Anexo 1.
Instalación y configuración de aplicaciones y servicios.

Autor: **Antonio José Fortis Malagón**

Tutor: **Antonio Mancheño Bautista**

Consultor UOC: **Jordi Massager Pla**

Índice

1. Introducción

En este anexo, vamos a detallar el proceso de instalación y configuración de un sistema servidor GNU/Linux, en concreto, se instalará la distribución Debian 6, que ha sido la elegida para instalar y configurar sobre ella los diferentes servicios y aplicaciones que se citan en el proyecto.

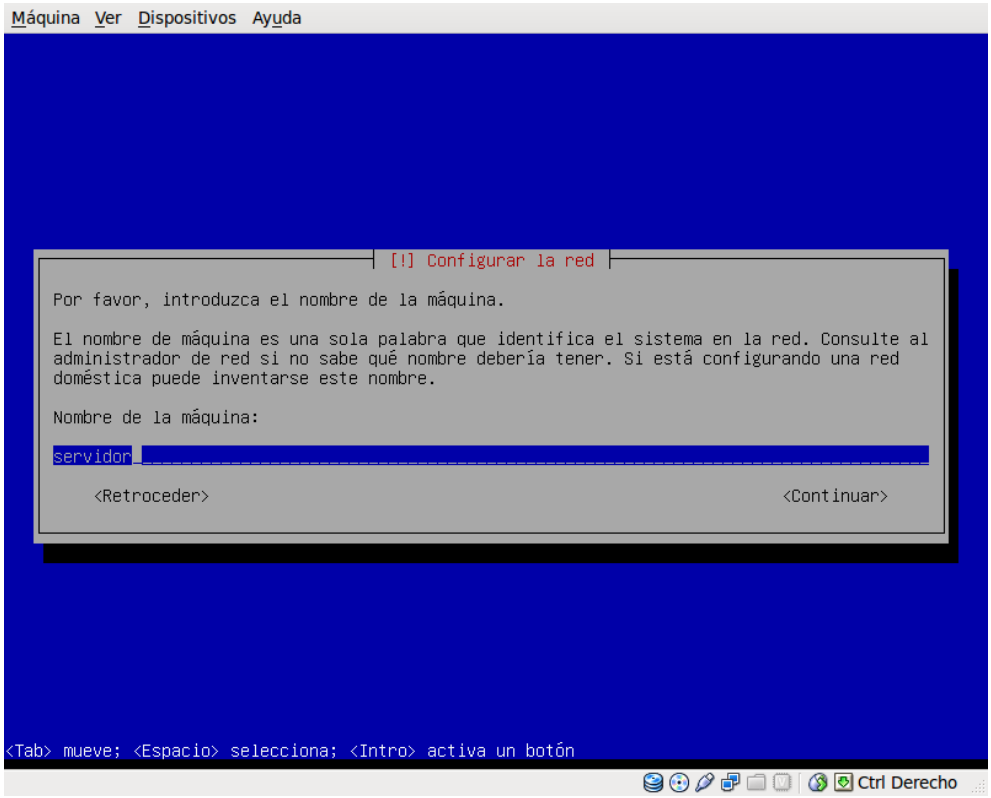
Dicha documentación, servirá de base para que el administrador del sistema pueda acceder a ella y consultar todo el procedimiento seguido para la puesta en funcionamiento del servidor. En ella se va a detallar el proceso de instalación del sistema operativo y de cada una de las aplicaciones que ofrecen los diferentes servicios, indicando para cada una de ellas, la configuración que se ha realizado.

Para la instalación del sistema operativo, se ha utilizado **VirtualBox** como máquina virtual, ya que se trata de un software de código abierto que puede ser usado libremente.

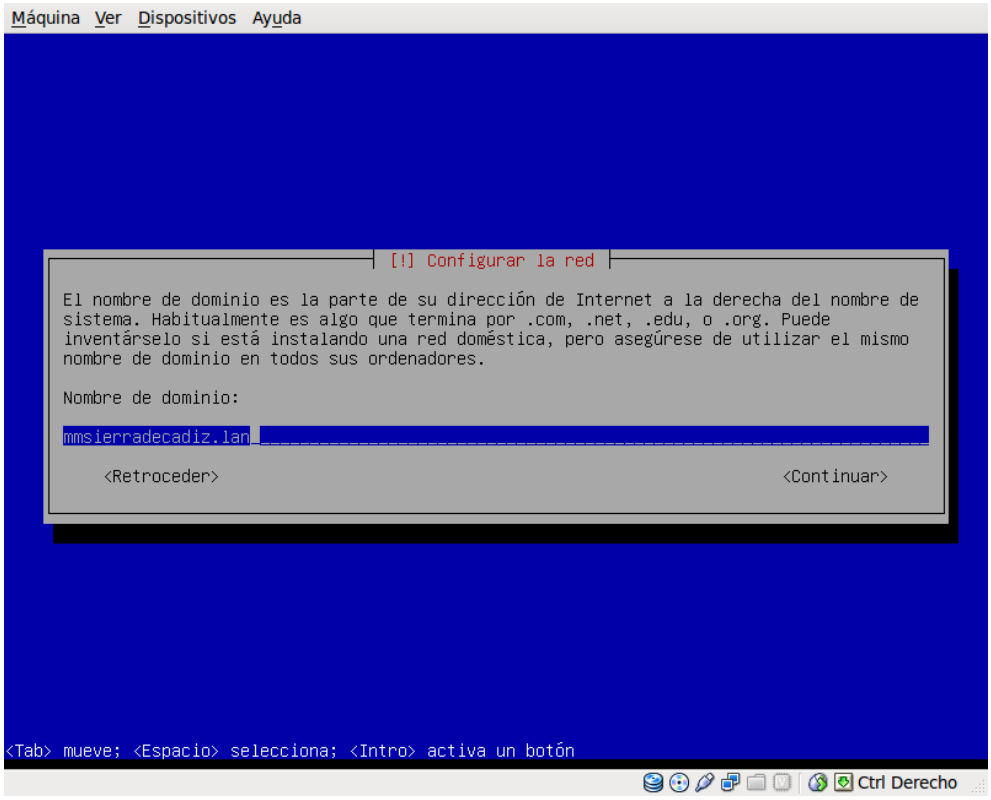
2. Instalación del sistema operativo

Lo primero que vamos a hacer, es introducir un DVD que contenga una imagen de Debian 6, y una vez reconocido e iniciado el proceso de instalación, tendremos que indicarle el Lenguaje que deseamos, la Zona de Ubicación y la Distribución del teclado.

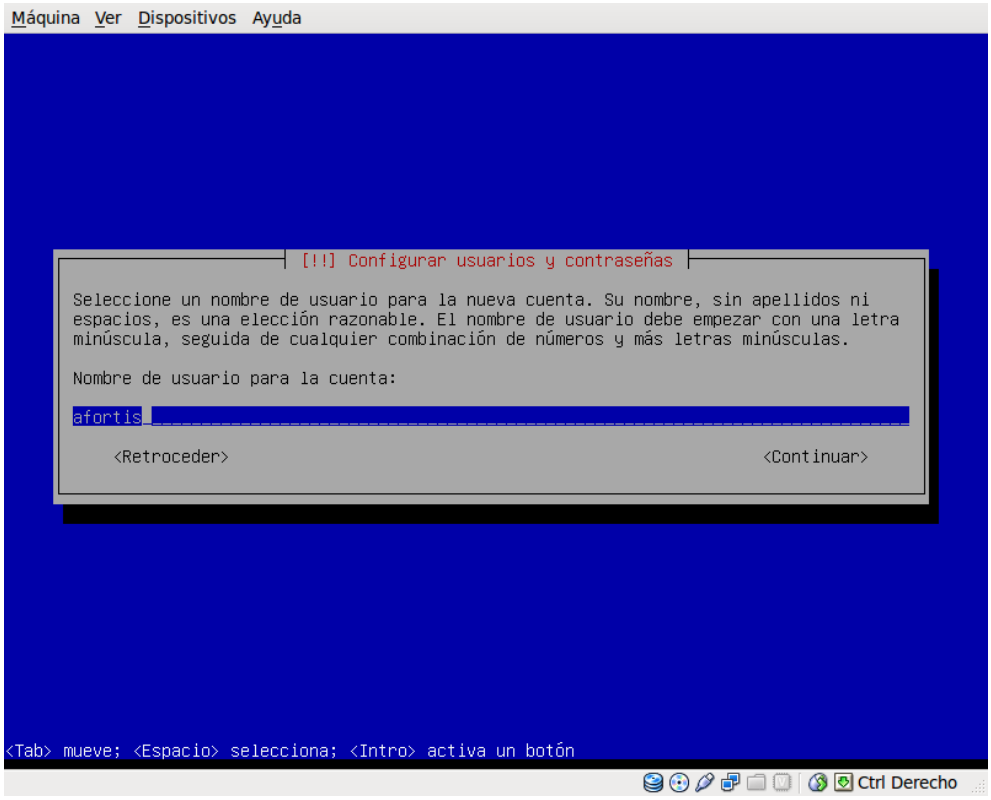
El siguiente paso, será asignar un nombre a la máquina, en cuyo caso se ha elegido el nombre **servidor**, tal y como podemos ver en la siguiente captura:



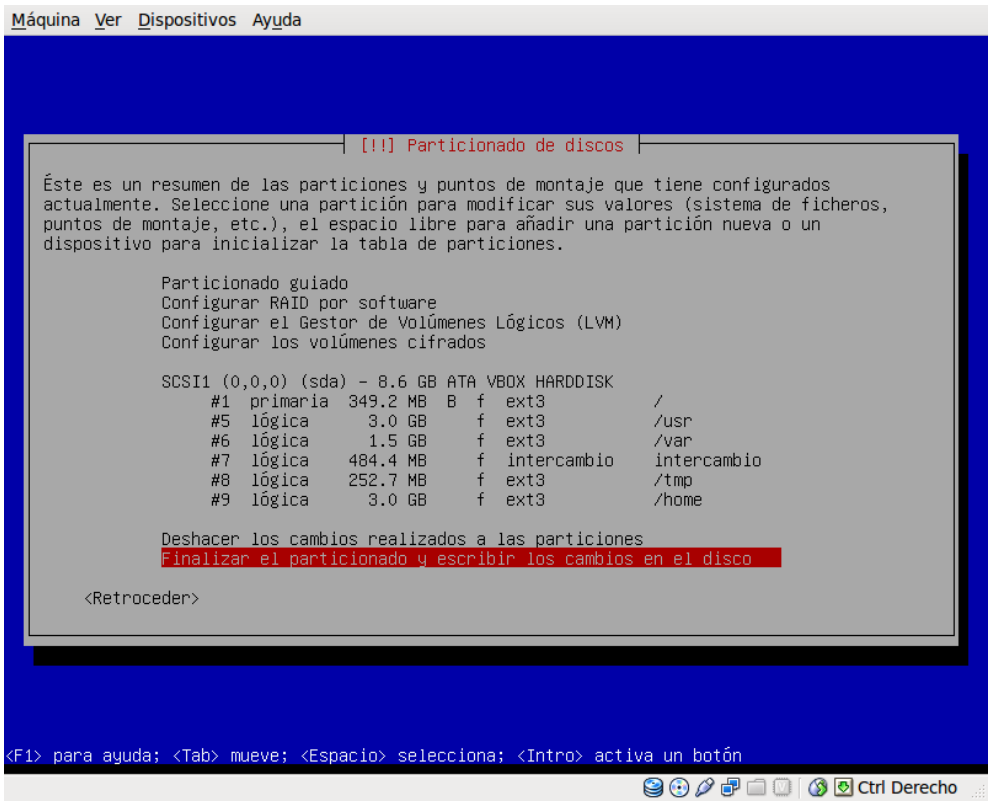
Además, habrá que elegir un nombre de dominio, siendo el elegido **mmsierradecadiz.lan**



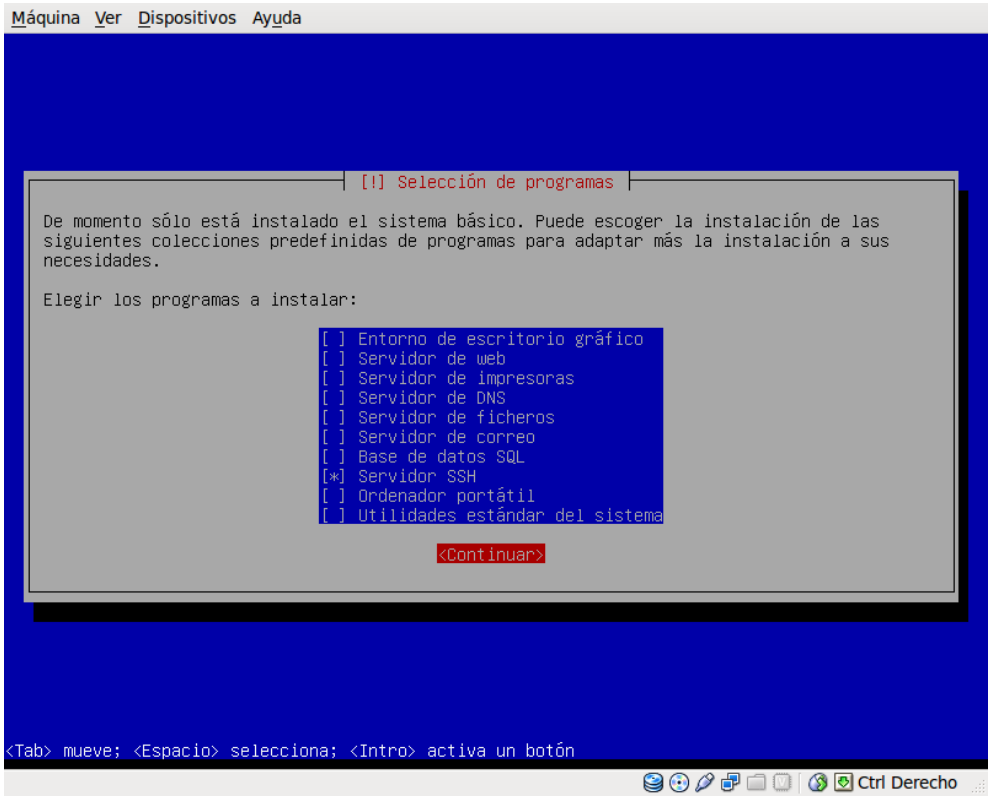
A continuación, debemos de indicar la contraseña de **root**, además de generar un usuario con el cual trabajaremos. En este caso, el usuario elegido será **afortis**, tal y como podemos observar:



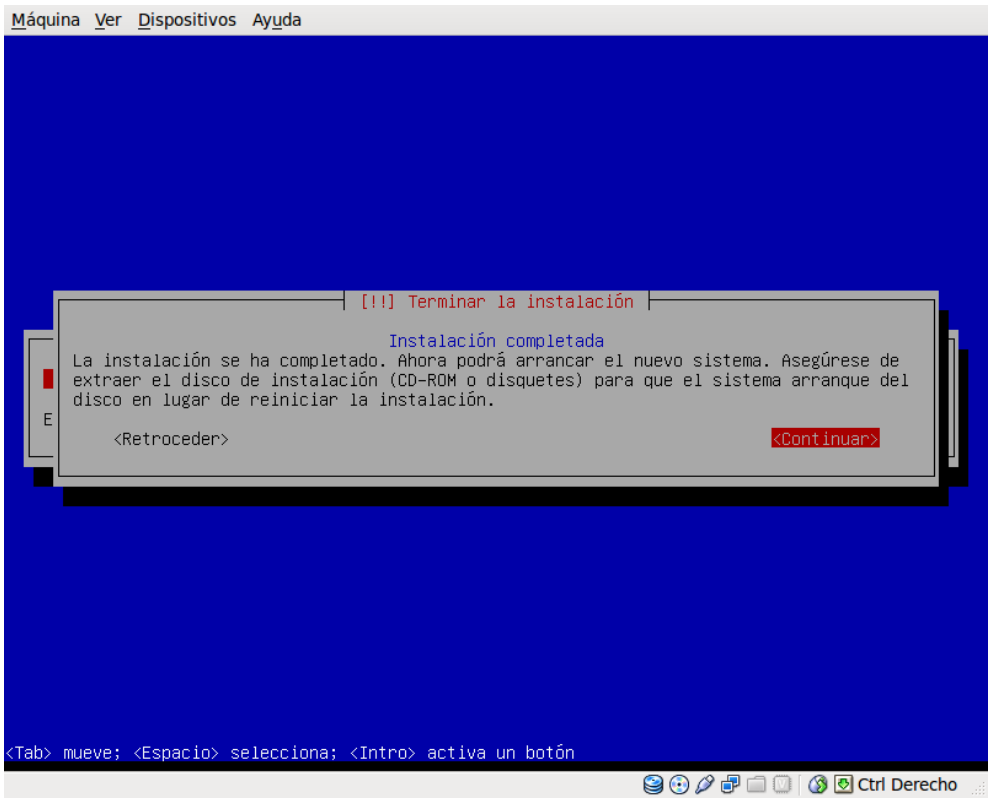
Seguidamente, deberemos de configurar la Zona horaria y seleccionar el Particionado de los discos, en cuyo caso se van a utilizar varias particiones tal y como podemos observar:



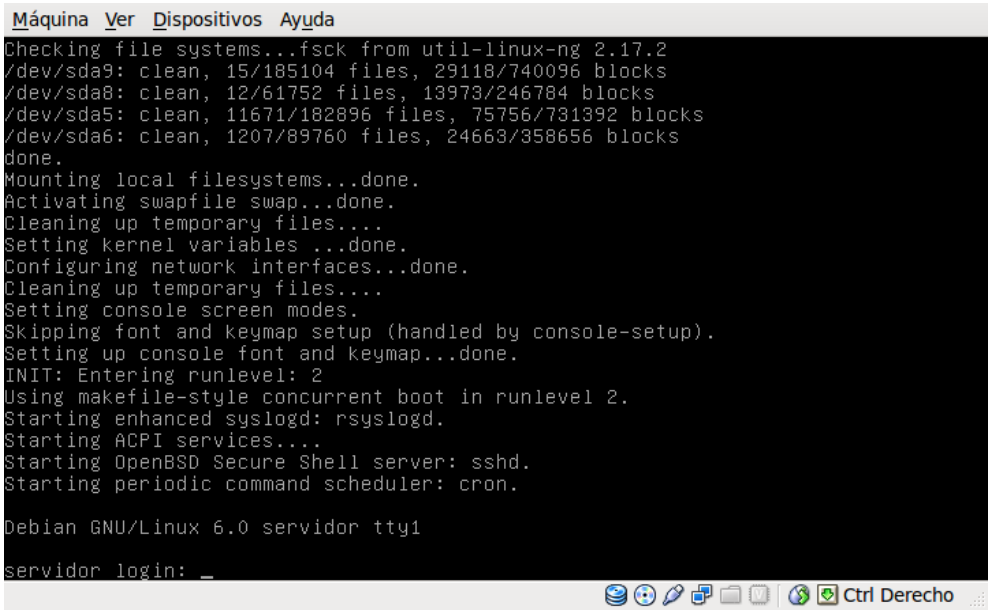
Una vez que se ha instalado el Sistema Base, debemos de seleccionar los programas que se van a instalar, en cuyo caso, no vamos a marcar ninguno, únicamente el servicio SSH, que nos permitirá conectarnos de forma segura y remota al servidor para administrarlo.



Finalizada la instalación de programas e instalado el Cargador de Arranque GRUB, tendremos el proceso de instalación finalizado.



Lo único que nos queda pendiente para comprobar la correcta instalación, será iniciar el sistema y ver que se carga correctamente.



Lo siguiente que vamos a realizar sobre el sistema es actualizarlo a través de Internet, para ello, haremos uso del gestor de paquetes **aptitude**, utilizando para ello los comandos:

- **aptitude update**: actualiza la lista de paquetes existentes en los repositorios.

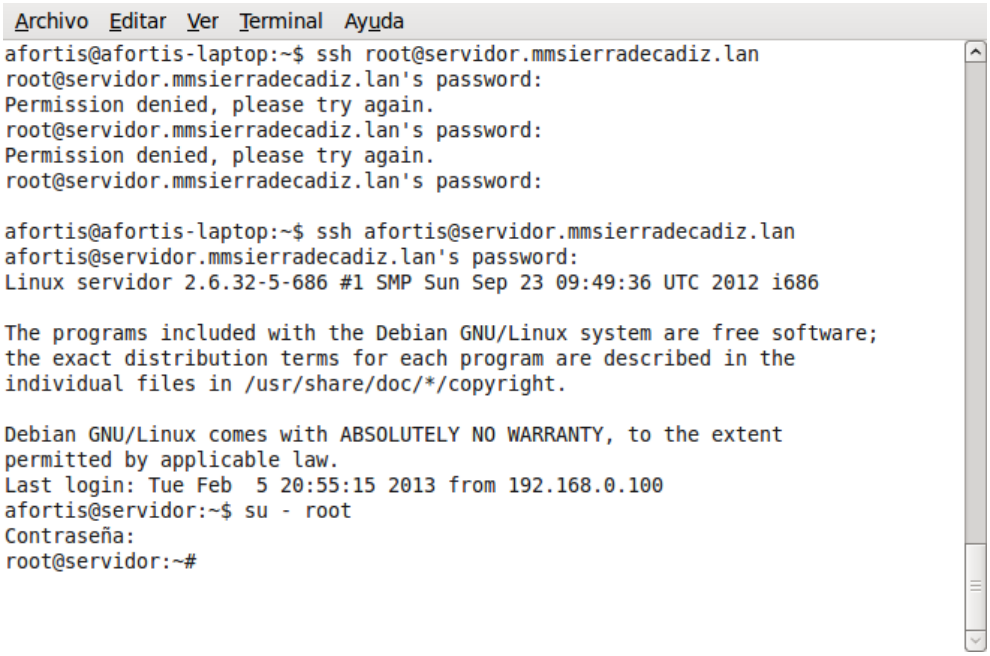
- **aptitude safe-upgrade**: una vez actualizada la lista de paquetes, permite instalar todas las actualizaciones disponibles, satisfaciendo todas las dependencias necesarias.

Una vez actualizado el sistema, para poder administrar el sistema de forma remota, se ha instalado el servidor SSH, el cual, es un protocolo de comunicaciones que encripta los datos intercambiados, haciendo imposible la violación de privacidad de la comunicación.

Para añadir un nivel de seguridad más a este servicio, vamos a realizar una modificación en el fichero de configuración del servicio SSH, el cual se encuentra en */etc/ssh/sshd_config* y nos va a permitir desactivar el login como root. Esta modificación, hay que realizarla en la siguiente sección:

```
#[...]

# Authentication:
LoginGraceTime 120
PermitRootLogin no
StrictModes yes

#[...]
```

De esta forma, para poder conectarnos al sistema, habrá que hacer un login con un usuario normal (en este caso **afortis**), y una vez que nos hemos identificado y accedido, adquirimos los privilegios de **root** según se puede observar en la siguiente imagen:



3. Instalación y configuración del Servicio HTTP

En este apartado vamos a detallar el proceso de instalación de un Servidor HTTP, junto con una serie de complementos que le añadirán una mayor funcionalidad. El servidor elegido es **Apache2** y los complementos que se le van a añadir son **SSL, PHP5** y **MySQL**. Además, para establecer una conexión segura y de confianza vamos a generar un certificado que respalde la identidad del servidor.

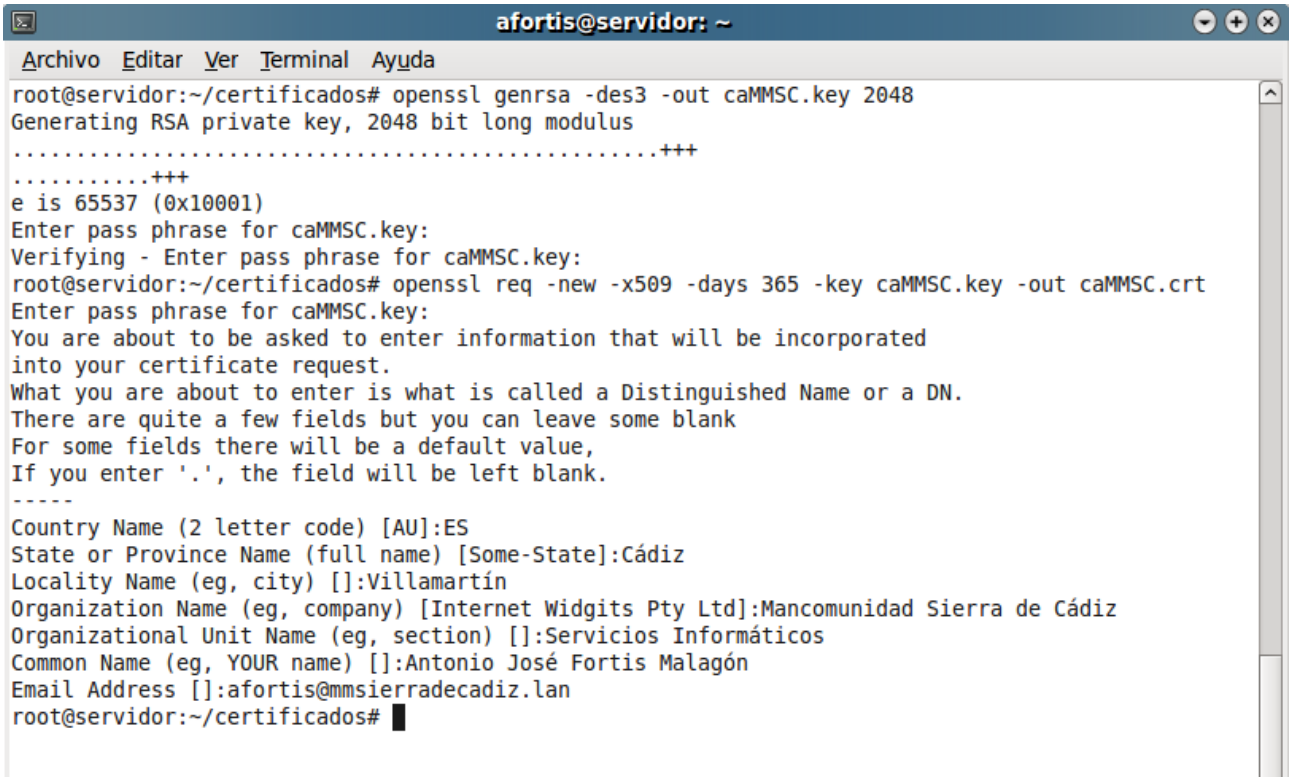
3.1. Certificado SSL auto-firmado

Aunque los certificados son generalmente emitidos por entidades certificadoras (Certificate Authority) independientes y de confianza reconocida, para nuestro caso vamos a generar un certificado auto-firmado para una CA (Certificate Authority) creada por nosotros.

En primer lugar, vamos a generar el certificado de la CA (Certificate Authority) cifrado con una passphrase de 2048 bits y válido durante 365 días utilizando los siguientes comandos:

openssl genrsa -des3 -out caMMSC.key 2048
openssl req -new -x509 -days 365 -key caMMSC.key -out caMMSC.crt

dichos comandos, nos van a pedir una cierta información sobre la CA que paso a mostrar en la siguiente captura:



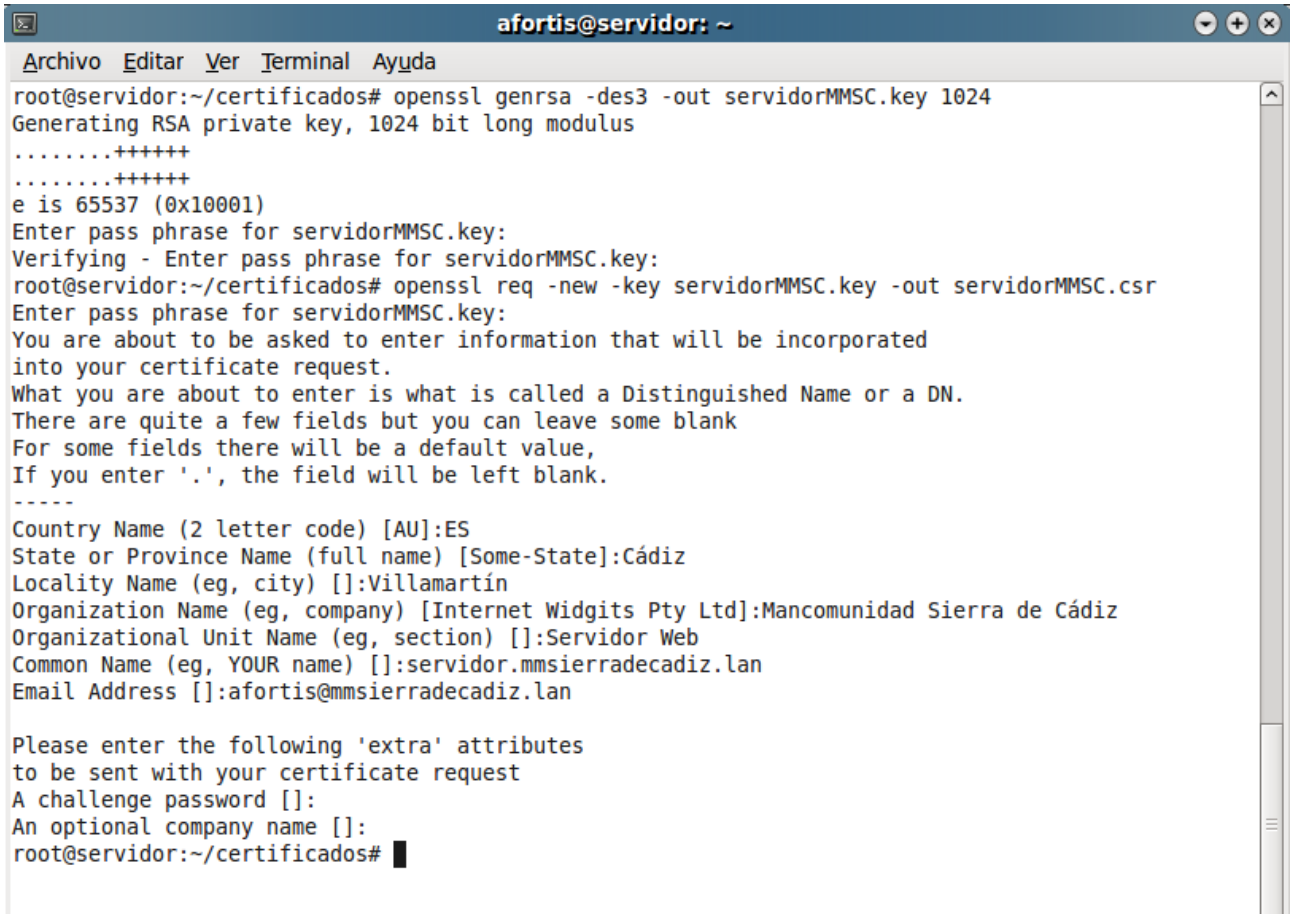
donde vamos a tener como resultado, el fichero **caMMSC.key** que corresponderá a la llave que usa la CA para firmar los certificados y el fichero **caMMSC.crt** que será el propio certificado de la CA.

A continuación, vamos a generar el certificado del servidor, utilizando para ello una clave de 1024 bits sin cifrar y que será firmado por la CA generada en el apartado anterior. Los comandos utilizados son los siguientes:

openssl genrsa -des3 -out servidorMMSC.key 1024

openssl req -new -key servidorMMSC.key -out servidorMMSC.csr

con la ejecución de dichos comandos, nos solicitará información sobre el servidor, que será la que muestro a continuación:



para posteriormente, firmar el requerimiento anterior (servidorMMSC.csr) con la CA que nos hemos creado. Para ello ejecuto el siguiente comando:

openssl x509 -req -days 365 -in servidorMMSC.csr -CA caMMSC.crt -CAkey caMMSC.key -set_serial 01 -out servidorMMSC.crt

que generará el certificado del servidor firmado por nuestra CA (servidorMMSC.crt) y la clave privada para el servidor (servidorMMSC.key).

Como la clave privada del servidor está encriptada y protegida por una contraseña, ésta debe de escribirse cada vez que se accede al servicio. Si queremos generar la clave de forma que no obliguemos al servidor a pedir la contraseña, debemos de crear una **clave insegura** que se usará para iniciar el servicio y no requerir la clave cada vez.

Los comandos utilizados para generar esta clave a partir de la clave que ya teníamos creada, y crear una copia de seguridad de la clave segura, son los siguientes:

openssl rsa -in servidorMMSC.key -out servidorMMSC.key.insecure

mv servidorMMSC.key servidorMMSC.key.secure

mv servidorMMSC.key.insecure servidorMMSC.key

En este caso, al no pedir la clave al iniciar el servicio, debemos de tener en cuenta que dicha clave será almacenada, por lo tanto, si alguna persona tiene acceso al fichero, podría desencriptar toda la transmisión. Para solucionar este problema, se aconseja hacer que el fichero que contiene la clave (servidorMMSC.key) tenga permiso solamente de lectura por el root, lo cual, lo conseguimos con el comando **chmod 400 servidorMMSC.key**, asegurándonos que es el root el propietario de dicho fichero.

La siguiente captura, muestra las últimas operaciones realizadas para generar el certificado del servidor firmado por nuestra CA, y eliminar la clave de forma que no sea requerida cada vez que se conecte con dicho servidor:

```
Archivo  Editar  Ver  Terminal  Ayuda
root@servidor:~/certificados# openssl x509 -req -days 365 -in servidorMMSC.csr -CA caMMSC.crt -CAk
ey caMMSC.key -set_serial 01 -out servidorMMSC.crt
Signature ok
subject=/C=ES/ST=C\xC3\xA1diz/L=Villamart\xC3\xADn/O=Mancomunidad Sierra de C\xC3\xA1diz/OU=Servid
or Web/CN=servidor.mmsierradecadiz.lan/emailAddress=afortis@mmsierradecadiz.lan
Getting CA Private Key
Enter pass phrase for caMMSC.key:
root@servidor:~/certificados# openssl rsa -in servidorMMSC.key -out servidorMMSC.key.insecure
Enter pass phrase for servidorMMSC.key:
unable to load Private Key
946:error:06065064:digital envelope routines:EVP_DecryptFinal_ex:bad decrypt:evp_enc.c:330:
946:error:0906A065:PEM routines:PEM_do_header:bad decrypt:pem_lib.c:428:
root@servidor:~/certificados# openssl rsa -in servidorMMSC.key -out servidorMMSC.key.insecure
Enter pass phrase for servidorMMSC.key:
writing RSA key
root@servidor:~/certificados# mv servidorMMSC.key servidorMMSC.key.secure
root@servidor:~/certificados# mv servidorMMSC.key.insecure servidorMMSC.key
root@servidor:~/certificados# chmod 400 servidorMMSC.key
root@servidor:~/certificados# ls -al
total 32
drwxr-xr-x 2 root root 4096 feb  6 20:18 .
drwx------ 4 root root 4096 feb  6 19:24 ..
-rw-r--r-- 1 root root 1944 feb  6 19:27 caMMSC.crt
-rw-r--r-- 1 root root 1751 feb  6 19:25 caMMSC.key
-rw-r--r-- 1 root root 1294 feb  6 19:56 servidorMMSC.crt
-rw-r--r-- 1 root root  781 feb  6 19:44 servidorMMSC.csr
-r-------- 1 root root  891 feb  6 20:17 servidorMMSC.key
-rw-r--r-- 1 root root  963 feb  6 19:42 servidorMMSC.key.secure
root@servidor:~/certificados#
```

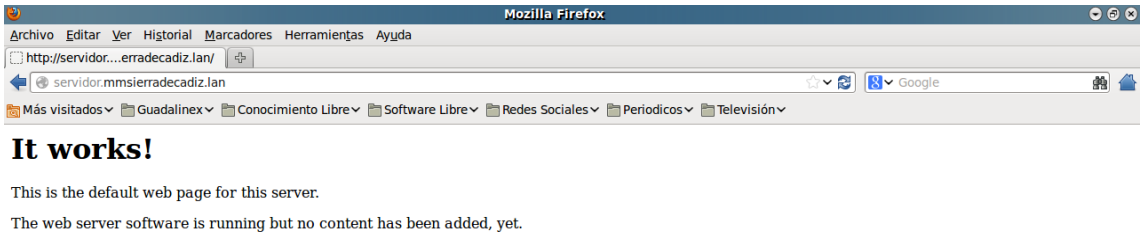
A continuación, veremos como arrancar el servidor Apache2 de forma que se pueda acceder tanto con el protocolo HTTP, como con HTTPS, mediante el certificado de servidor que hemos creado y que ha sido firmado por nuestra propia CA (Certificate Authority).

3.2. Apache2

Para instalar **Apache2** vamos a hacer uso de los repositorios, ejecutando como root el siguiente comando:

aptitude install apache2

Una vez que ha sido instalado, para comprobar su funcionamiento, desde el navegador web de una máquina cliente, introducimos el nombre **http://servidor.mmsierradecadiz.lan** y veremos que el servidor web funciona, tal y como se muestra en la siguiente captura.



3.3. Apache2 + SSL

Si deseamos que en nuestro servidor web se puedan establecer conexiones seguras y encriptadas, debemos de activar el soporte SSL. De esta forma, será posible enviar y recibir información importante con la certeza de que ésta no podrá ser interceptada por terceras personas.

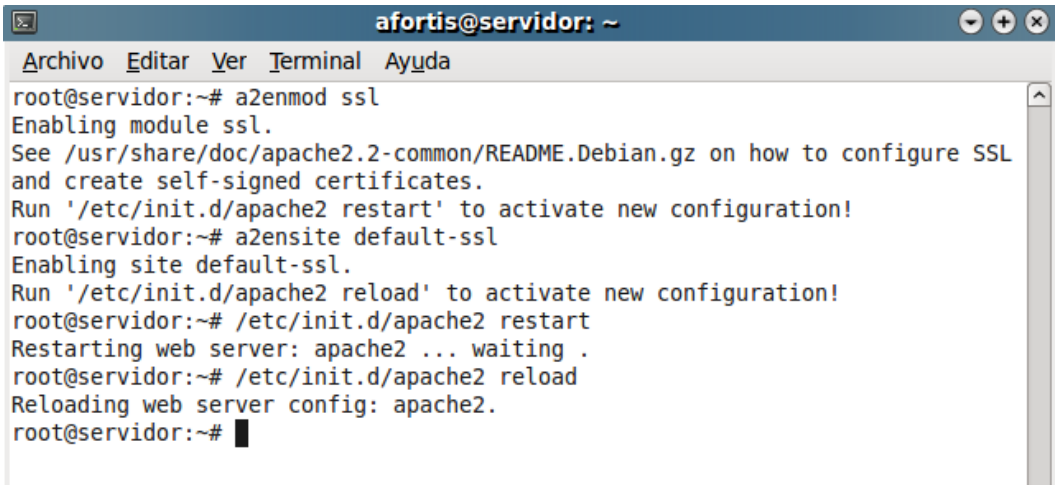
Durante la instalación de **Apache2** se crea una configuración para acceso seguro (HTTPS) de forma automática. Esta configuración deberá de ser modificada para incluir los certificados auto-firmados generados previamente.

El fichero donde se almacena la configuración y sobre el que vamos a realizar las modificaciones es */etc/apache2/sites-available/default-ssl* , quedando como se muestra a continuación:

```
# [...]

        #   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/apache2.2-common/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/ssl/certs/servidorMMSC.crt
        SSLCertificateKeyFile  /etc/ssl/private/servidorMMSC.key

# [...]
```

Aunque Apache2 está preparado para servir contenido HTTPS, aún hay que habilitar el módulo SSL y activar el sitio creado. Para ello ejecutaremos los comandos **a2enmod ssl** y **a2ensite default-ssl** respectivamente. Además, para que todos estos cambios surtan efecto, debemos de reiniciar el servicio, usando para ello el comando **/etc/init.d/apache2 restart**.

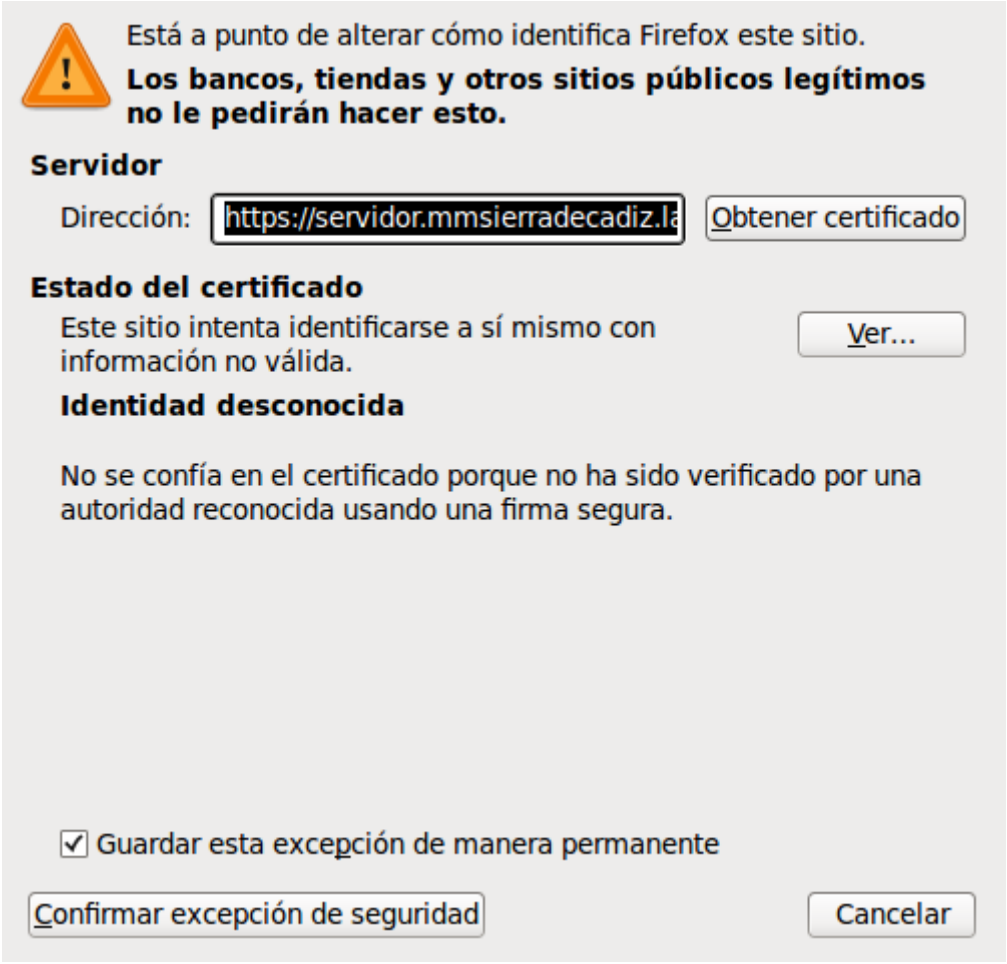
En la siguiente captura podemos ver la ejecución de dichos comandos:



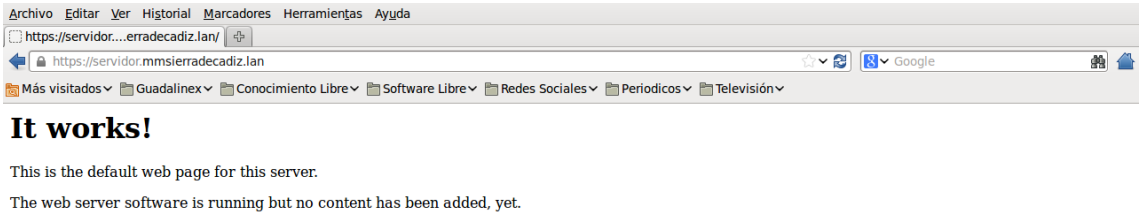
Si queremos verificar el correcto funcionamiento del servicio web seguro (HTTPS), deberemos de insertar en un navegador web la dirección https://servidor.mmsierradecadiz.lan y a continuación nos aparecerá una advertencia en la que se indica que la conexión es no confiable, debido a que dicho certificado (el certificado del servidor) ha sido firmado por una CA (la creada por nosotros) que no está reconocida, por lo tanto deberemos de confiar en ella para seguir adelante.

En la siguiente captura, se muestra la conexión realizada:

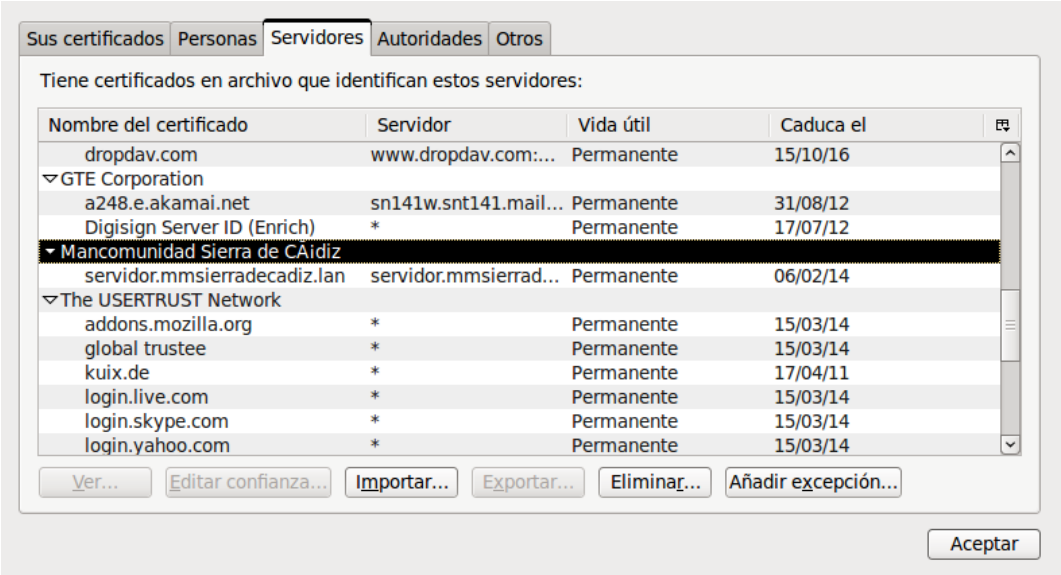
Una vez que aceptamos el certificado, tal y como podemos ver en la siguiente imagen



nos podemos conectar al servidor web a través de una conexión segura (HTTPS), lo cual vamos a ver a continuación:



Una vez establecida la conexión con el servidor desde el cliente, podemos observar en el navegador del cliente que el certificado del servidor ha sido añadido, lo cual quedaría reflejado así



e incluso, podemos llegar a consultarlo para que muestre la información que contiene.

3.4. *Apache2 + Php5*

Para expandir la funcionalidad del servidor web que estamos montando, necesito activar el soporte para php. Dicha activación requiero la instalación de una serie de paquetes y el reinicio del servidor, ejecutando para ello los siguientes comandos:

aptitude install php5 libapache2-mod-php5 para la instalación

/etc/init.d/apache2 restart para el reinicio del servidor apache2

Para verificar la instalación del soporte **php**, vamos a crear una pequeña página web que muestre las características de la instalación **php**. Esta página se creará en */var/www/phpinfo.php* y su contenido será:

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

Desde el navegador del cliente, introduciremos http://servidor.mmsierradecadiz.lan/php.info y comprobaremos que se ha instalado correctamente, tal y como vemos en la siguiente captura.

System	Linux servidor 2.6.32-5-686 #1 SMP Sun Sep 23 09:49:36 UTC 2012 i686
Build Date	Aug 6 2012 20:07:14
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php5/apache2
Loaded Configuration File	/etc/php5/apache2/php.ini
Scan this dir for additional .ini files	/etc/php5/apache2/conf.d
Additional .ini files parsed	/etc/php5/apache2/conf.d/mysql.ini, /etc/php5/apache2/conf.d/mysqli.ini, /etc/php5/apache2/conf.d/pdo.ini, /etc/php5/apache2/conf.d/pdo_mysql.ini, /etc/php5/apache2/conf.d/suhosin.ini
PHP API	20090626
PHP Extension	20090626
Zend Extension	220090626
Zend Extension Build	API220090626,NTS
PHP Extension Build	API20090626,NTS
Debug Build	no
Thread Safety	disabled
Zend Memory Manager	enabled
Zend Multibyte Support	disabled

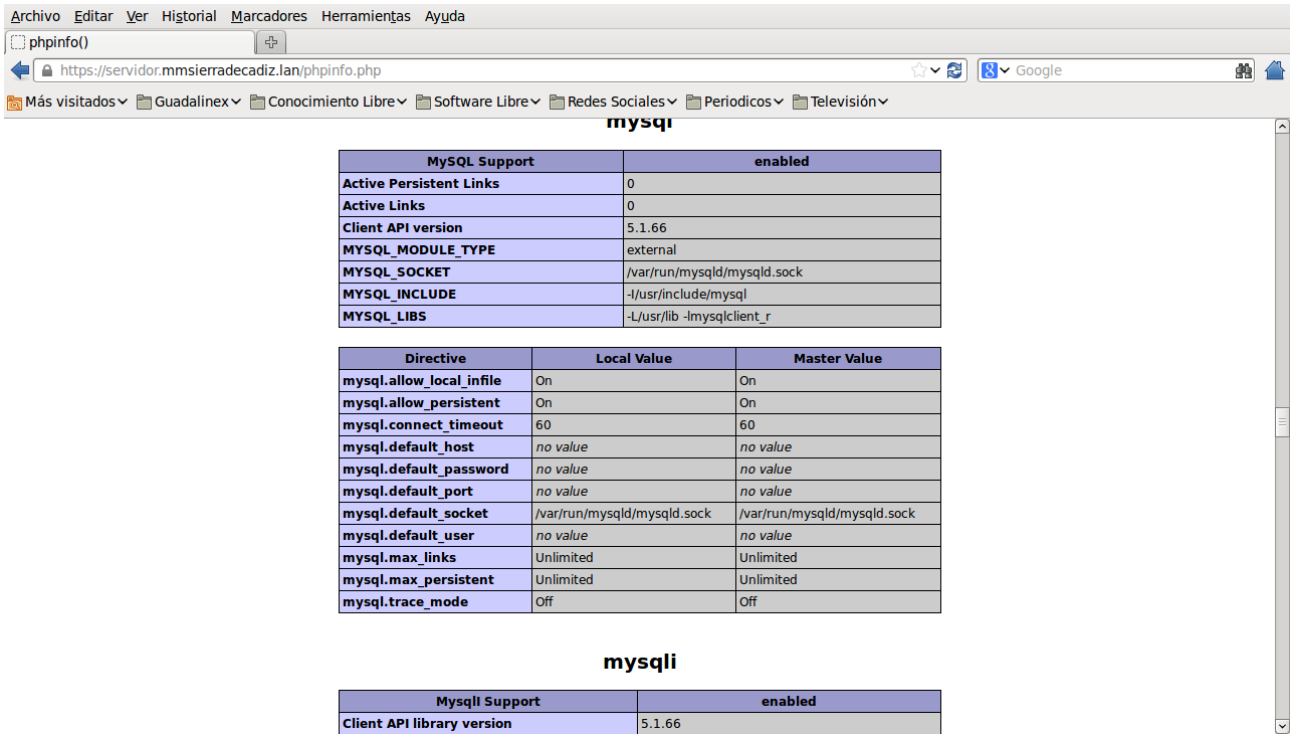
3.5. *Apache2 + Php5 + MySQL*

Para completar la funcionalidad del servidor web, vamos a activar el soporte **MySQL** del php, instalando para ello los paquetes necesarios y reiniciando de nuevo el servidor.

aptitude install php5-mysql para la instalación

/etc/init.d/apache2 restart para el reinicio del servidor apache2

Para verificar la instalación del soporte **MySQL** del php, procederemos de la misma manera que en el caso anterior, es decir, cargaremos el mismo fichero (*phpinfo.php*) en el navegador web de un cliente, introduciendo la dirección https://servidor.mmsierradecadiz.lan/phpinfo.php (en este caso se ha utilizado la conexión segura) y veremos que la instalación ha sido correcta.



3.6. *Servicio FTP*

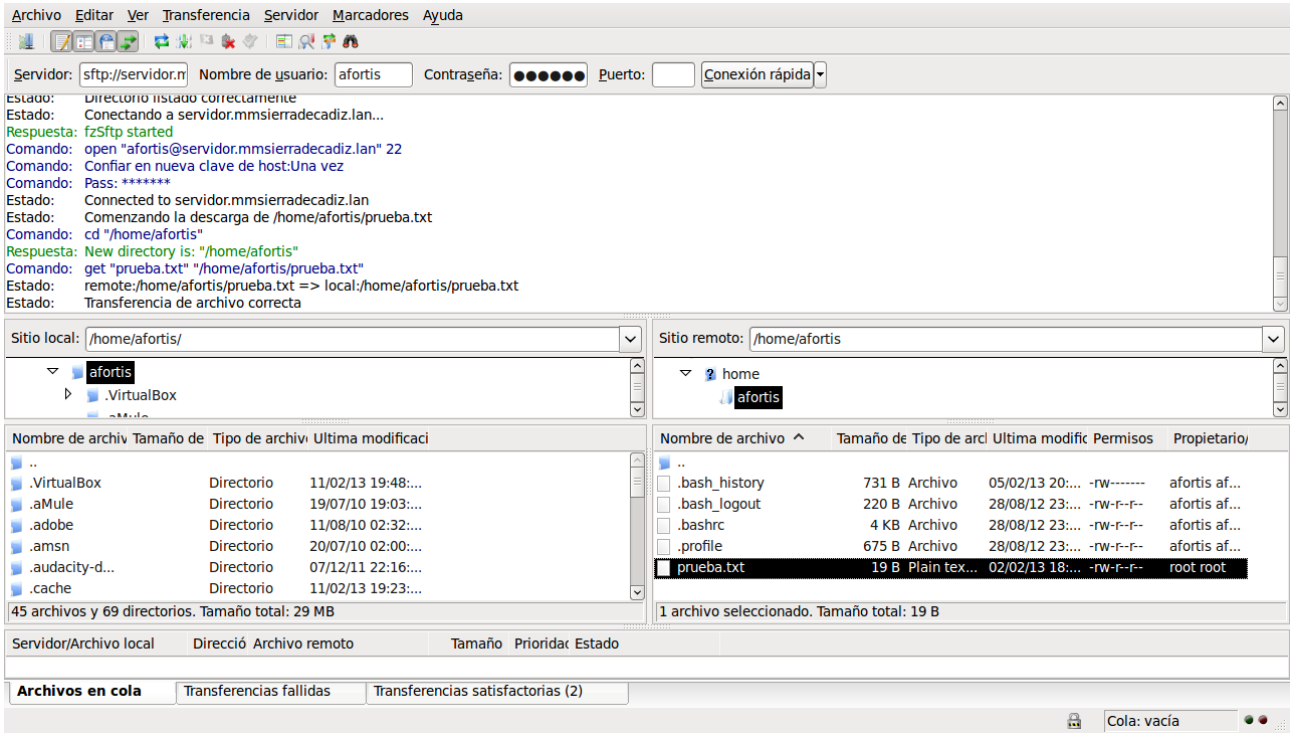
Como servicio de transferencia de ficheros vamos a utilizar la aplicación **SFTP** (también conocida como **SSH File Transfer Protocol**).

Aunque se trata de un protocolo del nivel de aplicación que proporciona la funcionalidad necesaria para la transferencia y manipulación de ficheros sobre un flujo de datos, es completamente diferente al protocolo FTP (Files Transfer Protocol). SFTP fué construido desde cero y añade la características FTP a SSH, ya que se utiliza conjuntamente con él para proporcionar la seguridad a los datos. Por lo tanto, la seguridad no la provee directamente el protocolo SFTP, sino SSH.

Algunas de las características que podemos destacar de dicho protocolo son:

- SFTP es más seguro que FTP .
- Si dispones de SSH en tu sistema por defecto ya puedes utilizar SFTP, no se necesita configuración adicional de servicios ni otro puerto .
- La transmisión de datos siempre siempre segura encapsulando la conexión mediante SSH .
- El protocolo incluye operaciones para permisos, manipulación de atributos, bloqueo de ficheros y mucho más.
- Se puede utilizar tanto a nivel de línea de comandos como mediante clientes gráficos .
- La mayor parte de los clientes de FTP modernos soportan el protocolo SFTP.

Para la instalación y configuración del servicio, indicar que una vez que tenemos el paquete **openssh-server** (el cual fue añadido durante el proceso de instalación del sistema operativo), dicho servicio ya se encuentra operativo, por lo tanto, lo único que vamos a realizar a continuación, es mostrar una captura donde podemos observar su funcionamiento una vez que ha sido configurado en un cliente FTP, como es el caso de **FileZilla**.



3.7. *Servicio WebDAV*

El protocolo **WebDAV** (Web-based Distributed Authoring and Versioning) es una extensión del protocolo HTTP que soporta escritura y permite modificar archivos en un servidor web. Esto hace posible no sólo la creación y actualización de contenidos de sitios web de una forma fácil, sino el uso de otras aplicaciones, como la creación de calendarios compartidos, la centralización de los marcadores de Firefox o incluso el acceso a los documentos alojados en el servidor desde un explorador de archivos, con la función de “carpetas compartidas”.

Para su instalación debemos de indicar que ya se realizó al instalar el servidor Apache2, ya que con él fueron añadidos los módulos necesarios. Seguidamente, para el proceso de configuración, vamos a realizar los siguientes pasos:

1. Crear el directorio donde se almacenará el contenido, estando dicho directorio en el sistema de ficheros del servidor, y utilizando para ello el comando **mkdir -p /var/www/webdav**.

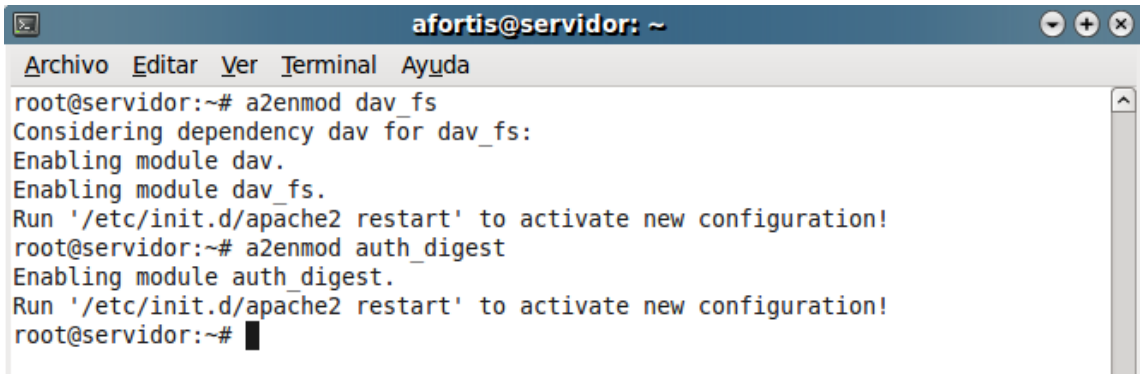
2. Una vez creado, permitimos que apache sea el propietario de dicho directorio y le asignamos permisos de escritura para el grupo, ejecutando para ello los siguientes comandos **chown www-data /var/www/webdav** y **chmod g+w /var/www/webdav** respectivamente.

3. El siguiente paso será añadir la localización de **WebDAV** a la configuración de **Apache2**, además de limitar el acceso a **WebDAV** sólo a los usuarios autorizados. Para ello, vamos a definir en el servidor web una localización protegida (webdav), creando o editando el fichero */etc/apache2/conf.d/webdav* con la siguiente información:

Alias /webdav /var/www/webdav

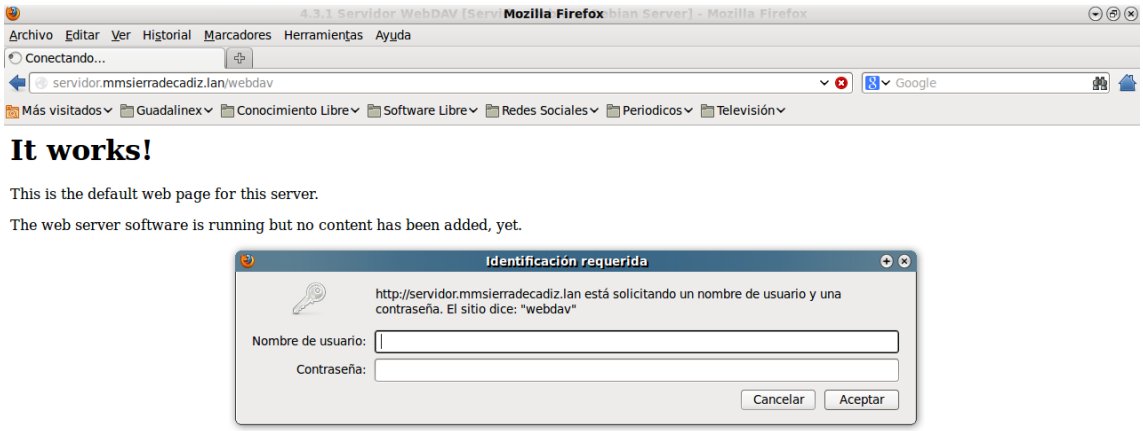
```
<Location /webdav>
  DAV On
  AuthType Digest
  AuthName "webdav"
  AuthUserFile /etc/apache2/webdav.passwd
  Require valid-user
</Location>
```

4. Posteriormente, deberemos de activar el módulo **dav_fs** y el de autenticación **auth_digest**, ejecutando para ello los comandos que se muestran a continuación:



5. Además, debe de crearse el fichero de autorizaciones para el recurso **webdav** con los usuarios que se han definido y sus respectivas contraseñas. Para la generación de la contraseña, utilizamos el comando **htdigest -c /etc/apache2/webdav.passwd webdav afortis** , siendo **webdav** el directorio donde se almacena la información de WebDav y **afortis** el nombre del usuario creado. Una vez ejecutado dicho comando, se nos pedirá una contraseña por dos veces que se que guardará en el fichero webdav.passwd de forma encriptada.

6. Por último, debemos de recargar el fichero de configuración de Apache con el comando **/etc/init.d/apache2 reload** para que lea la nueva configuración, y ya nos podemos conectar desde un cliente a http://servidor.mmsierradecadiz.lan/webdav previa autenticación, tal y como podemos ver en la imagen.



Y una vez autenticados, accedemos al contenido de dicho directorio, en este caso un fichero de prueba.

4. Instalación y configuración del Servicio de Correo

El servicio de correo electrónico que hemos instalado, se compone de un **Servicio SMTP** (Postfix), se trata de un protocolo de red utilizado para el intercambio de mensajes de correo electrónico entre computadoras u otros dispositivos. Un **Servicio IMAP** (Dovecot), es un protocolo de aplicación de acceso a mensajes electrónicos almacenados en un servidor. Por último, un **Servicio WebMail** (Roundcube) que nos permita consultar nuestros correos electrónicos de manera fácil, práctica y accesible desde cualquier lugar.

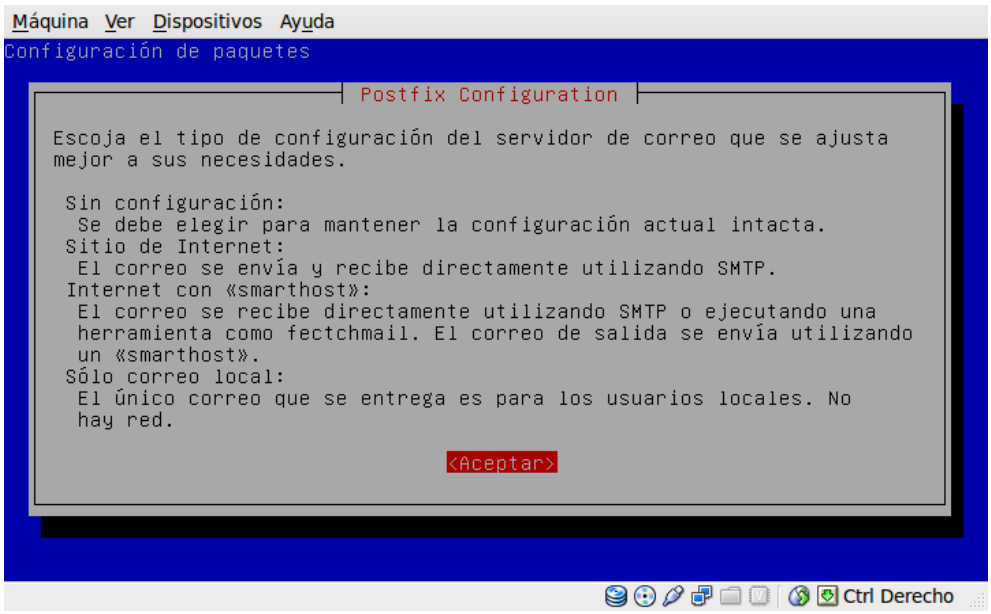
4.1. *Servicio SMTP (Simple Mail Transfer Protocol)*

Como aplicación para este servicio hemos utilizado **Postfix** y el proceso de instalación y configuración ha sido el siguiente:

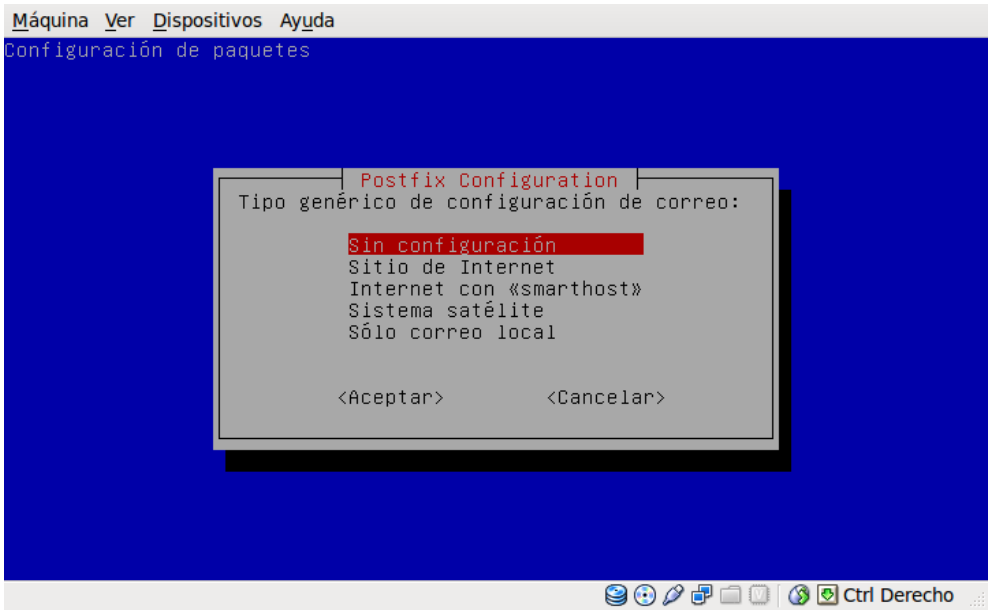
1. Instalamos los paquetes necesarios mediante el siguiente comando:

   aptitude install postfix postfix-tls sasl2-bin

Una vez iniciado el proceso de instalación, nos aparecerá una pantalla indicando que especifiquemos el tipo de configuración para el servidor, tal y como podemos ver en la imagen:



Como la configuración la vamos a realizar manualmente sobre diferentes ficheros, elegimos la opción **Sin Configuración**, procediendo posteriormente a la edición y modificación de los ficheros necesarios.



2. El primer fichero que vamos a modificar es el fichero **main.cf**, el cual vamos a obtenerlo de **/usr/lib/postfix/main.cf** y copiarlo en **/etc/postfix/main.cf**. Una vez realizada la copia, procedemos a editarlo y configurarlo, realizando sobre él las siguientes modificaciones.

```
# línea 59: descomentar
mail_owner = postfix

# línea 76: descomentar y especificar el nombre del servidor
myhostname = servidor.mmsierradecadiz.lan

# línea 83: descomentar y especificar el dominio del servidor
mydomain = mmsierradecadiz.lan

# línea 104: descomentar
myorigin = $mydomain

# línea 118: descomentar
inet_interfaces = all

# línea 166: descomentar
mydestination = $myhostname, localhost.$mydomain, localhost, $mydomain

# línea 209: descomentar
local_recipient_maps = unix:passwd.byname $alias_maps

# línea 268: descomentar y especificar la red
mynetworks = 127.0.0.0/8, 192.168.0.0/24

# línea 388: descomentar
alias_maps = hash:/etc/aliases

# línea 399: descomentar
alias_database = hash:/etc/aliases

# línea 421: descomentar ( usar Maildir )
home_mailbox = Maildir/

# línea 526: descomentar
header_checks = regexp:/etc/postfix/header_checks
# añadir: chequea el cuerpo del mensaje
body_checks = regexp:/etc/postfix/body_checks

# línea 552: comentar la primera línea y añadir la que aparece a
continuación
# smtpd_banner = $myhostname ESMTP $mail_name (@@DISTRO@@)
smtpd_banner = $myhostname ESMTP

# línea 626: añadir
sendmail_path = /usr/sbin/postfix

# línea 631: añadir
newaliases_path = /usr/bin/newaliases

# línea 636: añadir
mailq_path = /usr/bin/mailq
```

```
# línea 642: añadir
setgid_group = postdrop

# línea 646: comentar
# html_directory =

# línea 650: comentar
# manpage_directory =

# línea 655: comentar
# sample_directory =

# línea 659: comentar
# readme_directory =

# añadir al final de fichero para limitar el tamaño de los correos a 10
MBytes
message_size_limit = 10485760

# añadir a continuación para limitar el tamaño del buzón a 1 GByte
mailbox_size_limit = 1073741824


# añadir para autenticación SMTP-Auth
smtpd_sasl_type = dovecot
smtpd_sasl_path = private/auth-client
smtpd_sasl_auth_enable = yes
smtpd_sasl_security_options = noanonymous
smtpd_sasl_local_domain = $myhostname
smtpd_client_restrictions = permit_mynetworks,reject_unknown_client,permit
smtpd_recipient_restrictions =
permit_mynetworks,permit_auth_destination,permit_sasl_authenticated,reject
```

3. El segundo fichero a modificar es el fichero **header_checks**, el cual vamos a editar desde su ubicación actual que es **/etc/postfix/header_checks**, añadiendo sobre él lo siguiente:

```
# rechaza las direcciones de correo que no están especificadas ( vacías )
/^From:.*<#.*@.*>/ REJECT
/^Return-Path:.*<#.*@.*>/ REJECT
```

4. El tercer fichero a modificar es el fichero **body_checks**, el cual vamos a editar desde su ubicación actual que es **/etc/postfix/body_checks**, añadiendo sobre él lo siguiente:

```
# rechaza los mensajes que incluyen la dirección especificada en el cuerpo
# del mensaje
/^(|[^>].*)example.com/ REJECT
```

5. Por último, actualizamos el fichero **/etc/aliases** de la base de datos con el comando:

   newaliases

   y reiniciamos el servicio para habilitar los cambios realizados en los ficheros de configuración, ejecutando para ello el siguiente comando.

   /etc/init.d/postfix restart

4.2. *Servicio IMAP (Internet Message Access Protocol)*

Como aplicación para este servicio hemos utilizado **Dovecot** y el proceso de instalación y configuración ha sido el siguiente:

1. Instalamos los paquetes necesarios mediante el siguiente comando:

   aptitude install dovecot-common dovecot-imapd

2. A continuación, editamos el fichero de configuración **dovecot.conf** que se encuentra en la ubicación **/etc/dovecot/dovecot.conf** y realizamos sobre él las siguientes modificaciones:

```
# línea 53: descomentar and cambiar ( autenticación con texto plano )
disable_plaintext_auth = no

# línea 95: descomentar and cambiar ( permite uso de certificados )
ssl = yes

# línea 230: descomentar y añadir
mail_location = maildir:~/Maildir

# línea 893: añadir
mechanisms = plain login

# línea 1120: cambiar como viene a continuación
socket listen {
    #master {
          # Master socket provides access to userdb information. It's
typically
      # used to give Dovecot's local delivery agent access to userdb so it
      # can find mailbox locations.
      #path = /var/run/dovecot/auth-master
      #mode = 0600
      # Default user/group is the one who started dovecot-auth (root)
      #user =
      #group =
    #}
    client {
      # The client socket is generally safe to export to everyone. Typical
      # use is to export it to your SMTP server so it can do SMTP AUTH
      # lookups using it.
      #path = /var/run/dovecot/auth-client
      path = /var/spool/postfix/private/auth-client
      mode = 0660
      user = postfix
      group = postfix
      #mode = 0660
    }
  }
}
```

3. Por último, reiniciamos el servicio para habilitar los cambios realizados en el fichero de configuración, ejecutando para ello el siguiente comando.

   /etc/init.d/dovecot restart

4.3. *Configurar Postfix y Dovecot para SSL*

Después de haber configurado los servicios de SMTP e IMAP, vamos a añadirle la opción de usar los **certificados SSL** que habíamos generado anteriormente, añadiendo con ello otro nivel más de seguridad.

Los pasos que hemos seguido para su instalación y configuración han sido:

1. Editamos el fichero **/etc/postfix/main.cf**, añadiendo la siguiente información:

```
# añadir al final del fichero para configurar Postfix para SSL
smtpd_use_tls = yes
smtpd_tls_cert_file = /etc/ssl/certs/servidorMMSC.crt
smtpd_tls_key_file = /etc/ssl/private/servidorMMSC.key
smtpd_tls_session_cache_database = btree:${data_directory}/smtpd_scache
```

2. Además, también debemos de editar el fichero **/etc/postfix/master.cf** y realizar en él la siguiente modificación:

```
# líneas 17, 18: descomentar
smtps     inet  n       -       -       -       -       smtpd
  -o smtpd_tls_wrappermode=yes
```

3. También será necesario editar el fichero **/etc/dovecot/dovecot.conf** y realizar en él las siguientes modificaciones:

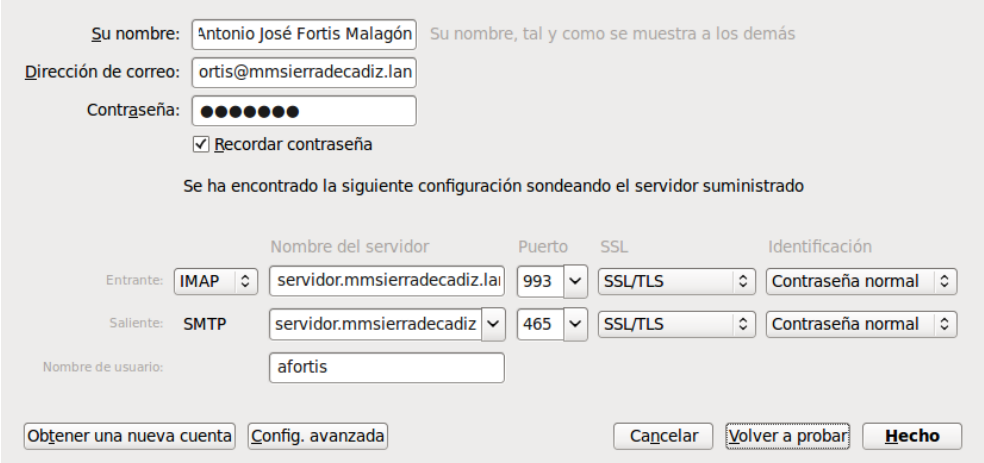
```
# línea 95: descomentar
ssl =  yes

# líneas 100, 101: descomentar y especificar certificado
ssl_cert_file = /etc/ssl/certs/servidorMMSC.crt
ssl_key_file = /etc/ssl/private/servidorMMSC.key
```

4. Por último, reiniciamos los servicios SMTP e IMAP, ejecutando para ellos los comandos:

   /etc/init.d/postfix restart

   /etc/init.d/dovecot restart

Una vez realizadas dichas configuraciones, podemos configurar un cliente de correo como **Thunderbird** y comprobar que el Servicio de Correo funciona correctamente.



Una vez realizada la configuración, al acceder al cliente **Thunderbird**, nos pedirá que aceptemos nuestro certificado, tal y como podemos ver en la siguiente captura:

Por último, junto al usuario afortis@mmsierradecadiz.lan, se ha creado otro usuario en el servidor, llamado usuario@mmsierradecadiz.lan y se ha comprobado el intercambio de correos electrónicos, tal y como podemos comprobar.

4.4. *Servicio WebMail*

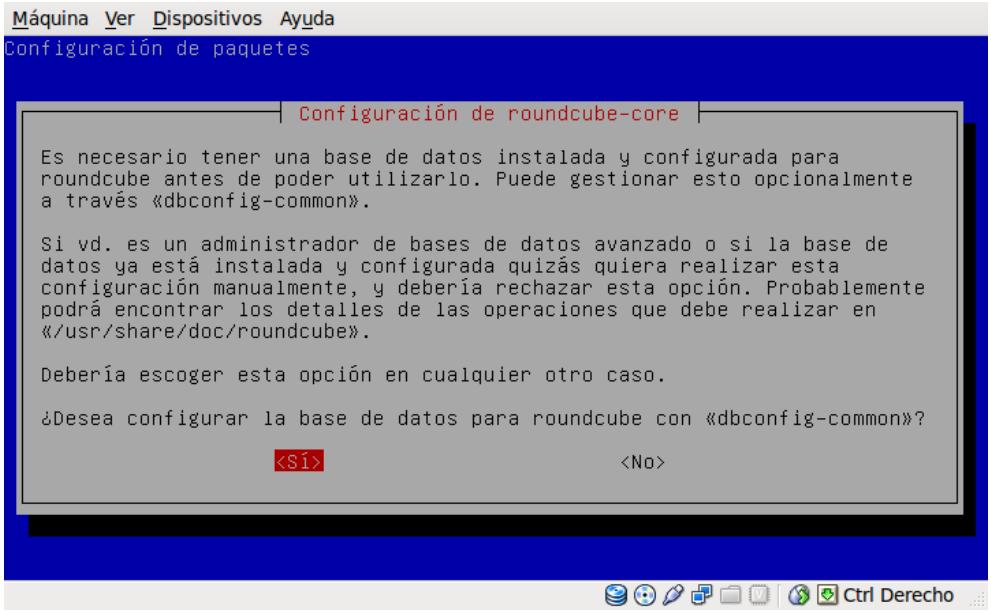
El siguiente paso será configurar un servicio de WebMail, para el cual hemos elegido **RoundCube**, que nos va a permitir consultar nuestro correo electrónico de manera fácil, práctica, confortable y accesible, desde cualquier lugar.

Para su instalación y configuración hemos realizado lo siguiente:

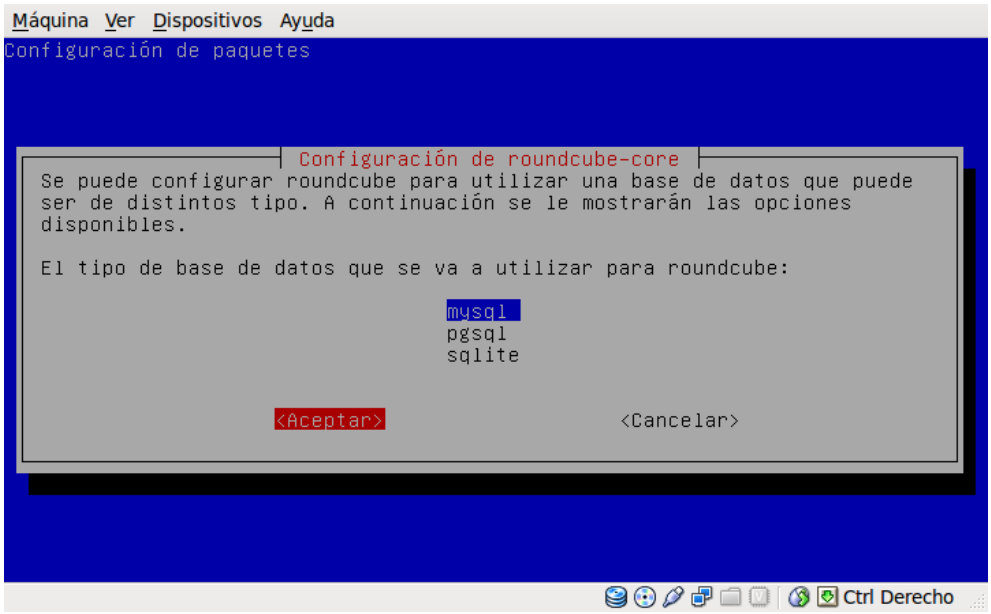
1. Instalamos los paquetes necesarios, mediante el comando:

   aptitude install roundcube roundcube-mysql

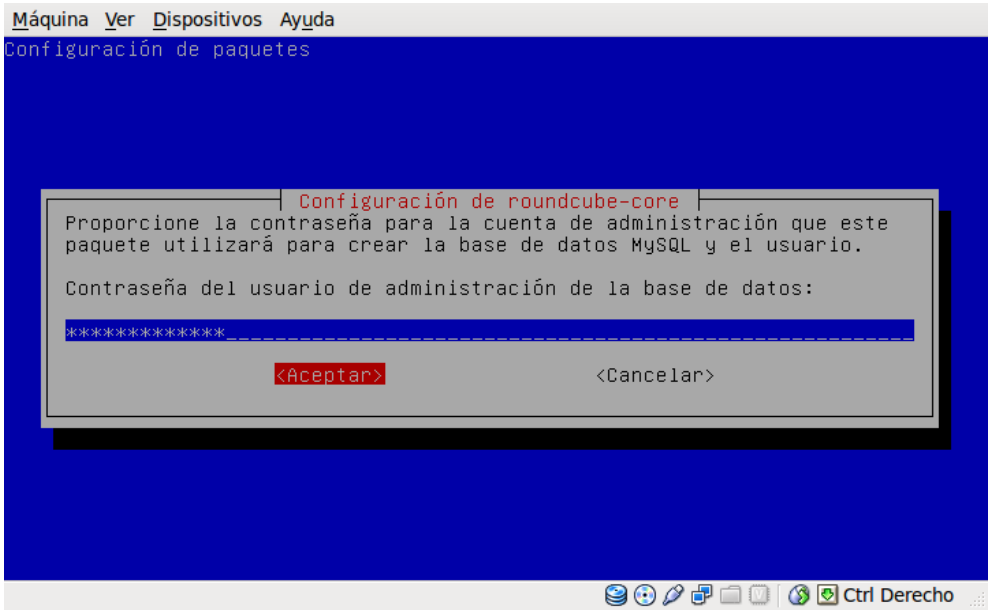
2. Nos indica si queremos configurar la base de datos que tenemos instalada para roundcube, a lo que respondemos que Sí



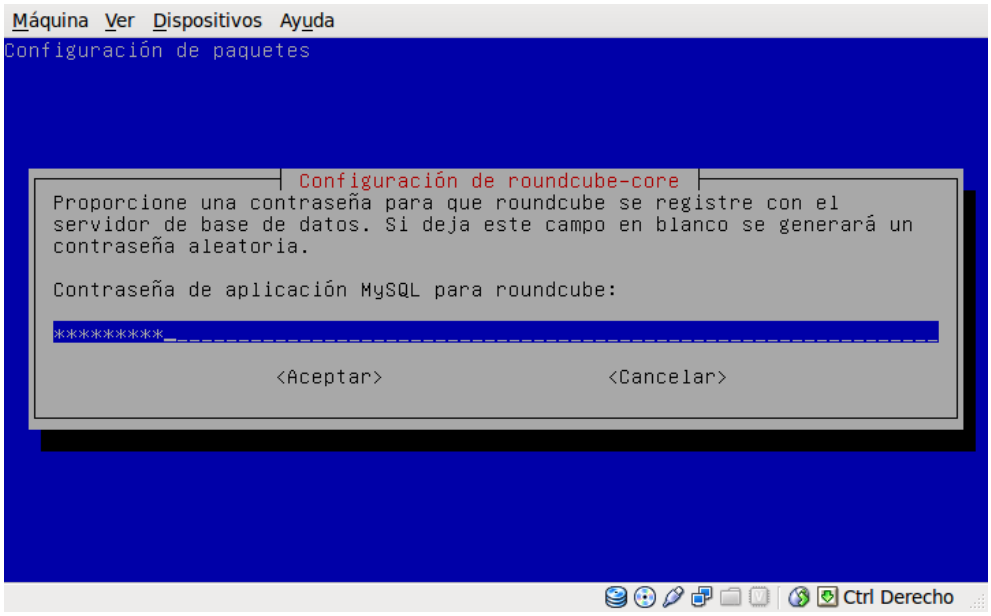
3. A continuación, nos pide que seleccionemos la base de datos que vamos a utilizar, siendo en nuestro caso mysql, tal y como podemos ver.



4. El siguiente paso será especificar la clave de root de MySQL para poder crear la base de datos y el usuario, según podemos ver en la siguiente captura.



5. Y por último, asignar una contraseña a la base de datos que se ha creado.



Una vez instalada la aplicación, debemos de realizar las configuraciones necesarias para ajustar la aplicación a nuestro servidor de correo, para lo cual, tenemos que editar el fichero **/etc/roundcube/main.inc.php** y realizar sobre él las siguientes modificaciones:

```
# línea 66: especifica el servidor IMAP con soporte para SSL
$rcmail_config['default_host'] = 'ssl://servidor.mmsierradecadiz.lan';

# línea 69: especifica el puerto IMAP para el soporte SSL
$rcmail_config['default_port'] = 993;

# línea 87: indica el nombre de dominio
$rcmail_config['mail_domain'] = 'mmsierradecadiz.lan';
```

```
# línea 102: especifica el servidor SMTP con soporte para SSL
$rcmail_config['smtp_server'] = 'ssl://servidor.mmsierradecadiz.lan';

# línea 105: especifica el puerto SMTP para el soporte SSL
$rcmail_config['smtp_port'] = 465;

# línea 109: RoundCube usa como usuario SMTP el utilizado en el login
$rcmail_config['smtp_user'] = '%u';

# línea 113: RoundCube usa como password SMTP el utilizado en el login
$rcmail_config['smtp_pass'] = '%p';

# línea 123: establece SMTP HELO host con el nombre de nuestro servidor
$rcmail_config['smtp_helo_host'] = 'servidor.mmsierradecadiz.lan';

# línea 168: establece el lenguaje a español
$rcmail_config['language'] = 'es_ES';

# línea 180: asigna a UserAgent el mensaje de cabecera en los envíos
$rcmail_config['useragent'] = 'RoundCube Webmail/'.RCMAIL_VERSION;

# línea 183: indica el título que tendrá la página web de RoundCube
$rcmail_config['product_name'] = 'Servidor Mancomunidad Sierra de Cadiz';

# línea 218: indica la codificación de caracteres por defecto
$rcmail_config['default_charset'] = 'UTF-8';
```

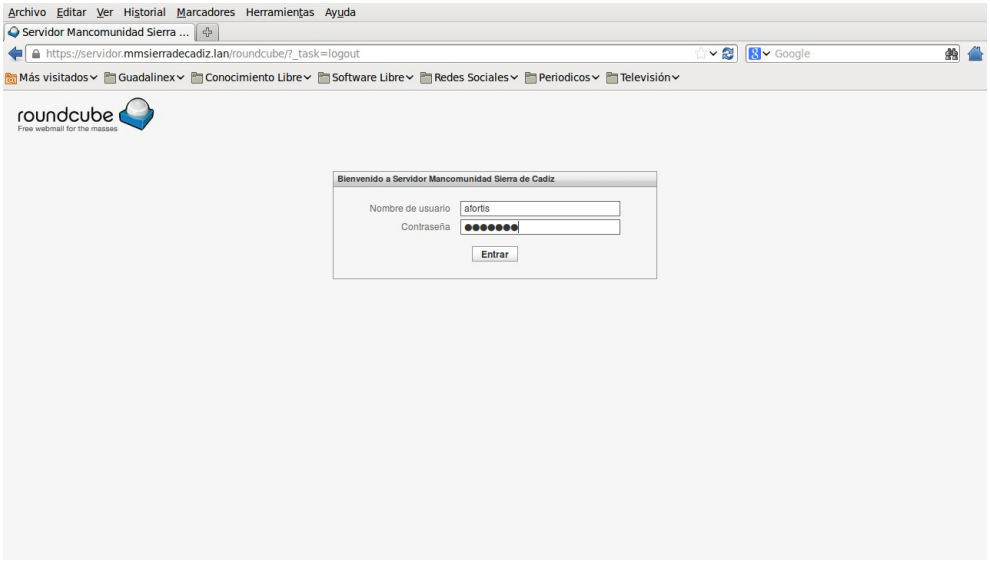
El siguiente fichero a modificar, es **/etc/roundcube/apache.conf**, en el cual sólo vamos a realizar una modificación.

```
# línea 4: descomentarla
Alias /roundcube /var/lib/roundcube
```

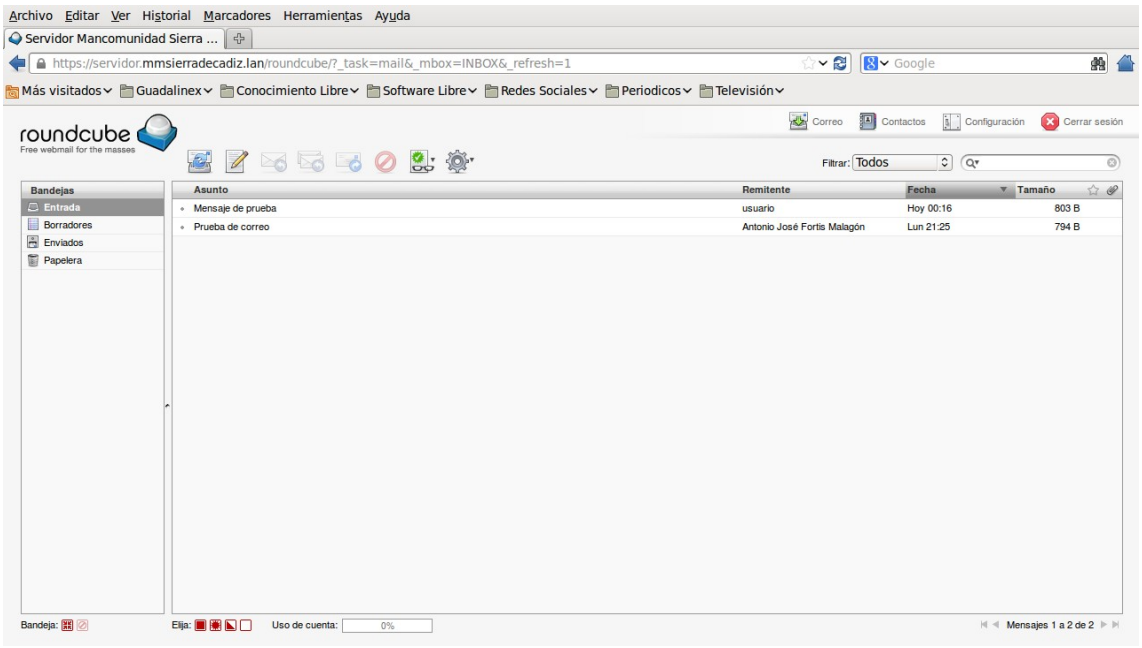
Para que los cambios en los ficheros surtan efecto, habrá que reiniciar el servidor Apache, utilizando para ello el comando que se especifica a continuación.

/etc/init.d/apache2 restart

Una vez reiniciado es servidor Apache, podemos comprobar el funcionamiento del servicio, introduciendo la dirección http://servidor.mmsierradecadiz.lan/roundcube.



Y una vez que nos hemos identificado, se puede comprobar como se accede a nuestro correo electrónico a través de WebMail.



4.5. *Filtros de correo (anti-virus, anti-spam)*

Para terminar con el servicio de correo electrónico, vamos a instalar y configurar una serie de filtros que van a añadir seguridad al servicio. Este sistema de filtros va a permitir implementar soluciones anti-virus y anti-spam, tal y como vamos a describir en los siguientes pasos:

1. Instalación de los paquetes necesarios ejecutando el siguiente comando.

   aptitude install clamav-daemon amavisd-new spamassassin

2. Editar el fichero **/etc/default/spamassassin** y realizar sobre él la siguiente modificación.

```
# línea 8: establecer a “1” para habilitar filtro anti-spam
ENABLED=1
```

3. Editar el fichero **/etc/amavis/amavisd.conf** y realizar sobre él las siguientes modificaciones.

```
# línea 66: descomentar
$MYHOME = '/var/lib/amavis';

# línea 71: especificar nombre de dominio
$mydomain = 'mmsierradecadiz.lan';

# línea 73: descomentar y especificar nombre del servidor
$myhostname = 'servidor.mmsierradecadiz.lan';

# línea 77,78: cambiar
$daemon_user  = 'amavis';
$daemon_group = 'amavis';

# línea 626: comentar ( no notifica si un virus es detectado )
#$virus_admin = "virusalert\@$mydomain";
```

```
# línea 1934: descomentar y añadir
['ClamAV-clamd',
  \&ask_daemon, ["CONTSCAN {}\n", "/var/run/clamav/clamd.ctl"],
  qr/\bOK$/m, qr/\bFOUND$/m,
  qr/^.*?: (?!Infected Archive)(.*) FOUND$/m ],
```

4. Editar el fichero **/etc/amavis/conf.d/15-content_filter_mode** y realizar sobre él las siguientes modificaciones.

```
# línea 13: descomentar
@bypass_virus_checks_maps = (
   \%bypass_virus_checks, \@bypass_virus_checks_acl, \
$bypass_virus_checks_re);

# línea 24: descomentar ( sólo si usamos filtro anti-spam )
@bypass_spam_checks_maps = (
   \%bypass_spam_checks, \@bypass_spam_checks_acl, \
$bypass_spam_checks_re);
```

5. Editar el fichero **/etc/postfix/main.cf** y añadir al final la siguiente línea.

```
# Configurar amavis en Postfix
content_filter=smtp-amavis:[127.0.0.1]:10024
```

6. Editar el fichero **/etc/postfix/master.cf** y añadir al final las siguientes líneas.

```
# Configurar amavis en Postfix
smtp-amavis unix -       -       n       -      2 smtp
      -o smtp_data_done_timeout=1200
      -o smtp_send_xforward_command=yes
      -o disable_dns_lookups=yes
127.0.0.1:10025 inet n     -      n      -      - smtpd
      -o content_filter=
      -o local_recipient_maps=
      -o relay_recipient_maps=
      -o smtpd_restriction_classes=
      -o smtpd_client_restrictions=
      -o smtpd_helo_restrictions=
      -o smtpd_sender_restrictions=
      -o smtpd_recipient_restrictions=permit_mynetworks,reject
      -o mynetworks=127.0.0.0/8
      -o strict_rfc821_envelopes=yes
      -o smtpd_error_sleep_time=0
      -o smtpd_soft_error_limit=1001
    -o smtpd_hard_error_limit=1000
```

Para finalizar, ejecutamos los comandos que se muestran a continuación:

touch /etc/mailname

chmod -R 775 /var/lib/amavis/tmp

usermod -G amavis clamav

Y reiniciamos los servicios que se han visto implicados para recargar las nuevas configuraciones que se han establecido.

/etc/init.d/clamav-daemon restart

/etc/init.d/spamassassin start

/etc/init.d/postfix restart

/etc/init.d/amavis restart

En la siguientes captura podemos ver como quedarían tras su ejecución en el servidor.

```
root@servidor:~# touch /etc/mailname
root@servidor:~# chmod -R 775 /var/lib/amavis/tmp
root@servidor:~# usermod -G amavis clamav
root@servidor:~# /etc/init.d/clamav-daemon restart
Stopping ClamAV daemon: clamd.
Starting ClamAV daemon: clamd .
root@servidor:~# /etc/init.d/spamassassin start
Starting SpamAssassin Mail Filter Daemon: spamd.
root@servidor:~# /etc/init.d/postfix restart
Stopping Postfix Mail Transport Agent: postfix.
Starting Postfix Mail Transport Agent: postfix.
root@servidor:~# /etc/init.d/amavis restart
Stopping amavisd: amavisd-new.
Starting amavisd: amavisd-new.
root@servidor:~#
```

Junto con el resultado del análisis de un correo electrónico que se ha enviado, donde se puede comprobar que ha sido analizado con dichas herramientas para comprobar la existencia de algún tipo de malware.

Archivo Editar Ver Ayuda

```
Return-Path: <usuario@mmsierradecadiz.lan>
X-Original-To: afortis@mmsierradecadiz.lan
Delivered-To: afortis@mmsierradecadiz.lan
Received: from localhost (localhost [127.0.0.1])
    by servidor.mmsierradecadiz.lan (Postfix) with ESMTP id 0280D6E33A
    for <afortis@mmsierradecadiz.lan>; Tue, 19 Feb 2013 01:14:49 +0100 (CET)
X-Virus-Scanned: Debian amavisd-new at mmsierradecadiz.lan
Received: from servidor.mmsierradecadiz.lan ([127.0.0.1])
    by localhost (servidor.mmsierradecadiz.lan [127.0.0.1]) (amavisd-new, port 10024)
    with ESMTP id KZcamUMbJiqP for <afortis@mmsierradecadiz.lan>;
    Tue, 19 Feb 2013 01:14:45 +0100 (CET)
Received: from [192.168.0.100] (unknown [192.168.0.100])
    by servidor.mmsierradecadiz.lan (Postfix) with ESMTPSA id 70CC56E337
    for <afortis@mmsierradecadiz.lan>; Tue, 19 Feb 2013 01:14:45 +0100 (CET)
Message-ID: <5122C3F6.7000402@mmsierradecadiz.lan>
Date: Tue, 19 Feb 2013 01:14:46 +0100
From: usuario <usuario@mmsierradecadiz.lan>
User-Agent: Mozilla/5.0 (X11; Linux i686; rv:17.0) Gecko/20130107 Thunderbird/17.0.2
MIME-Version: 1.0
To: afortis@mmsierradecadiz.lan
Subject: =?ISO-8859-1?Q?Env=EDo_de_un_correo_analizado_con_lo?=
 =?ISO-8859-1?Q?s_filtros=2E?=
Content-Type: text/plain; charset=ISO-8859-1; format=flowed
Content-Transfer-Encoding: 8bit

Vamos a enviar un correo desde el usuario "usuario" al usuario "afortis"
verificando que se han instalado los filtros Anti-Virus y Anti-Spam en
el Servidor de Correo.

Para ello, se visualizarán las propiedades del correo enviado y se podrá
comprobar que ha sido analizado mendiante dichos filtros.
```

5. Servicios de Seguridad

Dentro de los Servicios de Seguridad que vamos a instalar y configurar en nuestro servidor, se encuentran los siguientes:

- Herramienta para la prevención de Ataques por Fuerza Bruta (Fail2Ban).
- Herramienta para la Detección de Rootkits (Rootkit Hunter).
- Herramienta para Auditoría de Seguridad (Debsecan).

5.1. Ataques por Fuerza Bruta (Fail2Ban)

Fail2Ban es una aplicación que analiza continuamente los ficheros log y bloquea las direcciones Internet, de donde se hayan originado varias tentativas fallidas de acceso con contraseña inválida. Además, es extremadamente eficaz en la prevención de ataques de fuerza bruta y ataques de negación de servicio (DoS).

El procedimiento que he seguido para su instalación y configuración, ha sido el siguiente:

1. Instalación de los paquetes necesarios mediante el comando:

   aptitude install fail2ban whois

2. Para realizar las configuraciones necesarias, se aconseja trabajar sobre una copia local del fichero **/etc/fail2ban/jail.conf**, para lo cual, copiamos dicho fichero con el nombre **/etc/fail2ban/jail.local** y sobre él realizaremos las siguientes modificaciones.

   cp /etc/fail2ban/jail.conf /etc/fail2ban/jail.local

3. Definimos cuales serán las direcciones IP que no estarán sujetas a las restricciones y especificamos por cuanto tiempo estarán bloqueadas las direcciones de donde provengan las amenazas (600 seg), junto con el número de tentativas permitidas (3 tentativas).

   ```
   ignoreip = 127.0.0.1 192.168.0.0/24
   bantime  = 600
   maxretry = 3
   ```

4. Indicamos la dirección de correo electrónico que recibirá las alertas.

   ```
   destemail = afortis@mmsierradecadiz.lan
   ```

5. Configuramos las acciones a realizar cuando se detecte un posible ataque. En este caso, la dirección IP del atacante es bloqueada y se envía un correo electrónico al administrador del sistema.

   ```
   # ACTIONS
   #

   # Default banning action (e.g. iptables, iptables-new,
   # iptables-multiport, shorewall, etc) It is used to define
   # action_* variables. Can be overriden globally or per
   # section within jail.local file
   banaction = iptables-multiport
   ```

```
# email action. Since 0.8.1 upstream fail2ban uses sendmail
# MTA for the mailing. Change mta configuration parameter to mail
# if you want to revert to conventional 'mail'.
mta = postfix

# Default protocol
protocol = tcp

#
# Action shortcuts. To be used to define action parameter

# The simplest action to take: ban only
action_ = %(banaction)s[name=%(__name__)s, port="%(port)s", protocol="%
(protocol)s]

# ban & send an e-mail with whois report to the destemail.
action_mw = %(banaction)s[name=%(__name__)s, port="%(port)s", protocol="%
(protocol)s]
              %(mta)s-whois[name=%(__name__)s, dest="%(destemail)s",
protocol="%(protocol)s]

# ban & send an e-mail with whois report and relevant log lines
# to the destemail.
action_mwl = %(banaction)s[name=%(__name__)s, port="%(port)s", protocol="%
(protocol)s]
               %(mta)s-whois-lines[name=%(__name__)s, dest="%
(destemail)s", logpath=%(logpath)s]

#Choose default action.  To change, just override value of 'action' with
#the interpolation to the chosen action shortcut (e.g.   action_mw,
#action_mwl, etc) in jail.local globally (section [DEFAULT]) or per
#specific section
action = %(action_)s
```

6. Se establecen los parámetros de los diferentes servicios que se desean proteger.

```
# JAILS

# [ ... ]

[ssh]

enabled = true
port  = ssh
filter      = sshd
logpath  = /var/log/auth.log
maxretry = 6

# [ ... ]
```

7. Y finalmente, reiniciamos el servicio con el comando que indicamos a continuación.

   /etc/init.d/fail2ban restart

5.2. *Detección de Rootkits (Rootkit Hunter)*

Rootkit Hunter es un sistema de detección de rootkits que alerta por correo electrónico al administrador del sistema, en caso de que detecte alteraciones en el sistema que indiquen la presencia de un rootkit.

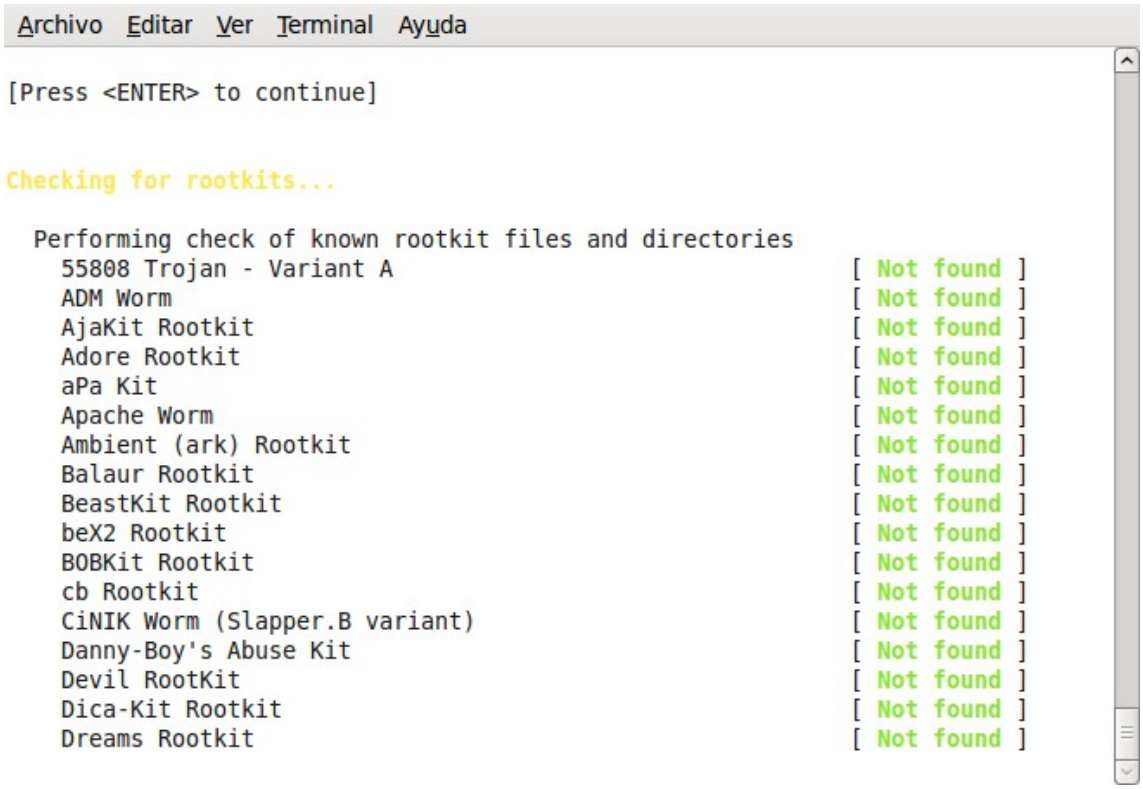
El procedimiento que he seguido para su instalación y configuración, ha sido el siguiente:

1. Instalación de los paquetes necesarios mediante el comando:

   aptitude install rkhunter

2. La configuración predeterminada del paquete **rkhunter** que se encuentra en el fichero **/etc/rkhunter.conf** suele ser suficiente. Además, el archivo **/etc/default/rkhunter** define que las actualizaciones de la base de datos tengan una periodicidad semanal, la verificación de rootkits diaria y que los resultados sean enviados por correo electrónico al administrador del sistema.

3. Para poder ver su funcionamiento, adjuntamos una captura de pantalla donde se puede observar los resultados obtenidos tras realizar un análisis del sistema al ejecutar el comando **rkhunter –check**.



5.3. *Auditoría de Seguridad (Debsecan)*

Debsecan efectúa una evaluación de la seguridad del sistema y relata las vulnerabilidades conocidas y asociadas a los paquetes instalados en el sistema, notificando al administrador del sistema los resultados.

El procedimiento que he seguido para su instalación y configuración, ha sido el siguiente:

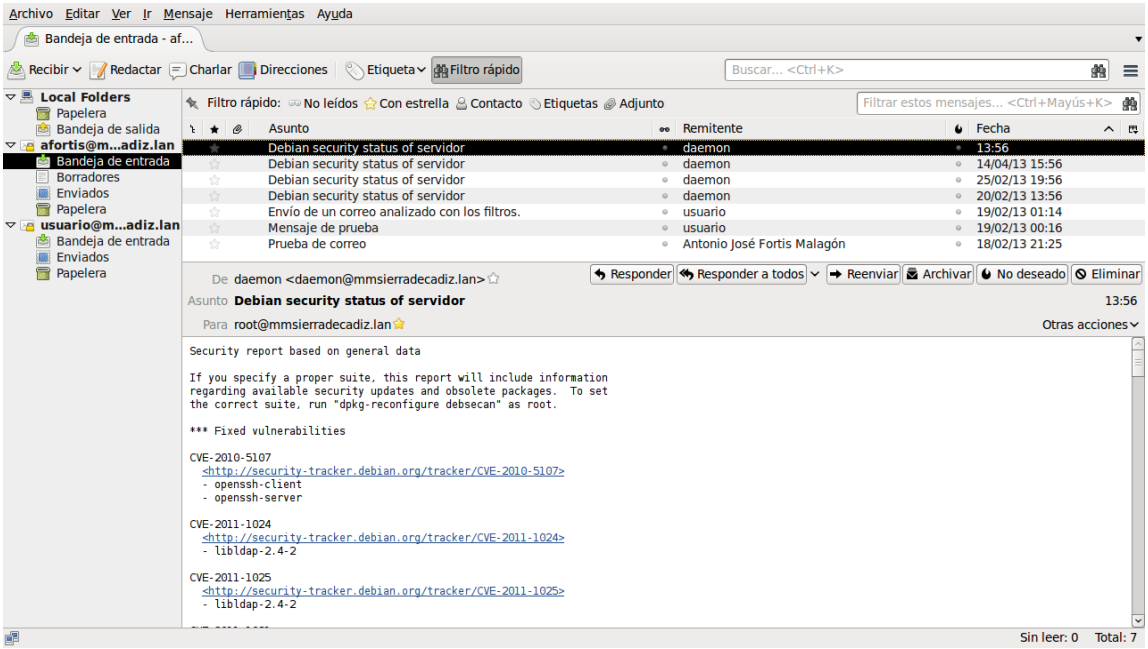
1. Instalación de los paquetes necesarios mediante el comando:

   aptitude install debsecan

2. Debsecan puede configurarse para ser ejecutado diariamente a través de una tarea programada (cron). De esta forma, sus resultados se enviarán por correo electrónico al administrador del sistema. El comando que debemos de ejecutar es el siguiente.

   debsecan-create-cron

3. Un ejemplo de utilización en el que se puede observar el envío de un correo electrónico al administrador del sistema, con los resultados del análisis de seguridad, lo podemos ver en la siguiente captura.



6. Instalación Plataforma de Formación (Moodle)

La plataforma de formación **Moodle**, es un Sistema de Gestión de Cursos de Código Abierto (Open Source Course Management System, CMS), conocido también como Sistema de Gestión del Aprendizaje (Learning Management System, LMS) cuyo objetivo es facilitar a los educadores una herramienta para crear sitios web dinámicos en línea para sus estudiantes.

Para su instalación, es necesario un Servidor Web con soporte para PHP y MySQL, el cual ya lo tenemos disponible, por lo tanto, vamos a detallar los pasos que hemos seguido para su instalación:

1. Nos descargamos de su página web la última versión estable disponible, que en este caso era **moodle-2.4.1.tgz** y procedemos a descomprimirlo en el directorio **/var/www**, utilizando para ello el siguiente comando.

   tar xzvf moodle-2.4.1.tgz -C /var/www

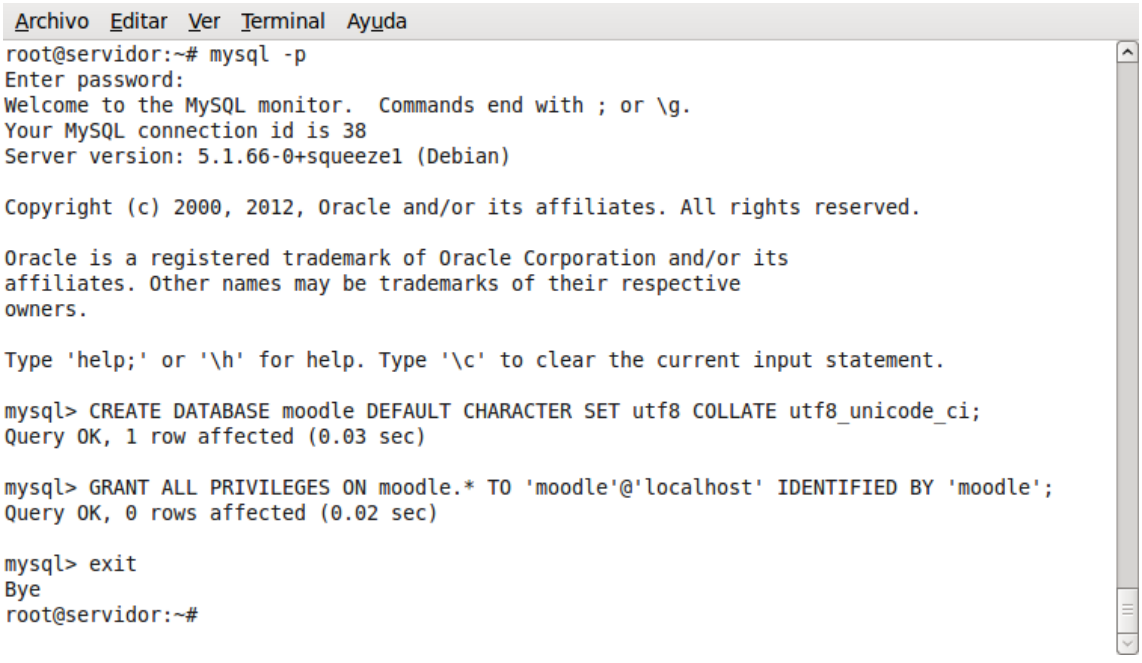
2. El siguiente paso, será crear el directorio **moodledata** y hacerlo accesible al Servidor Web, para ello, ejecutamos los siguientes comandos.

   mkdir /var/moodledata

   chown www-data /var/moodledata

   chown www-data /var/www/moodle

3. A continuación, debemos de crear la base de datos que utilizará Moodle (moodle), para ello, como administrador de MySQL, ejecutamos los comandos que se muestran en la siguiente captura.

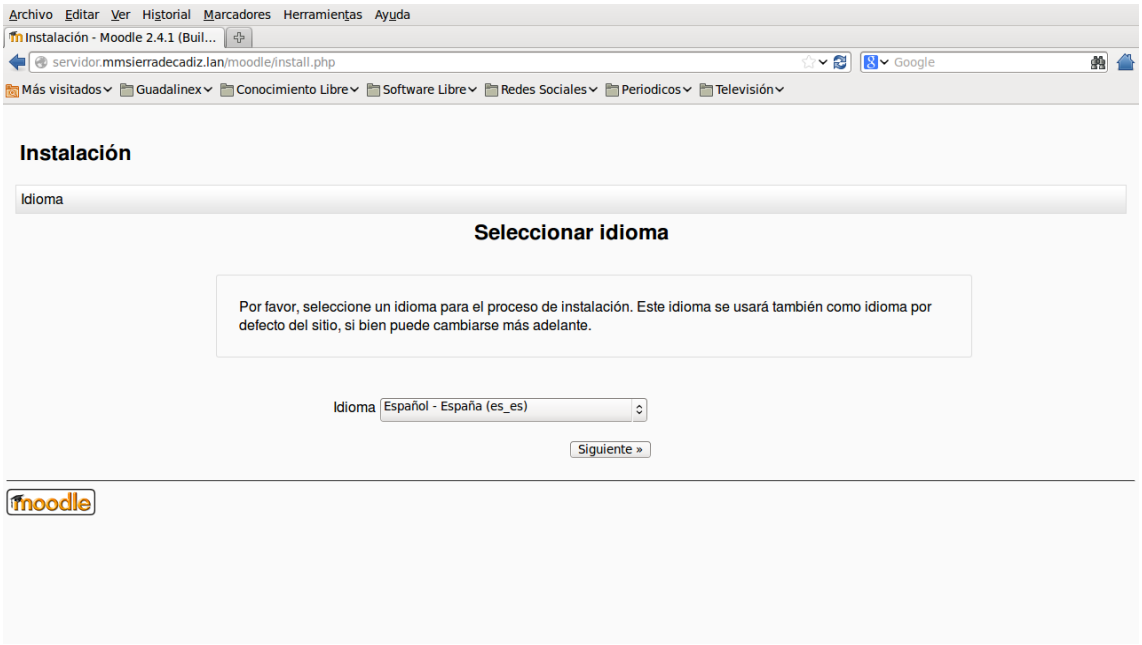


4. Una vez realizadas dichas operaciones, procedemos a reiniciar el Servidor Web para que todos los cambios realizados surtan efecto.

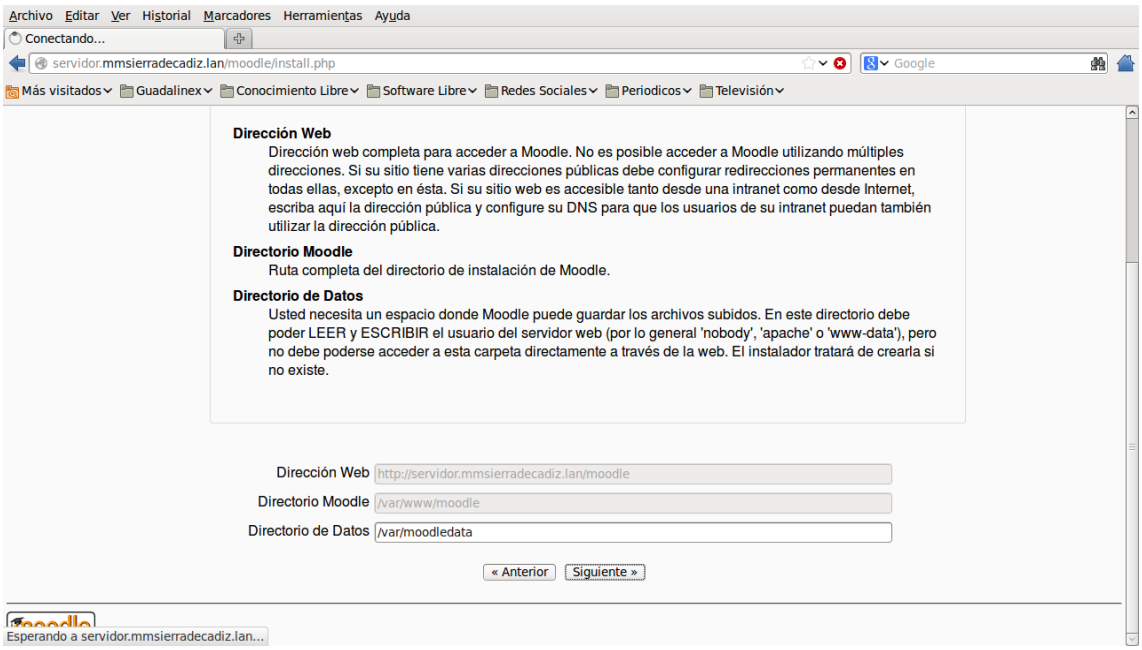
   /etc/init.d/apache2 restart

Una vez finalizada la primera parte de la instalación, para finalizar el proceso, debemos de acceder desde un navegador a la dirección donde está ubicada la plataforma **Moodle** (http://servidor.mmsierradecadiz.lan/moodle) y continuar con los siguientes pasos.

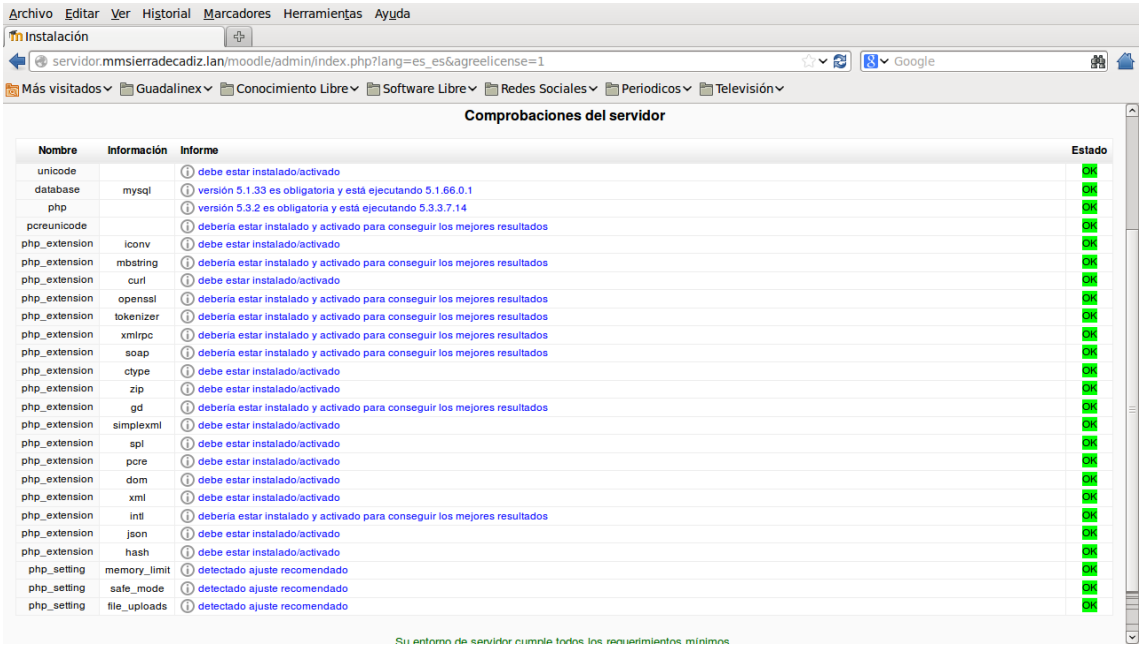
5. En primer lugar, debemos de seleccionar el idioma en el que deseamos que se muestren las opciones de la plataforma, en nuestro caso, **Español-España (es_es)**.



6. El siguiente paso, será definir los directorios que vamos a utilizar, los cuales ya vienen definidos, pero que se pueden modificar.



7. Una vez definidos los directorios, se hacen las **comprobaciones en el Servidor Web** para ver si se cumplen todos los requisitos, siendo dicha comprobación satisfactoria en este caso.



8. Por último, una vez que se ha dado de alta un usuario, podemos ver como quedaría la aplicación ya instalada y preparada para su personalización.

7. Instalación Almacenamiento Virtual (OwnCloud)

El sistema de Almacenamiento Virtual **OwnCloud**, es un software que proporciona un área de almacenamiento independiente de la ubicación de los datos (almacenamiento en la nube). Se creó como una aplicación basada en Software Libre que fuese una alternativa a los proveedores de cloud comerciales, por lo tanto, OwnCloud puede ser instalado en un servidor privado sin ningún tipo de coste adicional.

Para su instalación, es necesario un Servidor Web con soporte para PHP y MySQL, el cual ya lo tenemos disponible, por lo tanto, vamos a detallar los pasos que hemos seguido para su instalación:

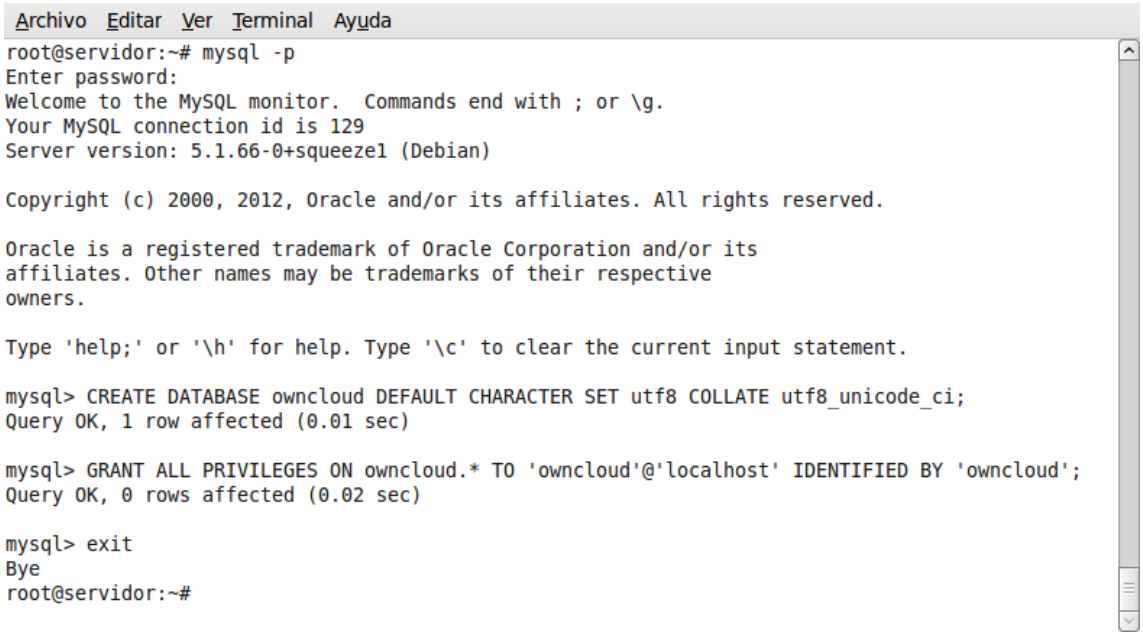
1. Nos descargamos de su página web la última versión estable disponible, que en este caso era **owncloud-latest.tar.bz2** y procedemos a descomprimirlo en el directorio **/var/www**, utilizando para ello el siguiente comando.

   tar -xvf owncloud-latest.tar.bz2 -C /var/www

2. El siguiente paso, será hacer dicho directorio accesible al Servidor Web, para ello, ejecutamos el siguiente comando.

   chown -R www-data:www-data /var/www/owncloud

3. A continuación, debemos de crear la base de datos que utilizará OwnClud (owncloud), para ello, como administrador de MySQL, ejecutamos los comandos que se muestran en la siguiente captura.

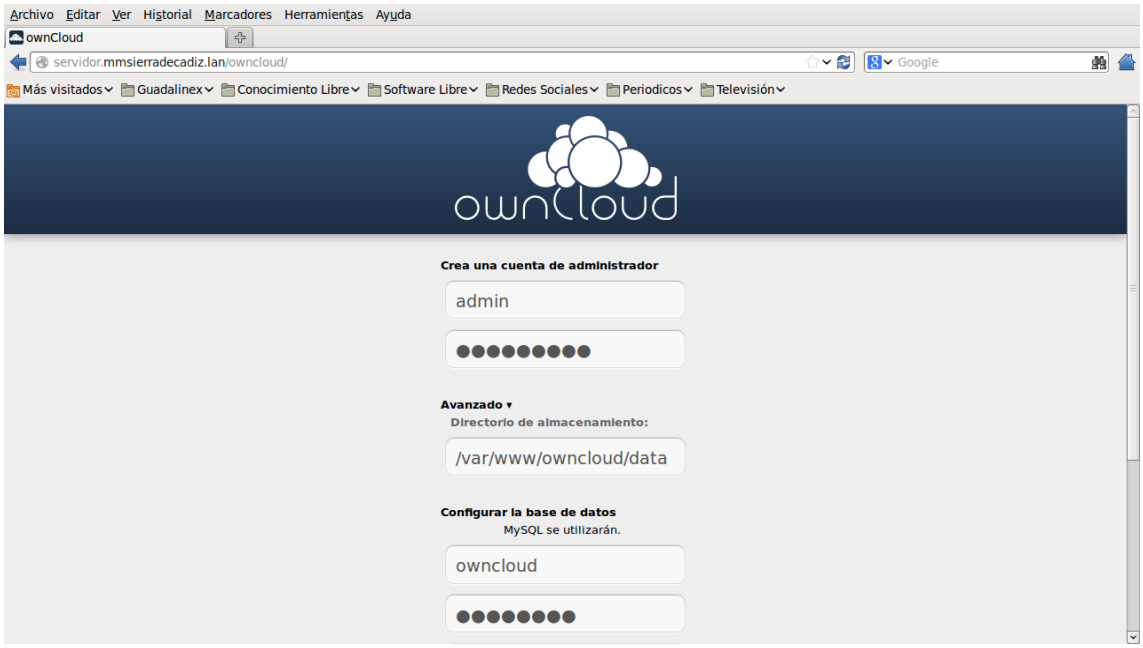


4. Una vez realizadas dichas operaciones, procedemos a reiniciar el Servidor Web para que todos los cambios realizados surtan efecto.

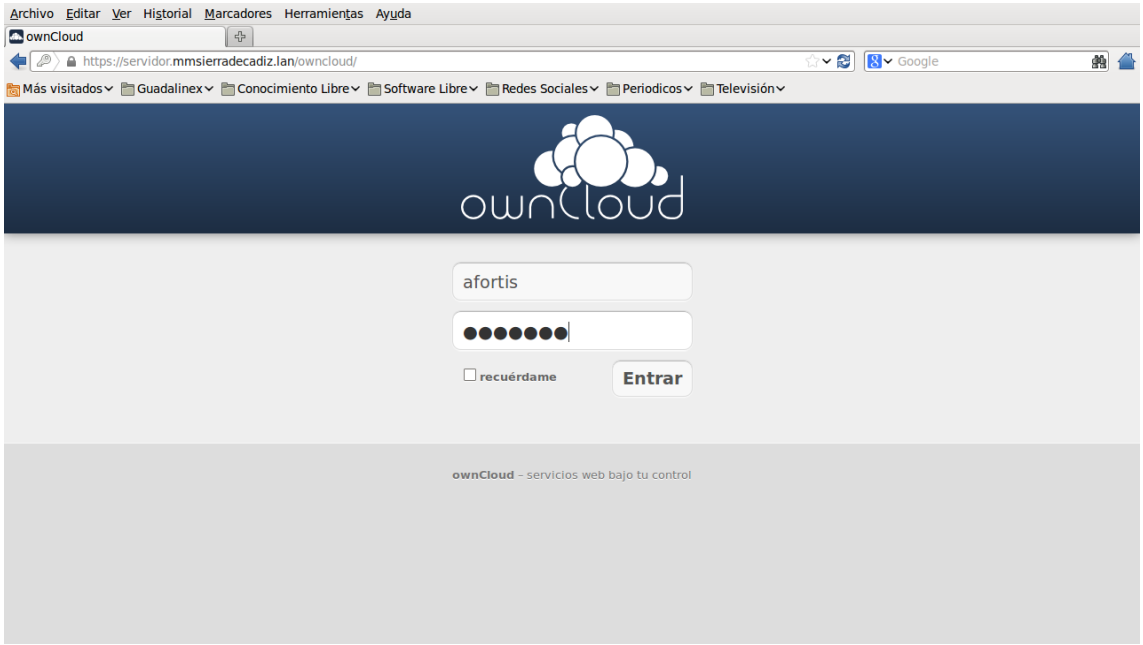
   /etc/init.d/apache2 restart

Una vez finalizada la primera parte de la instalación, para finalizar el proceso, debemos de acceder desde un navegador a la dirección donde está ubicada la plataforma **OwnClud** (http://servidor.mmsierradecadiz.lan/owncloud) y continuar con los siguientes pasos.

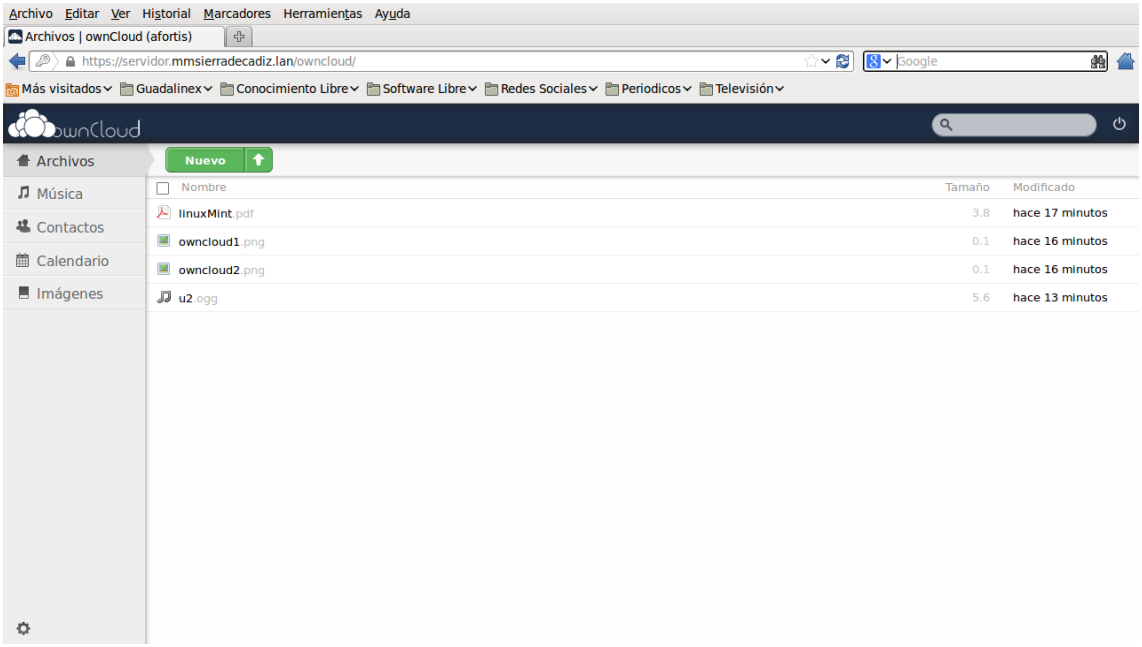
5. En primer lugar, debemos de crear una cuenta de administrador y especificar el directorio de almacenamiento y la base de datos que hemos creado para la aplicación.



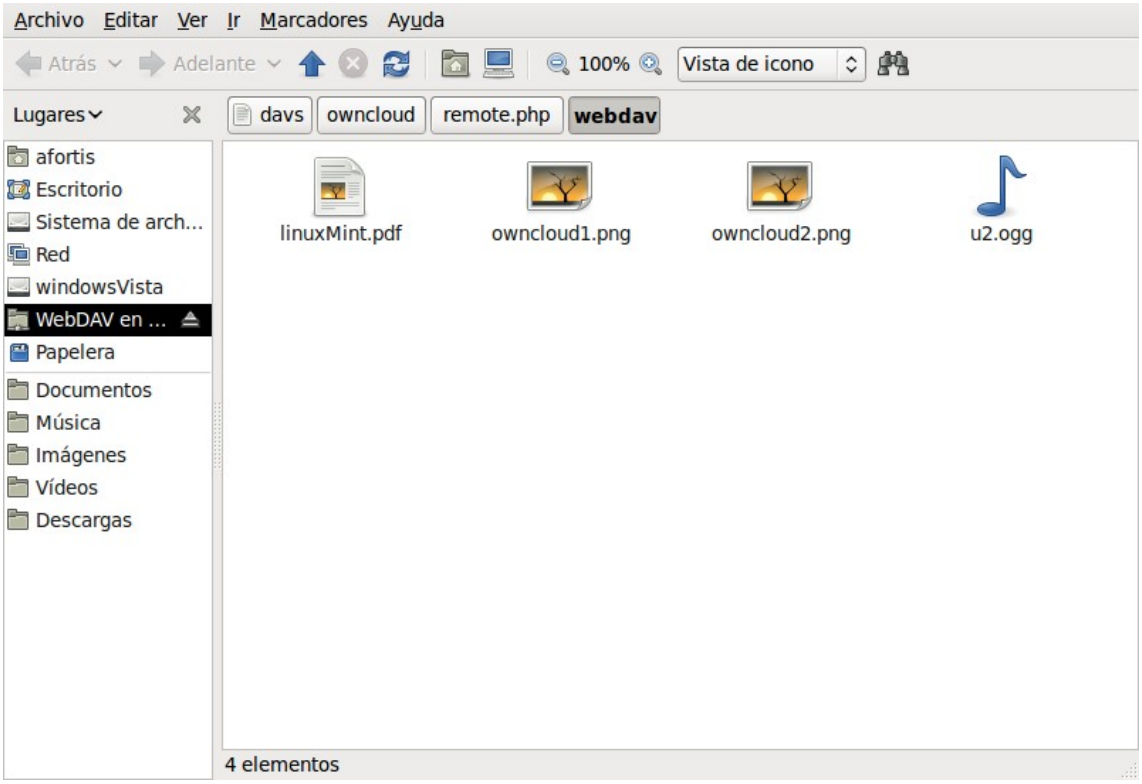
6. A continuación, una vez finalizado el proceso de instalación, se dará de alta un usuario, a través del cual accederemos a la aplicación.



7. Una vez que nos hemos identificado, podemos subir información a la plataforma, tal y como se puede ver, siendo accesible dicha información desde cualquier lugar.



8. Por último, también es posible configurar OwnCloud a través de WebDAV de forma que nos aparezca como una unidad en nuestro sistema, y de esta forma, sea más fácil acceder a nuestra información.



8. Configuración del Cortafuegos

El siguiente servicio que vamos a configurar en nuestro servidor es un cortafuegos, el cual, bloqueará los accesos no autorizados, permitiendo al mismo tiempo, comunicaciones seguras.

Para implementar dicho servicio, se ha hecho uso de **iptables**, el cual, es un componente del framework Netfilter, disponible para el núcleo Linux y que permite interceptar y manipular paquetes de red.

A continuación, se adjunta el código del script que se ha creado junto con los comentarios que explican su funcionamiento.

```
#!/bin/sh
## SCRIPT de IPTABLES para el Servidor de la Mancomunidad
## Script con Política por defecto Denegar

echo -n Aplicando Reglas de Firewall...

## FLUSH de reglas
iptables -F
iptables -X
iptables -Z
iptables -t nat -F

## Establecemos politica por defecto: DROP
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP

## Empezamos a abrir, porque ahora esta TODO denegado.
## Debemos decir de manera explicita qué es lo que queremos abrir.

# Operar en localhost sin limitaciones
iptables -A INPUT -i lo -j ACCEPT
iptables -A OUTPUT -o lo -j ACCEPT

# A nuestra IP le dejamos todo
iptables -A INPUT -s 192.168.0.101 -j ACCEPT
iptables -A OUTPUT -d 192.168.0.101 -j ACCEPT

#Permitimos conectarnos a nuestra máquina por SSH y tranferencia de ficheros
#por sFTP
iptables -A INPUT -p tcp -m tcp --sport 22 -m state --state RELATED,ESTABLISHED
-j ACCEPT
iptables -A OUTPUT -p tcp -m tcp --dport 22 -j ACCEPT

# Uno de los servicios que ofrece nuestra máquina es Servidor Web, por tanto,
# todo paquete entrante se acepta para ese puerto y los salientes vinculados se
aceptan.
iptables -A INPUT -p tcp -m tcp --dport 80 -j ACCEPT
iptables -A OUTPUT -p tcp -m tcp --sport 80 -m state --state RELATED,ESTABLISHED
-j ACCEPT
```

```
# Completamos el Servidor Web para ofrecer además conexiones seguras ( Servicio
HTTPS )
iptables -A INPUT -p tcp -m tcp --dport 443 -j ACCEPT
iptables -A OUTPUT -p tcp -m tcp --sport 443 -m state --state
RELATED,ESTABLISHED -j ACCEPT

#Otro de los servicios ofrecidos por nuestra máquina es Servidor de Correo,
#permitiendo el envío y la recepción de los mismos a través de SMTP e IMAP para
#los trabajadores
iptables -A INPUT -p tcp -m iprange --src-range 192.168.0.102-192.168.0.254 -d
192.168.0.101 --dport 25 -j ACCEPT
iptables -A OUTPUT -p tcp -m iprange --dst-range 192.168.0.102-192.168.0.254 -s
192.168.0.101 --sport 25 -j ACCEPT

iptables -A INPUT -p tcp -m iprange --src-range 192.168.0.102-192.168.0.254 -d
192.168.0.101 --dport 143 -j ACCEPT
iptables -A OUTPUT -p tcp -m iprange --dst-range 192.168.0.102-192.168.0.254 -s
192.168.0.101 --sport 143 -j ACCEPT

#También se permite el uso de los servicios de correo electrónico, de manera
#segura sobre SSL
iptables -A INPUT -p tcp -m iprange --src-range 192.168.0.102-192.168.0.254 -d
192.168.0.101 --dport 465 -j ACCEPT
iptables -A OUTPUT -p tcp -m iprange --dst-range 192.168.0.102-192.168.0.254 -s
192.168.0.101 --sport 465 -j ACCEPT

iptables -A INPUT -p tcp -m iprange --src-range 192.168.0.102-192.168.0.254 -d
192.168.0.101 --dport 993 -j ACCEPT
iptables -A OUTPUT -p tcp -m iprange --dst-range 192.168.0.102-192.168.0.254 -s
192.168.0.101 --sport 993 -j ACCEPT


# Permitimos que la maquina también pueda salir a la web
iptables -A INPUT -p tcp -m tcp --sport 80 -m state --state RELATED,ESTABLISHED
-j ACCEPT
iptables -A OUTPUT -p tcp -m tcp --dport 80 -j ACCEPT

# Y pueda consultar webs seguras
iptables -A INPUT -p tcp -m tcp --sport 443 -m state --state RELATED,ESTABLISHED
-j ACCEPT
iptables -A OUTPUT -p tcp -m tcp --dport 443 -j ACCEPT

# Permitimos la consulta a un primer DNS
iptables -A INPUT -s 80.58.61.250 -p udp -m udp --sport 53 -j ACCEPT
iptables -A OUTPUT -d 80.58.61.250 -p udp -m udp --dport 53 -j ACCEPT

# Permitimos la consulta a un segundo DNS
iptables -A INPUT -s 80.58.61.254 -p udp -m udp --sport 53 -j ACCEPT
iptables -A OUTPUT -d 80.58.61.254 -p udp -m udp --dport 53 -j ACCEPT

echo " OK . Verifique que lo que se aplica con: iptables -L -n"

# Fin del script
```

9. Configuración de la Copia de Seguridad

Por último, también se ha decidido implementar un servicio de copias de seguridad, el cual, copiará los ficheros de configuración de los diferentes servicios, junto con las bases de datos y demás directorios que se estimen oportunos y se especifiquen.

A continuación, se adjunta el código del script que se ha creado junto con los comentarios que explican su funcionamiento.

```
#!/bin/sh
#
#Script que realiza copia de seguridad de las configuraciones que contienen los
#diferentes servicios instalados en el servidor, junto con las bases de datos
#que existen en el sistema.
#
#Una vez realizada la copia, comprime dicha información y envía un e-mail al
#administrador indicando que la copia de seguridad se ha realizado
#correctamente.

NAME="backup"                               # Nombre script.
#Descripción: Script de backup de ficheros de configuracón, datos y Bases de
#Datos.
: ${DATE:=$(date +'%Y-%m-%d')}              # Variable para Fecha.
: ${TIME:=$(date +'%R')}                    # Variable para Hora.
: ${WORK_DIR:=/seguridad/backups/$DATE}  # Directorio de trabajo actual.
: ${LOG_FILE:=/seguridad/backups/$DATE/record.log}   # Archivo de log.
: ${ADMIN1:=afortis@mmsierradecadiz.lan}       # Email de Administrador
: ${LINE:="--------------------------------------------------------------------"}
: ${DB_PASS:=administrador}                    # Password de MySQL.

# Creando directorio donde se trabajará, si es que no existe.
if [ ! -d "`dirname $LOG_FILE`" ] ; then mkdir -p "`dirname $LOG_FILE`"; fi

# Cambiando a directorio donde trabajaremos.
cd $WORK_DIR

# Copiando /ETC/ hacia directorio de trabajo actual.
cp -Rv /etc/ $WORK_DIR > $LOG_FILE
echo $LINE >> $LOG_FILE

# Creando carpeta para logs.
mkdir $WORK_DIR/logs

# Copiando LOGs.
cp /var/log/apache2/ $WORK_DIR/logs/ -Rv >> $LOG_FILE && echo $LINE >> $LOG_FILE
cp /var/log/aptitud* $WORK_DIR/logs/ -v >> $LOG_FILE && echo $LINE >> $LOG_FILE
cp /var/log/auth* $WORK_DIR/logs/ -v >> $LOG_FILE && echo $LINE >> $LOG_FILE
cp /var/log/daemon* $WORK_DIR/logs/ -v >> $LOG_FILE && echo $LINE >> $LOG_FILE
cp /var/log/dmes* $WORK_DIR/logs/ -v >> $LOG_FILE && echo $LINE >> $LOG_FILE
cp /var/log/kern* $WORK_DIR/logs/ -v >> $LOG_FILE && echo $LINE >> $LOG_FILE
cp /var/log/mail* $WORK_DIR/logs/ -v >> $LOG_FILE && echo $LINE >> $LOG_FILE
cp /var/log/message* $WORK_DIR/logs/ -v >> $LOG_FILE && echo $LINE >> $LOG_FILE
cp /var/log/mysql/ $WORK_DIR/logs/ -Rv >> $LOG_FILE && echo $LINE >> $LOG_FILE
cp /var/log/mysql.* $WORK_DIR/logs/ -v >> $LOG_FILE && echo $LINE >> $LOG_FILE
cp /var/log/sys* $WORK_DIR/logs/ -v >> $LOG_FILE && echo $LINE >> $LOG_FILE
```

```
# Creando carpeta para Bases de Datos.
mkdir $WORK_DIR/dbs

# Exportando DBs.

mysqldump --opt -hlocalhost -uroot -p"$DB_PASS" information_schema >
"$WORK_DIR/dbs/information_schema.sql"
mysqldump --opt -hlocalhost -uroot -p"$DB_PASS" moodle >
"$WORK_DIR/dbs/moodle.sql"
mysqldump --opt -hlocalhost -uroot -p"$DB_PASS" mysql >
"$WORK_DIR/dbs/mysql.sql"
mysqldump --opt -hlocalhost -uroot -p"$DB_PASS" owncloud >
"$WORK_DIR/dbs/owncloud.sql"
mysqldump --opt -hlocalhost -uroot -p"$DB_PASS" roundcube >
"$WORK_DIR/dbs/roundcube.sql"

# Creando carpeta para otros datos del sistema.
mkdir $WORK_DIR/datos

# Uso de la utilidad Rsync para copiar otros datos del sistema.
# El fichero lista_directorios contiene los directorios a copiar.
rsync -av --delete --prune-empty-dirs --include-from=lista_directorios.txt /
$WORK_DIR/datos

# Saliendo un nivel más arriba (/seguridad/backups/).
cd ..

# Comprimiendo directorio de trabajo actual.
tar czvf "$DATE.tar.gz" "$DATE"

# Preparando información para enviar por email.
touch data.info
echo "El archivo $DATE.rar tiene un tamaño de:" > data.info
du -bsh $DATE.rar >> data.info
echo $LINE >> data.info
cat $LOG_FILE >> data.info
tar czvf data.info.tar.gz data.info


# Eliminando directorio una vez comprimido.
rm -R "$WORK_DIR"

# Enviando emails a administradores.
mail -s "[ScriptBackup] | Status de Backup." $ADMIN1 < data.info.tar.gz

echo "                  ########################"
echo "                  ## BACKUP REALIZADO! ##"
echo "                  ########################"

# Fin del script.
```