan this dir for additional .ini files	/etc/php/7.0/apache2/conf.d
Additional .ini files parsed	/etc/php/7.0/apache2/conf.d/10-mysqldb.ini, /etc/php/7.0/apache2/conf.d/10-opcache.ini, /etc/php/7.0/apache2/conf.d/10-pdo.ini, /etc/php/7.0/apache2/conf.d/15-xml.ini, /etc/php/7.0/apache2/conf.d/20-calendar.ini, /etc/php/7.0/apache2/conf.d/20-ctype.ini, /etc/php/7.0/apache2/conf.d/20-dom.ini, /etc/php/7.0/apache2/conf.d/20-exif.ini, /etc/php/7.0/apache2/conf.d/20-fileinfo.ini, /etc/php/7.0/apache2/conf.d/20-

4. Documenta como se inicia, re-inicia y detiene el servidor web Apache, y también como se comprueba que la configuración es correcta, pero desde la línea de comandos. Si conoces varios comandos para realizar dichas tareas, indícalos.

- Para iniciar el servidor podemos usar el comando “*service apache2 start*” o “*apache2ctl restart*” .
- Reiniciar el servidor podemos usar el comando “*service apache2 restart*” o “*apache2ctl restart*” .
- Para detener el servidor usaremos el comando “*service apache2 stop*” o “*apache2ctl stop*” .
- Para comprobar que la sintaxis de la configuración es correcta lo haremos con el comando “*apache2ctl configtest*” . Nos debe salir un mensaje con Syntax OK, en caso contrario nos saldrá información del error.

```
root@ignacio-Lenovo-B590:/home/ignacio# apache2ctl configtest
Syntax OK
root@ignacio-Lenovo-B590:/home/ignacio#
```

5. Documenta como hacer que el servidor web Apache2, escuche el puerto 808X (donde X es un número entre 0 y 9 que corresponde al último número de tu DNI o NIE, excluyendo la letra obviamente), además del puerto 80 (tiene que escuchar ambos puertos). Indica como se podría comprobar que efectivamente Apache escucha dicho puerto a través del navegador web y a través de la línea de comandos.

Teniendo en cuenta que mi último número del DNI es el 8, el puerto que queremos escuchar el es 8088 y el 80.

Para ello nos iremos a la carpeta /etc/apache2 con permisos de superusuario para poder editar el archivo ports.conf. Lo abrimos con el editor de texto y añadimos la línea *Listen 8088* debajo de la línea *Listen 80*:



```
ports.conf
/etc/apache2

# If you just change the port or add more ports here, you will likely also
# have to change the VirtualHost statement in
# /etc/apache2/sites-enabled/000-default.conf

Listen 80
Listen 8088

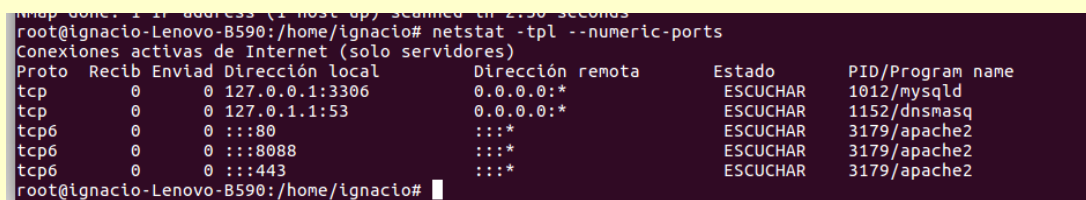
<IfModule ssl_module>
    Listen 443
</IfModule>

<IfModule mod_gnutls.c>
    Listen 443
</IfModule>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
```

Guardamos y comprobamos con *apache2ctl configtest* en terminal que la configuración es correcta, si es así, reiniciamos el servidor para que se efectúen los cambios con *apache2ctl restart*.

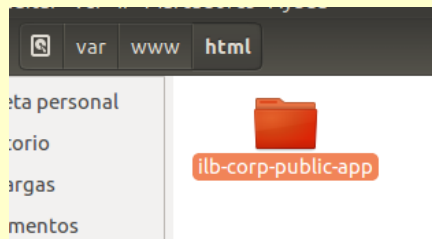
Comprobamos desde el navegador de la propia máquina que se han efectuado los cambios en la configuración poniendo la dirección <http://localhost:80>, luego la <http://localhost:8088> y debe salirnos la página de It's work de Apache, podemos comprobar que si ponemos otro puerto, por ejemplo el 8087 no se nos abre la página solicitada, nos da error. Para comprobar desde consola que puertos escucha Apache lo podemos hacer con el comando "*netstat -tln --numeric-ports*", también lo podríamos comprobar con nmap, con el comando *nmap -sT -O localhost*:



```
root@ignacio-Lenovo-B590:/home/ignacio# netstat -tln --numeric-ports
Conexiones activas de Internet (solo servidores)
Proto Recib Envíad Dirección local Dirección remota Estado PID/Program name
tcp 0 0 127.0.0.1:3306 0.0.0.0:* ESCUCHAR 1012/mysqld
tcp 0 0 127.0.1.1:53 0.0.0.0:* ESCUCHAR 1152/dnsmasq
tcp6 0 0 :::80 :::* ESCUCHAR 3179/apache2
tcp6 0 0 :::8088 :::* ESCUCHAR 3179/apache2
tcp6 0 0 :::443 :::* ESCUCHAR 3179/apache2
root@ignacio-Lenovo-B590:/home/ignacio#
```

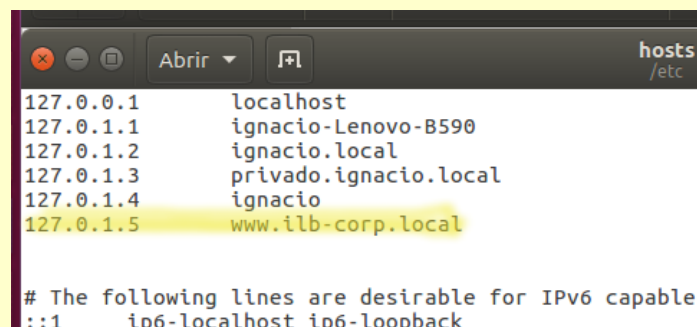
6. Configura un host virtual, basado en nombre, que funcione bajo el nombre **www.xyz-corp.local**, donde xyz serán las iniciales de tu nombre y apellidos (por ejemplo, si tu nombre es "Julia García Muñoz", el nombre sería **www.jgm-corp.local**). Este host virtual atenderá peticiones en el puerto 80. La aplicación web, en este caso un script php detallado al principio de la tarea, deberá estar alojada en un directorio del servidor llamado "**xyz-corp-public-app**", en una ubicación adecuada, donde nuevamente xyz son las iniciales de tus nombres y apellidos. Para este paso recuerda incluir además de la configuración necesaria (incluido los cambios realizados en el archivo **/etc/hosts**), los comandos que has utilizado para completar el ejercicio.

Lo primero que haremos será crear el directorio con la aplicación web, lo haremos en la ruta **/var/www/html** (recuerda que debemos hacer todo el ejercicio con los permisos root) y como dice el enunciado será **ilb-corp-public-app**:



En el guardaremos la aplicación que teníamos en el enunciado de la tarea, bajo el nombre **index.php**.

Seguidamente nos iremos a la ubicación **/etc** y buscamos el archivo **hosts**, lo editamos añadiendo esta línea con la entrada de dominio:



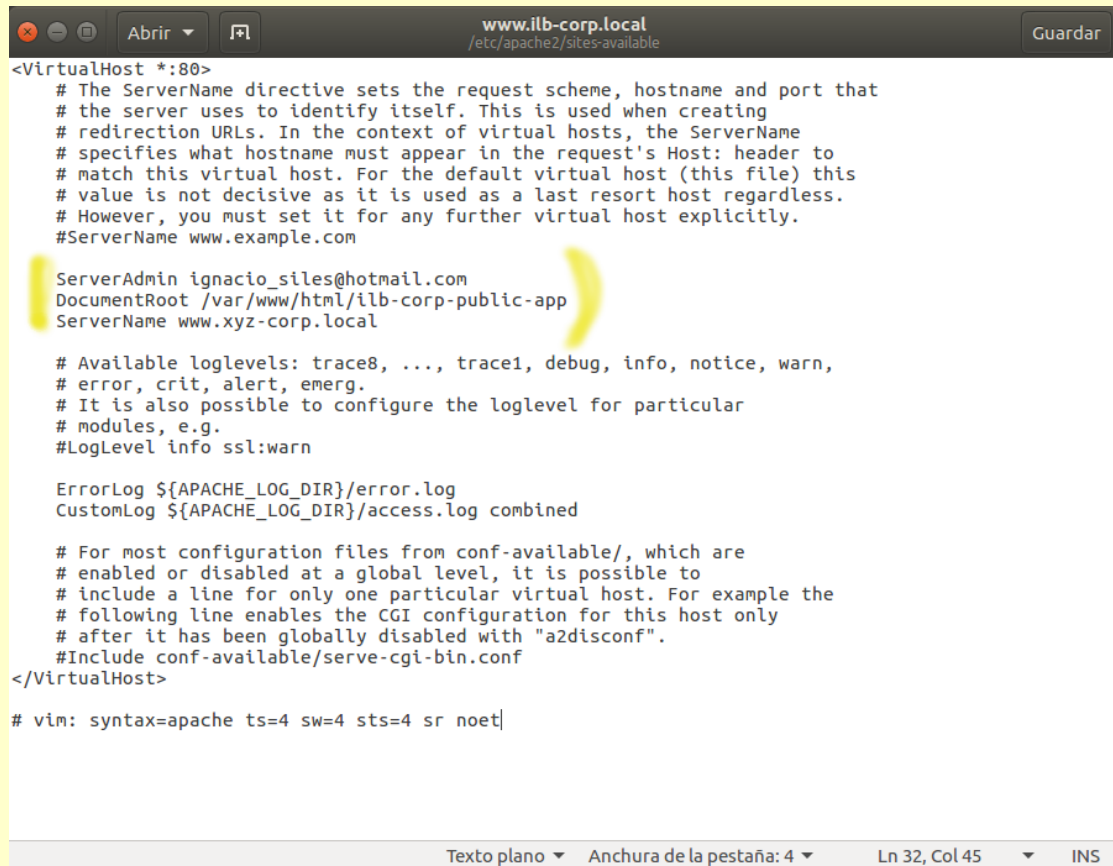
Guardamos.

Nos vamos al directorio donde se definen los virtualHosts, **/etc/apache2/sites-available** y creamos un archivo para el virtualHost **ilb-corp.conf**.

Creamos el archivo con el comando

"gedit /etc/apache2/sites-available/ilb-corp.conf", que nos abra el archivo directamente en el editor gedit.

Abrimos el archivo de la misma ubicación llamado 000-default.conf, lo copiamos y lo pegamos en el archivo que creamos anteriormente. Lo modificamos de manera que quede de la siguiente manera:



```
<VirtualHost *:80>
# The ServerName directive sets the request scheme, hostname and port that
# the server uses to identify itself. This is used when creating
# redirection URLs. In the context of virtual hosts, the ServerName
# specifies what hostname must appear in the request's Host: header to
# match this virtual host. For the default virtual host (this file) this
# value is not decisive as it is used as a last resort host regardless.
# However, you must set it for any further virtual host explicitly.
#ServerName www.example.com

ServerAdmin ignacio_siles@hotmail.com
DocumentRoot /var/www/html/ilb-corp-public-app
ServerName www.xyz-corp.local

# Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
# error, crit, alert, emerg.
# It is also possible to configure the loglevel for particular
# modules, e.g.
#LogLevel info ssl:warn

ErrorLog ${APACHE_LOG_DIR}/error.log
CustomLog ${APACHE_LOG_DIR}/access.log combined

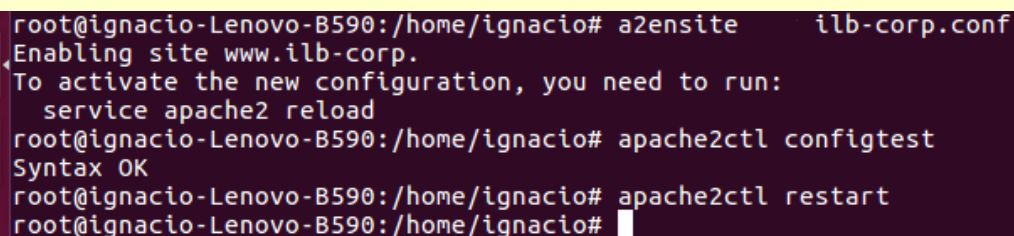
# For most configuration files from conf-available/, which are
# enabled or disabled at a global level, it is possible to
# include a line for only one particular virtual host. For example the
# following line enables the CGI configuration for this host only
# after it has been globally disabled with "a2disconf".
#Include conf-available/serve-cgi-bin.conf
</VirtualHost>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet|
```

Las modificaciones realizadas solo han sido efectuadas en las marcas amarillas, en el ServerAdmin he puesto mi email para que aparezca en los páginas de error como contacto para el posible usuario. El DocumentRoot la ruta donde esta alojada la página web, en mi caso /var/www/html/ilb-corp-public-app. Y el ServerName es la dirección que ponemos en el navegador para cargar nuestra página.

Ahora tenemos que habilitar el virtualHost, para ello usaremos el comando *a2ensite ilb-corp.conf*.

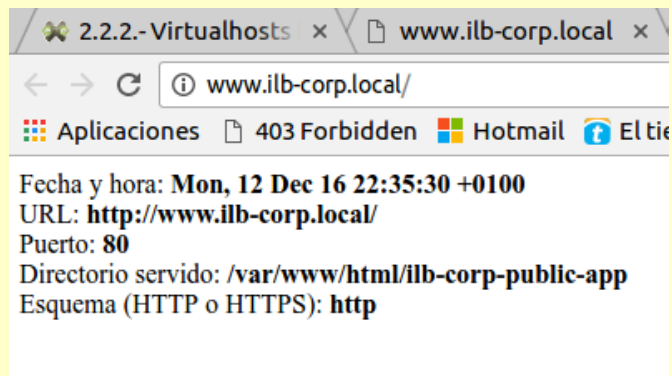
Comprobamos que la sintaxis de la configuración del servidor sea correcta y si es así restauramos el servidor:



```
root@ignacio-Lenovo-B590:/home/ignacio# a2ensite ilb-corp.conf
Enabling site www.ilb-corp.
To activate the new configuration, you need to run:
service apache2 reload
root@ignacio-Lenovo-B590:/home/ignacio# apache2ctl configtest
Syntax OK
root@ignacio-Lenovo-B590:/home/ignacio# apache2ctl restart
root@ignacio-Lenovo-B590:/home/ignacio#
```

Ahora tenemos que habilitar el

Nos vamos a nuestro navegador e introducimos la dirección www.ilb-corp.local, se nos tiene que cargar la página que hicimos index.php:

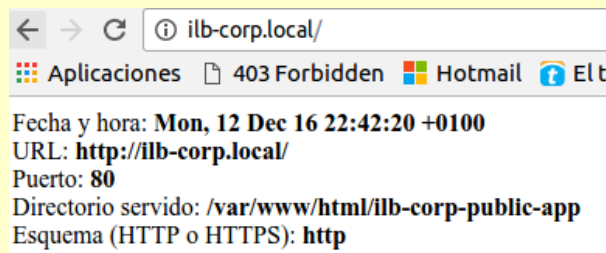


7. Ahora es el paso de hacer accesible la página web del apartado anterior, también a través de xyz-corp.local (por si a algún usuario se le olvida poner www). No olvides nuevamente documentar los cambios realizados en los archivos de configuración, así como todos los comandos utilizados.

Para esto tendremos que volver a editar el archivo hosts, que editamos en el ejercicio anterior, añadiendo en la línea introducida en el caso anterior, un espacio y **ilb-corp.local**

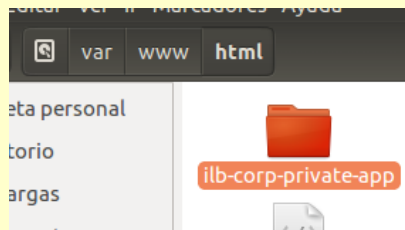
Guardamos y nos vamos a donde definimos el virtualhost ilb-corp.conf. Lo abrimos para editarlo y añadimos la línea ServerAlias ilb-corp.local, es la manera de indicar que también aceptaría ese nombre.

Comprobamos que funciona en el navegador:



8. **Ahora tendrás que crear otro host virtual, basado también en nombre, que funcione bajo el nombre `private.xyz-corp.local` (donde `xyz` sigue las reglas anteriores), y que solo responda al puerto `808X` (donde `X` será un número entre 0 y 9 que corresponde con el último número de tu DNI o NIE). La aplicación web, en este caso también un script php detallado al principio, deberá estar alojada en un directorio del servidor llamado `"xyz-corp-private-app"`.**

Para poder hacer esto tenemos que tener el puerto 8088 escuchando como ya hicimos en el ejercicio 5. Una vez hecho esto, crearemos otro directorio en la ruta `/var/www/html` llamada **`ilb-corp-private-app`**:



Copiamos el script `index.php` de la carpeta creada en el ejercicio anterior y la pegamos en esta que acabamos de crear.

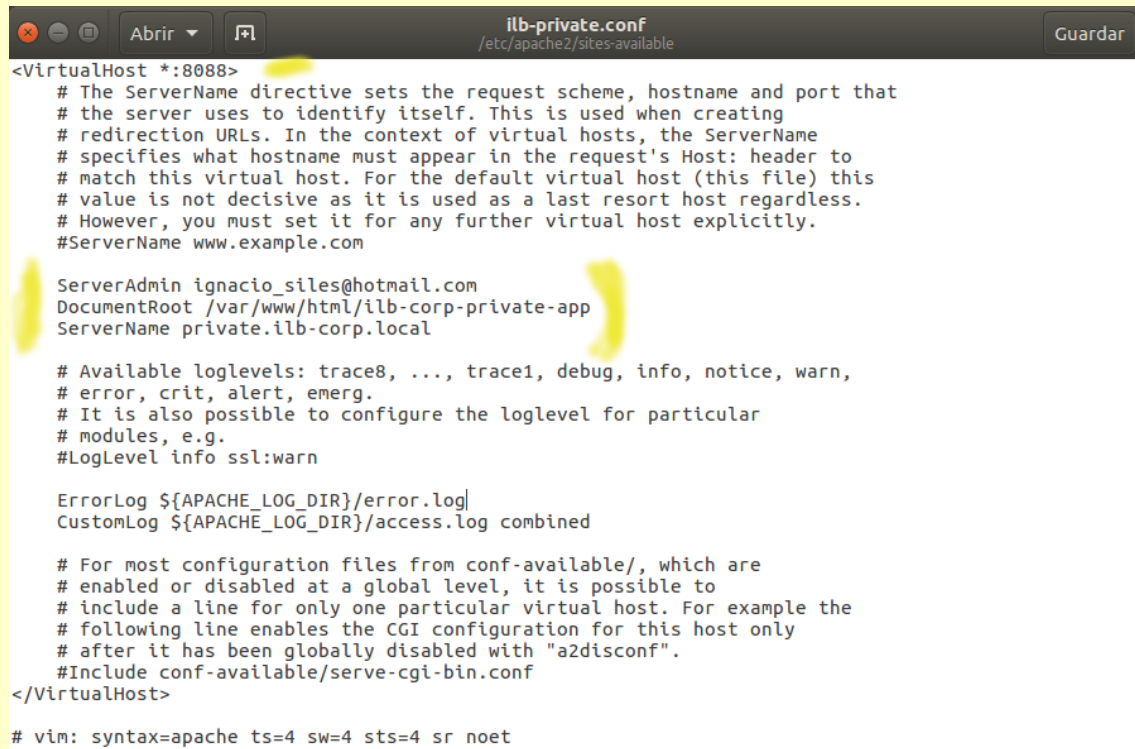
De nuevo volvemos a editar el archivo `hosts` añadiendo la siguiente línea debajo de las que añadimos en los ejercicios anteriores:

```
127.0.1.7      private.ilb-corp.local
```

Guardamos y nos vamos a `/etc/apache2/sites-available` donde copiaremos el archivo `000-default.conf` y lo pegaremos en esa misma ubicación con otro nombre y misma extensión, por ejemplo **`ilb-private.conf`**. Lo abrimos para editarlo, lo primero será modificar la primera línea (`<VirtualHost *:8088>`) estamos indicando que el `virtualHost` acogerá las peticiones de esas cualquier IP por el puerto 8088.

Ponemos el `DocumentRoot` `/var/www/html/ilb-corp-private-app` y el `ServerName` `private.ilb-corp.local`

Guardamos y desde consola habilitamos el virtual host con el comando *`a2ensite ilb-private.conf`*.



```
<VirtualHost *:8088>
# The ServerName directive sets the request scheme, hostname and port that
# the server uses to identify itself. This is used when creating
# redirection URLs. In the context of virtual hosts, the ServerName
# specifies what hostname must appear in the request's Host: header to
# match this virtual host. For the default virtual host (this file) this
# value is not decisive as it is used as a last resort host regardless.
# However, you must set it for any further virtual host explicitly.
#ServerName www.example.com

ServerAdmin ignacio_siles@hotmail.com
DocumentRoot /var/www/html/ilb-corp-private-app
ServerName private.ilb-corp.local

# Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
# error, crit, alert, emerg.
# It is also possible to configure the loglevel for particular
# modules, e.g.
#LogLevel info ssl:warn

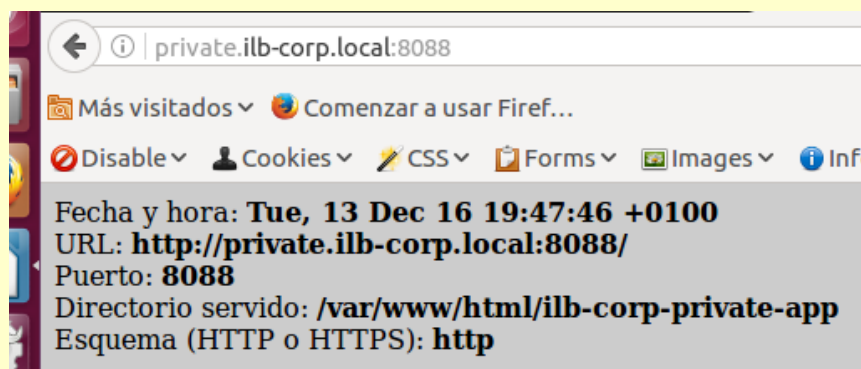
ErrorLog ${APACHE_LOG_DIR}/error.log
CustomLog ${APACHE_LOG_DIR}/access.log combined

# For most configuration files from conf-available/, which are
# enabled or disabled at a global level, it is possible to
# include a line for only one particular virtual host. For example the
# following line enables the CGI configuration for this host only
# after it has been globally disabled with "a2disconf".
#Include conf-available/serve-cgi-bin.conf
</VirtualHost>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
```

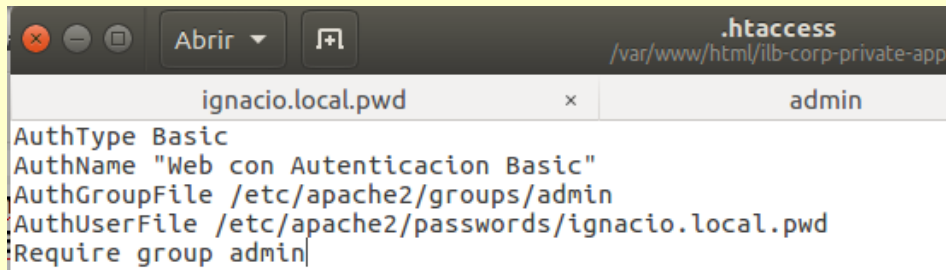
Comprobamos la sintaxis de configuración de apache con `apache2ctl configtest` si esta todo correcto restauramos el servidor con `apache2ctl restart`.

Comprobamos que funciona si en la barra de direcciones de nuestro navegador se nos abre nuestra pagina con la dirección `http://private.ilb-corp.local:8088`, y que en cambio, no lo hace si ponemos la misma dirección sin el puerto y otro puerto.



9. La aplicación web del ejercicio 8 solo será accesible por aquellas personas de la empresa con perteneciente al grupo "admin". Para ello configura en apache una autenticación HTTP Basic para el directorio correspondiente a través de un archivo ".htaccess".

Para esta tarea nos iremos al directorio de nuestra aplicación `/var/www/html/ilb-corp-private-app` y dentro de esta crearemos el archivo `.htaccess` con el comando `gedit .htaccess` y escribimos esto:

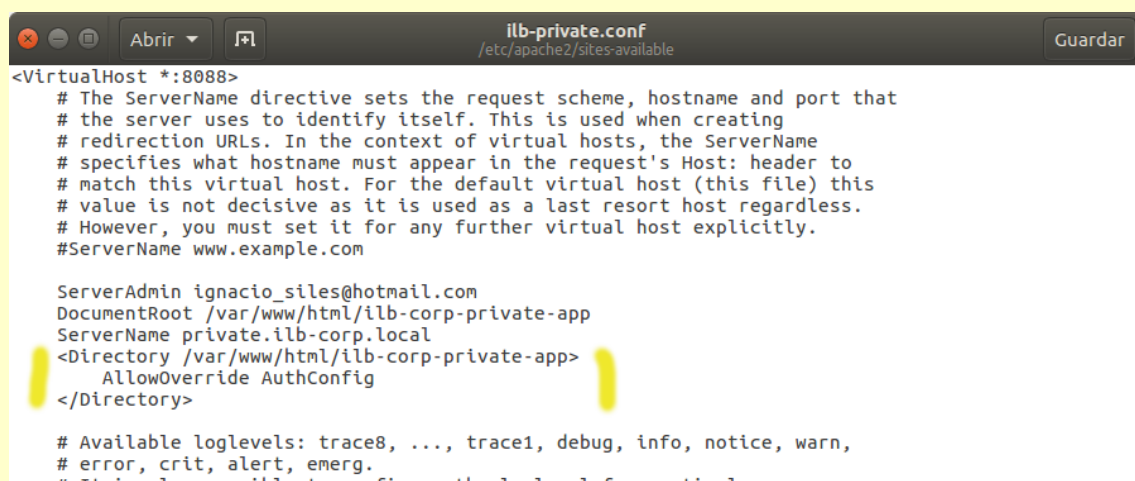
A screenshot of a gedit window titled ".htaccess" with the path "/var/www/html/ilb-corp-private-app". The window shows a table with two columns: "ignacio.local.pwd" and "admin". Below the table, the following configuration is entered:

```
AuthType Basic
AuthName "Web con Autenticacion Basic"
AuthGroupFile /etc/apache2/groups/admin
AuthUserFile /etc/apache2/passwords/ignacio.local.pwd
Require group admin
```

AuthType Basic nos indica el tipo de autenticación, en este caso Basic. **AuthName** especifica el nombre del dominio que se muestra al solicitar la autorización.

AuthGroupFile indica el archivo de texto y la ubicación del mismo que contiene la lista de grupos de usuarios. **AuthUserFile** es el archivo de texto y la ruta del mismo que contiene la lista de usuarios y contraseñas.

Ahora tendremos que ir al virtualHost (`/etc/apache2/sites-available/ilb-private.conf`) y abrirlo para editarlo, añadiendo las líneas marcadas en amarillo:

A screenshot of a gedit window titled "ilb-private.conf" with the path "/etc/apache2/sites-available". The window shows the configuration for a virtual host. The following lines are highlighted in yellow:

```
<Directory /var/www/html/ilb-corp-private-app>
    AllowOverride AuthConfig
</Directory>
```

Con esto estamos indicando que ese directorio se aplica la directiva `AllowOverride AuthConfig` para que se tomen en cuenta las directivas de los archivos `.htaccess` referidas a las directivas de autenticación.

Guardamos.

Ahora tenemos que crear el archivo de contraseñas. Lo ubicaremos en la ruta que ya indicamos en el `.htaccess (/etc/apache2/passwords)`, y lo haremos con la utilidad `htpasswd`, escribiendo el siguiente comando para CREAR dicho archivo:

```
htpasswd -c /etc/apache2/passwords/ignacio.local.pwd admin
```

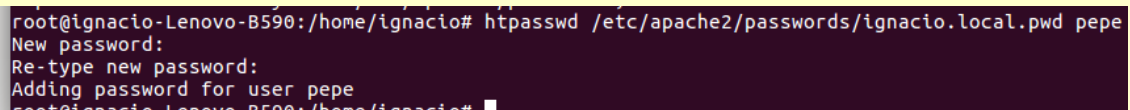
Al ejecutar el comando nos pedirá la contraseña (en mi caso `admin`, igual que el usuario).

Ahora tendremos que crear un archivo de grupo que asocie nombres de grupo con una lista de usuarios perteneciente a ese grupo. Lo crearemos en el directorio que indicamos a la directiva `AuthGroupFile` en el archivo `.htaccess (/etc/apache2/groups)` con el nombre **admin** como indicamos a la directiva `Require group admin` del archivo `.htaccess`. El formato es muy sencillo:



```
admin /etc/apache2/groups
admin: admin pepe
```

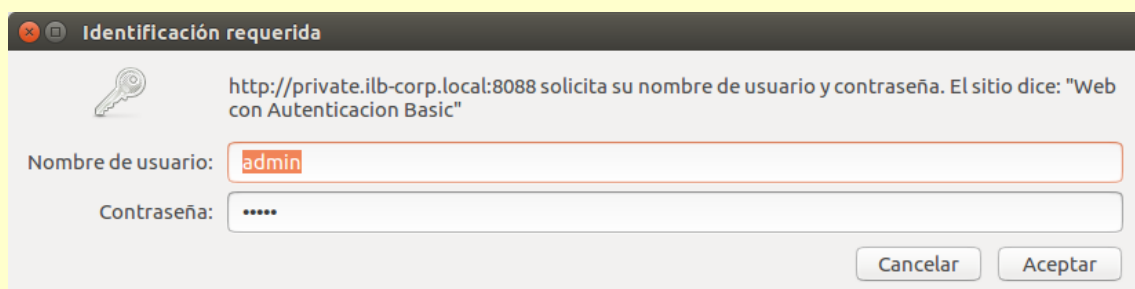
Si queremos agregar un usuario a un archivo de contraseñas ya existente como el realizado anteriormente, ejecutaremos el siguiente comando en la terminal (crearemos el usuario `pepe` con contraseña `pepe`):



```
root@ignacio-Lenovo-B590:/home/ignacio# htpasswd /etc/apache2/passwords/ignacio.local.pwd pepe
New password:
Re-type new password:
Adding password for user pepe
root@ignacio-Lenovo-B590:/home/ignacio#
```

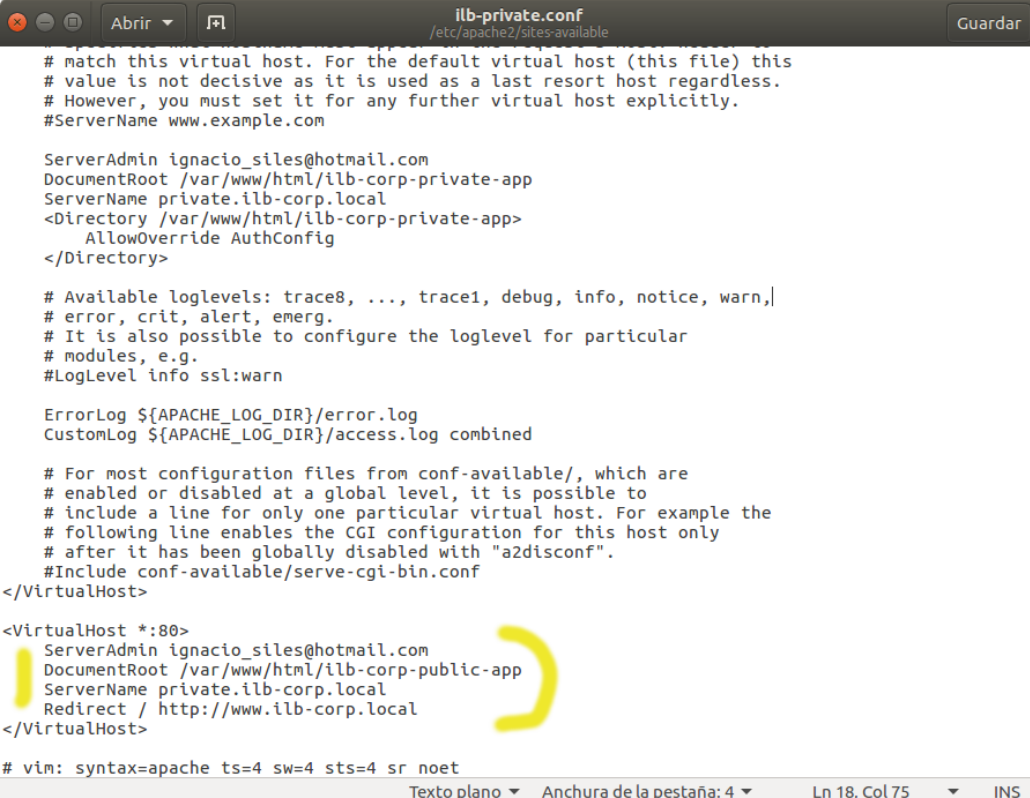
Comprobamos la sintaxis de configuración de apache con `apache2ctl configtest` si esta todo correcto restauramos el servidor con `apache2ctl restart`.

Nos vamos al navegador y accedemos al sitio privado <http://private.ilb-corp.local:8088>, nos tiene que pedir las claves e introduciendo la de alguno de los dos usuarios creados debemos acceder:



10. Si alguien por error teclea la URL “<http://private.xyz-corp.local>” en lugar de “<http://private.xyz-corp.local:808X>”, se le redireccionará a “<http://www.xyz-corp.local>”. Para realizar esta parte usa un nuevo VirtualHost donde se utilice la directiva Redirect (puedes incluir otro VirtualHost en el archivo de configuración correspondiente a private.xyz-corp.local).

Tal y como nos aconseja el enunciado nos iremos al archivo de configuración del sitio private.ilb-corp.local (/etc/apache2/sites-available/ilb-private.conf) y lo abrimos para editarlo. Añadimos la siguientes líneas (amarillo):



```
# match this virtual host. For the default virtual host (this file) this
# value is not decisive as it is used as a last resort host regardless.
# However, you must set it for any further virtual host explicitly.
#ServerName www.example.com

ServerAdmin ignacio_siles@hotmail.com
DocumentRoot /var/www/html/ilb-corp-private-app
ServerName private.ilb-corp.local
<Directory /var/www/html/ilb-corp-private-app>
    AllowOverride AuthConfig
</Directory>

# Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
# error, crit, alert, emerg.
# It is also possible to configure the loglevel for particular
# modules, e.g.
#LogLevel info ssl:warn

ErrorLog ${APACHE_LOG_DIR}/error.log
CustomLog ${APACHE_LOG_DIR}/access.log combined

# For most configuration files from conf-available/, which are
# enabled or disabled at a global level, it is possible to
# include a line for only one particular virtual host. For example the
# following line enables the CGI configuration for this host only
# after it has been globally disabled with "a2disconf".
#Include conf-available/serve-cgi-bin.conf
</VirtualHost>

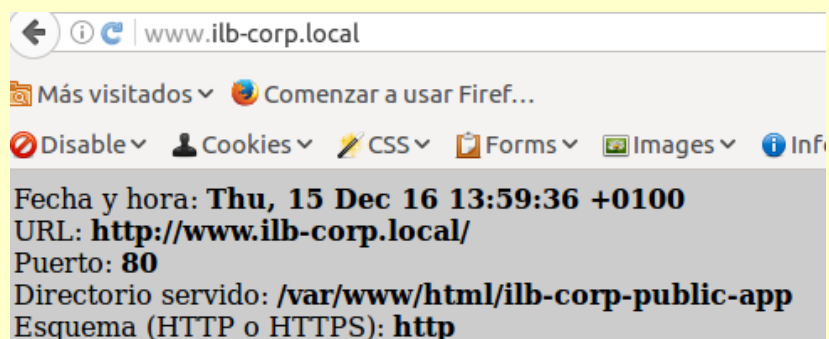
<VirtualHost *:80>
    ServerAdmin ignacio_siles@hotmail.com
    DocumentRoot /var/www/html/ilb-corp-public-app
    ServerName private.ilb-corp.local
    Redirect / http://www.ilb-corp.local
</VirtualHost>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
```

Estamos creando un virtualHost a continuación del anterior que escucha peticiones de cualquier IP por el puerto 80 con el ServerName private.ilb-corp.local y el DocumentRoot con la ubicación de la web a la que vamos a redireccionar (www.ilb-corp.local).

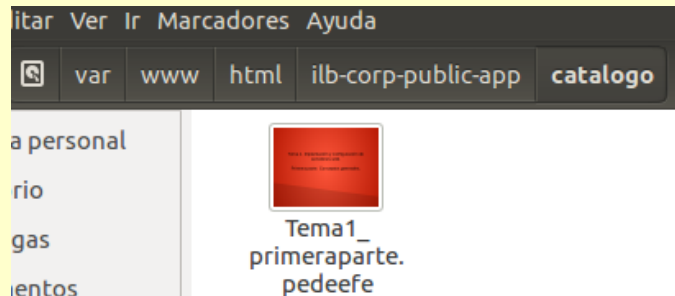
La directiva Redirect nos lleva de la URL indicada como primer parametro (/ , la dirección del ServerName) a la segunda URL.

Lo comprobamos en el navegador, como introduciendo la dirección <http://private.ilb-corp.local> nos lleva a <http://www.ilb-corp.local>:



11. En una carpeta de la web www.xyz-corp.local, habrá una carpeta que contendrá archivos PDF con extensión “.pedeefe”. Dicha carpeta se llamará "catalogo" y para ella hay que configurar el tipo MIME de forma que el contenido servido sea reconocido como archivos “PDF”.

Nos vamos a la carpeta de la ubicación de la web www.ilb-corp.local, `/var/www/html/ilb-corp-public-app` y creamos la carpeta catalogo. Dentro de esta tendremos archivos .pdf con la extensión .pedeefe:



Una vez hecho esto, nos vamos al archivo de configuración de la web www.ilb-corp.local y en el añadimos las siguientes líneas (amarillo):

```
ilb-corp.conf
/etc/apache2/sites-available

VirtualHost 127.0.1.5:80>
# The ServerName directive sets the request scheme, hostname
# the server uses to identify itself. This is used when crea
# redirection URLs. In the context of virtual hosts, the Ser
# specifies what hostname must appear in the request's Host:
# match this virtual host. For the default virtual host (thi
# value is not decisive as it is used as a last resort host
# However, you must set it for any further virtual host exp
#ServerName www.example.com

ServerAdmin ignacio_siles@hotmail.com
DocumentRoot /var/www/html/ilb-corp-public-app
ServerName www.ilb-corp.local
ServerAlias ilb-corp.local
<Directory /var/www/html/ilb-corp-public-app/catalogo>
  <Files *.pedeefe>
    ForceType application/pdf
  </Files>
</Directory>
```

Estamos indicando dentro del VirtualHost que en el directorio catalogo los archivos que cumplen el patrón *.pedeefe sean tratados como tipos .pdf gracias a la directiva ForceType y los parámetros MIME application/pdf.

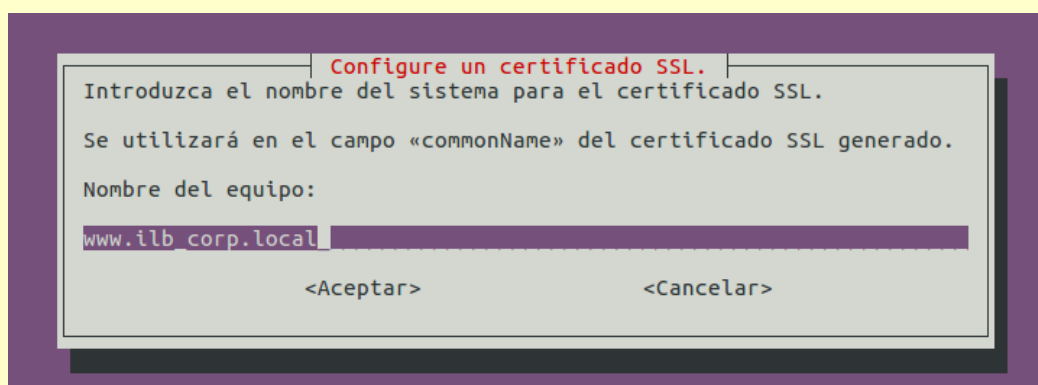
Para comprobar que funciona nos vamos al navegador y si introducimos la URL <http://www.ilb-corp.local/catalogo>, nos aparecerá la siguiente página:



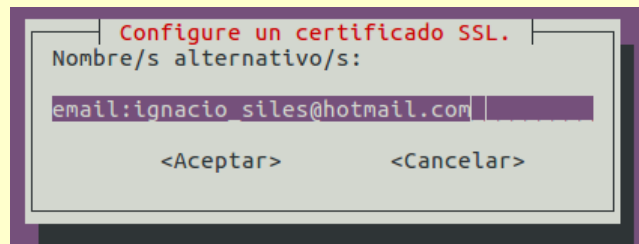
12. Configura el protocolo HTTPS, para que, aquellos usuarios que quieran acceder de forma segura a la web www.xyz-corp.local puedan hacerlo. Para esta parte es necesario que demuestres que dicho servicio está funcionando correctamente (por ejemplo, con una captura de pantalla del navegador que muestre la máxima información posible, donde se vea la información del certificado de seguridad y de fondo la página web).

Primero tendremos que asegurarnos que tenemos habilitado el modulo ssl, si no es así, lo hacemos con el comando `"a2enmod ssl"`. Si tenemos hecho esto, tendremos que crear nuestro certificado, para ello tendremos que tener instalado openssl (`apt-get install openssl`).

Y una vez instalado creamos el certificado con esta herramienta. Creamos el directorio donde guardar el certificado, en mi caso en la ruta `/etc/apache/certificados` `"mkdir /etc/apache2/certificados"`, y seguidamente el comando `"make-ssl-cert /usr/share/ssl-cert/ssleay.cnf /etc/apache2/certificados/ilb-corp.pem"`. Que es crear un certificado dentro de la carpeta certificados con nombre de archivo **ilb-corp.pem**, nos pedirá el nombre del DNS de la web:



También nos pedirá un DNS alternativo, en mi caso solo he indicado un email:



Ya tendremos nuestro propio certificado, ahora tendremos que crear un virtualHost en la carpeta **sites-available**. Copiaremos el contenido del archivo default-ssl.conf y pegaremos en el archivo de configuración del sitio, **ilb-corp.conf**, a continuación del virtualHost creado anteriormente, y editamos las siguientes líneas:

```
ilb-corp-ssl.conf
# If you have a dynamic shared object (DSO) support you can build
# the module as a DSO module. See the documentation for more
# information.
#Include conf-available/serve-cgi-bin.conf
</VirtualHost>

<IfModule mod_ssl.c>
  <VirtualHost *:443>
    ServerAdmin ignacio_siles@hotmail.com

    DocumentRoot /var/www/html/ilb-corp-public-app

    # Available loglevels: trace8, ..., trace1, debug, info, notice, warn
    # error, crit, alert, emerg.
    # It is also possible to configure the loglevel for particular
    # modules, e.g.
    #LogLevel info ssl:warn

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

    # For most configuration files from conf-available/, which are
    # enabled or disabled at a global level, it is possible to
    # include a line for only one particular virtual host. For example the
    # following line enables the CGI configuration for this host only
    # after it has been globally disabled with "a2disconf".
    #Include conf-available/serve-cgi-bin.conf

    # SSL Engine Switch:
    # Enable/Disable SSL for this virtual host.
    SSLEngine on

    # A self-signed (snakeoil) certificate can be created by installing
    # the ssl-cert package. See
    # /usr/share/doc/apache2/README.Debian.gz for more info.
    # If both key and certificate are stored in the same file, only the
    # SSLCertificateFile directive is needed.
    SSLCertificateFile /etc/apache2/certificados/ilb-corp.pem
    SSLCertificateKeyFile /etc/apache2/certificados/ilb-corp.pem
```

El virtualHost aceptara peticiones desde cualquier IP por el puerto 443, Ponemos nuestro email al ServerAdmin, el DocumentRoot con la ubicación de la web. Y las directivas SSLCertificateFile y SSLCertificateKeyFile con la ubicación y archivo del certificado.

Buscamos con Firefox la dirección con la URL [https:// www.ilb-corp.local](https://www.ilb-corp.local) y si es la primera vez nos dirá que el certificado no es de confianza porque no viene firmado por

una AC contenida en la lista que posee el navegador. Lo que haremos será añadir una excepción de seguridad con esta web.



13. Configurar los archivos de log tanto para www.xyz-corp.local como para private.xyz-corp.local, de manera que se cumpla con las siguientes premisas:

1. Debe haber un log de acceso personalizado, con un nombre de archivo llamado xyz-public_access.log y xyz-private_access.log respectivamente (donde xyz sigue las premisas ya vistas anteriores) que tenga common como formato de log (alias_logformat o nickname).

En el archivo de configuración de la aplicación pública (/etc/apache2/ilb-corp.conf), agregamos la siguiente línea (amarillo).

```
ServerAdmin ignacio_siles@hotmail.com
DocumentRoot /var/www/html/ilb-corp-public-app
ServerName www.ilb-corp.local
ServerAlias ilb-corp.local
CustomLog /var/log/ilb-public_access.log common
<Directory /var/www/html/ilb-corp-public-app/catalogo>
    <Files *.pedeefe>
        ForceType application/pdf
    </Files>
</Directory>
```

En la directiva CustomLog indicamos la ruta del archivo personalizado ilb-public_access.log y seguidamente el formato del registro, en este caso common. Para la aplicación privada en el mismo lugar del archivo /etc/apache2/ilb-private.conf añadimos la siguiente línea:

CustomLog /var/log/ilb-private_access.log common

2. Ambos logs deben rotarse cada 15 días y mantenerse un periodo de un año de logs (después de ese año, que es lo legalmente establecido, se descartan).

Para poder llevar a cabo la rotación usaremos los Piped Logs, es decir escribiremos la información de los registros a través de otro proceso. Debemos de saber cuantos archivos necesitaremos para el registro en un año, si deseamos que la rotación se haga cada 15 días tendremos que calcular $365/15 = 25$ archivos necesitaremos.

Sabiendo esto y que 15 días son 1296000 segundos, modificaremos la directiva CustomLog del apartado anterior de la siguiente forma:

```
CustomLog "|$rotatelog -n 25 /var/log/ilb-public_access.log 1296000" common
```

Esto lo hemos hecho en la aplicación pública, en la privada lo haremos así:

```
CustomLog "|$rotatelog -n 25 /var/log/ilb-private_access.log 1296000" common.
```

14. **Configurar apache para que al acceder a una carpeta llamada “config”, tanto en la web www.xyz-corp.local como en private.xyz-corp.local, retorne un error 404 de recurso no encontrado.**

Para comprobar que funciona creare una carpeta **/config** en **/var/www/html/ilb-corp-public-app** y otra en **/var/www/html/ilb-corp-private-app** en la que insertaremos un archivo de ejemplo, si accedemos desde el navegador sin mas a este directorio nos aparecerá esto:



Sin embargo si indicamos la directiva en los dos virtualHost la siguiente directiva: **Redirect 404 /config**, nos saldra la siguiente página de error:

