

---

# Práctica Final

De

# Administración de Sistemas

---

<sup>name</sup>  
DANIEL HERAS QUESADA



UNIVERSIDAD DE SALAMANCA

---

# Índice

<b>1. Introducción</b>	<b>3</b>
<b>2. Configuración inicial</b>	<b>4</b>
<b>3. Directorio de apuntes</b>	<b>4</b>
<b>4. Copias de seguridad</b>	<b>5</b>
<b>5. Conexión WEB segura</b>	<b>6</b>
<b>6. Base de datos</b>	<b>9</b>
<b>7. Servidor de correo</b>	<b>10</b>
7.1. Cliente de correo . . . . .	11
<b>8. Perl y CGI</b>	<b>13</b>
8.1. Registro de usuarios . . . . .	13
8.2. Fichero de condiciones . . . . .	14
8.3. Monitorización . . . . .	14
8.4. Inicio de sesión . . . . .	16
8.5. Borrado de usuarios . . . . .	18
8.6. Log de accesos . . . . .	19
<b>9. Blog privado</b>	<b>19</b>
<b>10. Bibliografía</b>	<b>20</b>

## 1. Introducción

Para la construcción del sistema exigido por el enunciado, partiré de una instalación de Debian 11 en una máquina virtual. La elección de la versión del sistema es clara, al tratarse de una distribución estable como es Debian, la mejor opción suele ser optar por el último de sus productos. La elección de Debian como sistema base radica en su popularidad en los sistemas servidor y en el motivo de esta popularidad: su estabilidad y resistencia a fallos. Descartamos por el mismo motivo otras distribuciones populares como Arch, cuya inestabilidad puede convertirse en un problema en tiempo real. Durante toda la extensión de este documento iré indicando los programas y técnicas usadas para cumplir con los requisitos del sistema, que deberá ser funcional y de fácil mantenimiento. Prestaré especial atención a los permisos concedidos a los diferentes usuarios sobre los ficheros más delicados, y buscaré facilitar la interacción con el sistema mediante las interfaces WEB. El grueso del sistema se basará en el desarrollo de scripts de PERL y en su interacción con un conjunto de documentos html.

**Funcionamiento.** El sistema planteado permitirá al usuario acceder mediante varios protocolos de conexión: *HTTP*, *HTTPS*, *SSH*, *SFTP* y *FTP*. Dejando para *HTTP* y *HTTPS* las tareas de registro y borrado de la cuenta de usuario. Este protocolo aportará también acceso a una página de información que mostrará el estado de los diferentes servicios del sistema y una página principal dónde la empresa contratante podrá incluir información adicional.



**Usuarios.** El sistema concibe 2 tipos de usuarios: profesores y alumnos. La diferencia entre estos es mínima, puesto que únicamente cambia en el tipo de acceso al directorio de apuntes, sobre el que los profesores tendrán control pero los alumnos únicamente podrán leer los archivos allí compartidos. Ambos tendrán limitado el acceso al disco a 80M y serán avisados al llegar a 60M. El resto de servicios del sistema tendrán igual distribución a ambos.

## 2. Configuración inicial

**Hostname.** Defino el nombre de host de la máquina como el nombre de la supuesta empresa, en este caso 'TrueSight'.

```
TrueSight
/etc/hostname
```

**Actualizar.** El primer paso para mejorar la experiencia de instalación del servidor actualizando el sistema.

```
# apt-get update
# apt-get upgrade
```

**Utilidades.** Necesario será también la instalación de paquetes necesarios para la administración del servidor.

```
# apt-get vim neovim net-tools
```

**SSH y SFTP.** Usaremos estos protocolos como medio de conexión de configuración remota del sistema. La instalación consistirá en instalar el paquete y habilitar la conexión remota del usuario 'root'.

```
PermitRootLogin yes
/etc/ssh/sshd_config
```

**Quotas.** El proceso de configuración de quotas es igual que el los ejercicios de clase, he seguido es guía. Lo único relevante es la configuración de quotas a nivel de grupo. Esto evita que tenga que escribir un script de Perl para su gestión.

```
root@TrueSight:~# setquota -g alumno 60M 80M 0 0 /
```

Comprobamos la correcta configuración de estas

```
root@TrueSight:~# quota -g alumno -vs
Disk quotas for group alumno (gid 1002):
  Filesystem    space   quota   limit   grace   files   quota   limit   grace
  /dev/vda1      0K    61440K  81920K           0         0       0
root@TrueSight:~# quota -g profesor -vs
Disk quotas for group profesor (gid 1003):
  Filesystem    space   quota   limit   grace   files   quota   limit   grace
  /dev/vda1      4K    61440K  81920K           1         0       0
```

## 3. Directorio de apuntes

Crear un directorio de apuntes implica crear un directorio y dar los permisos necesarios, dando propiedad del mismo al grupo de profesores y permisos de lectura a cualquier otro miembro del sistema. Para ello es útil tener creados ya los grupos necesarios:

```
alumno:x:1002:
profesor:x:1003:
root@TrueSight:/etc#
```

```
root@TrueSight:~# mkdir /mnt/apuntes
root@TrueSight:~# cd /mnt/apuntes/
root@TrueSight:/mnt/apuntes# cd ..
root@TrueSight:/mnt# ls
apuntes
root@TrueSight:/mnt# chgrp profesor apuntes/
root@TrueSight:/mnt# chmod 774 apuntes/
```

#### 4. Copias de seguridad

**Script.** Para efectuar las copias de seguridad escribo un script muy sencillo que permita hacer copias de los directorios *'home'* comprimidas o sin comprimir. El destino de estas copias será una carpeta en el directorio *'root'*, así no habrá peligro de acceso inadecuado a estas copias de seguridad.

```
8 #!/bin/sh
7
6 for dir in /home/*; do
5     if [ $2 -eq 0 ]; then
4         rsync -r $dir /root/BackUp
3     else
2         tar -cf "/root/BackUp/${dir:6}.tar" $dir
1     fi
9 done
~
~
INSERT backup.sh[+]
```

Para conseguir que este script se ejecute periódicamente usaré CRON, por ello hago la instalación del paquete y modifico el archivo de CRON del usuario *root*". El ejecutable lo deberá ejecutar el *root* por ser el único que tiene y deberá tener permisos de acceso a las carpetas personales, además que la copia se hará en el directorio */root*".

```
# apt-get install cron
```

```
# crontab -e
```

```
0 5 * * * bash /etc/Scripts/backup.sh > /dev/null 2>&1
crontab.WWmnLX/crontab [ +]
```

## 5. Conexión WEB segura

**Apache.** El primer paso antes de plantear la seguridad de una conexión web es instalar el propio servidor, que será apache.

```
# apt-get install apache2
```

Para garantizar la seguridad de la conexión deberemos habilitar las conexiones HTTPS, que aprovechen la capa segura de sockets o SSL, que son unos protocolos de seguridad para proteger el tráfico. Esta capa se activa con un módulo de SSL que viene con el sistema base:

```
# a2enmod ssl
```

Deberemos reiniciar apache para que los cambios tengan lugar

```
# service apache2 restart
```

**Certificados SSL.** Para poder hacer uso de este módulo necesitaremos un certificado y una clave, estos los guardaré en el directorio `/etc/apache2/ssl`.

```
# sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /etc/apache2/ssl/apache.key  
-out /etc/apache2/ssl/apache.crt
```

Rellenaremos el contenido de los campos solicitados, poniendo atención al campo *Common Name* pues será dónde debemos rellenar el nombre del dominio en caso de tenerlo o la ip del servidor en nuestro caso.

```
writing new private key to '/etc/apache2/ssl/apache.key'  
-----  
You are about to be asked to enter information that will be incorporated  
into your certificate request.  
What you are about to enter is what is called a Distinguished Name or a DN.  
There are quite a few fields but you can leave some blank  
For some fields there will be a default value,  
If you enter '.', the field will be left blank.  
-----  
Country Name (2 letter code) [AU]:ES  
State or Province Name (full name) [Some-State]:CyL  
Locality Name (eg, city) []:Salamanca  
Organization Name (eg, company) [Internet Widgits Pty Ltd]:USAL  
Organizational Unit Name (eg, section) []:USAL  
Common Name (e.g. server FQDN or YOUR name) []:192.168.122.235  
Email Address []:dani.heras@usal.es  
root@admin:~#
```

**Configuración de Apache.** Tomaré como base el fichero `/etc/apache2/sites-available/default-ssl.conf`, y añadiré las líneas necesarias para que el servidor reconozca la ubicación de los certificados.

```
IfModule mod_ssl.c>
<VirtualHost _default_:443>
    ServerAdmin dani.heras@usal.es
    ServerName 192.168.122.235

    DocumentRoot /var/www/html

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

    SSLEngine on

    SSLCertificateFile      /etc/apache2/ssl/apache.crt
    SSLCertificateKeyFile   /etc/apache2/ssl/apache.key

    #SSLCertificateChainFile /etc/apache2/ssl.crt/server-ca.crt

    #SSLCACertificatePath /etc/ssl/certs/
    #SSLCACertificateFile /etc/apache2/ssl.crt/ca-bundle.crt

    #SSLCARevocationPath /etc/apache2/ssl.crl/
    #SSLCARevocationFile /etc/apache2/ssl.crl/ca-bundle.crl

    #SSLVerifyClient require
    #SSLVerifyDepth 10

    #SSLOptions +FakeBasicAuth +ExportCertData +StrictRequire
    <FilesMatch "\.(cgi|shtml|phtml|php)$">
        SSLOptions +StdEnvVars
    </FilesMatch>
    <Directory /usr/lib/cgi-bin>
        SSLOptions +StdEnvVars
    </Directory>

</VirtualHost>
</IfModule>
```

Figura 1: /etc/apache2/sites-available/default-ssl.conf

Será necesario activar la configuración antes editada y reiniciar el servidor:

```
# a2ensite default-ssl.conf
# service apache2 restart
```

**Comprobación.** Podemos comprobar el éxito de los cambios accediendo con *"https"* a la dirección ip.

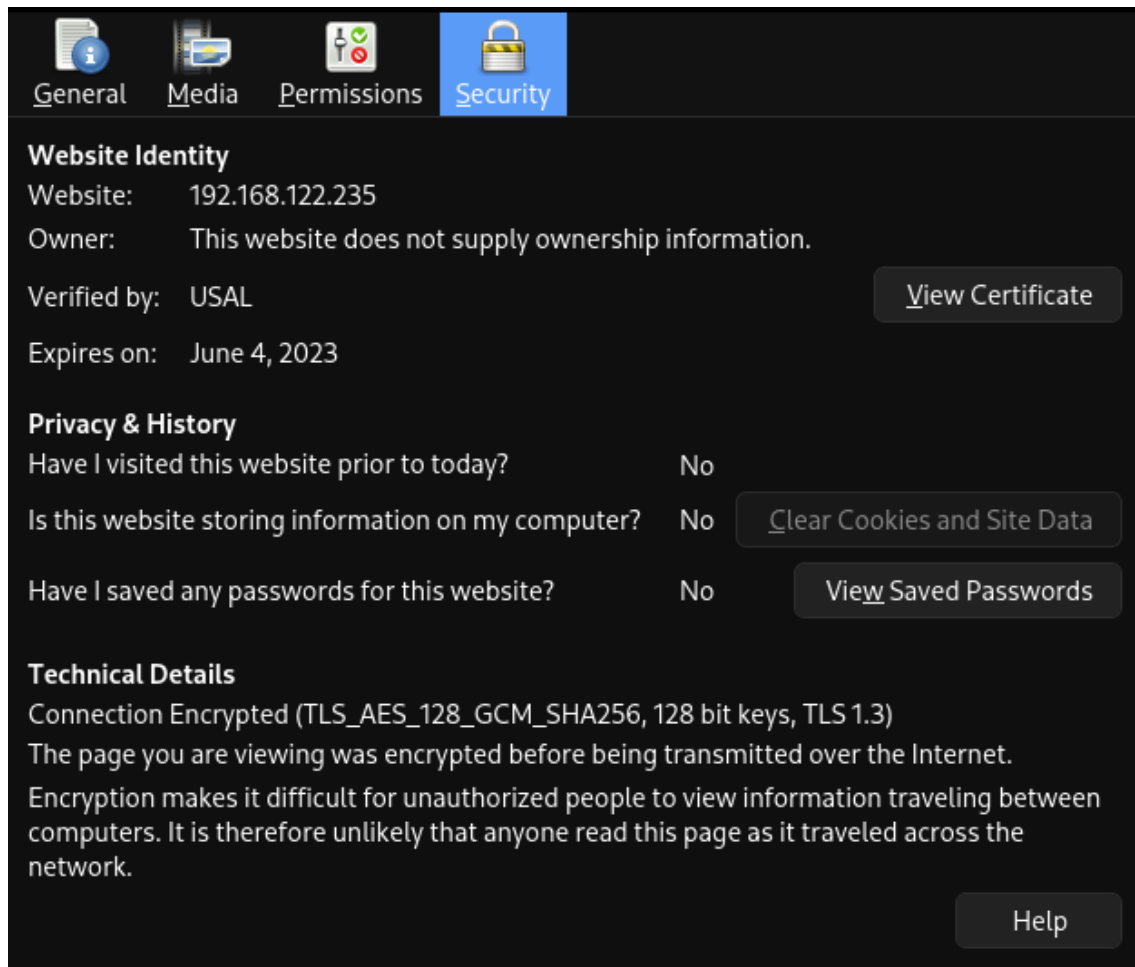


Figura 2: Certificado SSL en navegador.

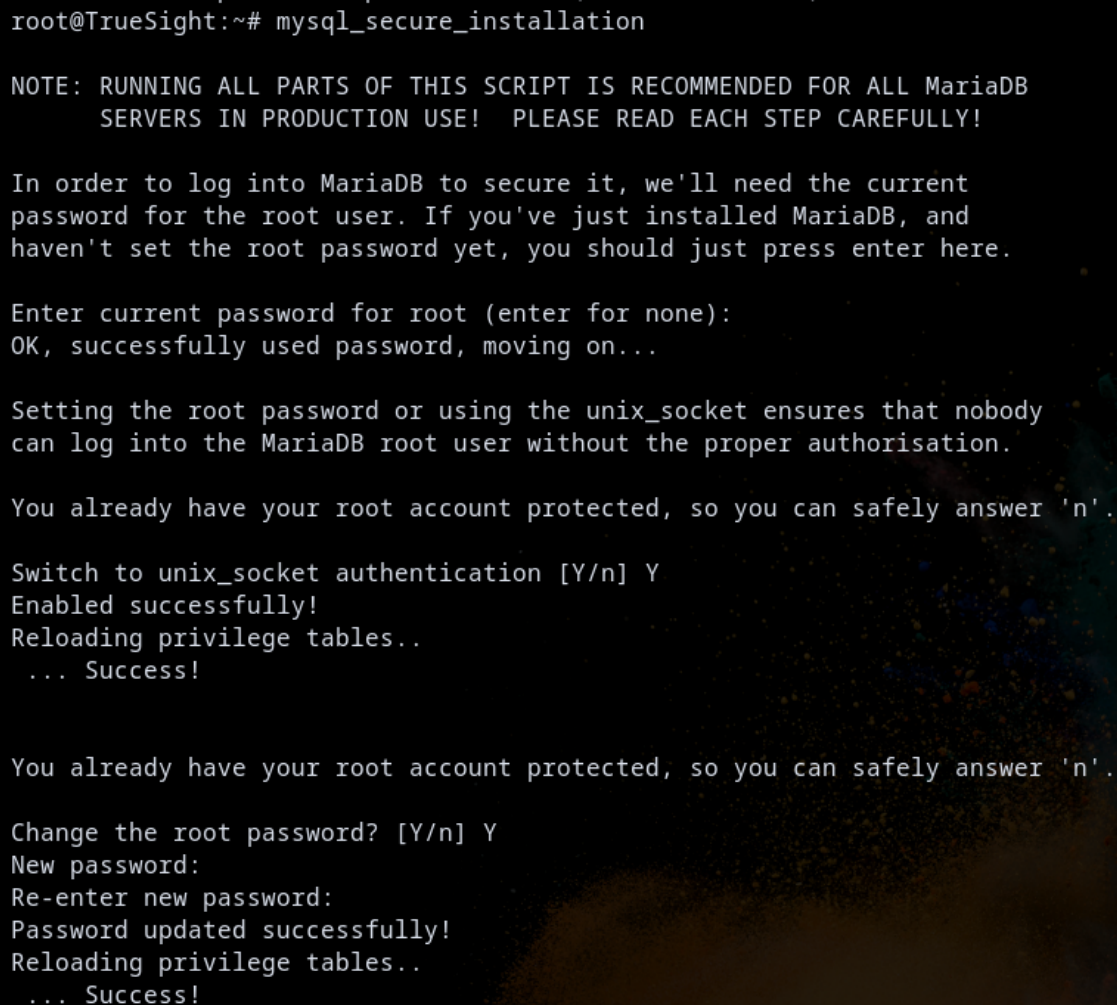


## 6. Base de datos

**Instalación.** El primer paso es instalar los paquetes necesarios, que son el cliente y el servidor.

```
# apt-get install mariadb-server mariadb-client
```

Con el paquete ya en el sistema hacemos la configuración inicial.

A screenshot of a terminal window showing the output of the 'mysql\_secure\_installation' script. The script starts with a note recommending running all parts for production use. It then prompts for the current root password, which is successfully used. Next, it asks to switch to unix\_socket authentication, which is enabled successfully. It then prompts to change the root password, which is also done successfully. The background of the terminal has a dark, starry space theme.

```
root@TrueSight:~# mysql_secure_installation

NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB
      SERVERS IN PRODUCTION USE!  PLEASE READ EACH STEP CAREFULLY!

In order to log into MariaDB to secure it, we'll need the current
password for the root user. If you've just installed MariaDB, and
haven't set the root password yet, you should just press enter here.

Enter current password for root (enter for none):
OK, successfully used password, moving on...

Setting the root password or using the unix_socket ensures that nobody
can log into the MariaDB root user without the proper authorisation.

You already have your root account protected, so you can safely answer 'n'.

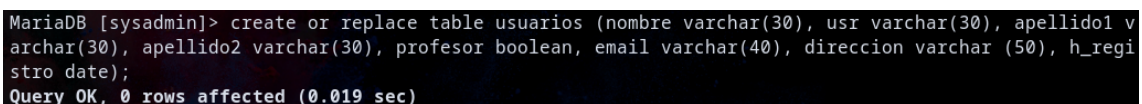
Switch to unix_socket authentication [Y/n] Y
Enabled successfully!
Reloading privilege tables..
... Success!

You already have your root account protected, so you can safely answer 'n'.

Change the root password? [Y/n] Y
New password:
Re-enter new password:
Password updated successfully!
Reloading privilege tables..
... Success!
```

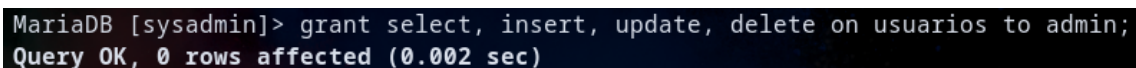
Figura 3: Configuración de MariaDB.

**Configuración.** Hecha ya la instalación y teniendo en cuenta que en futuro necesitaremos tener un base de datos creada y una tabla de usuarios definida, hacemos ya esos procesos.

A screenshot of the MariaDB command line interface. The user 'sysadmin' has executed a 'create or replace table' command to create a table named 'usuarios' with various columns. The output shows 'Query OK, 0 rows affected (0.019 sec)'.

```
MariaDB [sysadmin]> create or replace table usuarios (nombre varchar(30), usr varchar(30), apellidol v
archar(30), apellido2 varchar(30), profesor boolean, email varchar(40), direccion varchar (50), h_regi
stro date);
Query OK, 0 rows affected (0.019 sec)
```

Daremos los permisos necesarios a un usuario creado llamado '*admin*', este usuario será el que accederá a la base de datos desde CGI.

A screenshot of the MariaDB command line interface. The user 'sysadmin' has executed a 'grant' statement to give select, insert, update, and delete permissions to the 'admin' user on the 'usuarios' table. The output shows 'Query OK, 0 rows affected (0.002 sec)'.

```
MariaDB [sysadmin]> grant select, insert, update, delete on usuarios to admin;
Query OK, 0 rows affected (0.002 sec)
```

## 7. Servidor de correo

**Instalación.** Instalar el servidor de correo localmente es sencillo, únicamente deberemos instalar el paquete correspondiente y seguir los pasos de una configuración inicial. Solo es necesario recalcar la elección de servidor local en las opciones de configuración.

```
root@TrueSight:~# apt install postfix
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
Paquetes sugeridos:
  procmail postfix-mysql postfix-pgsql postfix-ldap postfix-pcre postfix-lmdb
  postfix-sqlite sasl2-bin | dovecot-common resolvconf postfix-cdb mail-reader ufw
  postfix-doc
Se instalarán los siguientes paquetes NUEVOS:
  postfix
0 actualizados, 1 nuevos se instalarán, 0 para eliminar y 0 no actualizados.
Se necesita descargar 1.540 kB de archivos.
Se utilizarán 4.283 kB de espacio de disco adicional después de esta operación.
Des:1 http://deb.debian.org/debian bullseye/main amd64 postfix amd64 3.5.6-1+b1 [1.540 kB]
Descargados 1.540 kB en 2s (744 kB/s)
Preconfigurando paquetes ...
Seleccionando el paquete postfix previamente no seleccionado.
(Leyendo la base de datos ... 38847 ficheros o directorios instalados actualmente.)
Preparando para desempaquetar .../postfix_3.5.6-1+b1_amd64.deb ...
Desempaquetando postfix (3.5.6-1+b1) ...
Configurando postfix (3.5.6-1+b1) ...
Añadiendo el grupo 'postfix' (GID 114) ...
Hecho.
Añadiendo el usuario del sistema 'postfix' (UID 107) ...
Añadiendo un nuevo usuario 'postfix' (UID 107) con grupo 'postfix' ...
No se crea el directorio personal '/var/spool/postfix'.
Creating /etc/postfix/dynamicmaps.cf
Añadiendo el grupo 'postdrop' (GID 115) ...
Hecho.
```

Figura 4: Instalación de PostFix.

Editaremos el fichero `/etc/postfix/virtual` cuando queramos añadir direcciones a usuarios.

**POP/IMAP.** Necesitaremos también la instalación de dovecot para esta gestión y, más concretamente, para el correcto funcionamiento del programa web RoundCube que más adelante instalaremos. Instalamos con la siguiente orden:

```
# apt -y install dovecot-core dovecot-pop3d dovecot-imapd
```

Y configuraremos modificando el fichero de configuración apropiado: `/etc/dovecot/conf.d/10-master.conf`.

```
# Postfix smtp-auth
unix_listener /var/spool/postfix/private/auth {
  mode = 0666
  user = postfix
  group = postfix
}
```

Figura 5: `/etc/dovecot/conf.d/10-master.conf`.

## 7.1. Cliente de correo

**Pasos previos.** Para la instalación de un cliente web de correo, necesitaremos instalar las dependencias primero. Como ya hemos dejado PostFix y Dovecot configurados en el paso anterior, nos dirigimos directamente a la instalación del lenguaje base del cliente, que es PHP y sus dependencias para funcionar en la web.

```
# apt -y install php php-cgi libapache2-mod-php php-common php-pear php-mbstring
```

Deberemos después, activar este módulo en Apache:

```
root@TrueSight:~# a2enconf php7.4-cgi
Enabling conf php7.4-cgi.
To activate the new configuration, you need to run:
    systemctl reload apache2
root@TrueSight:~#
```

**Base de datos.** Debemos configurar la base de datos de manera que RoundCube pueda usarla, para ello crearemos la base de datos con el nombre apropiado y daremos privilegios al usuario usado por este sistema:

```
root@TrueSight:~# mysql
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 30
Server version: 10.5.15-MariaDB-0+deb11u1 Debian 11

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> create database roundcube;
Query OK, 1 row affected (0.000 sec)

MariaDB [(none)]> grant all privileges on roundcube.* to roundcube@'localhost' identified by
'labii';
Query OK, 0 rows affected (0.002 sec)

MariaDB [(none)]> flush privileges;
Query OK, 0 rows affected (0.001 sec)
```

**Instalación.** La instalación se realiza con la herramienta deseada de gestión de paquetes:


```
# apt -y install roundcube roundcube-mysql
```

**Configuración de Apache.** Los pasos previos se han encargado de habilitar la configuración de RoundCube para Apache, pero voy a hacer un paso extra para el sencillo acceso a esta interfaz: añadir un alias al fichero de configuración.

```
Alias /roundcube /var/lib/roundcube/public_html
/etc/apache2/conf-enabled/roundcube.conf
```

Resulta en la instalación completa:


🔒 <https://192.168.122.235/roundcube/>



LOGIN

Roundcube Webmail

Desde esta interfaz podremos hacer el login:



LOGIN

Roundcube Webmail

## 8. Perl y CGI

### 8.1. Registro de usuarios

**Gestión de usuarios.** Para añadir un usuario al sistema desde la página web, emplearé los módulos de PERL *CGI* y *Linux::usermod*. Aunque deberé realizar una serie de pasos previos:

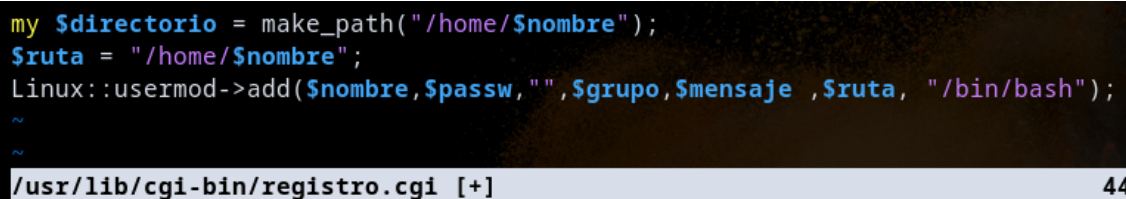
1. Instalar los módulos mediante la consola de CPAN:

```
# perl -MCPAN -e shell
■ cpan[n]> install CGI
■ cpan[n]> install Linux::usermod
■ cpan[n]> install File::Path
■ cpan[n]> install File::chown
■ cpan[n]> install DBI
■ cpan[n]> install Email::Send::SMTP::Gmail
■ cpan[n]> install Time::Piece::MySQL
■ cpan[n]> install local::lib
■ cpan[n]> install File:Slurper
```

2. Activo el módulo de apache de *CGI*, de esta manera permitiré la interacción con la web

```
# a2enmod cgi
```

3. Deposito los archivos ejecutables en el directorio */usr/lib/cgi-bin/*



```
my $directorio = make_path("/home/$nombre");
$ruta = "/home/$nombre";
Linux::usermod->add($nombre,$passwd,"",$grupo,$mensaje,$ruta,"/bin/bash");
~
~
/usr/lib/cgi-bin/registro.cgi [+]
```

4. Doy permisos del ejecutable al usuario web, que será quien lo ejecute:

```
# chmod a+x registro.cgi
```

5. Añado al usuario *www-data* al grupo *shadow*, esto es necesario para la creación de usuarios.

```
# usermod -a -G shadow www-data
```

6. Y, por último, doy propiedad del directorio */home* a *www-data*. Este paso es necesario para la gestión de las carpetas de usuario desde scripts.

```
# chown www-data /home
```

Con este sistema en mente puedo ya hacer una página web que contenga un formulario y ejecute el script creado anteriormente.

```
<form class="pure-form" method="POST" action="/cgi-bin/registro.cgi">
  <fieldset class="formulario">
    <input name="usr" placeholder="Nombre de usuario" />
    <input name="nombre" placeholder="Nombre" />
    <input name="apel1" placeholder="Apellido 1" />
    <input name="ap2" placeholder="Apellido 2" />
    <input name="passw" placeholder="Contraseña" />
    <input type="email" name="email" placeholder="Correo electrónico"/>
    <input name="dir" placeholder="Direccion postal"/>
    <label ><input type="checkbox" name="esProfesor" style="display: inline-block !important;"
> Profesor</label>
    <br>
    <button type="submit" class="pure-button pure-button-primary">Registrar</button>
  </fieldset>
</form>
```

## 8.2. Fichero de condiciones

**PERL.** La gestión de este fichero la haré mediante el módulo de PERL llamado *'File::Copy'*, este me permitirá hacer la copia del fichero desde una carpeta origen (*'/etc/condiciones.txt'*) hasta el directorio del usuario recientemente creado. Añado así las siguientes líneas al fichero de registro de usuarios de PERL:

```
copy("/etc/condiciones.txt", "$ruta/condiciones.txt") or die "Error al copiar las condiciones
: $!";
```

Además, deberemos hacer propietario del fichero al usuario *www-data* para su correcta gestión.

## 8.3. Monitorización

**Bash.** Para la monitorización de los procesos arrancados en el sistema necesito un sistema fiable por el que mostrar información en la web. Para conseguir esto decido hacer una inserción de html desde un script de CGI y asociar este a un botón de la interfaz principal. Este hará dos cosas:

**Primero.** Ejecutará un script muy simple que compruebe el estado de los servicios de interés:

```
root@TrueSight:~# cat /etc/Scripts/servicios.sh
#!/bin/sh

servicios=("sshd" "apache2" "postfix" "mariadb")
dir="/var/www/html/servicios/"
for i in "${servicios[@]}"
do
    activo=$(systemctl is-active $i)
    if [[ "$activo" == "active" ]]; then
        echo 1 > "$dir$i"
    else
        echo 0 > "$dir$i"
    fi
done
```

Figura 6: */etc/Scripts/servicios.sh*

**Perl.** Escribiré un script CGI que se encargue de mostrar la página web, escribiendo el código HTML de la misma dentro de la estructura *'print qq()'* de CGI. Así podré hacer cambios estéticos de forma sencilla, como cambiar los colores de los mensajes.

```
for ($i = 0; $i < @servicios; $i++){  
    my $servicio = $nombresServicios[$i];  
    if($servicios[$i] != 1){  
        print qq(  
            <div class="elemento">  
                <label class="servicio_inactivo">$servicio: Inactivo</label>  
            </div>);  
    }else{  
        print qq(  
            <div class="elemento">  
                <label class="servicio_activo">$servicio: Activo</label>  
            </div>);  
    }  
}
```

**Web.** El script de PERL generará la siguiente salida por pantalla, mostrando los servicios del sistema con un indicador textual y otro de color indicando su activación.



Figura 7: Sección resultante de página web por las inserciones anteriores



## 8.4. Inicio de sesión

**Script.** Iniciar sesión requiere una autenticación de los datos del usuario en el sistema, para ello usaré el módulo *'Authen::PAM'* de PERL. Si los datos son correctos, redirigiré a la web personal. En caso de producirse un error en el sistema o la lectura de unos datos incorrectos, se direccionará a una página de error específica.

```
sub my_conv_func {
    my @res;
    while ( @_ ) {
        my $code = shift;
        my $msg = shift;
        my $ans = '';

        $ans = $username if ($code == PAM_PROMPT_ECHO_ON());
        if ($code == PAM_PROMPT_ECHO_OFF()){
            $ans = $password
        }

        push @res, (PAM_SUCCESS(),$ans);
    }
    push @res, PAM_SUCCESS();
    return @res;
}

if (!ref($pamh = new Authen::PAM("passwd", $username, \&my_conv_func))) {
    print "Authen::PAM error de inicio\n";
    print $q->redirect('/errorInicio.html');
    exit 1;
}

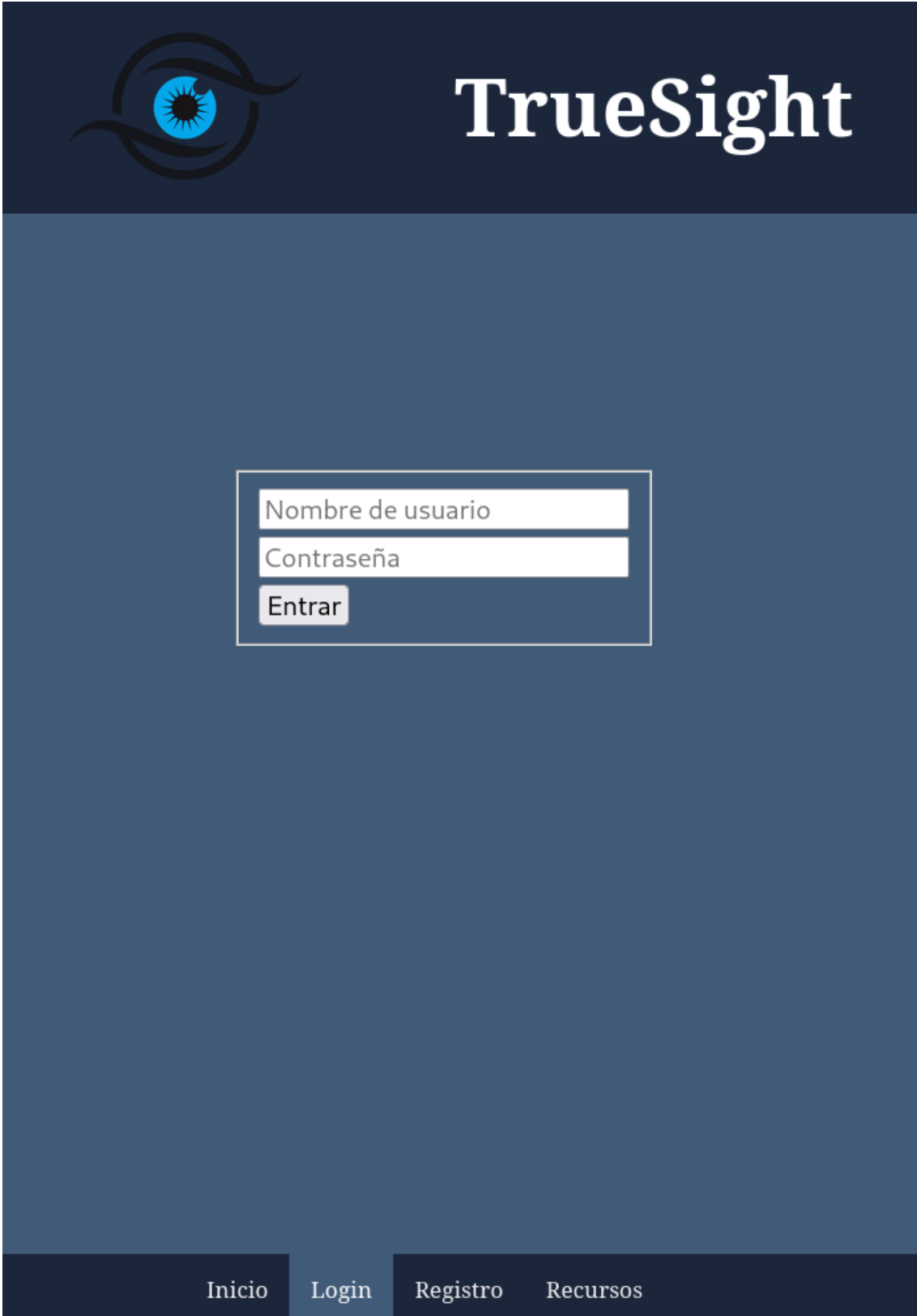
my $res = $pamh->pam_authenticate;

if($res == PAM_SUCCESS()){
    $session = CGI::Session->new();
    $session->save_param($q);
    $session->expires("+15m");
    $session->flush();
    print $q->redirect('/personal.html');
}else{
    print $q->redirect('/errorInicio.html');
}

/usr/lib/cgi-bin/inicio.cgi 53,1
```

**Web.** Para la lectura de esos datos usaré un formulario de html con el script anterior asociado a su botón de envío.





The image shows a login page for a system named 'TrueSight'. The header is dark blue with a stylized eye logo on the left and the text 'TrueSight' in white on the right. The main body is a medium blue color. In the center, there is a white rectangular box containing a login form. The form has two input fields: 'Nombre de usuario' (Username) and 'Contraseña' (Password). Below these fields is a button labeled 'Entrar' (Enter). At the bottom of the page, there is a dark blue navigation bar with four links: 'Inicio' (Home), 'Login', 'Registro' (Registration), and 'Recursos' (Resources). The 'Login' link is highlighted with a lighter blue background.

# TrueSight

Nombre de usuario

Contraseña

Entrar

[Inicio](#) [Login](#) [Registro](#) [Recursos](#)

No es necesaria ninguna acción adicional, asumiendo la instalación adecuada de los módulos de PERL.

## 8.5. Borrado de usuarios

**Script.** El borrado de usuarios lo haré mediante un script de PERL que usará elementos de los anteriores, primero deberá comprobar las credenciales del usuario y luego efectuar el borrado del sistema y de la base de datos. Una vez terminado el proceso, se redireccionará al usuario a la página principal.

```
if (!ref($pamh = new Authen::PAM("passwd", $username, \&my_conv_func))) {  
    print "Authen::PAM error de inicio\n";  
    print $q->redirect('/errorBorrado.html');  
    exit 1;  
}  
  
my $res = $pamh->pam_authenticate;  
  
if($res == PAM_SUCCESS()){  
    Linux::usermod->del($username);  
  
    my $dbh = DBI->connect('DBI:MariaDB:database=sysadmin;host=localhost','admin','labii',  
        '{ RaiseError => 1, PrintError => 0 }');  
    my $sth;  
    $sth = $dbh->prepare("delete from usuarios where usr=?") or die;  
    $sth->execute($username) or die;  
    $dbh->disconnect();  
  
    print $q->redirect('/inicio.html');  
}else{  
    print $q->redirect('/errorBorrado.html');  
}  
}
```

/usr/lib/cgi-bin/borraUsuario.cgi 27,1 Final

**Web.** Los datos se recogerán desde un formulario web, dónde se leerán nombre y contraseña para confirmar la identidad del usuario a borrar.



A web form with a dark blue background and a white border. The text 'Introduce los datos del usuario a borrar:' is written in large white font. Below the text are two white input fields: 'Nombre de usuario' and 'Contraseña'. At the bottom is a white button labeled 'Entrar'.

## 8.6. Log de accesos

**Bash.** Para gestionar los accesos escribiré un script en bash que añada una al final de un fichero de log ubicado en `/var/log/login.log` con la información horaria y el nombre de usuario.

```
root@TrueSight:~# cat /etc/Scripts/log.sh
#!/bin/sh
a=$(date)
echo "$a : $1" >> /var/log/login.log
```

**Perl.** Debemos ejecutar adecuadamente este script después del inicio de sesión del usuario, así añadimos la ejecución a nuestro script de Perl.

```
my $res = $pamh->pam_authenticate;

if($res == PAM_SUCCESS()){
    $session = CGI::Session->new();
    $session->save_param($q);
    $session->expires("+15m");
    $session->flush();
    system("/bin/bash /etc/Scripts/log.sh $username");
    print $q->redirect('/personal.html');
}else{
    print $q->redirect('/errorInicio.html');
}
```

## 9. Blog privado

**Userdir.** La solución que propongo es usar el módulo de userdir de Apache. Para ello activo el módulo adecuado:

```
root@TrueSight:~# a2enmod userdir
Enabling module userdir.
To activate the new configuration, you need to run:
systemctl restart apache2
root@TrueSight:~# systemctl restart apache2
```

Así cualquier usuario podrá incluir su blog personal en una carpeta llamada `public_html` en su directorio personal. Para asistir este proceso crearé la carpeta automáticamente desde el script de registro de perl.

```
make_path("$directorio/public_html");
print $q->redirect('/exitoInicio.html');
```

Figura 8: `/usr/lib/cgi-bin/registro.cgi`

## 10. Bibliografía

- Creación de certificados SSL.
- Instalación de PERL.
- Instalación de Postfix.
- Añadir entradas a la base de datos desde PERL.
- Módulo DBI de PERL.
- Gestión de permisos en SQL.
- Lectura de archivos desde PERL.
- Llamar a un script de bash desde PERL.
- Instalación de Roundcube.
- Instalación de RoundCube.
- Autenticación PAM.
- Autenticación PAM, FAQ.
- CGI-Session.
- Instalación de Dovecot.
- Instalación de las dependencias PHP.
- Instalación de RoundCube.
- Configuración del módulo userdir de Apache.