Universidad Internacional de La Rioja
Máster universitario en Seguridad Informática

Montaje de una nube segura para uso de una Intranet OTAN y Sistemas de Mando y Control

Trabajo Fin de Máster

presentado por: Ibáñez Vázquez, Jorge Manuel

Director/a: Martínez Herráiz, José Javier

Ciudad: Sanlúcar de Barrameda
Fecha: 21 de septiembre de 2017

Agradecimientos

Cuando comencé este proyecto hace casi un año no fui capaz de ver su envergadura real, como un escalador, vi la cima mucho más cerca de lo que realmente estaba.

Tener que compaginar el trabajo, sabiendo la dedicación tan exclusiva que tiene la forma de vida que elegí, y siendo un campo en el día a día que no me permite trabajar con material referente a todo lo estudiado durante este máster fue otro peldaño más hasta el objetivo final.

Indudablemente han sido unos meses donde los pasos fueron lentos pero seguros, más pesados de lo esperado, pero donde una vez alcanzada la cima es hora de reconocer y dar gracias, porque ni el mejor escalador llega al final sin ayuda.

Sin lugar a dudas mis mayores gracias son para mi familia, mis padres Manuel y Antonia por darme apoyo, por decirme que descansara de vez en cuando, por los ánimos que me daban a sabiendas de la dificultad de todo lo que llevaba encima. A mis hermanos Rubén y María, mis cuñados Charo y José María y mi amigo Álvaro porque siempre me alentaban con sus palabras y sabían que compartir una cerveza era la mejor forma posible de recuperar fuerzas para continuar con los estudios.

A mis compañeros de trabajo y en especial a mi Capitán Manuel por apoyarme en los momentos que necesitaba tiempo, por reconocer y entender el esfuerzo extra que debía realizar para sacar este máster y porque incluso en momentos de horas tardías cuando todos descansaban y él me veía estudiar me animaba.

Y por supuesto a dos mujercitas que están en mi corazón pues ese es su hogar. Dos mujercitas que marcaron mi vida y me hicieron ser mejor persona y querer seguir siéndolo día a día, y porque aunque una de ellas tuvo que irse, en mi vivirá eternamente, mi Laura y mi Paula.

Gracias, simplemente gracias por estar ahí a todos.

Resumen

Debido a la realidad actual, en la cual España cada vez realiza más operaciones y al uso, por lo tanto conjunto, de sistemas de Mando y Control en una red que debe ser global e interconectada entre los diferentes Ejércitos, se hace necesario el uso de una plataforma en la nube, que facilite el acceso desde cualquier zona de operaciones sea cual sea su ubicación en el planeta. Además, esta nube securizada debe permitir el empleo de correo interno, almacenaje de documentación, videoconferencias…, y todo aquello que sea necesario y facilite el cumplimiento de la misión.

Junto a esto, debemos considerar la evolución del uso por parte de las distintas administraciones del Estado Español de software libre, en cuyo caso el Ministerio de Defensa no está exento, por lo tanto se usará para la implantación y securización de la nube software igualmente libre, ownCloud y servidores Linux Ubuntu.

Palabras Clave: ownCloud – software libre – Mando y Control

Abstract

Due to the current reality, in which Spain increasingly carries out more operations and therefore the use of control and command systems in a network that must be global and interconnected between the different Armies, it becomes necessary to use A platform in the cloud, that facilitates access from any area of operations regardless of its location on the planet. In addition, this secured cloud should allow the use of internal mail, storage of documentation, videoconferences ..., and everything that is necessary and facilitate the fulfillment of the mission.

Along with this, we must consider the evolution of the use by different administrations of the Spanish State of free software, in which case the Ministry of Defense is not exempt, therefore it will be used for the implementation and securization of the equally free software cloud , Ownclown and Linux Ubuntu servers.

Keywords: ownCloud – free software – Command and Control

Lista de figuras

Lista de acrónimos

ARIET……..Aplicación de Recursos Informáticos del Ejército de Tierra

SIGLE……..Sistema Integral de Gestión Logística

SIMACET……Sistema de Mando y Control del Ejército de Tierra

SIMENDEF……Sistema de Mensajería de Defensa

ZO……Zona de Operaciones

Índice

1. Introducción

La siguiente memoria presenta el trabajo realizado para poder instalar y configurar un servicio de almacenaje en la nube, de forma que esta pudiera ser usado por miembros de los Fuerzas Armadas de España, tanto en territorio nacional pero sobre todo en ZO.

Uno de los mayores retos con los que se enfrentan las Fuerzas Armadas, y en concreto las Unidades desplegadas en el extranjero, es la gestión de la información de manera rápida y fiable. Esta gestión de información incluye la recopilación de informes desde personal, medios, rutas o inteligencia hasta reparaciones, municiones, repuestos, combustible o la misma alimentación de las tropas desplegadas.

Para facilitar esta tarea, en este trabajo he implementado un almacenaje en la nube mediante el uso de software libre, teniendo en cuenta las normativas vigentes y siempre dentro de unos marcos de seguridad OTAN. Tratando así de usar un entorno lo más intuitivo y sencillo para el usuario, el cual puede tener acceso a estos recursos desde accesos directos en sus escritorios, ya que sobre los equipos de estos no habría que realizar ningún tipo de instalación ni configuración, realizándose todas las operaciones sobre un servidor Apache y sobre el servidor de dominio de la Intranet militar.

1.1. Motivación y antecedentes

Para todo militar el despliegue en ZO es una de sus facetas más importantes. Igual que un médico estudia y se prepara para poder salvar la vida en caso de extrema urgencia a pacientes o un bombero debe ser capaz de templar el ánimo y resistir a su instinto de supervivencia cuando debe entrar entre las llamas de un incendio, el militar se instruye y se adiestra con su unidad, en un proceso continuo que dura desde que ingresa a formar parte de las Fuerzas Armadas hasta el día de su pase a la reserva, siendo en las misiones en el extranjero donde debe aplicar todos sus conocimientos y experiencia para el cumplimiento de la misión.

La Intranet es la herramienta principal que el militar usa en su día a día. En ella existen multitud de aplicaciones que permiten realizar distintas gestiones o simplemente realizar su trabajo diario. Entre ellas podemos contar con el PORTAL PERSONAL, en el cual podemos encontrar toda la vida militar de un soldado, desde cursos, medallas, destinos, ascensos, nóminas junto a muchas más opciones. También tenemos el SIMENDEF, como sistema de

mensajería interna sin clasificar, por el que realmente se transmiten las órdenes entre unidades y/o la propia unidad. También existen aplicaciones para la gestión de incidencias informáticas ICIS, para la gestión del material informático ARIET o para el mantenimiento, transportes y abastecimiento, que permite solicitar piezas de repuesta y diverso material para de vida y funcionamiento para los vehículos y medios usados por nuestro ejército, conocido como SIGLE.

Lo realmente curioso es que no exista aún un sistema de almacenaje en la nube realizado con software libre, a pesar de lo evidente de su utilidad y del poco gasto económico y de recursos que supondría su implantación y uso por parte de todos los miembros de las Fuerzas Armadas. Sin olvidar por supuesto la seguridad debido a los, cada vez más recientes, distintos ciberataques que somos conocedores gracias a las noticias y por supuesto a muchos más que no llegan a nuestros oídos.

Por lo tanto por estas razones anteriormente indicadas, en este Trabajo Final de Máster pretendo definir el procedimiento adecuado para el montaje de una nube segura para uso de una Intranet OTAN y Sistemas de Mando y Control, un ejemplo de ello sería SIMACET.

Algunas de sus características las cito a continuación:

- Gestión y configuración totalmente transparente al usuario final.
- Cumplimiento de las distintas normas y recomendaciones de seguridad para almacenaje en la nube, pudiendo obtener así certificación OTAN.
- Consulta y uso desde cualquier ZO.
- El acceso se realizará mediante el propio usuario de dominio perteneciente a los miembros de las Fuerzas Armadas.

1.2. Organización de la memoria

Esta memoria se encuentra estructurada en 7 capítulos, los cuales son descritos brevemente a continuación:

Capítulo 1. Introducción

Este capítulo contiene la presentación de la memoria y la justificación de su realización, teniendo en cuenta para ellos los antecedentes actuales en el almacenaje en la nube que usa el Ejército Español.

Capítulo 2. Descripción del proyecto

Este capítulo hace referencia a los objetivos que se pretenden conseguir, como al alcance y las herramientas y software empleados durante la realización de este TFM.

Capítulo 3. Análisis del sistema

Se describen en este capítulo los usuarios del sistema y se describen los requisitos necesarios, tantos funcionales como no funcionales.

Capítulo 4. Normativas en vigor a aplicar

Aquí se encuentra toda la instalación y configuración del servidor y las distintas aplicaciones necesarias para su securización basándome en las guías CCN-STIC que se nombran y en la documentación oficial de éstas, pudiendo consultarse en la bibliografía y anexos..

Capítulo 5. Conclusiones y trabajo futuro

Ya en el capítulo final se exponen las conclusiones extraídas de este Trabajo Final de Máster y se proponen las líneas a seguir para continuar trabajando con este campo. En concreto sería adecuado realizar una auditoría exhaustiva de la seguridad del servidor, siendo esto la continuación de este proyecto.

2. Descripción del proyecto

2.1. Objetivos

Los objetivos a alcanzar al finalizar este trabajo son los siguientes:

- Instalar y configurar un lugar de almacenaje en la nube mediante la aplicación owncloud.
- Securizar los datos almacenados en la nube cumpliendo los estándares exigidos por la OTAN.
- Definir la metodología a usar para una futura implementación de este proyecto en la Intranet de los distintos Ejércitos.

A título personal los objetivos son:

- Familiarizarme con la aplicación ownCloud.
- Asentar los conocimientos adquiridos durante la realización de este máster en el uso de servidores y su securización.
- Adquirir mayor conocimiento sobre las normativas y recomendaciones de seguridad en información exigidas por la OTAN y distintos organismos internacionales.

2.2. Alcance del TFM

Como indiqué anteriormente, si este proyecto fuera finalmente implementado por parte de las Fuerzas Armadas, supondría una nueva herramienta que facilitaría el trabajo realizado por nuestros soldados cuando se encontrasen desplegados en misiones internacionales. Facilitaría toda la gestión, uso y comunicaciones realizadas mediante la Intranet en el extranjero y suponiendo un coste mínimo para las arcas del estado al usar software libre.

Así, por lo tanto a la finalización de este trabajo tendremos:

- Un espacio de almacenamiento en la nube securizado.
- Una metodología actualizada a implementar para su realización y uso, permitiendo el acceso desde redes públicas.
- Una guía como referencia de software totalmente gratuito y seguro para el montaje de un servidor UBUNTU.

2.3. Herramientas empleadas

Este trabajo tiene tanto un enfoque teórico como práctico, por lo tanto he tenido que hacer uso de distintas aplicaciones software que a continuación nombraré y describiré su utilidad:

- **VIRTUALBOX:** es un software de virtualización para arquitecturas x86/amd64 sobre la que instalaremos nuestro servidor Ubuntu
- **UBUNTU 16.04.02 SERVER AMD64:** es el servidor virtualizado en VirtualBox sobre el que se instalará y configurará ownCloud X y todas las herramientas de seguridad necesarias.
- **OWNCLOUD X:** es una aplicación de software libre del tipo Servicio de alojamiento de archivos, que permite el almacenamiento en línea y aplicaciones en línea.
- **GANTPROJECT:** software libre también como los anteriores utilizada para la creación del Diagrama de Gantt.
- **ARGOUML:** herramienta CASE (Computer Aided Software Engineering) de software libre que he utilizado para la especificación de la aplicación SW (o sistema software) con el Lenguaje Unificado de Modelado (UML). En nuestro caso se ha empleado para la creación de los Diagramas de Caso de Uso que especifican los requisitos funcionales de la aplicación (sistema).
- **OMNIGRAFFLE:** herramienta utilizada para la creación de diagramas en general. Se ha utilizado para la creación de la Estructura de Descomposición del Proyecto.

2.4. Estructura de Descomposición del Proyecto

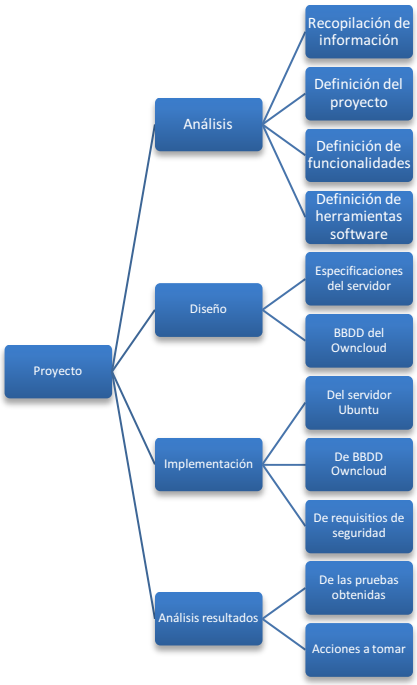


Figura 1: 2.4. Descomposición del Proyecto

2.5. Diagrama de Gant

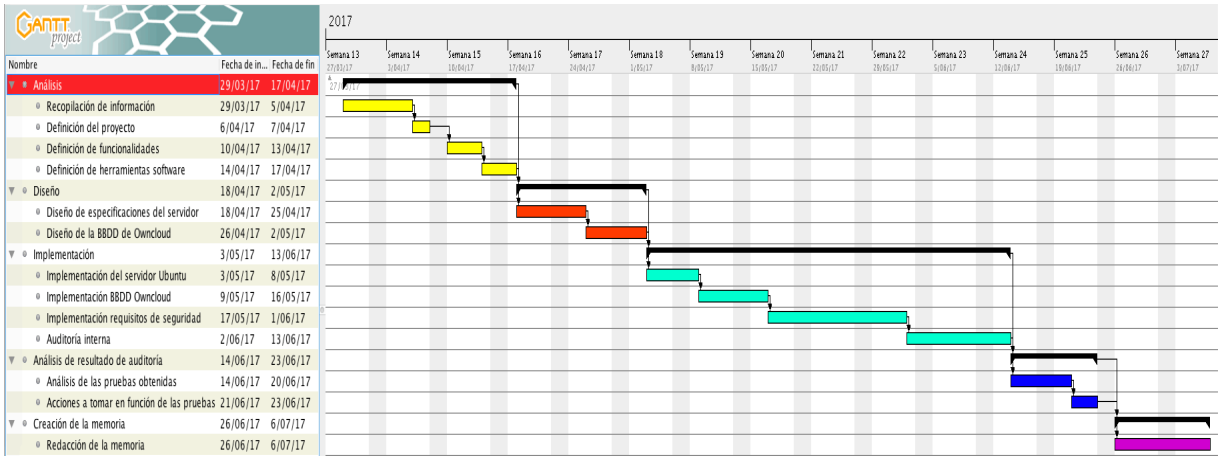


Figura 2: 2.5. Diagrama de Gant del TFM

3. Análisis del sistema

3.1. Requerimientos funcionales

En este apartado se usan los Casos de Uso para explicar los requisitos de los servicios que el servidor **ownCloud** va a proveer, además de las distintas interacciones entre este y su entorno.

3.1.1. Actores

Realmente sólo existe un tipo de actor que interactuará con nuestro servidor **ownCloud**, al que simplemente llamaremos "**User**".

- **User**

  Este tipo de usuario tras loguearse tendrá acceso a los datos almacenados en nuestras carpetas, pudiendo consultar y modificar aquella información para la cual tenga los permisos adecuados.

3.1.2. Diagramas de Casos de Uso

En la siguiente figura se muestra el Diagrama de Casos de Uso en el que puede observarse las interacciones que el usuario **User** puede realizar con el servidor **ownCloud**. Para poder realizar estas interacciones el usuario debe primero loguearse de forma correcta en el sistema.

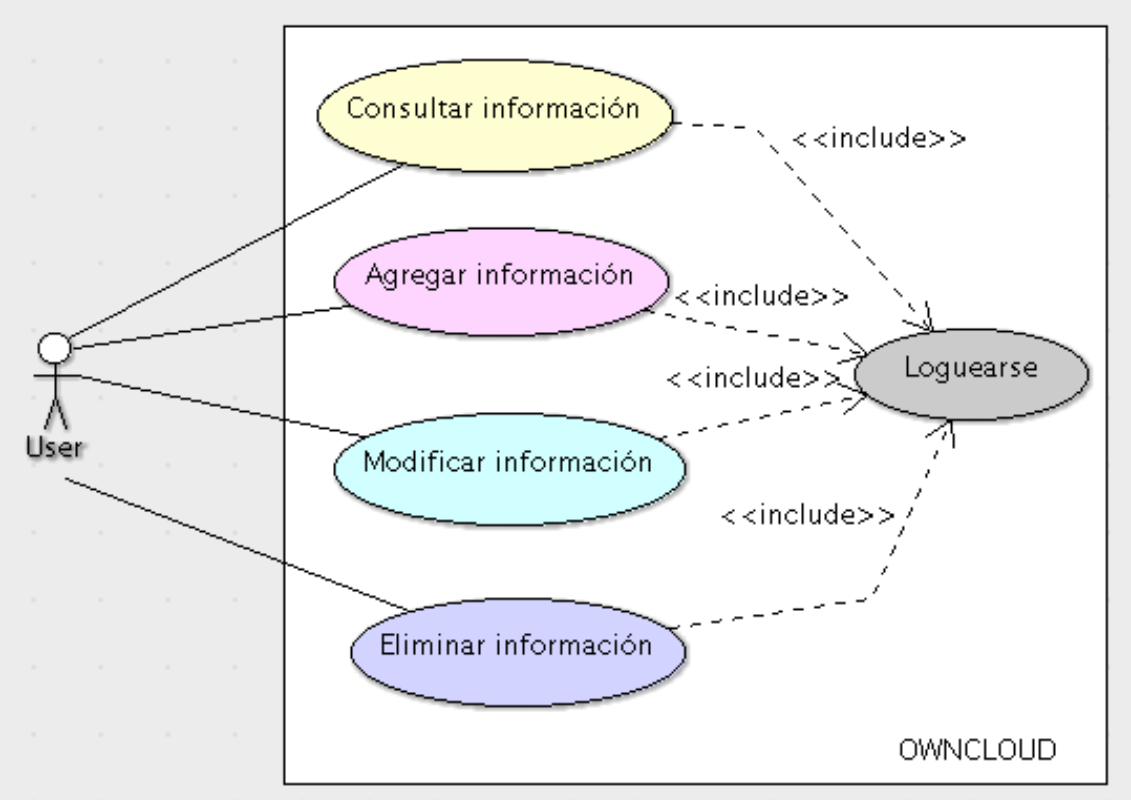


Figura 3: 3.1.2. Diagrama de Casos de Uso

3.1.3. Especificación de Casos de Uso

En este apartado se explica los Casos de Uso del diagrama anterior:

- Caso de Uso: **Loguearse**
  - Definición: proceso en el cual el usuario debe introducir su nombre y clave para poder confirmar su identidad. Se podría considerar más una subfuncionalidad que una funcionalidad, ya que todas las actividades se realizan tras esta.
  - Actores: User
  - Precondiciones: el usuario debe pertenecer a la base de datos.
  - Escenario principal:
    1) El usuario User introduce su nombre y clave.
    2) OwnCloud comprueba que los datos son correctos.
    3) OwnCloud permite el acceso.
  - Escenario alternativo:

    1.a) El usuario User introduce su nombre y una clave incorrecta.

    2.a) Owncloud comprueba que los datos son incorrectos.

3.a) Owncloud no permite el acceso.

4.a) Owncloud mostrará por pantalla que existe un error.

1.b) El usuario User introduce un nombre incorrecto y clave.

2.b) Owncloud comprueba que el usuario no existe en la base de datos.

3.b) Owncloud no permite el acceso.

4.b) Owncloud muestra por pantalla que el usuario es incorrecto.

- Caso de Uso: **Consultar información**
  - Definición: proceso mediante el cual el usuario puede consultar la información existente en el servidor de almacenaje en la nube, siendo esto en función de los privilegios que tenga a las distintas carpetas.
  - Actores: User
  - Precondiciones: el usuario debe estar logueado:
  - Escenario principal:
    1) El usuario User trata de consultar las carpetas que desea.
    2) OwnCloud muestra por pantalla el contenido de las carpetas.
  - Escenario alternativo:
    1) El usuario User trata de consultar las carpetas que desea.
    2) OwnCloud no muestra por pantalla el contenido de las carpetas por falta de privilegios.

- Caso de Uso: **Agregar información**
  - Definición: proceso mediante el cual el usuario puede agregar información nueva en el servidor de almacenaje en la nube, siendo esto en función de los privilegios que tenga a las distintas carpetas.
  - Actores: User
  - Precondiciones: el usuario debe estar logueado:
  - Escenario principal:
    1) El usuario User trata de agregar información a las carpetas que desea.
    2) OwnCloud le permite agregar información de las carpetas.
  - Escenario alternativo:
    1) El usuario User trata de agregar las carpetas que desea.
    2) OwnCloud no le permite agregar información de las carpetas por falta de privilegios.

- Caso de Uso: **Modificar información**
  - Definición: proceso mediante el cual el usuario puede modificar la información existente en el servidor de almacenaje en la nube, siendo esto en función de los privilegios que tenga a las distintas carpetas.
  - Actores: User
  - Precondiciones: el usuario debe estar logueado:
  - Escenario principal:
    1) El usuario User trata de modificar las carpetas que desea.
    2) OwnCloud le permite modificar información de las carpetas.
  - Escenario alternativo:
    1) El usuario User trata de modificar las carpetas que desea.
    2) OwnCloud no le permite modificar información de las carpetas por falta de permisos.

- Caso de Uso: **Eliminar información**
  - Definición: proceso mediante el cual el usuario puede eliminar la información existente en el servidor de almacenaje en la nube, siendo esto en función de los privilegios que tenga a las distintas carpetas.
  - Actores: User
  - Precondiciones: el usuario debe estar logueado:
  - Escenario principal:
    1) El usuario User trata de eliminar las carpetas que desea.
    2) OwnCloud le permite eliminar las carpetas.
  - Escenario alternativo:
    1) El usuario User trata de eliminar las carpetas que desea.
    2) OwnCloud no le permite eliminar información de las carpetas por falta de permisos.

3.2. Requerimientos no funcionales

- Seguridad

  Debido a la información que puede contener el servidor es necesario restringir el acceso mediante una base de datos que contenga a los usuarios. Sus privilegios serán asignados por parte del controlador de dominio (esto no forma parte de este trabajo), así que en todo momento consideramos que están bien asignados por parte del organismo responsable del Ejército.

- Portabilidad

  Esta herramienta además de poder usarse en los navegadores web puede gestionarse y tener acceso mediante aplicaciones para Android y IOS, junto a un cliente de escritorio para Windows, Linux y Mac OS X.

- Usabilidad

  La sencillez y intuición son siempre importantes para que el usuario pueda desenvolverse con soltura durante todo el tiempo que haga uso de ownCloud.

- Escalabilidad

  El proyecto tendrá un base de datos dedicada de usuarios limitada, pero esta puede ser aumentada según las necesidades de uso. Esta base de datos es sencilla de implementar y gestionar pues se realiza por SQL, permitiendo por lo tanto el manejo de multitud de bases de datos e información.

- Mantenimiento

  Las medidas de securización se podrán ir ampliando o modificando según los requerimientos en cada situación (nuevas carpetas, nuevos usuarios, nuevas normativas…).

4. Implementación mediante normativas CNN-STIC en vigor a aplicar

En este apartado se realiza la implementación completa de nuestro servidor, usando como referencia las guías CCN-STIC en vigor, de hecho cada capítulo se encuentra estructura según la guía empleada. Por supuesto hay pasos e información no extraída de estas guías, como puede ser la instalación y configuración de ownCloud, pero se ha incluido en un capítulo concreto para seguir el orden lógico necesario de implementación de todo el sistema. También existe información de estas guías que son recomendaciones y estudios por lo que veremos que en la implementación tampoco aparecen aunque se han tenido en cuenta. Las guías usadas son las siguientes:

- 430 – Herramientas de seguridad
- 431 – Herramientas de análisis de vulnerabilidades
- 612 – Seguridad en Debian
- 660 – Securización de proxies
- 661 – Securización en firewalls de aplicación web (Modsecurity)
- 665 – Configuración segura de SSH
- 671 – Configuración segura de Servidores Web Apache
- 820 – Protección contra Denegación de Servicio
- 823 – Utilización de servicios en la nube

Además para este trabajo se ha instalado una máquina virtual de Ubuntu Server mediante VirtualBox. Para ello las especificaciones que se le dieron fueron básicos, pues es un entorno de prueba. Como es lógico para dar servicios a un gran organismo estas especificaciones deberían ser mucho mayor. Las usadas fueron:

Sistema Operativo: Ubuntu Server 16.04.02 AMD64.

Número de procesadores: 1 procesador.

Memoria RAM: 4098 Mb.

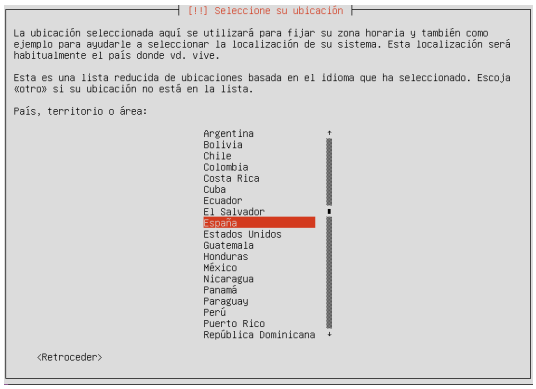
Almacenamiento: 30 Gb.

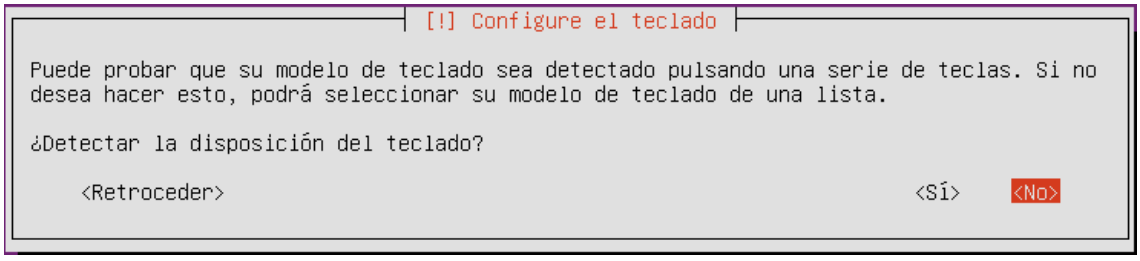
Red: adaptador puente (para las actualizaciones e instalaciones necesarias desde internet y acceso exterior a nuestro servidor). Se le configurará un IP fija.

4.1. 612 – Configuración segura de Debian y 671 – Configuración segura de servidores web Apache

4.1.1. Instalación de un servidor UBUNTU LAMP

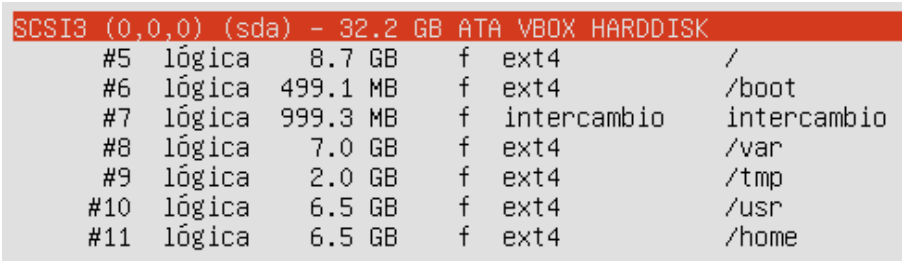
Una vez comenzamos una instalación estándar de nuestro Linux Server, lo primero que nos solicitará será el idioma para la instalación y el del teclado.



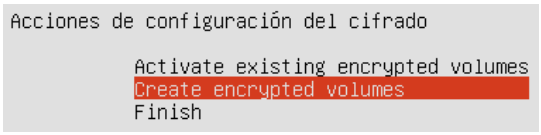


Tras esto comenzará la instalación de diversos componentes, la detección de hardware y la configuración de la red. Indicaremos el nombre que tendrá nuestro servidor, el usuario y su clave. Después de esto nos solicitará si queremos cifrar nuestra carpeta, así que por seguridad confirmaremos e introduciremos una nueva clave.

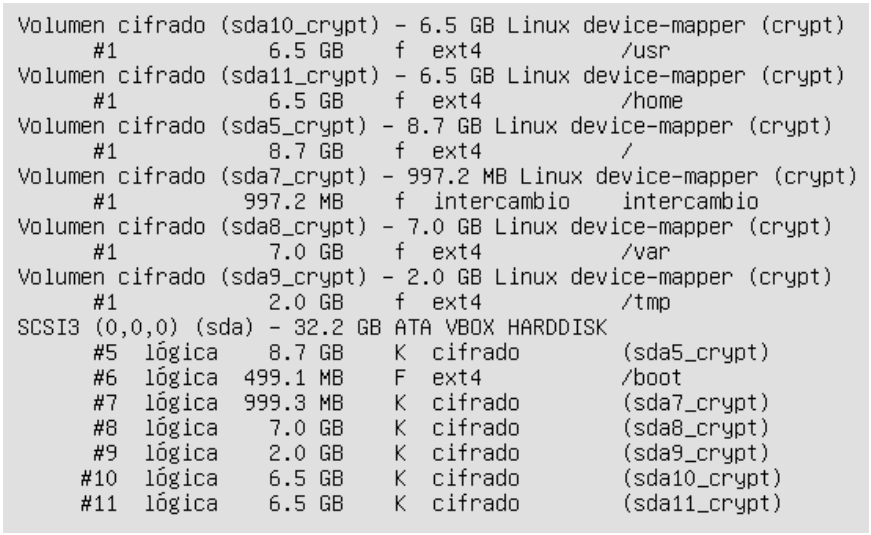
Lo siguiente será la configuración del disco duro. Para la mayor seguridad posible lo haremos de forma Manual, separando en varias carpetas que además cifraremos, quedando así.



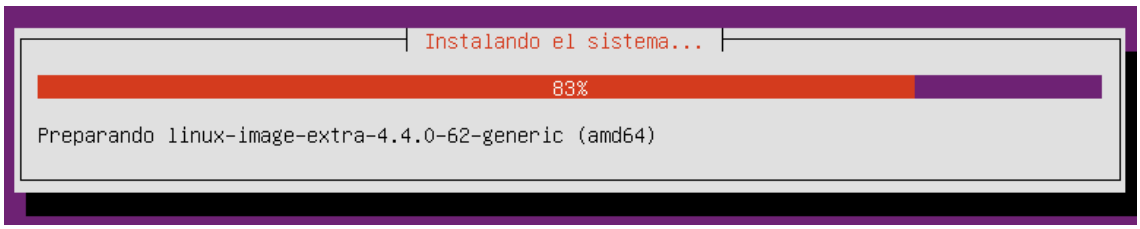
Y tras la configuración de las particiones lógicas como hemos visto anteriormente, procederemos a su cifrado desde la misma ventana, marcando todas ellas.



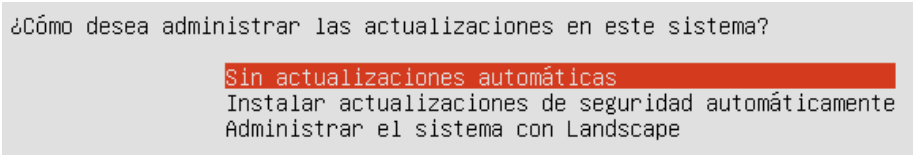
Y por lo tanto quedando finalmente como se muestra.



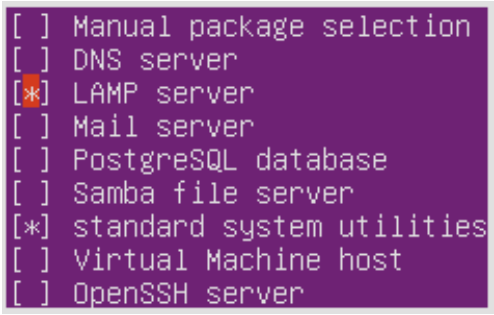
Continuaremos con la instalación eligiendo siguiente en el resto de ventanas hasta que comience la instalación del sistema.



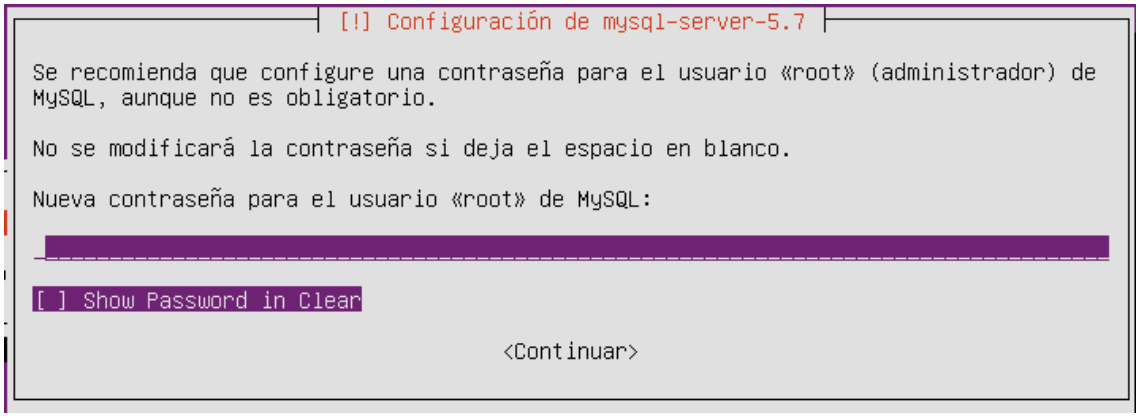
A lo largo de esta nos pedirá que configuraremos el proxy, opción de la cual pasaremos. En el caso de actualizaciones automáticas si la configuraremos de forma positiva.



En un determinado momento nos solicitará que herramientas queremos instalar siendo aquí donde deberemos marcar LAMP server tal como se puede ver en la imagen a continuación.



Al seleccionar que instale LAMP, será necesario introducir la clave para el usuario “root” de MySQL (también se usara para phpMyAdmin).



Al final de la instalación nos preguntará si queremos instalar el gestor de arranque Grub, el cual nos resultará útil si tenemos varios sistemas operativos instalados en nuestra máquina.

En esta edición de Ubuntu ya cuenta con las librerías necesarias para poder utilizar PHP de manera correcta así que no hará falta instalarlas pero si actualizar los repositorios introduciendo por teclado con usuario root lo indicado.

```
# apg-get upgrade
```

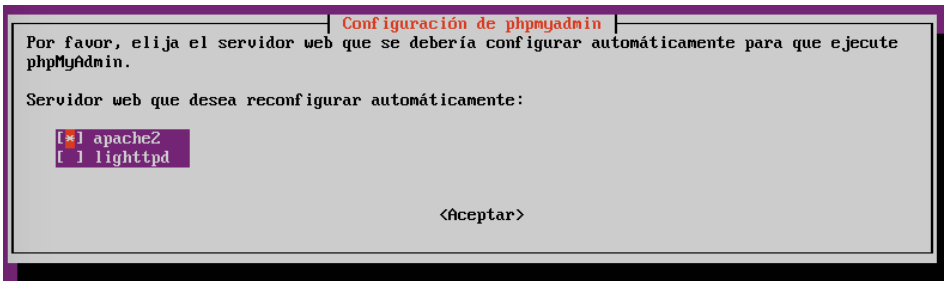
Y tras esto actualizaremos nuestro sistema servidor:

```
# apg-get update
```

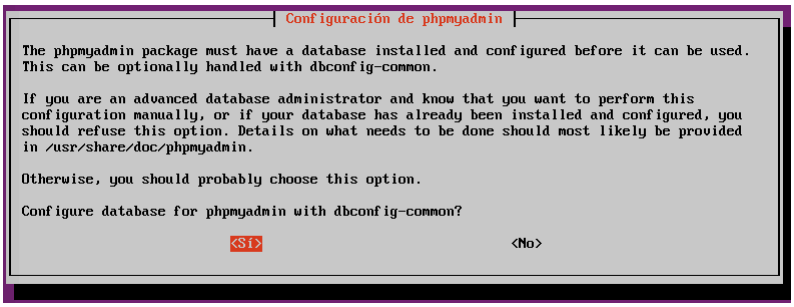
Ahora tocará instalar PhpMyAdmin para poder gestionar de manera más sencilla nuestras bases de datos MySQL.

```
# apg-get isntall phpmyadmin
```

Y en la ventana que nos saldrá deberemos elegir nuestro servidor anteriormente instalado apache2 y pulsamos ENTER.



Seguidamente nos mostrará una ventana que nos indica la necesidad de tener una base de datos instalada y configurada, pues sin ella no podremos usar **PhpMyAdmin**. Por lo tanto seleccionamos la opción SI y pulsamos ENTER, lo cual hará que se instale y configure de forma automática.



Tras estos procesos toca comprobar que nuestro servidor funciona correctamente para lo que crearemos una pequeña página en la que incluiremos un script PHP. Este archivo deberemos crearlo en la ruta que se indica a continuación, respecto al nombre puede se cualquiera pero con la extensión PHP.

```
# nano /var/www/html/info.php
```

El texto que escribiremos en dicho archivo una vez abierto el editor de texto será algo como el que se muestra.

```
  GNU nano 2.5.3        Archivo: /var/www/html/info.php

<html>
<body>
<h1>Página prueba funcionamiento PHP</h1>
<?php
phpinfo();
?>
</body>
</html>
```

Ahora desde nuestro navegador trataremos de acceder, escribiendo la siguiente URL donde **localhost** debe corresponder a la IP de nuestro servidor (a no ser que configuremos el archivo hosts en la ruta **/etc/** añadiendo como línea nuestra IP y **localhost** a continuación).

http://localhost/info.php

Y deberíamos poder ver una ventana similar a la siguiente. En caso que no se pueda se debe cambiar el grupo y propietario de **info.php** mediante **chown www-data:www-data <ruta>/info.php** y ajustar sus permisos **chmod 644 <ruta>/info.php** para que pueda ser leído por cualquier usuario.

Página prueba funcionamiento PHP

PHP Version 7.0.18-0ubuntu0.16.04.1

System	Linux tfm 4.4.0-62-generic #83-Ubuntu SMP Wed Jan 18 14:10:15 UTC 2017 x86_64
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/7.0/apache2
Loaded Configuration File	/etc/php/7.0/apache2/php.ini
Scan this dir for additional .ini files	/etc/php/7.0/apache2/conf.d
Additional .ini files parsed	/etc/php/7.0/apache2/conf.d/10-mysqlnd.ini, /etc/php/7.0/apache2/conf.d/10-opcache.ini, /etc/php/7.0/apache2/conf.d/10-pdo.ini, /etc/php/7.0/apache2/conf.d/15-xml.ini, /etc/php/7.0/apache2/conf.d/20-calendar.ini, /etc/php/7.0/apache2/conf.d/20-ctype.ini, /etc/php/7.0/apache2/conf.d/20-dom.ini, /etc/php/7.0/apache2/conf.d/20-exif.ini, /etc/php/7.0/apache2/conf.d/20-fileinfo.ini, /etc/php/7.0/apache2/conf.d/20-ftp.ini, /etc/php/7.0/apache2/conf.d/20-gd.ini, /etc/php/7.0/apache2/conf.d/20-gettext.ini, /etc/php/7.0/apache2/conf.d/20-iconv.ini, /etc/php/7.0/apache2/conf.d/20-json.ini, /etc/php/7.0/apache2/conf.d/20-mbstring.ini, /etc/php/7.0/apache2/conf.d/20-mcrypt.ini, /etc/php/7.0/apache2/conf.d/20-mysqli.ini, /etc/php/7.0/apache2/conf.d/20-pdo_mysql.ini, /etc/php/7.0/apache2/conf.d/20-phar.ini, /etc/php/7.0/apache2/conf.d/20-posix.ini, /etc/php/7.0/apache2/conf.d/20-readline.ini, /etc/php/7.0/apache2/conf.d/20-shmop.ini, /etc/php/7.0/apache2/conf.d/20-simplexml.ini, /etc/php/7.0/apache2/conf.d/20-sockets.ini, /etc/php/7.0/apache2/conf.d/20-sysvmsg.ini, /etc/php/7.0/apache2/conf.d/20-sysvsem.ini, /etc/php/7.0/apache2/conf.d/20-sysvshm.ini, /etc/php/7.0/apache2/conf.d/20-tokenizer.ini, /etc/php/7.0/apache2/conf.d/20-wddx.ini, /etc/php/7.0/apache2/conf.d/20-xmlreader.ini, /etc/php/7.0/apache2/conf.d/20-xmlwriter.ini, /etc/php/7.0/apache2/conf.d/20-xsl.ini
PHP API	20151012
PHP Extension	20151012
Zend Extension	320151012
Zend Extension Build	API320151012,NTS
PHP Extension Build	API20151012,NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	disabled
Zend Memory Manager	enabled
Zend Multibyte Support	provided by mbstring

Puesto que el script **phpinfo()** nos muestra información sobre el PHP instalado. Y haremos lo mismo con **PhpMyAdmin**.

http://localhost/phpmyadmin/

CONFIGURACIÓN DE UNA IP ESTÁTICA

Para disponer de acceso exterior a nuestro servidor ownCloud se hace necesario una IP fija o estática, básicamente por dos motivos:

1. La IP de nuestro servidor puede ser cualquiera, pues cada vez que lo arrancamos se produce una asignación dinámica. Si no disponemos de una IP fija no podemos garantizar el acceso permanente a este.
2. La IP fija nos facilita la conexión a cualquier equipo de nuestra red, pues los tenemos individualmente identificados.

Por seguridad sería conveniente realizar una copia de seguridad del fichero **/etc/network/interfaces**, por lo que escribiremos en nuestro terminal lo siguiente:

```
# cp /etc/network/interfaces /etc/network/interfaces.bak
```

Tras realizar la la copia de seguridad podemos empezar con la configuración de nuestra IP fija.

Los pasos a seguir serían los siguientes:

1. Necesitamos saber cual es la interfaz que usa nuestro servidor así que mediante la orden **sudo ifconfig –a** sabremos cual es.

```
tfm@tfm:~$ sudo ifconfig -a
[sudo] password for tfm:
enp0s3    Link encap:Ethernet  direcciónHW 08:00:27:2f:71:2a
          Direc. inet:192.168.0.20  Difus.:192.168.0.255  Másc:255.255.255.0
          Dirección inet6: fe80::a00:27ff:fe2f:712a/64 Alcance:Enlace
          ACTIVO DIFUSIÓN FUNCIONANDO MULTICAST  MTU:1500  Métrica:1
          Paquetes RX:4256 errores:0 perdidos:0 overruns:0 frame:0
          Paquetes TX:96 errores:0 perdidos:0 overruns:0 carrier:0
          colisiones:0 long.colaTX:1000
          Bytes RX:1110275 (1.1 MB)  TX bytes:9473 (9.4 KB)

lo        Link encap:Bucle local
          Direc. inet:127.0.0.1  Másc:255.0.0.0
          Dirección inet6: ::1/128 Alcance:Anfitrión
          ACTIVO BUCLE FUNCIONANDO  MTU:65536  Métrica:1
          Paquetes RX:170 errores:0 perdidos:0 overruns:0 frame:0
          Paquetes TX:170 errores:0 perdidos:0 overruns:0 carrier:0
          colisiones:0 long.colaTX:1
          Bytes RX:12418 (12.4 KB)  TX bytes:12418 (12.4 KB)
```

2. Una vez realizado, tenemos la información que necesitamos (la marcada con el recuadro rojo) para seguir con la configuración de nuestra red con IP fija. Tecleamos en la terminal lo siguiente:

```
# nano /etc/network/interfaces
```

3. Ya en el editor de textos tendremos que reemplazar el contenido existente por el que se muestra en la siguiente captura de pantalla.

```
  GNU nano 2.5.3   Archivo: /etc/network/interfaces

# This file describes the network interfaces available on your s$
# and how to activate them. For more information, see interfaces$

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto enp0s3
iface enp0s3 inet static
        address 192.168.0.20
        netmask 255.255.255.0
        network 192.168.0.0
        broadcast 192.168.0.255
        gateway 192.168.0.1
        dns-nameservers 192.168.0.1
```

Una breve explicación de cada uno de los parámetros que se muestran en la captura de pantalla es mostrado a continuación:

"**auto enp0s3**": Inicia la interfaz **enp0s3** durante la secuencia de arranque del ordenador. Esta interfaz es la que ya vimos al analizar la configuración de red de nuestro servidor.

"**iface enp0s3 inet static**": Con este comando lo que estamos indicando es que una vez levantada la interfaz **enp0s3** se asigne una IP fija o estática del tipo IPV4 a nuestro ordenador.

"**address: 192.168.0.20**": Se debe pone la IP que quiero queremos se asigne al servidor como IP fija o estática, que aunque puede ser cualquiera dentro de nuestra red le podemos asignar directamente la que ya trae por defecto.

"**netmask: 255.255.255.0": ** Aquí la máscara que corresponde a nuestra red, como en este caso es tipo C su máscara es la indicada. Para redes mayores usaremos de tipo B o incluso A.

"**network: 192.168.0.0**": La red en la que se encuentra nuestro servidor.

"**broadcast: 192.168.0.255**": Esta dirección se podrá usar para comunicarse y enviar paquetes a la totalidad de equipos que forman parte de una misma red. La dirección broadcast es la dirección más alta de la red.

"**gateway: 192.168.0.1**": Corresponde a la dirección de IP de nuestro router.

"**dns-nameservers: 192.168.0.1**:" También usaremos la IP de nuestro router, así será el encargado de resolver las peticiones DNS de nuestro ordenador sean resultas mediante los DNS de mi ISP. Podríamos usar por ejemplo los DNS de google, reemplazándola por 8.8.8.8.

4. Y por último aplicamos los cambios a la configuración, reiniciando nuestra red.

```
# service networking restart
```

4.1.2. Instalación de ownCloud

Una vez instalado y configurado nuestro servidor procederemos a instalar ownCloud desde la terminal del servidor. Todos las instrucciones deben escribirse como **root** o incluyendo **sudo** delante de estas.

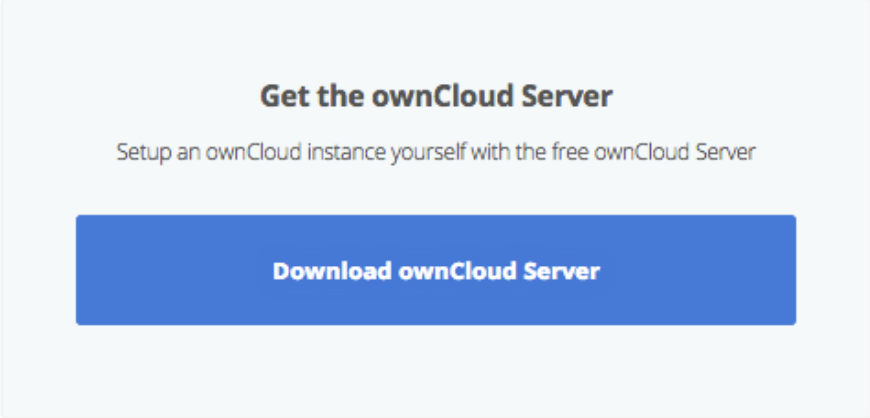
Para empezar creamos una carpeta con el nombre de **owncloud** y nos trasladamos a ella.

```
# mkdir /home/owncloud

# cd /home/owncloud
```

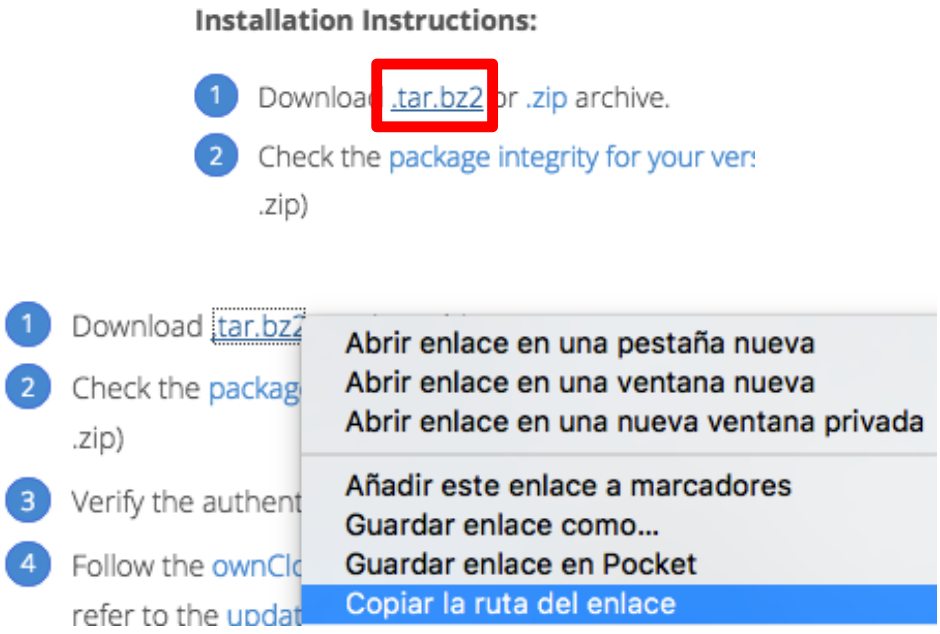
Para ello necesitamos saber la URL exacta de descarga, para lo cual deberemos ir con nuestro navegador a la página oficial de descarga, siendo actualmente la que se muestra.

https://owncloud.org/install/#instructions-server

Pulsamos sobre **Download ownCloud Server**.



Una vez en la siguiente página pondremos nuestro ratón encima de **.tar.bz2** y pulsando segundo botón, elegimos la opción **Copiar la ruta del enlace**.



Ahora ya si, situándonos en el terminal deberemos escribir **wget** y pegar la ruta que hemos copiado.

```
# wget https://download.owncloud.org/community/owncloud-10.0.2.tar.bz2
```

```
tfm@tfm:/home/owncloud$ sudo wget https://download.owncloud.org/community/owncloud-10.0.2.tar.bz2
[sudo] password for tfm:
--2017-07-07 17:40:36--  https://download.owncloud.org/community/owncloud-10.0.2.tar.bz2
Resolviendo download.owncloud.org (download.owncloud.org)... 46.4.80.187, 136.243.124.46, 138.201.139.178, ...
Conectando con download.owncloud.org (download.owncloud.org)[46.4.80.187]:443... conectado.
Petición HTTP enviada, esperando respuesta... 200 OK
Longitud: 30963258 (30M) [application/x-bzip2]
Grabando a: “owncloud-10.0.2.tar.bz2”

owncloud-10.0.2. 100%[=======>]  29,53M  4,76MB/s    in 5,9s

2017-07-07 17:40:42 (5,04 MB/s) - “owncloud-10.0.2.tar.bz2” guardado [30963258/30963258]
```

Lo siguiente será descomprimir el archivo descargado.

```
# tar –xvf owncloud-10.0.2.tar.bz2
```

Y ahora movemos ownCloud a nuestro servidor web, finalizando así la instalación.

```
# mv owncloud /var/www/html/
```

CONFIGURACIÓN DE OWNCLOUD

Para que el funcionamiento de ownCloud sea correcto tendremos que otorgar los permisos adecuados a las distintas carpetas donde se encuentre instalado. Comenzaremos con la carpeta **/var/www/html/owncloud,** dando permiso total a su propietario y al resto solo de lectura y ejecución mediante **0755**.

```
# chmod 0755 /var/www/html/owncloud –R
```

Tras esto asignamos un grupo y un usuario a la carpeta de **owncloud,** tanto el grupo como usuario se llaman **www-data.**

```
# chown –R www-data:www-data  /var/www/html/owncloud/
```

Después de todo esto necesitamos una carpeta para almacenar los datos de los usuarios. Por seguridad esta carpeta estará en un ruta distinta a la usada para la instalación. En nuestro caso creamos la nueva carpeta **/home/owncloudddatos**

```
# mkdir /home/owncloudddatos
```

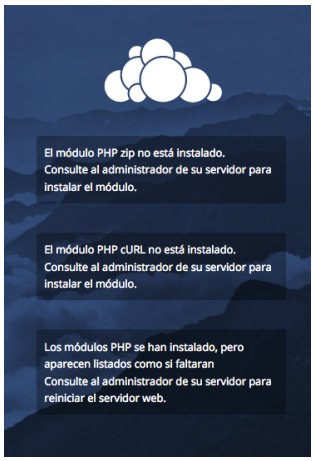
Y de nuevo le damos los permisos adecuados.

```
# chmod 0755 /home/owncloudddatos –R
```

E igual para el grupo y usuario.

```
# chown –R www-data:www-data  /home/owncloudddatos/
```

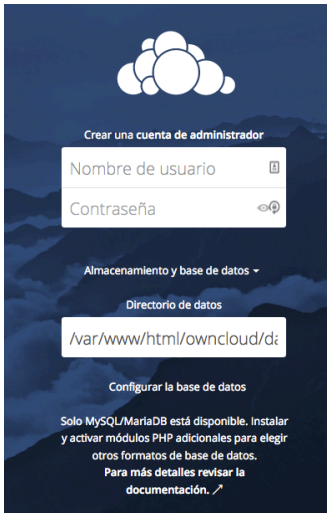
Ahora podremos comprobar si nuestro servidor ownCloud está funcionando, escribiendo en nuestro navegador la IP de Apache **y /owncloud/**

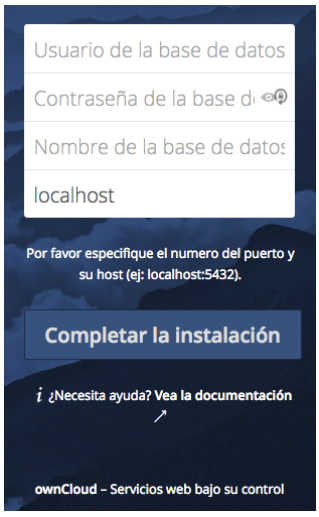


Como podemos ver tenemos un fallo pues nos faltan dos módulos por instalar, así que procederemos de la siguiente manera.

```
# apt-get install php7.0-zip php7.0-curl
```

Y por lo tanto ya vemos que nos funciona, pero aún nos falta algo para completar la instalación.

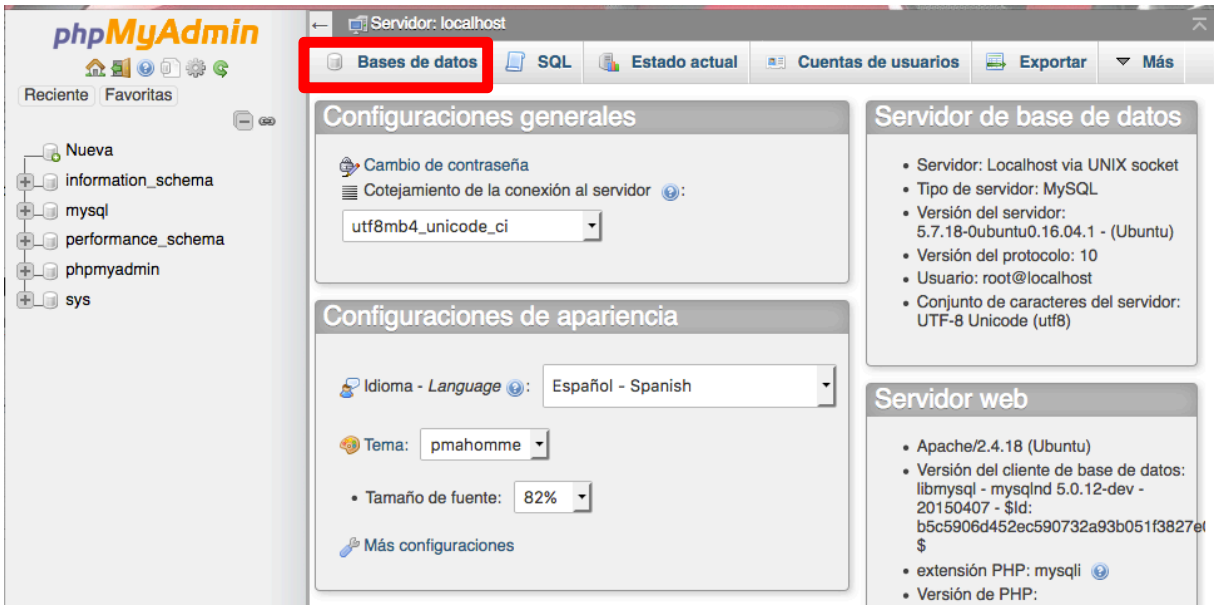


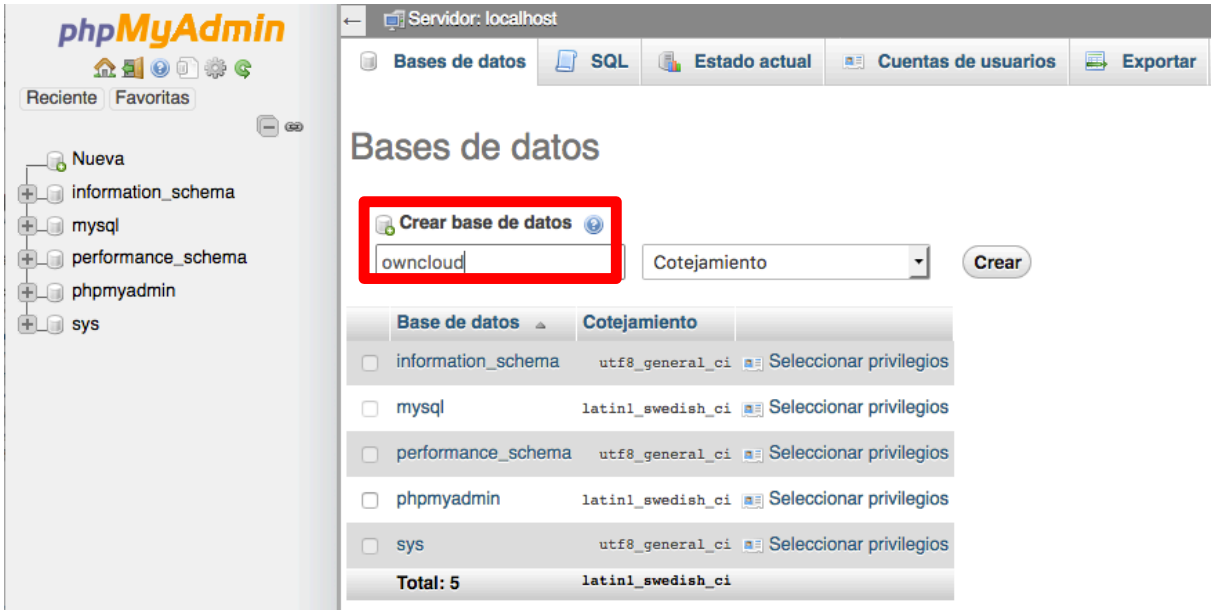


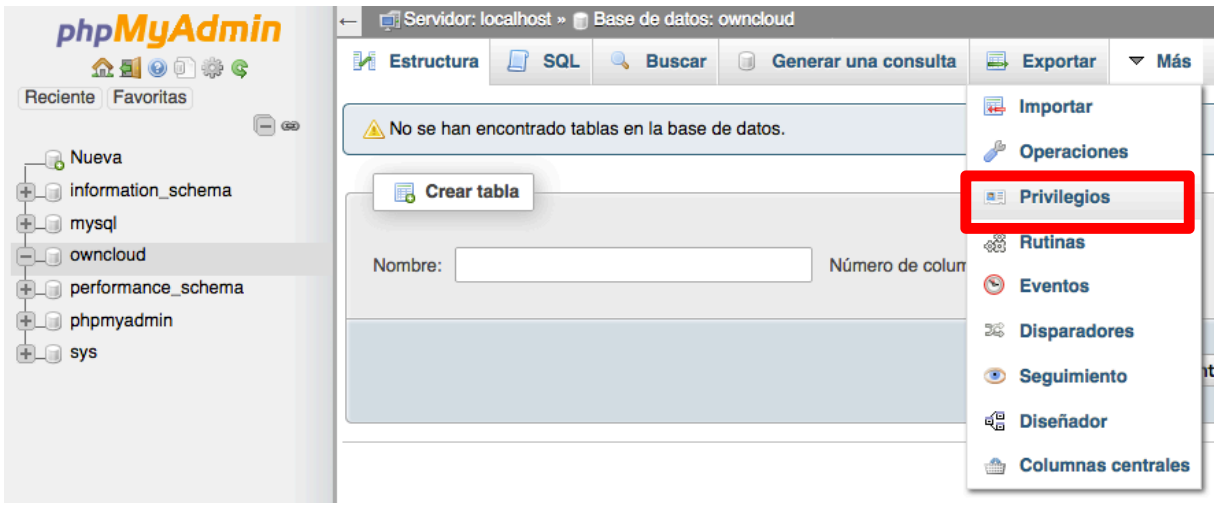
CONFIGURACIÓN DE LA BBDD PARA OWNCLOUD

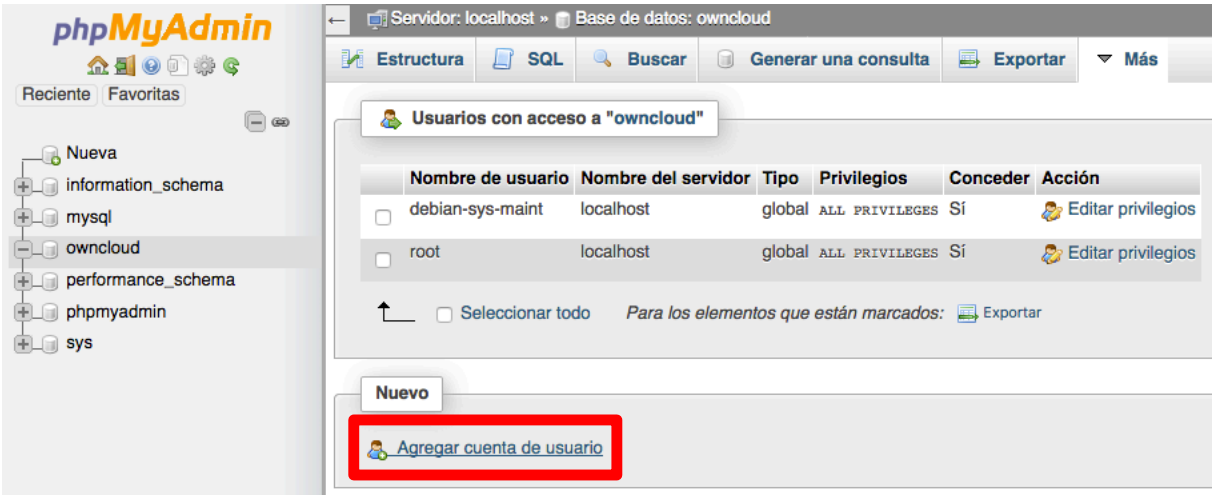
Para finalizar la instalación y configuración es necesario crear un usuario para la BBDD de MySQL. Esto lo haremos de forma visual mediante phpMyAdmin.

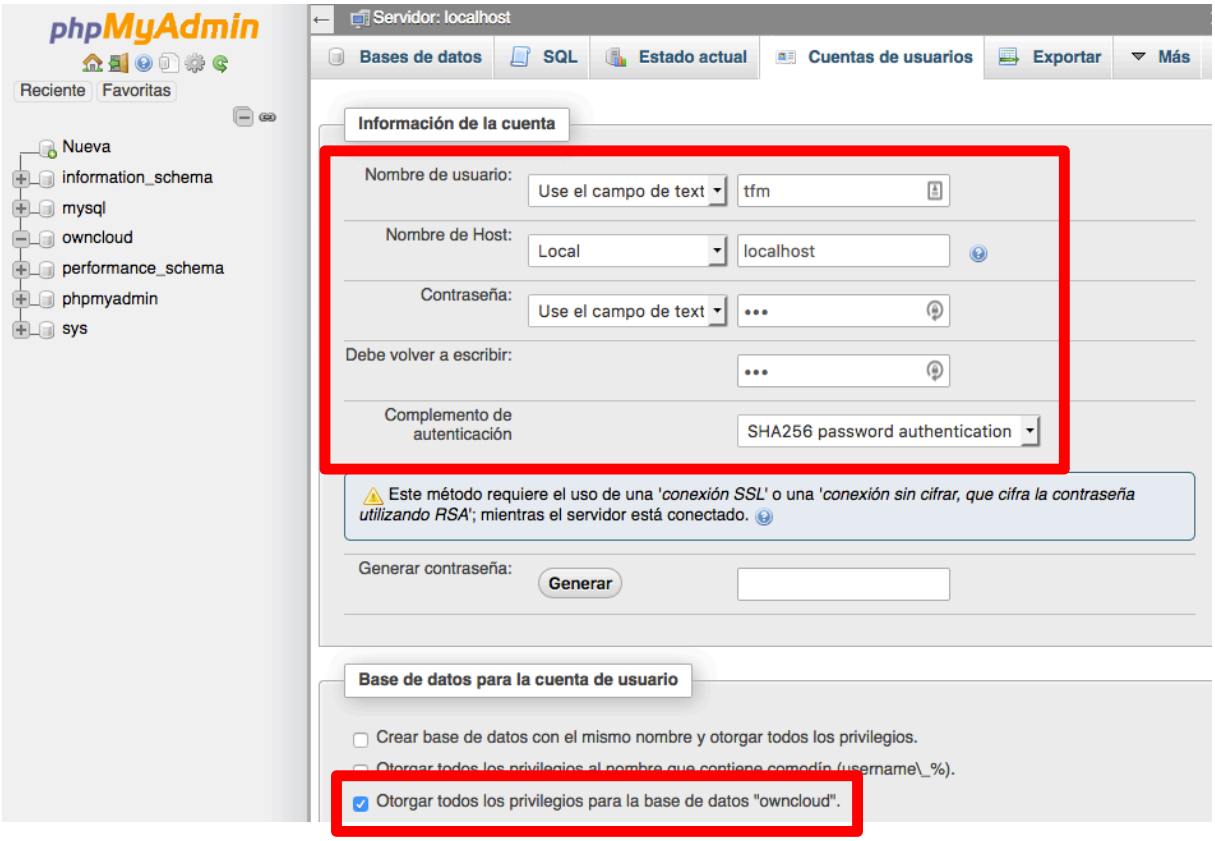
http://localhost/phpmyadmin/




phpMyAdmin
Reciente
Favoritas
Nueva
information_schema
mysql
performance_schema
phpmyadmin
sys
Servidor: localhost
Bases de datos
SQL
Estado actual
Cuentas de usuarios
Exportar
Bases de datos
Crear base de datos
owncloud
Cotejamiento
Crear
Base de datos
Cotejamiento
information_schema utf8_general_ci Seleccionar privilegios
mysql latin1_swedish_ci Seleccionar privilegios
performance_schema utf8_general_ci Seleccionar privilegios
phpmyadmin latin1_swedish_ci Seleccionar privilegios
sys utf8_general_ci Seleccionar privilegios
Total: 5 latin1_swedish_ci


phpMyAdmin
Reciente
Favoritas
Nueva
information_schema
mysql
owncloud
performance_schema
phpmyadmin
sys
Servidor: localhost » Base de datos: owncloud
Estructura
SQL
Buscar
Generar una consulta
Exportar
Más
No se han encontrado tablas en la base de datos.
Crear tabla
Nombre:
Importar
Operaciones
Privilegios
Rutinas
Eventos
Disparadores
Seguimiento
Diseñador
Columnas centrales


phpMyAdmin
Reciente
Favoritas
Nueva
information_schema
mysql
owncloud
performance_schema
phpmyadmin
sys
Servidor: localhost » Base de datos: owncloud
Estructura
SQL
Buscar
Generar una consulta
Exportar
Más
Usuarios con acceso a "owncloud"
Nombre de usuario
Nombre del servidor
Tipo
Privilegios
Conceder
Acción
debian-sys-maint
localhost
global
ALL PRIVILEGES
Sí
Editar privilegios
root
localhost
global
ALL PRIVILEGES
Sí
Editar privilegios
Seleccionar todo
Para los elementos que están marcados:
Exportar
Nuevo
Agregar cuenta de usuario


phpMyAdmin
Reciente
Favoritas
Nueva
information_schema
mysql
owncloud
performance_schema
phpmyadmin
sys
Servidor: localhost
Bases de datos
SQL
Estado actual
Cuentas de usuarios
Exportar
Más
Información de la cuenta
Nombre de usuario:
Use el campo de text
tfm
Nombre de Host:
Local
localhost
Contraseña:
Use el campo de text
Debe volver a escribir:
Complemento de autenticación
SHA256 password authentication
Este método requiere el uso de una 'conexión SSL' o una 'conexión sin cifrar, que cifra la contraseña utilizando RSA'; mientras el servidor está conectado.
Generar contraseña:
Generar
Base de datos para la cuenta de usuario
Crear base de datos con el mismo nombre y otorgar todos los privilegios.
Otorgar todos los privilegios para la base de datos "owncloud".

Usuarios con acceso a "owncloud"

	Nombre de usuario	Nombre del servidor	Tipo	Privilegios	Conceder	Acción
☐	debian-sys-maint	localhost	global	ALL PRIVILEGES	Sí	Editar privilegios
☐	root	localhost	global	ALL PRIVILEGES	Sí	Editar privilegios
☐	tfm	localhost	específico para la base de datos	ALL PRIVILEGES	No	Editar privilegios

Y por último yéndonos de nuevo a la página inicial de ownCloud y rellenando los datos que nos solicita para poder crear la cuenta de administración:

- Crear una cuenta de administrador:
  - Nombre de usuario: usuario administrador para ownCloud. He usado **rootown**.
  - Contraseña: la clave para nuestro usuario administrador de ownCloud.
- Directorio de datos: la dirección creada, en nuestro caso **/home/owncloudddatos/**
- Usuario de la base de datos: el usuario que hemos creado mediante phpMyAdmin **tfm**.
- Contraseña de la base de datos: la que hayamos puesto.
- Localhost: lo dejamos tal cual está y pondremos además el puerto que usa. Para ello haremos una consulta SQL.

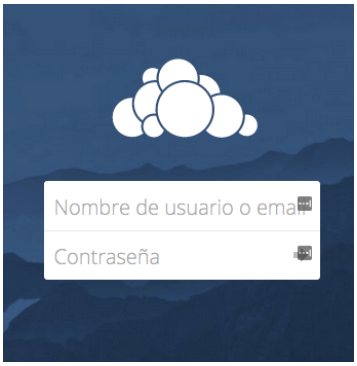
Ejecute la o las consultas SQL en el servidor "localhost":

```
1 SHOW VARIABLES WHERE Variable_name IN ('hostname','port')
```

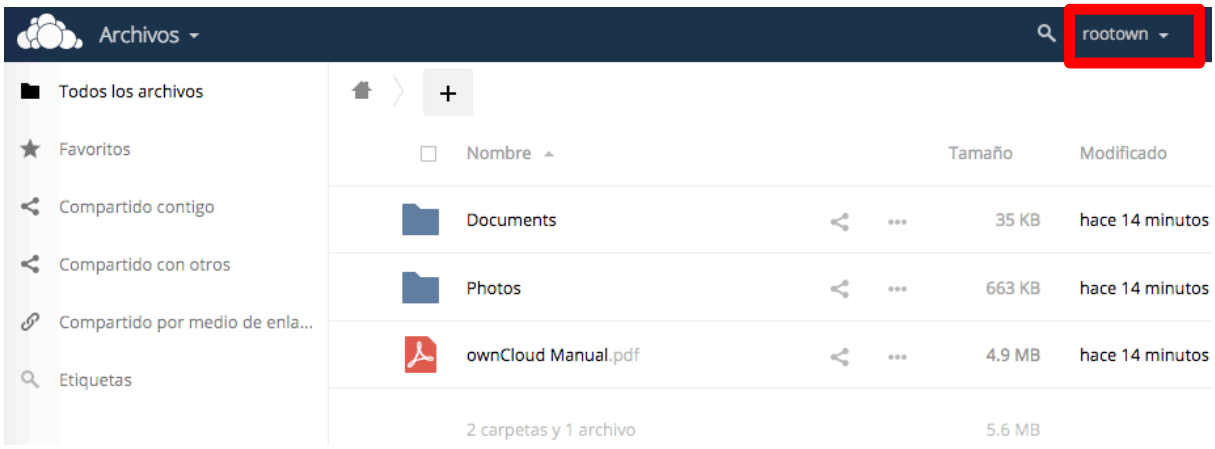
Y como resultado deberíamos obtener algo así.

Variable_name	Value
hostname	tfm
port	3306

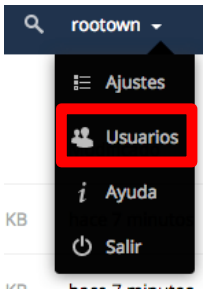
Por lo tanto una vez finalizado podremos ver la ventana de entrada para los usuarios. La primera vez lo haremos con el usuario administrador que hayamos creado.



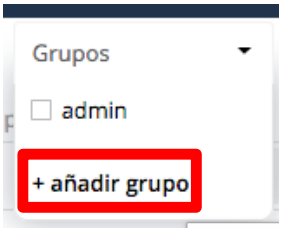
Y una vez introducimos un usuario y su clave que ya existen en la base de datos **ownCloud** pasaremos a sus carpetas en la nube.

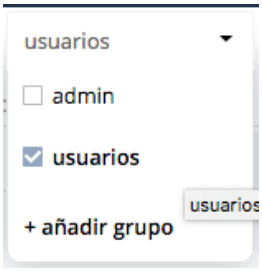


Ahora crearemos algún usuario. Así que pulsando en la esquina superior derecha sobre nuestro administrador **rootown** se desplegará un nuevo menú.



A continuación pulsamos sobre usuarios y pasaremos a una nueva ventana. En ésta deberemos pulsar sobre **Grupos** y luego **+añadir grupo** para crear uno propio que llamaremos usuarios.

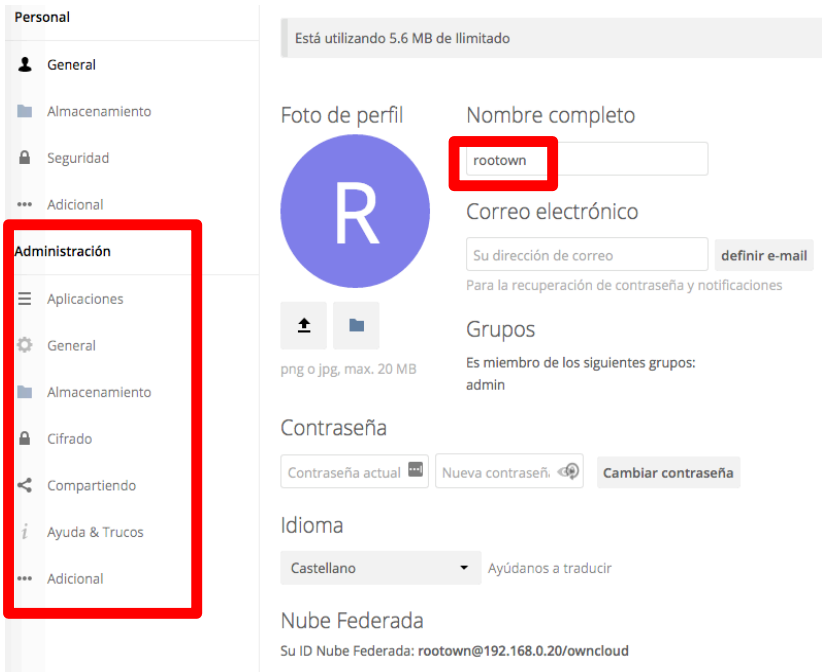


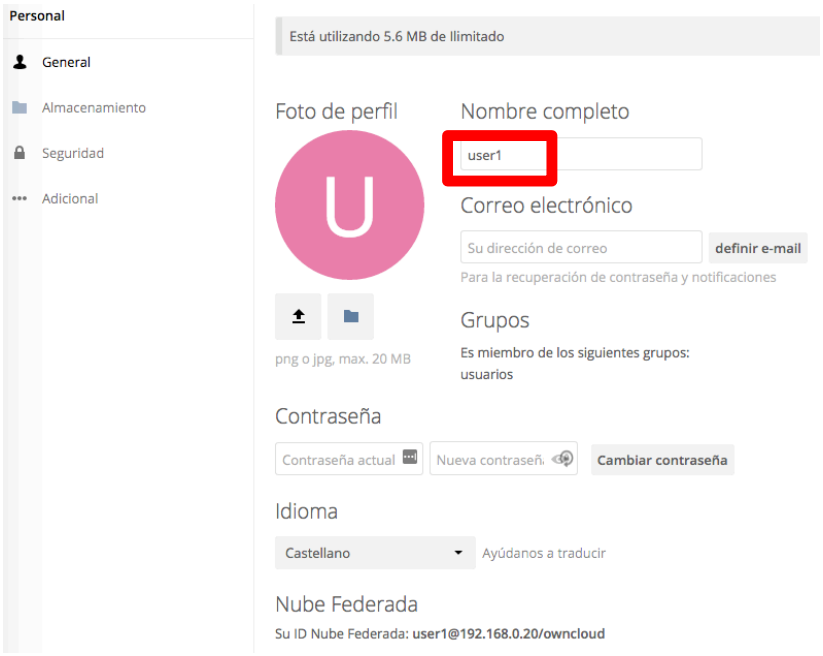


Ahora sí podremos crear nuestros usuarios tal como se muestra.

Y veremos que tras crearlos nos aparecerán en el listado de usuarios que podemos ver.

Entrando ya en la opción de **Ajustes** de cada usuario vemos una clara diferencia entre el usuario **rootown** y los del grupo **usuarios.**

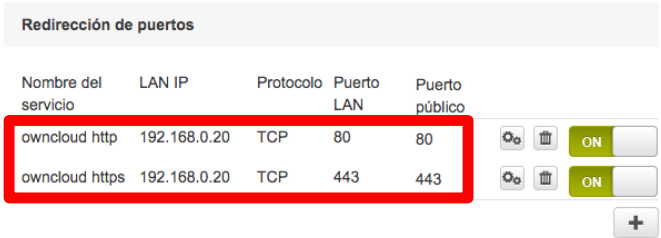




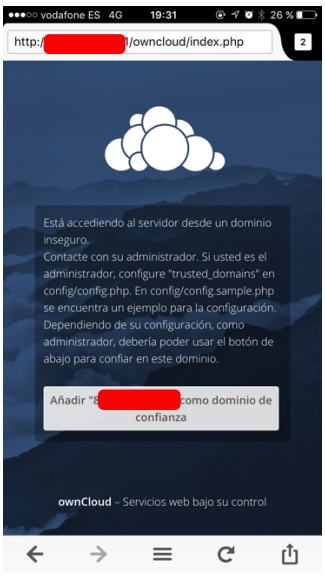
CONFIGURACIÓN DEL ROUTER PARA PERMITIR ACCESO EXTERIOR AL SERVIDOR OWNCLOUD

Lo siguiente será permitir los accesos desde el exterior a nuestro servidor tras haberle dado una IP fija. Para ello debemos acceder a nuestro router a través del navegador, que como norma general tendrá como IP la correspondiente al último octeto en 1. Para nuestra red es **192.168.0.1**.

Dependiendo del modelo de router que tengamos debemos configurar unas opciones u otras (virtual servers o abrir puertos), pero lo importante es tener claro el concepto y es habilitar la conexión desde el exterior a unos puertos determinados de nuestro servidor a través de nuestra IP pública. En mi caso particular dispongo de un router muy básico para lo cual simplemente he abierto los puertos **80** para **HTTP** y el **443** para **HTTPS.**



Y al acceder a nuestro servidor desde el exterior (en este ejemplo es un móvil) obtenemos el resultado es el esperado.



Como vemos nos da un aviso en el que nos indica que el dominio no es de confianza. Para solucionar este error deberemos escribir en la terminal del servidor lo siguiente.

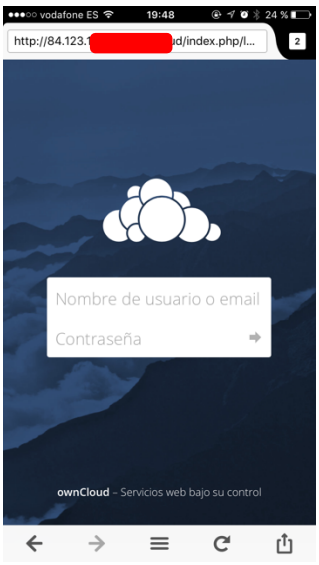
```
# nano /var/www/html/owncloud/config/config.php
```

Una vez dentro de este archivo, debemos de localizar donde aparece **trusted_domains** y en este apartado añadir la instrucción correspondiente al dominio de confianza, ya sea una dirección IP pública o un dominio que tengamos contratado (corresponde a las XXXXXX).

1 => 'XXXXXXXX' ,

```
  GNU nano 2.5.3  Archivo: /var/www/html/owncloud/config/config.php Modificado

<?php
$CONFIG = array (
  'instanceid' => 'oclzzk9yraov',
  'passwordsalt' => 'EJhd6ZedQep/fUhcwF/xoitr0xA9RK',
  'trusted_domains' =>
  array (
    0 => '192.168.0.20',
    1 => 'XXXXXXXXX' ,
  ),
  'datadirectory' => '/home/owncloud dat os',
  'overwrite.cli.url' => 'http://192.168.0.20/owncloud',
  'dbtype' => 'mysql',
  'version' => '10.0.2.1',
  'dbname' => 'owncloud',
  'dbhost' => 'localhost',
  'dbtableprefix' => 'oc_',
  'mysql.utf8mb4' => true,
  'dbuser' => 'tfm',
```

Salvamos y reseteamos el servidor y ya tenemos nuestro acceso desde el exterior sin ningún error.



CONFIGURACIÓN DE SSL Y HTTPS

En este capítulo se explicarán todos los pasos para la instalación de los paquetes necesarios y su posterior configuración con el objetivo de permitir solo el acceso a nuestra nube a través de de HTTPS. Para ellos será necesario crear los certificados pertinentes y configurar posteriormente tanto nuestro servidor como ownCloud.

INSTALACIÓN Y CONFIGURACIÓN INICIAL

Comenzaremos instalando el paquete **openssl** para luego activar este módulo en Apache.

```
# apt-get install openssl

# a2enmod ssl
```

Ahora nos iremos al archivo **/etc/ssl/openssl.cnf** y modificaremos las líneas que se indican para dejarlas tal como la captura, cuyo cometido se indica en ella.

dir = ./demoCA

default_days = 365

```
dir             = /root/sslcertauth     # ruta a los certificados
defaults_days   = 3650                  # días de validez de los certificados
```

Lo siguiente será crear los directorios necesarios y darles los permisos, además de los archivos **index.txt** que contendrá la base de datos de los certificados que se generen y el **serial** que se usa para generar el número de serie del certificado generado (le daremos 1000 en nuestro caso).

```
# mkdir /root/sslcertauth

# chmod 700 /root/sslcertauth

#cd /root/sslcertauth

#mkdir certs private newcerts

#echo 1000 > serial

# touch index.txt
```

Una vez hecho todo lo anterior procederemos a la creación de los certificados y la firma.

CREACIÓN DE CERTIFICADOS Y FIRMA

Lo primero será crear una autoridad de certificación, que nos es necesaria para generar un certificado, por ello creamos nuestra autoridad. Durante la creación nos pedirá que introduzcamos varios datos, entre ellos el más importante será el FQDN, es decir, nuestro dominio, para lo cual introduciremos el dominio que use nuestro servidor o si no tenemos la IP pública que tengamos.

```
# openssl req -new -x509 -days 3650 -extensions v3_ca \-keyout private/cakey.pem -out
cacert.pem \-config /etc/ssl/openssl.cnf
```

A continuación tocará la solicitud de firma. Nos pedirá una password para proteger la clave privada que estamos creando durante el proceso.

```
# openssl req -new -nodes \-out apache-req.pem \-keyout private/apache-key.pem \-config
/etc/ssl/openssl.cnf
```

Lo siguiente será crear el firmar nuestro certificado SSL. Nada más ejecutarlo nos pedirá que introduzcamos la clave privada que usamos durante la creación de la certificación. Además debemos responder yes a dos preguntas.

```
# openssl ca \-config /etc/ssl/openssl.cnf \-out apache-cert.pem \-infiles apache-req.pem
```

```
root@tfm:~/sslcertauth# openssl ca \-config /etc/ssl/openssl.cnf \-out apache-cert.pem \-inf
iles apache-req.pem
Using configuration from /etc/ssl/openssl.cnf
Enter pass phrase for /root/sslcertauth/private/cakey.pem:
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number: 4096 (0x1000)
        Validity
            Not Before: Aug 30 09:35:11 2017 GMT
            Not After : Aug 28 09:35:11 2027 GMT
        Subject:
            countryName               = ES
            stateOrProvinceName       = CADIZ
            organizationName          = UNIR
            organizationalUnitName    = UNIR
            commonName                = [redacted]
            emailAddress              = jorgemibavaz@gmail.com
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Comment:
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:
                35:0F:7D:FB:59:4D:6F:A9:4F:59:E2:38:F4:00:C1:71:88:AC:D9:97
            X509v3 Authority Key Identifier:
                keyid:17:46:8E:44:5B:1E:5C:06:57:9A:0D:D8:51:B9:32:FB:40:A8:71:DB

Certificate is to be certified until Aug 28 09:35:11 2027 GMT (3650 days)
Sign the certificate? [y/n]:y


1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
```

Por último en este apartado deberemos crear unos directorios y copiar las claves y certificaciones creados en estos.

```
# mkdir /etc/ssl/crt

# mkdir /etc/ssl/key

# cp /root/sslcertauth/apache-cert-pem /etc/ssl/crt

# cp /root/sslcerauth/private/apache-key.pem /etc/ssl/key
```

CONFIGURACIÓN DE APACHE Y OWNCLOUD PARA QUE USE SOLO HTTPS

Ya para finalizar este capítulo deberemos añadir unas líneas a dos archivos de nuestro servidor. Para **/etc/apache2/sites-available/default-ssl.conf**.

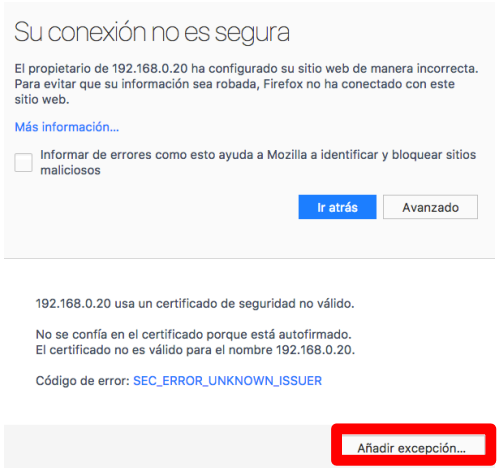
```
<IfModule mod_ssl.c>
        <VirtualHost _default_:443>

                ServerAdmin jorgemibavaz@gmail.com
                ServerName [redacted]
                DocumentRoot /var/www/html

                <Directory /var/www/html/owncloud>
                        AllowOverride All
                </Directory>

                <Directory /home/owncloud datos>
                        AllowOverride All
                </Directory>

                SSLEngine on

                SSLCertificateFile /etc/ssl/crt/apache-cert.pem
                SSLCertificateKeyFile /etc/ssl/key/apache-key.pem

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

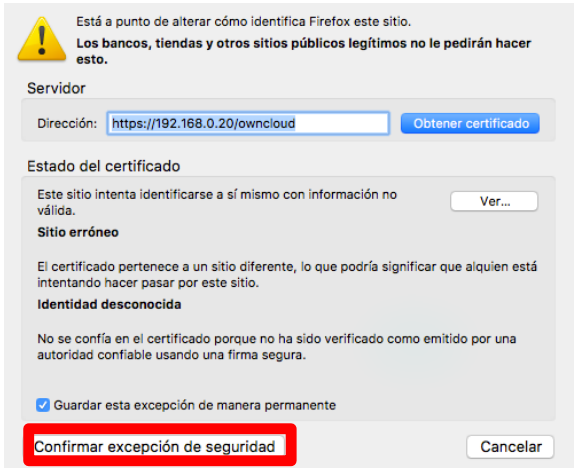
Tras esto crearemos un enlace simbólico del archivo **default-ssl.conf**. y reiniciamos nuestro servidor.

```
# ln -s /etc/apache2/sites-available/default-ssl.conf /etc/apache2/sites-enabled/default-ssl.conf

# service apache2 restart
```

Ya llegados a este punto lo que tendremos que hacer es acceder a nuestra nube a través de HTTPS, escribiendo la url https://192.168.0.20/owncloud.

Cómo es lógico nuestro navegador nos presentará unos avisos de seguridad pues estamos tratando de usar certificados creados por nosotros. Lo que debemos hacer es pulsar sobre **Avanzado** y posteriormente en la ventana siguiente en **Confirmar excepción de seguridad**, con lo cual nos abrirá la página de logueo de ownCloud pero ya sí usando HTTPS.



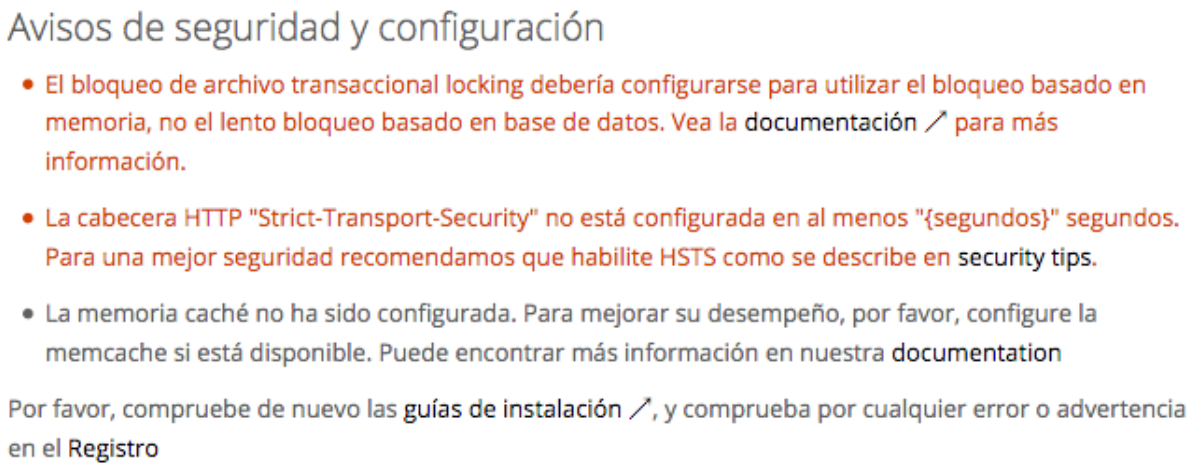


Lo único que nos faltaría para cerrar este apartado es que nuestro servidor reenvíe toda petición de HTTP por HTTPS y para ello en **/etc/apache2/sites-available/000-default.conf** añadimos la siguiente línea para que fuerce dicho encaminamiento.

```
Redirect permanent / https://192.168.0.20/
```

CONFIGURACIÓN DE MEMCACHE Y TRANSACTIONAL FILE LOCKING

Cuando entramos como el usuario **rootown** a ownCloud y nos vamos a **Administración**, opción **General** nos muestra por pantalla varios avisos (los rojos si son necesarios corregir y los negros es recomendable)



En este apartado corregiremos el referente a la memoria caché, que simplemente nos indica que si la gestionamos conseguiremos una carga mejor de las páginas, y el Transactional File Locking, es decir el archivo transaccional de bloqueo, el cual es el responsable de bloquear un archivo que esté ejecutando un usuario para que no exista un edición simultánea de este con todos los problemas que esto pudiera acarrear.

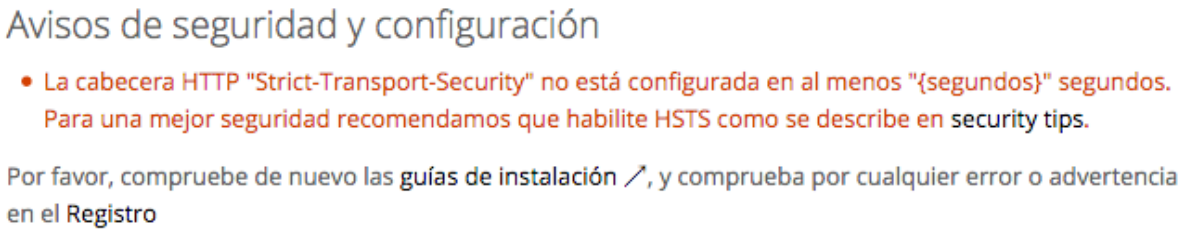
Por lo tanto como ya es de costumbre lo primero será instalar los paquetes que necesitamos.

```
# apt-get install redis-server php-redis
```

Tras esto en el archivo **/var/www/html/owncloud/config/config.php** deberemos añadir las siguientes líneas (esta configuración sería para una gran organización, para una pequeña podríamos omitir la línea que hace referencia a **memcache.distributed**.

```
'filelocking.enabled' => true,
'memcache.distributed' => '\OC\Memcache\Redis',
'memcache.locking' => '\OC\Memcache\Redis',
'memcache.local' => '\OC\Memcache\Redis',
'redis' => array(
      'host' => 'localhost',
      'port' => 6379,
      'timeout' => 0.0,
      ),
```

Tras añadir estas líneas deberemos reiniciar nuestro servidor y refrescar la página de nuestra nube, comprobando así que los avisos de seguridad han desaparecido.



HABILITACIÓN DE HSTS

HSTS lo que nos permite es evitar que los visitantes a nuestra nube ignoren los avisos de los certificados no válidos además de indicar a los navegadores que no acepten ningún tipo de conexión mediante HTTP a ownCloud.

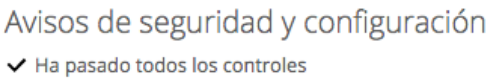
Para poder habilitar este servicio se nos hace necesario instalar y activar el módulo **mod_headers**, para lo que tendremos que hacer lo que se muestra.

```
# a2enmod headers
```

Por lo tanto lo siguiente es ya habilitar **HSTS**, para lo que deberemos abrir el archivo **/etc/apache2/sites-available/default-ssl.conf** y añadir las líneas en el recuadro en rojo. Se han agregado un comentario a las dos últimas líneas fuera del recuadro rojo para dejar bien claro a que hacen referencia y donde deben ir enmarcadas las líneas que añadimos.

```
        <IfModule mod_headers.c>
                Header always set Strict-Transport-Security "max-age=15552000; includeSubDomains"
        # cierra modulo mod_headers.c
        </IfModule>
    # cierra VirtualHost 443
    </VirtualHost>

# cierra modulo ssl_c
</IfModule>
```

Ahora reiniciamos el servidor para activar **mod_headers** y todos los cambios hechos en **defaul-ssl.conf**. Ya viendo de nuevo nuestra ventana de **Administración** comprobamos que ya no existe ningún aviso de seguridad.



4.1.3. Configuración de los paquetes y actualizaciones

Tras la finalización deberemos eliminar el directorio **/var/tmp** y pondremos un enlace simbólico a **/tmp.**

```
# rm –r /var/tmp

# ln –s /tmp /var/tmp
```

El siguiente paso es configurar los repositorios, para los cuales usaremos los oficiales del sistema operativo y software instalado, usando por lo general aquellos de la rama estable y descartando los demás (no oficiales, código fuente…, salvo que se justifique dicha necesidad). Nuestro archivo **/etc/apt/sources.list** debería quedar con solo estos repositorios (donde xenial es la versión de Ubuntu).

```
deb http://es.archive.ubuntu.com/ubuntu/ xenial main restricted
deb http://es.archive.ubuntu.com/ubuntu/ xenial-updates main restricted
deb http://security.ubuntu.com/ubuntu xenial-security main restricted
```

En esta edición de Ubuntu ya se cuenta con las librerías necesarias para poder utilizar PHP de manera correcta así que no hará falta instalarlas pero si actualizar los repositorios introduciendo por teclado con usuario root:

```
# apt-get update
```

Y tras esto actualizaremos nuestro sistema servidor.

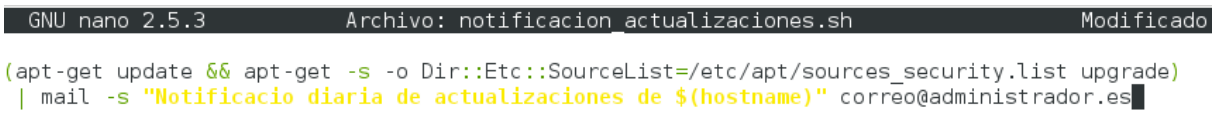
```
# apt-get upgrade && apt-get dist-upgrade
```

Y eliminamos los paquetes no necesarios.

```
# apt-get autoremove
```

Para configurar las actualizaciones crearemos un script el cual nos avisará cuando haya alguna de seguridad. Primero creamos en la ruta este archivo **etc/apt/sources_security.list** con solo el paquete de seguridad.

Y tras esto creamos el script /**etc/cron.daily/notificacion_actualizaciones.sh** donde el email **correo@administrador.es** debe corresponder a aquel donde queremos recibir los avisos.



Por último le damos los permisos necesarios para la ejecución del script.

```
# chmod +x /etc/cron.daily/notificacion_actualizaciones.sh
```

Y como resultado de nuestro script veremos que llega a nuestro email la información necesaria sobre actualizaciones (revisar la carpeta spam).

root miércoles, 6 sept 13:45

To: JORGE IBAÑEZ

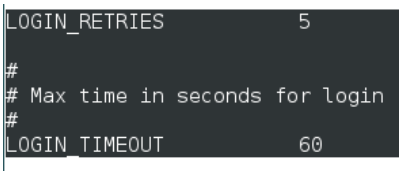
```
CRON-APT RUN [/etc/cron-apt/config]: Wed Sep  6 12:50:01 CEST 2017
CRON-APT SLEEP: 3278, Wed Sep  6 13:44:40 CEST 2017
CRON-APT ACTION: 0-update
CRON-APT LINE: /usr/bin/apt-get -o quiet=1 update -o quiet=2
CRON-APT ACTION: 3-download
CRON-APT LINE: /usr/bin/apt-get -o quiet=1 autoclean -y
Reading package lists...
Building dependency tree...
Reading state information...
CRON-APT LINE: /usr/bin/apt-get -o quiet=1 dist-upgrade -d -y -o APT::Get::Show-Upgraded=true
Reading package lists...
Building dependency tree...
Reading state information...
Calculating upgrade...
The following NEW packages will be installed:
  linux-headers-4.4.0-93 linux-headers-4.4.0-93-generic
  linux-image-4.4.0-93-generic linux-image-extra-4.4.0-93-generic
The following packages will be upgraded:
  libgd3 libsnapd-glib1 linux-firmware linux-generic linux-headers-generic
  linux-image-generic snapd-login-service
7 upgraded, 4 newly installed, 0 to remove and 0 not upgraded.
Need to get 0 B/108 MB of archives.
After this operation, 298 MB of additional disk space will be used.
Download complete and in download only mode
```

AUTENTICACIÓN

Para continuar con la securización debemos configurar unos parámetros respecto a las contraseñas para acceso al servidor. Como sabemos las contraseñas deben tener una longitud mínimos, minúsculas, mayúsculas, caracteres especiales y números. Nunca usaremos frases, palabras o números característicos que puedan ser adivinados por fuerza bruta. Igualmente debemos configurar la caducidad de estas contraseñas. Para ello en **/etc/login.def** debemos fijar un valor para **PASS_MAX_DAYS**. Con **PASS_WARN_AGE** indicaremos los días previos que el sistema avisará a los usuarios para que cambien su contraseña.

```
PASS_MAX_DAYS   99999
PASS_MIN_DAYS   0
PASS_WARN_AGE   7
```

También podemos configurar el número de intentos y el tiempo que durará la sesión antes de cerrarse, aunque si tenemos configurados PAM el número de intentos será de 3.



Para las cuentas existentes modificarlas con

```
# change –M <número días> -W <días previos aviso> <usuario>
```

Respecto a PAM es un framework que facilita una arquitectura de autenticación y puede y debe ser configurado para minimizar los riesgos a los que nuestro sistema está expuesto. Instalaremos el paquete **libpam-cracklib**.

Tras esto debemos modificar el fichero **/etc/pam.d/common-password** añadiendo las siguientes líneas o modificándolas de forma que queden tal cual.

```
# password requisite pam_cracklib.so retry=3 minlen=12 lcredit=-1 ucredit=-1 dcredit=-1 ocredit=-1 difok=3

# password [success=1 default=ignore] pam_unix.so obscure use_authtok try_first_pass sha512 remember=5
```

Las opciones de configuración son las siguientes:

- retry: número de intentos antes de devolver error.
- minlen longitud mínima de contraseña.
- difok: cambios de caracteres de la nueva contraseña en relación a la vieja.
- ucredit: caracteres en mayúscula que debe tener.
- lcredit: caracteres en minúscula que debe tener.
- dcredit: número de dígitos que debe tener la nueva contraseña.
- ocredit: número de dígitos que debe tener la contraseña

Y por último un signo – indica que como mínimo debe tener esa cantidad y un signo + indica que esa es la cantidad máxima que debe tener.

4.1.4. Restricciones de acceso

El primer acceso a configurar es el de la BIOS, para el cual deberemos poner una contraseña segura y desactivar los arranques desde discos externos y la red, siendo desde el disco duro la opción correcta.

Lo primero será proteger nuestro gestor de arranque Grub. Para ello escribiremos en consola.

```
# grub-mkpasswd-pbkdf2 sha512 remember=5
```

E introduciremos una contraseña de la cual obtendremos el hash.

```
tfm@tfm:~$ grub-mkpasswd-pbkdf2
Introduzca contraseña:
Reintroducir la contraseña
el hash PBKDF2 de su contraseña es grub.pbkdf2.sha512.10000.06FB40A8A240
C04C5FB01989D8B2B7DDE6C76B3579564A06DFCB5950F75CC29CBB1B5454B2E04296B88E
1D4EE133B5EB3B1AB01A2B107FC77BF3B8278954A3D1.93A7EE90E880DD336E5BD10901E
E35A1728AD69DA0A3C6DD853AB3F3877B7DFBDB8C16BDB53D2A3750669742C33EAE3E1CD
D332C64E262E23BF47FEB338E774F
```

Este hash deberemos pegarlo con las líneas a continuación en **/etc/default/grub.**

```
# set superusers="nombre"

# password_pbkdf2 nombre <hash generado>
```

Debemos deshabilitar el terminal gráfico editando el fichero /etc/default/grub para lo cual debemos dejar de comentar GRUB_TERMINAL=console

```
# Uncomment to disable graphical terminal (grub-pc only)
GRUB_TERMINAL=console
```

Y por último actualizaremos GRUB.

```
# update-grub
```

CONFIGURACIÓN MODO SINGLE-USER

Debemos configurar el modo **single-user**, el cual usa el sistema para administración en casos excepcionales y que si no se configura no tendrá contraseña alguna. Para ello escribiremos **sudo su** y posteriormente **passwd** en nuestro terminal tal como vemos en la captura (la cual nos avisa que nuestra clave no cumple los requisitos de seguridad). Además vemos que esto habilita al usuario **root**.

```
tfm@tfm:~$ sudo su
[sudo] password for tfm:
root@tfm:/home/tfm# passwd
Nueva contraseña:
CONTRASEÑA INCORRECTA: Es DEMASIADO corta.
CONTRASEÑA INCORRECTA: es demasiado sencilla
Vuelva a escribir la nueva contraseña:
passwd: password updated successfully
root@tfm:/home/tfm#
```

ASIGNACIÓN DE TIEMPO DE INACTIVIDAD A LA CONSOLA

Para asignar un tiempo de inactividad a la consola lo único que deberemos hacer es añadir al final de **/etc/profile** 3 líneas (300 hace referencia a segundos).

```
TMOUT=300
readonly TMOUT
export TMOUT
```

CONFIGURAR EL BLOQUEO DE PANTALLA EN LA CONSOLA

Esto debe configurarse para que los usuarios puedan bloquear sus sesiones de consola pero dejándola abierta. Lo primero sería instalar el paquete **screen**, aunque es probable que ya lo tenga mediante:

```
# apt-get install screen –no-install-recommends
```

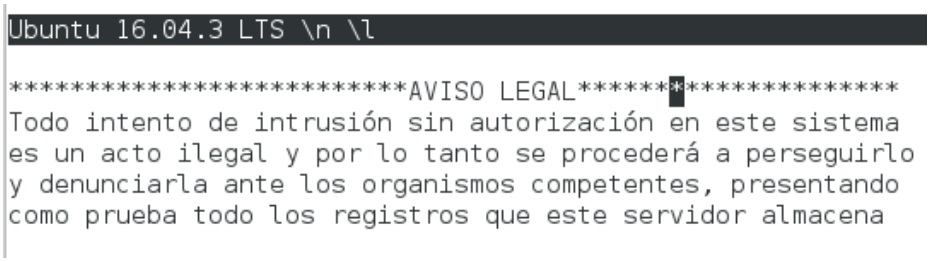
A partir de aquí para mantener la sesión pero cerrar el terminal escribiremos. Y para recuperarla pulsaremos “Control+a+d”.

```
# screen –S nombre_sesion
```

CONFIGURACIÓN DEL BANNER

El banner que se muestra al arrancar el sistema puede mostrar algún tipo de información demasiado explícita como el sistema operativo, versión y demás. Es conveniente eliminar toda esta información y añadir una advertencia legal que informe que el uso no autorizado será

perseguido y que todas las actividades del servidor se registran los archivos **etc/issue** y **etc/issue.net**. Como vemos hemos añadido un aviso legal y deberíamos quitar que muestra la información resaltada sobre nuestro sistema operativo.



DESHABILITAR MAGIC SYSRQ KEY (TECLA PETICIÓN DE SISTEMA)

Simplemente escribiremos en el terminal kernel.sysrq=0 para evitar las llamadas a acciones de bajo nivel.

DESHABILITAR CONTROL+ALT+SUP DESDE CONSOLA

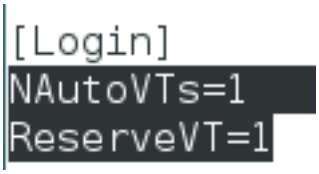
Deberemos escribir por consola ambos comandos, evitando así que un usuario malicioso sin estar logueado pueda reiniciar nuestro servidor.

```
# sudo systemctl mask ctrl-alt-del.target

# sudo systemctl daemon-reload
```

LIMITAR EL NÚMERO DE TERMINALES VIRTUALES Y RESTRICCIÓN DE LOGGINS

Por defecto trae 6 terminales, así que para dejar sólo una comentaremos las líneas de /etc/securetty dejando solo aquellos loggins que queramos dejar para el root a la consola.

Para limitar el número de terminales en **/etc/systemd/logind.conf** dejaremos los valores como en el los comandos. De esta forma solo podremos usar el primer terminal virtual.



CONFIGURACIÓN DE LOS USUARIOS

Tras la instalación se nos crea un usuario de administración el cual puede usar **su.** Este comando permite usar el intérprete de comandos de otro usuario sin salir de nuestra sesión mientras que **sudo** nos permita solo ejecutar un comando como si fuéramos otro

usuario. Como vemos ambos comandos pueden ser peligrosos usados por malas manos, así que su configuración también será muy importante.

- A pesar de que **sudo** solo permite ejecutar un comando puede tener un pequeño tiempo de gracia que deberemos eliminar añadiendo en **/etc/sudoers** la línea

```
Defaults:ALL timestamp_timeout=0
```

Además debemos limitar el uso de **su**. En nuestro caso es el usuario **tfm**, por lo que se hace necesario limitar las cuentas que puedan aumentar sus privilegios a root. Para ello crearemos el grupo **wheel** y añadiremos **tfm** (en la captura se observa como puede usar **su**)

```
tfm@tfm:/$
tfm@tfm:/$ sudo addgroup wheel
Añadiendo el grupo `wheel' (GID 1001) ...
Hecho.
tfm@tfm:/$ sudo adduser tfm wheel
Añadiendo al usuario `tfm' al grupo `wheel' ...
Adding user tfm to group wheel
Hecho.
tfm@tfm:/$
```

Tras esto debemos verificar que en **/etc/pam.d/su** aparece lo que se muestra.

```
auth       required   pam_wheel.so
```

- También debemos limitar el uso de sudo. Los pasos a seguir serás los siguientes:

1. Verificar que tenemos instalado el paquete **sudo**.

```
# sudo –V
```

2. Crear el grupo **admin** si no existiera.

```
# addgroup admin
```

3. Añadir nuestros usuarios al grupo **admin**.

```
# usermod –a –G admin <nombre_usuario>
```

4. Modificar el fichero **/etc/sudoers** mediante **visudo** y añadir la línea **%admin ALL=(ALL) ALL**

- Y por último debemos bloquear todas las cuentas del sistema para evitar el acceso a la Shell (excepto al usuario root). Siguiendo estos pasos:

1. Para mostrar los usuarios, sus UID y si tienen acceso a la Shell escribiremos:

```
# awk 'BEGIN {FS=":"} {print $1,$3,$7}' /etc/passwd
```

2. Lo siguiente será bloquear cada uno de los usuarios del sistema (aquellos cuya UID es inferior a 500).

```
# usermod –L <usuario>
```

3. Y finalizamos escribiendo en el terminal

```
# usermod –s /usr/sbin/nologin <usuario>
```

Que deshabilitará la Shell de cada usuario del sistema. En la captura vemos como los usuarios aparecen **nologin** excepto **root.**

```
root 0 /bin/bash
daemon 1 /usr/sbin/nologin
bin 2 /usr/sbin/nologin
sys 3 /usr/sbin/nologin
sync 4 /usr/sbin/nologin
```

RESTRINGIR LOS ACCESOS POR RED

Usamos TCP WRAPPER en el control de acceso para los servicios de red del sistema usando los ficheros de configuración **/etc/hosts.allow** y **/etc/hosts.deny.** Su funcionamiento es de esta manera:

1. Se inicia un intento de conexión por lo que busca en **/etc/host.allow** alguna regla, si existe la permite.
2. Si no la encuentra busca en **/etc/host.deny**, si existe la bloquea.
3. Si no encuentra ninguna regla permitirá la conexión.

Como vemos por defecto no bloque ningún intento de conexión. Lo que haremos será configurarlo para permitir conexiones remotas para servicios públicos y limitar los servicios privados. TCP WRAPPER se usa para los servicios que usan la librería **libwrap,** si quisiéramos saber cuales son estos escribiríamos.

```
# apt-cache rdepends libwrap0
```

Por lo tanto para saber con exactitud si un servicio es público o privado debemos:

1. Para obtener la ruta/ejecutable de los servicios usamos lo siguiente.

```
# whereis <nombre_servicio>
```

2. Luego escribimos para ver si el servicio es público o privado.

```
# strings /ruta/ejecutable/servicio | grep libwrap
```

3. Si nos devuelve algún mensaje es un servicio privado y por lo tanto deberemos configurarlo.

4. Para configurar los servicios privados y públicos añadimos a **/etc/hosts.all**

servicio_público: ALL

servicio_privado: .dominio.local

Y en **/etc/hosts.deny** añadimos:

ALL: ALL

5. Para comprobar el funcionamiento de TCP WRAPPER, usamos el comando **tcdmatch** con el cual podremos ver su comportamiento.

```
# tcpdmatch
```

En este ejemplo hemos configurado el servicio **gdm3** para que solo acepte conexiones de las direcciones IP de las redes locales 192.168.0.0-192.168.1.255 en el archivo **/etc/hosts.allow**

```
gdm3: 192.168.0.0/255.255.254.0
ALL: localhost
```

Vemos que para 192.168.2.4 es denegada.

```
root@tfm:/home/tfm# tcpdmatch gdm3 192.168.0.4
client:   address  192.168.0.4
server:   process  gdm3
access:   granted
root@tfm:/home/tfm# tcpdmatch gdm3 192.168.2.4
client:   address  192.168.2.4
server:   process  gdm3
access:   denied
root@tfm:/home/tfm# tcpdmatch gdm3 192.168.1.4
client:   address  192.168.1.4
server:   process  gdm3
access:   granted
```

4.1.5. Reducción de la superficie de ataque

Reducir la superficie de ataque es tener instalado y ejecutar la menor cantidad posible de software puesto que así se reducen las posibles vulnerabilidades. Para esto se hace muy necesario desinstalar aquellos paquetes y eliminar todos los servicios innecesarios, configurar los permisos de archivos y directorios, proteger los ejecutables y fortalecer el kernel.

ELIMINACIÓN DE LOS PAQUETES INNCESARIOS

Deberemos ver todos los paquetes que tenemos instalados mediante

```
# dpkg -l
```

Una vez sepamos que tenemos procederemos a desinstalar aquellos innecesarios. Tras esto hay que desinstalar las dependencias innecesarias.

```
# apt-get autoremove
```

DESHABILITAR SERVICIOS INNECESARIOS

Además esta guía recomienda deshabilitar algunos servicios como se pueden observar en el cuadro.

Nombre del servicio	Descripción	Acción
acpid	Configuración avanzada e interfaz de energía. Se utiliza para gestión de la energía.	Deshabilitar.
cron	Ejecuta trabajos programados	Dejar activo.
dns-clean	Limpia la información de DNS en conexiones por red telefónica (dial-up).	Deshabilitar.
rc.local	Inicio de los scripts locales	Dejar activo.
grub-common	Muestra información en el menú de GRUB.	Dejar activo.
ondemand	Configura el gestor de escala de frecuencia de CPU debajo demanda.	Dejar activo.
pppd-dns	Limpia el fichero /resolv.conf cuando el sistema falla.	Deshabilitar.
rmnologin	Elimina el fichero /etc/nologin al inicio.	Dejar activo.
rsync	Permite la copia de ficheros a o desde máquinas remotas.	Deshabilitar.
rsyslog	Permite el envío de mensajes.	Dejar activo.
ssh	Servidor de SSH.	Deshabilitar en caso de tener acceso físico.
stopbootlogd	Para el demonio de registro de *logs* enviados a la consola.	Dejar activo.
watchdog	Inicia el demonio de "watchdog".	Deshabilitar.
wd_keepalive	Inicia el demonio "keepalive" de "watchdog".	Deshabilitar.

Figura 4. 4.1.4. Cuadro de servicios a deshabilitar guía CCN-STIC 612

Para ellos es bueno saber cuales son los servicios que tenemos listándolos, donde todos aquellos que lleven un + significa que están activos y el – que no lo están.

```
# service --status-all
```

Para deshabilitar los servicios basados en **/etc/init.d** que se recomiendan deberemos escribir la siguiente instrucción:

```
# update-rc.d –f <nombre_servicio> remove
```

Para deshabilitar los servicios basados en **Upstart** debemos editar **/etc/init/servicio** y comentar la línea:

```
#start on runlevel [2345] and not-container
stop on runlevel [!2345]
```

Además para monitorizar las conexiones rechazadas y los errores debemos verificar que en **/etc/rsyslog.d/50-default.conf** se encuentra la línea **auth. Authpriv.* /var/log/auth.log** sino habrá que añadirla.

COMPROBACIÓN DE PERMISOS EN DIRECTORIOS Y FICHEROS

En este capítulo, como bien indica su nombre, comprobaremos los permisos en directorios y ficheros, ya no solo para prevenir un mal uso o acceso no autorizado por usuarios del sistema a alguna información, sino sobre todo por evitar debilidades que puedan ser usadas por atacantes.

PERMISOS DE LOS FICHEROS PASSWD, SHADOW, GROUP Y GSHADOW

El fichero **shadow** contiene información sobre el hash de las claves de los usuarios, al cual debemos restringir sus permisos para que un usuario malicioso no hago uso de este y obtenga contraseñas de usuarios. Para el caso de **passwd** es distinto pues muchas aplicaciones acceden a este con permiso de lectura.

```
# cd /etc
# chown root:root passwd shadow group gshadow
# chmod 644 passwd group
# chmod 400 shadow gshadow
```

VERIFICAR SI EL BIT STICKY ESTÁ ACTIVADO

Debemos comprobar que ninguno de los directorios con permiso de escritura tienen activado el bit **sticky** mediante terminal, para evitar que un usuario con permiso de escritura modifique o borre algún archivo del que no es propietario. En caso de que sea usado por una aplicación, será necesario comprobar la documentación referente para el directorio en concreto.

PARTICIÓN es el punto de montaje de una partición, así si tengo en **/dev/sda5** el directorio **/usr** este deberemos escribir en el comando.

```
# find PARTICION –xdev –type d \( -perm -0002 –a ¡ -perm -1000 \) -print
```

En el caso que mostrara alguna información habría que corregir dicho directorio.

```
# chmod –t /DIRECTORIO
```

VERIFICAR FICHEROS CON PERMISOS GLOBALES DE ESCRITURA

Para los archivos también deberemos evitar que tengan permisos globales de escritura por lo que deberemos buscarlos en cada partición.

```
# find PARTICION –xdev –type f –perm -0002 –print
```

En caso de que alguna de las particiones muestre información como puede verse en la captura de pantalla procederemos a cambiar los permisos de los archivos mostrados.

```
root@tfm:/etc# find /tmp -xdev -type f -perm -0002 -print
root@tfm:/etc# find /usr -xdev -type f -perm -0002 -print
root@tfm:/etc# find /home -xdev -type f -perm -0002 -print
root@tfm:/etc# find /boot -xdev -type f -perm -0002 -print
root@tfm:/etc# find /var -xdev -type f -perm -0002 -print
/var/crash/.lock
```

Para usamos el siguiente comando.

```
# chmod o-w FICHERO
```

Y podemos comprobar que ya no aparece con permisos de escritura globales.

```
root@tfm:/etc# chmod o-w /var/crash/.lock
root@tfm:/etc# find /var -xdev -type f -perm -0002 -print
root@tfm:/etc#
```

ENCONTRAR FICHEROS EJECUTABLES CON PERMISOS SUID/SGID ACTIVADOS

Los permisos SUID (usuario) y SGID (grupo) permiten que un usuario ejecute dicho archivo con los permisos de los usuarios y grupos que son propietarios de él en vez de con los permisos del usuario que los está ejecutando.

```
# find PARTICION –xdev \( -perm -4000 –o –perm -2000 \) –type f –print
```

La siguiente tabla muestra algunos servicios que por defecto traen en Debian y Ubuntu Server activados y la recomendación sobre deshabilitarlos o no. De forma que para hacerlo pasaremos por la terminal.

```
# chmod –s FICHERO
```

Fichero	SUID/SGID	Perteneciente	¿Deshabilitar?
/usr/lib/openssh/ssh-keysign	uid root	SSH	Sí
/usr/sbin/exim4	uid root	Correo	Sí
/bin/mount	uid root	Sistema	No
/bin/umount	uid root	Sistema	No
/bin/ping	uid root	Red	No
/usr/bin/su	uid root	Autenticación	No
/sbin/unix chkpwd	uid root	Autenticación PAM	No
/usr/bin/chage	uid root	Expiración de contraseñas	Sí
/usr/bin/chsh	uid root	Información de usuario	Si
/usr/bin/crontab	uid / gid root	Cron	Sí
/usr/bin/gpasswd	uid root	Autenticación	No
/usr/bin/locate	gid slocate	Base de datos local	No
/usr/bin/newgrp	uid root	Autenticación	No
/usr/bin/passwd	uid root	Autenticación	No
/usr/bin/rcp	uid root	RSH	Sí
/usr/bin/rlogin	uid root	RSH	Sí
/usr/bin/rsh	uid root	RSH	Sí
/usr/bin/wall	gid tty	Mensajes de consola	Sí
/usr/bin/write	gid tty	Mensajes de consola	Sí

Figura 5. 4.1.4. Cuadro de servicios a deshabilitar guía CCN-STIC 612

ENCONTRAR FICHEROS SIN PROPIETARIO O GRUPO VÁLIDO

Para cada partición debemos encontrar aquellos ficheros que no tengan propietario o grupo válido según el comando que se muestra.

```
# find PARTICION –xdev \( -nouser –o –nogroup \)  –print
```

En el caso que se muestre alguna información será necesario asignarles un usuario y/o grupo, a pesar de que dichos ficheros no sean directamente explotables, pero es preferible hacerlo así.

VERIFICAR QUE LOS DIRECTORIOS CON PERMISO DE ESCRITURA GLOBALES TIENEN UN PROPIETARIO ADECUADO

Un fallo de seguridad es que exista un propietario de un directorio escribible por todos los usuarios distinto del root o un usuario del sistema (aquellos cuyo UID es inferior a 1000), ya que este permitiría a ese usuario modificar o eliminar algún archivo colocado por otro usuario. En caso de obtener algún directorio con este error se le deberá asignar un propietario adecuado.

```
# find PARTICION –xdev –type d –perm -0002 –uid +1000 –print
```

COMPROBACIÓN DE EJECUTABLES DEL SISTEMA

Aquí se busca configurar el sistema operativo para que no existan fugas de información debido a desbordamientos de búffer y la ejecución de binarios.

RUTAS ABSOLUTAS DE EJECUTABLES

La variable $PATH contiene la ruta de los ficheros ejecutables más comunes. Un atacante podría ejecutar esta ruta, de forma que apuntara a un directorio donde se encontrara un fichero malicioso llamado igual que un comando de uso en la terminal. Para evitar que esto ocurra debemos comprobar que dicha variable no contiene ".", ya que incluiría así la ruta local. Para ello simplemente haciendo lo que se muestra a configuración comprobamos las rutas que contiene $PATH.

```
# echo $PATH
```

Y vemos el resultado.

```
root@tfm:/home/tfm# echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr
/games:/usr/local/games
```

DESHABILITAR CORE DUMPS

Un fichero **core dumps** se crea cuando un ejecutable debido a un error para su ejecución, siendo una imagen en la memoria de este. Pueden contener información muy sensible además de ocupar gran cantidad de espacio. Todo esto hace necesario deshabilitarlos para todos los usuarios añadiendo a **/etc/security/limits.conf** la siguiente línea.

```
*       hard    core    0
```

También deberemos configurar /etc/sysctl.conf y añadir esto para evitar que los **core dumps** se realicen para ejecutables con el bit SUID activado.

```
fs.suid_dumpable = 0
```

PROTEGER FRENTE A DESBORDAMIENTOS DE BÚFFER

Para proteger nuestro sistema contra el desbordamiento de búffer debemos hacer dos cosas. Una de ellas es habilitar "**Execute Disable**" (**XD**, para procesadores Intel) o "**No**

```
# cat /proc/cpuinfo
```

Execute" (**NX**, para procesadores AMD) en sistemas x86-32. Para ello deberemos comprobar si nuestro procesador soporta estas funcionalidades

Si al ejecutar ese comando nos muestra por pantalla **nx** o **pae** en el campo **flags** significa que soporta PAE y NX.

```
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep mtr
r pge mca cmov pat pse36 clflush mmx fxsr sse sse2 syscall nx rdt
scp lm constant_tsc rep_good nopl xtopology nonstop_tsc pni pclmu
lqdq monitor ssse3 cx16 sse4_1 sse4_2 x2apic popcnt aes xsave avx
 rdrand hypervisor lahf_lm
```

Y por lo tanto debemos instalar el paquete **linux-image-generic-pae** si el procesador fuera x86-32, en nuestro caso es de 64 bits y por lo tanto no será necesario. Además habría que comprobar que en la BIOS está activada, ya que algunas permiten activar o desactivar esta funcionalidad.

El segundo paso será usar direcciones de pila aleatorias que nos protegerá de ataques intencionados por desbordamiento de búffer en los que se busque la ejecución de un código malicioso, casi siempre buscando ejecutar una Shell remota. Para ellos deberemos irnos a **/etc/sysctl.conf** y editar lo que se indica.

```
kernel.randomize_va_space = 1
```

COMPROBACIÓN DE LAS CUENTAS DE USUARIOS

Este apartado se dedica a una correcta configuración de la seguridad en las cuentas de usuarios.

ELIMINAR LAS CUENTAS DE USUARIOS INNECESARIAS

Lo primero que haremos será eliminar los usuarios que recomienda la guía CCN-STIC que estamos usando.

Usuario	Descripción	Acción
proxy	Usuario asociado a algunos servidores Proxy (como squid).	Eliminar el usuario.
www-data	Usuario asociado a servidores web (como Apache).	Eliminar el usuario.
list	Usuario asociado al servicio de listas de distribución de correo electrónico.	Eliminar el usuario.
irc	Usuario asociado a algunos servidores de IRC.	Eliminar el usuario.
gnats	Usuario asociado al servicio de control de bugs (errores) del sistema operativo GNU.	Eliminar el usuario.
uucp	Usuario asociado al protocolo UUCP (Unix to Unix CoPy).	Eliminar el usuario.
news	Usuario asociado al servicio de news (Usenet News).	Eliminar el usuario.
lp	Usuario asociado al servicio de impresión.	Eliminar el usuario.

Figura 6. 4.1.4. Cuadro de usuarios a eliminar según guía CCN-STIC 612

Para ello lo haremos de esta forma.

```
# userdel USUARIO
```

COMPROBAR QUE NO EXISTEN CUENTAS SIN CONTRASEÑA

A continuación comprobamos que todos los usuarios tienen clave. En caso que obtengamos algún resultado deberemos bloquear el acceso de los usuarios resultantes a la Shell como ya hemos visto anteriormente.

```
# awk –F: ‘($2 == “”) {print}’ /etc/shadow
```

VERIFICAR QUE NO EXISTEN CUENTAS NO ROOT CON UID IGUAL A 0

Solo un usuario debe tener un UID igual a 0, por lo tanto se hace necesario comprobar en **/etc/passwd** la UID de todos los usuarios.

```
# awk –F: ‘($3 == “0”) {print}’ /etc/passwd
```

Y vemos el resultado.

```
root@tfm:/home/tfm# awk -F: '($3=="0") {print}' /etc/passwd
root:x:0:0:root:/root:/bin/bash
```

CREACIÓN DE UN GRUPO ÚNICO A CADA USUARIO

Aquí lo que se hace necesario es aclarar que al crear un usuario nunca deberemos usar la opción **–g** al comando **useradd**, pues sin ella se crea un grupo único por cada usuario.

SECURIZAR LOS FICHEROS DE CONFIGURACIÓN DE SESIÓN PARA LAS CUENTAS DE USUARIO

Para evitar que un atacante tenga acceso a la información de los distintos ficheros, sobre todo las cuentas de administración se deberán seguir los siguientes pasos para securizarlos.

Comenzaremos comprobando los permisos de los directorios del usuario. Es muy importante que el directorio home no tenga permisos de escritura para miembros del grupo o

```
# ls –ld /home/USUARIO
```

permisos de lectura para todos los usuarios. Haremos esto para cada usuario humano del sistema.

Así podemos observar como nuestro usuario humano **tfm** tiene permisos de lectura, escritura y ejecución sobre el directorio, pero como en la parte perteneciente a los usuarios del grupo (los tres primeros guiones tras x) y la parte perteneciente al resto de usuarios (los tres últimos guiones) no tiene ningún tipo de permiso, por lo tanto la configuración de seguridad es la idónea.

```
root@tfm:/home/tfm# ls -ld /home/tfm
drwx------ 16 tfm tfm 4096 ago 19 19:00 /home/tfm
```

Nos aseguramos que los miembros del grupo no tienen permiso de escritura.

```
# chmod g-w /home/USUARIO
```

Ni el resto de usuarios permisos de lectura.

```
# chmod o-rwx /home/USUARIO
```

Haremos lo mismo para los ficheros ocultos de los usuarios.

```
# ls –la /home/USUARIO
```

En el caso que mostrara alguno deberíamos corregirlo.

```
# chmod go-w /home/USUARIO/FICHERO
```

```
drwx------ 16 tfm  tfm  4096 ago 19 19:00 .
drwxr-xr-x  5 root root 4096 ago 11 20:27 ..
-rw-------  1 tfm  tfm  1684 ago 13 23:56 .bash_history
-rw-r--r--  1 tfm  tfm   220 ago 11 20:27 .bash_logout
-rw-r--r--  1 tfm  tfm  3771 ago 11 20:27 .bashrc
drwx------  2 root root 4096 ago 11 20:46 .cache
drwx------ 14 tfm  tfm  4096 ago 14 16:45 .config
drwxr-xr-x  2 tfm  tfm  4096 ago 11 20:57 Descargas
drwxr-xr-x  2 tfm  tfm  4096 ago 11 20:57 Documentos
lrwxrwxrwx  1 tfm  tfm    29 ago 11 20:27 .ecryptfs -> /home/.ecryptfs/tfm/.ecryptfs
drwxr-xr-x  2 tfm  tfm  4096 ago 11 20:57 Escritorio
drwxr-xr-x  2 tfm  tfm  4096 ago 11 20:57 .fontconfig
drwx------  2 tfm  tfm  4096 ago 11 20:58 .gconf
-rw-------  1 tfm  tfm  4284 ago 19 19:00 .ICEauthority
drwxr-xr-x  2 tfm  tfm  4096 ago 11 20:57 Imágenes
drwxr-xr-x  3 tfm  tfm  4096 ago 11 20:57 .local
drwxr-xr-x  2 tfm  tfm  4096 ago 11 20:57 Música
drwxrwxr-x  2 tfm  tfm  4096 ago 12 20:12 .nano
drwxr-xr-x  2 tfm  tfm  4096 ago 11 20:57 Plantillas
lrwxrwxrwx  1 tfm  tfm    28 ago 11 20:27 .Private -> /home/.ecryptfs/tfm/.Private
-rw-r--r--  1 tfm  tfm   655 ago 11 20:27 .profile
drwxr-xr-x  2 tfm  tfm  4096 ago 11 20:57 Público
-rw-r--r--  1 tfm  tfm     0 ago 11 20:29 .sudo_as_admin_successful
drwxr-xr-x  2 tfm  tfm  4096 ago 11 20:57 Vídeos
```

Como podemos observar tenemos 3 casos. Los dos marcados en recuadros azules son enlaces que no serán necesario, solo deberemos corregir los permisos del directorio que se encuentra en el recuadro rojo.

Lo siguiente será configurar **umask** a 077 por defecto, haciendo que los ficheros creados no sean accesibles por ningún otro usuario. Para esto deberemos editar varios ficheros, **/etc/login.defs** y **/root/.bashrc** añadiendo la línea mostrada.

```
umask 077
```

Y por último eliminar el historial teniendo que añadir a los ficheros **.bashrc**, **bash_profile** y **/etc/profile** lo mostrado. Lo general es que **bash_history** esté en la carpeta personal de cada usuario.

```
shred /home/tfm/.bash_history
```

OPCIONES DE MONTAJE DEL SISTEMA DE FICHEROS

El aplicar ciertas opciones a las particiones nos darán un plus de seguridad. Para ello en el archivo **/etc/fstab** deberemos configurarlo de la siguiente manera:

- Añadir la opción **nodev** a las particiones que no son **raíz /**.
- Añadir la opción **nosuid** a las particiones que no son **raíz /** ni **/usr**.
- Añadir la opción **nodev**, **nosuid**, **noexec** a las particiones extraíbles.

Esta es la configuración inicial de nuestro archivo **/etc/fstab**.

```
/dev/mapper/sda5_crypt /                                ext4    errors=remount-ro 0       1
# /boot was on /dev/sda6 during installation
UUID=6fc9d7d6-4a1b-4134-93db-a156e418c843 /boot         ext4    defaults          0       2
/dev/mapper/sda11_crypt /home                           ext4    defaults          0       2
/dev/mapper/sda9_crypt /tmp                             ext4    defaults          0       2
/dev/mapper/sda10_crypt /usr                            ext4    defaults          0       2
/dev/mapper/sda8_crypt /var                             ext4    defaults          0       2
/dev/mapper/sda7_crypt none                             swap    sw                0       0
```

Y así debería quedar el archivo tras aplicar las recomendaciones indicadas.

```
/dev/mapper/sda5_crypt /                                ext4    errors=remount-ro       0     1
# /boot was on /dev/sda6 during installation
UUID=6fc9d7d6-4a1b-4134-93db-a156e418c843 /boot         ext4    defaults,nodev,nosuid   0     2
/dev/mapper/sda11_crypt /home                           ext4    defaults,nodev,nosuid   0     2
/dev/mapper/sda9_crypt /tmp                             ext4    defaults,nodev,nosuid   0     2
/dev/mapper/sda10_crypt /usr                            ext4    defaults,nodev          0     2
/dev/mapper/sda8_crypt /var                             ext4    defaults,nodev,nosuid   0     2
/dev/mapper/sda7_crypt none                             swap    sw                      0     0
```

DESHABILITAR IDSPOSISTIVOS USB ALMACENAMIENTO MASIVO

Una medida seguridad básica en cualquier sistema informático es deshabilitar el montaje automático de los USB. Para esto editaremos **/etc/modprobe.d/blacklist.conf** y añadiremos esta línea.

```
blacklist usb-storage
```

SECURIZACIÓN DE LOS PARÁMETROS DE RED

En este apartado securizaremos la pila TCP/IP. Los siguientes parámetros que añadiremos a **/etc/sysctl.conf** son solo para aquellos dispositivos que no deben realizar funciones como router, firewall o gateway.

```
net.ipv4.conf.all.accept_source_route = 0
net.ipv4.conf.all.accept_redirects = 0
net.ipv4.conf.all.secure_redirects = 0
net.ipv4.conf.all.log_martians = 1
net.ipv4.conf.default.accept_source_route = 0
net.ipv4.conf.default.accept_redirects = 0
net.ipv4.conf.default.secure_redirects = 0
net.ipv4.icmp_echo_ignore_broadcasts = 1
net.ipv4.icmp_ignore_bogus_error_messages = 1
net.ipv4.tcp_syncookies = 1
net.ipv4.conf.all.rp_filter = 1
net.ipv4.conf.default.rp_filter =1
```

Como aun no se está usando IPv6 también lo deshabilitaremos añadiendo a **sysctl.conf**.

```
net.ipv6.conf.all.disable_ipv6 = 1
net.ipv6.conf.default.disable_ipv6 = 1
net.ipv6.conf.lo.disable_ipv6 = 1
```

4.1.6. Configuración segura de servicios

SERVICIO SSH

Como siempre tendremos acceso físico al servidor es recomendable no tener el paquete que nos permite las conexiones en remoto mediante SSH. Si fuera necesario tener un servidor SSH instalado deberíamos limitar las direcciones IP que se pueden conectar mediante el firewall y establecer unas configuraciones de seguridad en el archivo **/etc/ssh/sshd_config** siendo todo lo siguiente:

- Comprobar que dicho archivo tiene una línea que pone **Protocol 2**.
- Añadir las siguientes líneas cuya acciones son las que se indica:
    1. Incluir los usuarios que pueden conectarse al servidor en remoto.
    2. Aumentar el número de bits de cifrado para las comunicaciones.
    3. Configurar el tiempo de cierre de una sesión a uno más seguro (por ejemplo 5 minutos o menos).
    4. Deshabilitar la emulación RSH.
    5. Deshabilitar la autenticación basadas en host.
    6. Deshabilitar el acceso del root en remoto.
    7. Deshabilitar el acceso mediante cuentas con claves vacías.

8. Deshabilitar autenticación mediante usuario y clave y habilitar mediante clave pública/privada. Para ellos además crearemos las claves en el sistema cliente y copiaremos el contenido al fichero **.ssh/authorized-keys** del directorio personal de dicho usuario que se encontrará en el servidor.

```
# ssh-keygen
```

9. Cambiar el puerto por defecto a usar para la conexión remota.

```
1 AllowUser tfm
2 ServerkeyBits 4096
3 ClientAliveInterval 300
  ClientAliveCountMax 0
4 IgnoreRhosts 0
5 HostbasedAuthentication no
6 PermitRootLogin no
7 PermitEmptyPasswords no
8 PubkeyAuthentication yes
  PasswordAuthentication no
  UsePAM no
9 Port 31892
```

SERVICIO CRON

CRON es un servicio que permite la ejecución programada de comandos o scripts, pudiendo cambiar la hora de ejecución en **/etc/crontab**. Este servicio debería de ser posible de usar solo por aquellos usuarios administradores, por lo tanto deberemos crear e introducir por consola para los ficheros **/etc/cron.allow** y **/etc/cron.deny** tal como se muestran.

```
 GNU nano 2.5.3              Archivo: cron.allow

root
```

```
 GNU nano 2.5.3              Archivo: cron.deny

ALL
```

NTP

El servicio **NTP** nos permite que todos los sistemas de nuestra red tengan la misma hora y fecha, lo cual es útil para llevar un control correcto de logs y eventos. Para esto deberemos instalar un nuevo paquete.

```
# apt-get install ntp --no-install-recommends
```

Y configurar el fichero **/etc/ntp.conf** con las líneas mostradas en la captura y reiniciamos el servicio.

```
restrict default kod nomodify notrap nopeer noquery
restrict 127.0.0.1
```

4.1.7. Monitorización del sistema

CONFIGURACIÓN DE RSYSLOG

En **rsyslog** tenemos centralizados el registro de sucesos. En una red sería conveniente tener centralizado todos los logs en un servidor, recogiendo así los logs de todos los dispositivos de nuestra red, para ello deberíamos configurarlos individualmente para que envíen sus logs al servidor indicado.

Además este servidor central para los logs deberemos configurarlo para que recepcione todos los mensajes syslog del resto de dispositivos de la red. En el archivo **/etc/rsyslog.conf** debemos quitar el comentario a las siguientes líneas.

```
# provides UDP syslog reception
module(load="imudp")
input(type="imudp" port="514")
```

Tras esto reiniciamos el servicio para que se apliquen los cambios. También es conveniente configurar la rotación de ficheros, es decir, el tiempo que estos permanecerán hasta que sean sobrescritos. Para nuestro caso guardaremos los logs durante 52 semanas.

```
# keep 52 weeks worth of backlogs
rotate 52
```

LOGCHECK

LOGCHECK es una aplicación que nos permite automatizar la revisión de los logs. Para ello enviará por email aquello que considere extraño o resúmenes cada cierto de tiempo según configuremos.

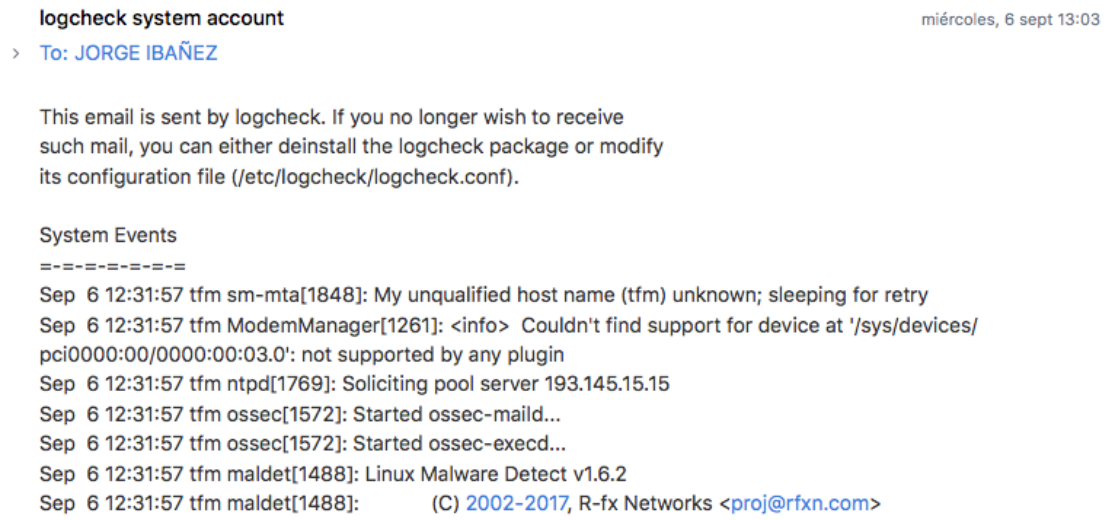
Comenzaremos con la instalación de su paquete.

```
# apt-get install logcheck --no-install-recommends
```

Deberemos indicar a que dirección queremos que envíe los emails, modificando el valor del campo **SENDMAILTO** en **/etc/logcheck/logcheck.conf**.

```
SENDMAILTO="usuario@dominio.com"
```

Esto es un fragmento de un correo enviado por **LOGCHECK** (por defecto estarán en la carpeta spam).



logcheck system account miércoles, 6 sept 13:03

› To: JORGE IBAÑEZ

This email is sent by logcheck. If you no longer wish to receive
such mail, you can either deinstall the logcheck package or modify
its configuration file (/etc/logcheck/logcheck.conf).

System Events
=-=-=-=-=-=-=-=
Sep 6 12:31:57 tfm sm-mta[1848]: My unqualified host name (tfm) unknown; sleeping for retry
Sep 6 12:31:57 tfm ModemManager[1261]: <info> Couldn't find support for device at '/sys/devices/pci0000:00/0000:00:03.0': not supported by any plugin
Sep 6 12:31:57 tfm ntpd[1769]: Soliciting pool server 193.145.15.15
Sep 6 12:31:57 tfm ossec[1572]: Started ossec-maild...
Sep 6 12:31:57 tfm ossec[1572]: Started ossec-execd...
Sep 6 12:31:57 tfm maldet[1488]: Linux Malware Detect v1.6.2
Sep 6 12:31:57 tfm maldet[1488]: (C) 2002-2017, R-fx Networks <proj@rfxn.com>

Si tuviéramos falsos positivos, estos deberían configurarse en **/etc/logcheck/ignore.d.server/**.

LOGWATCH

La misión de **LOGWATCH** es permitirnos estudiar aquellos logs que se consideran normales (entradas de usuarios, uso del disco, conexiones SSH, actividades programadas por CRON…). Esto nos permite ver si algún servicio es usado más de lo normal o por usuarios sin permisos.

```
# apt-get install logwatch --no-install-recommends
```

Tras ello copiaremos el fichero que se muestra a la ruta indicada.

```
# cp /usr/share/logwatch/default.conf/logwatch.conf /etc/logwatch/conf/
```

Como para **LOGCHECK,** deberemos modificar la dirección a la que queremos enviar el informe diario. En este caso modificaremos **/etc/logwatch/conf/logwatch.conf**.

```
TmpDir = /tmp
MailTo = usuario@dominio.com
```

4.1.8. Sistema de detección de intrusiones

OSSEC

OSSEC es un sistema de detección de intrusos basado en host que puede funcionar en local (1 solo servidor) o en modo agente/servidor (varios servidores en la red).

Para su instalación deberemos descargarnos sus fuentes de su página oficial e instalar el paquete **build-essential**. Descomprimimos el archivo descargado y lo instalamos.

```
# apt-get install build-essential --no-install-recommends

# wget https://codeload.github.com/ossec/ossec-hids/tar.gz/2.9.1

# tar zxvf 2.9.1

# cd ossec-hids-2.9.1

# ./install.sh
```

```
OSSEC HIDS v2.9.1 Guión de instalación - http://www.ossec.net

Usted esta por comenzar el proceso de instalación del OSSEC HIDS.
Usted debe tener un compilador de C previamente instalado en el sistema.

 - Sistema: Linux tfm 4.4.0-91-generic
 - Usuario: root
 - servidor: tfm


 -- Presione ENTER para continuar ó Ctrl-C para abortar. --


1- Que tipo de instalación Usted desea (servidor, agente, local ó ayuda)? servidor

  - Usted eligió instalación de Servidor.

2- Configurando las variables de entorno de la instalación.

 - Eliga donde instalar OSSEC HIDS [/var/ossec]:

    - La instalación se realizará en  /var/ossec .

3- Configurando el sistema OSSEC HIDS.

  3.1- Desea recibir notificación por correo electrónico? (s/n) [s]: s
   - Cuál es vuestra dirección de correo electrónico?
```

```
  3.2- Desea Usted agregar el servidor de integridad del sistema? (s/n) [s]: s

   - Ejecutando syscheck (servidor de integridad del sistema).

  3.3- Desea Usted agregar el sistema de detección de rootkit? (s/n) [s]: s

   - Ejecutando rootcheck (sistema de detección de rootkit).
strings: '/usr/bin/mail': No hay tal fichero

  3.4- Respuestas activas le permitirán ejecutar un comando
       específico en base a los eventos recibidos. Por ejemplo,
       Usted podra bloquear una dirección IP ó deshabilitar el acceso
       de un usuario específico.
       Más información en:
       http://www.ossec.net/en/manual.html#active-response

   - Desea Usted habilitar respuesta activa? (s/n) [s]: s


     - Respuesta activa habilitada.

   - Por omición, podemos habilitar el rechazo de servicio
     o el abandono del paquete por medio del Firewall.
     El rechazo agregara el ofendedor en el archivo etc/hosts.deny
     y el abandono bloquara la comunicación con el ofendedor en iptables
     (si el sistema fuera linux) ó ipfilter (si el sistema fuera
     Solaris, FreeBSD or NetBSD).

    - Las dos repuestas pueden ser utilizadas para detener un escaneo
      de fuerza bruta contra SSHD, escaneo de puertos y otras  formas
      de ataque. Por ejemplo Usted podra tambien agregar los ofensores
      de acuerdo a eventos registrados por medio de snort.


   - Desea Usted habilitar la respuesta desechar en el Firewall? (s/n) [s]: s

     - Respuesta desechar en el Firewall habilitada (local)  para niveles >= 6

   - Lista blanca para respuesta activa por omisión:
      - 192.168.0.1

   - Desea Usted agregar más IPs a la lista blanca? (s/n)? [n]: n

  3.5- Desea Usted habilitarsyslog remoto (puerto 514 udp)? (s/n) [s]: s

   - Syslog remoto habilitado.


   3.6- Estableciendo la configuración para analizar los siguientes registros:
     -- /var/log/auth.log
     -- /var/log/syslog
     -- /var/log/dpkg.log
     -- /var/log/apache2/error.log (apache log)
     -- /var/log/apache2/access.log (apache log)


  - Si Usted deseara monitorear algún otro registro, solo
    tendrá que editar el archivo ossec.conf y agregar una
    nueva entrada de tipo localfile.
    Cualquier otra pregunta de configuración podra ser
    respondida visitandonos en linea en http://www.ossec.net .

    --- Presione ENTER para continuar ---


  - Instalando el sistema
```

Tras esto eliminaremos el paquete **build-essential** (no es necesario para el funcionamiento de **OSSEC**) e iniciaremos el servicio **OSSEC**.

```
# apt-get remove --purge build-essential

# /var/ossec/bin/ossec-control start
```

4.1.9. Detección de virus, rootkits y webshells

ANTIVIRUS

Como es lógico un software de seguridad que debe tener cualquier sistema informático es un antivirus. En nuestro caso instalaremos **ClamAv**, antivirus de open source con protección contra trojanos, virus, malware y más agentes maliciosos (se deja enlace en ANEXOS).

Su instalación la realizaremos por el terminal escribiendo lo que se indica.

```
# apt-get install clamav clamav-base clamav-freshclam clamav-daemon --no-install-recommends
```

Para actualizar la base de antivirus lo haremos así.

```
# freshclam
```

Y para realizar algún scan con la siguiente instrucción obtendremos todas las opciones de configuración.

```
# clamscan --help
```

Un ejemplo sería esto.

```
root@tfm:/home/tfm/maldetect-1.6.2# clamscan -r -i /home/tfm/Descargas

----------- SCAN SUMMARY -----------
Known viruses: 6321699
Engine version: 0.99.2
Scanned directories: 1
Scanned files: 0
Infected files: 0
Data scanned: 0.00 MB
Data read: 0.00 MB (ratio 0.00:1)
Time: 14.546 sec (0 m 14 s)
```

DETECCIÓN DE ROOTKITS

El objetivo de un **rootkit** es obtener el control del equipo sin que este sea detectado. Por ello también se hace necesario instalar algún software que nos prevenga de este tipo de ataques.

Como nuestro servidor está montado sobre UBUNTU la herramienta que utilizaremos será **Chkrootkit**. El proceso de instalación será tan sencillo como se muestra.

```
# apt-get install chkrootkit --no-install-recommends
```

Y para ejecutarlo y realizar un test.

```
# chkrootkit -q
```

```
root@tfm:/home/tfm# chkrootkit -q

/lib/modules/4.4.0-91-generic/vdso/.build-id /lib/modules/4.4.0-62-generic/vdso/
.build-id
/lib/modules/4.4.0-91-generic/vdso/.build-id /lib/modules/4.4.0-62-generic/vdso/
.build-id
Possible Linux/Ebury - Operation Windigo installetd
```

Vemos como resultado que nos da un aviso por una posible infección maliciosa por Ebury, un troyano backdoor SSH. Este es instalado a nivel root sustituyendo los binarios relacionados con SSH o modificando la biblioteca compartida por SSH (**libkeyutils**).

A continuación comprobaremos si realmente estamos infectados o es un falso positivo. Dependiendo de la versión de Ebury, inferior a 1.5 o la 1.5 los pasos a realizar son distintos. Comenzaremos con la versión inferior a 1.5, siendo lo primero obtener un listado de los SHMs existentes y comprobando si el tamaño es mayor de 3 Mb y los permisos son 666.

```
# ipcs -m
```

```
root@tfm:/home/tfm# ipcs -m

---- Segmentos memoria compartida ----
key        shmid      propietario perms      bytes      nattch     estado
0x00000000 0          root       644        80         2
0x00000000 32769      root       644        16384      2
0x00000000 65538      root       644        280        2
0x3c81b7f5 163843     tfm        666        4096       0
0x00000000 491524     tfm        600        4194304    2          dest
0x00000000 524293     tfm        600        524288     2          dest
```

Vemos que para el SHM de **tfm** su permiso si es 666 pero su tamaño es inferior a 3 Mb. Aun así haremos más comprobaciones pues para la versión 1.3.5 los permisos de SHM se cambian a 600 y el tamaño del segmento debería ser bastante pequeño, caso que vemos que de nuevo el usuario **tfm** con distintos **shmid** tiene permiso de 600 pero su tamaño es bastante grande, así que podríamos descartarlo.

Pero más seguridad comprobaremos ahora la librería **libkeytuils,** cuyo tamaño normal se encuentra por debajo de los 15 kb, pero cuando es infectada este tamaño sobrepasa los 25 kb, por lo tanto esto es lo que debemos comprobar.

```
# find /lib* -type f –name libkeyutils.so* -exec ls –la {} \;
```

```
root@tfm:/home/tfm# find /lib* -type f -name libkeyutils.so* -exec ls -la {} \;
-rw-r--r-- 1 root root 14256 dic 10  2015 /lib/x86_64-linux-gnu/libkeyutils.so.1
.5
```

Con esto habríamos terminado la comprobación para las versiones de Ebury inferior a 1.5 y por lo tanto con total seguridad podemos descartar que tenemos una infección en ese rango, pero para certificar que es un falso positivo deberemos ahora comprobarlo para la versión 1.5.

Lo primero es comprobar si tenemos instalada la librería adicional **libns2.so**, pues se instala con estas versiones del rootkit.

```
# find /lib* -type f –name libns2.so
```

Si no se encuentra instalada no debería devolvernos nada por pantalla. Lo mismo ocurriría para el siguiente comando, en el cual tratamos de ver si se está usando un socket malicioso.

```
# netstat –nap | grep “@/proc/udevd”
```

Donde vemos que no nos devuelven nada como resultado y por lo tanto ya si podemos asegurar que tenemos un falso positivo por Ebury.

```
root@tfm:/home/tfm# find /lib* -type f -name libns2.so
root@tfm:/home/tfm# netstat -nap | grep "@/proc/udev"
root@tfm:/home/tfm#
```

DETECCIÓN DE WEBSHELLS

Para la detección de código malicioso en web tenemos varias herramientas disponibles gratuitas. Para nuestro servidor hemos instalado **Linux Malware Detector** conocida como **LMD**. A continuación indico todos los pasos en orden, pues la instalación no varía de otras que ya hemos realizado.

```
# wget http://www.rfxn.com/downloads/maldetect-current.tar.gz

# tar –xvf maldetect-current.tar.gz
```

Ahora deberemos saber como se ha nombrado el directorio en el que se ha descomprimido la aplicación para poder ejecutar el script de instalación que se encuentra en él.

```
# ls –l | grep maldetec
```

```
root@tfm:/home/tfm# ls -l |grep maldetect
drwxr-xr-x 3 root root    4096 jul 14 05:36 maldetect-1.6.2
-rw------- 1 root root 1605546 jul 14 06:45 maldetect-current.tar.gz
```

Ahora ya si podremos entrar en **maldetect-1.6.2** y ejecutar el script de instalación.

```
# cd maldetect-1.6.2

# ./install.sh
```

Tras finalizar la instalación podremos ver que se ha colocado un script en /etc/cron.daily para su ejecución diaria (además observamos que también está el correspondiente a nuestro **Chkrootkit**).

```
root@tfm:/home/tfm# cd /etc/cron.daily
root@tfm:/etc/cron.daily# ls
00logwatch  bsdmainutils      logrotate  notificacion_actualizaciones.sh
0anacron    chkrootkit        maldet     ntp
apache2     cracklib-runtime  man-db     passwd
apport      cron-apt          mdadm      popularity-contest
apt-compat  dpkg              mlocate    update-notifier-common
```

Debemos tener en cuenta que cuando realicemos algún escaneo en busca de virus con **ClamAv** obtendremos unos falsos positivos en relación a los archivos que se muestran a continuación, que forman parte de la instalación de **LMD**.

```
root@tfm:/etc/clamav# clamscan -r -i /home
/home/tfm/maldetect-1.6.2/files/clean/gzbase64.inject.unclassed: {HEX}gzbase64.i
nject.unclassed.15.UNOFFICIAL FOUND
/home/tfm/maldetect-1.6.2/files/sigs/rfxn.hdb: YARA.Safe0ver_Shell__Safe_Mod_Byp
ass_By_Evilc0der_php.UNOFFICIAL FOUND
/home/tfm/maldetect-1.6.2/files/sigs/rfxn.yara: {HEX}gzbase64.inject.unclassed.1
5.UNOFFICIAL FOUND
/home/tfm/maldetect-1.6.2/files/sigs/hex.dat: YARA.Safe0ver_Shell__Safe_Mod_Bypa
ss_By_Evilc0der_php.UNOFFICIAL FOUND
/home/tfm/maldetect-1.6.2/files/sigs/rfxn.ndb: YARA.Safe0ver_Shell__Safe_Mod_Byp
ass_By_Evilc0der_php.UNOFFICIAL FOUND
/home/tfm/maldetect-1.6.2/files/sigs/md5.dat: YARA.Safe0ver_Shell__Safe_Mod_Bypa
ss_By_Evilc0der_php.UNOFFICIAL FOUND
/home/tfm/maldetect-1.6.2/files/sigs/md5v2.dat: YARA.Safe0ver_Shell__Safe_Mod_By
pass_By_Evilc0der_php.UNOFFICIAL FOUND
/home/tfm/maldetect-current.tar.gz: YARA.Safe0ver_Shell__Safe_Mod_Bypass_By_Evil
c0der_php.UNOFFICIAL FOUND
```

4.1.10. Otros aspectos de seguridad

COPIAS DE SEGURIDAD Y RECUPERACIÓN

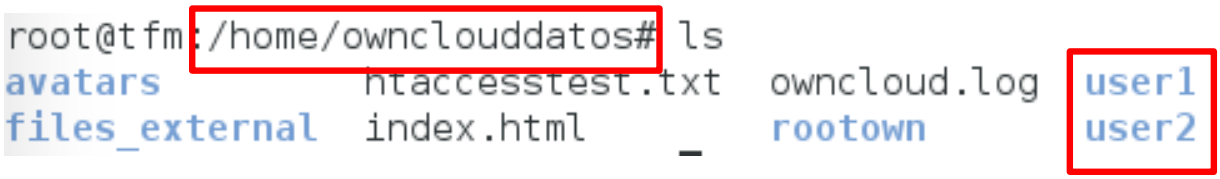
El tener un adecuado sistema de copias de seguridad asegura la supervivencia de nuestra información, elemento clave de nuestro servidor, junto a un correcto planeamiento que proporcione la recuperación en un tiempo razonable. Según la guía que estamos usando las copias de seguridad deberán realizarse de la siguiente manera:

- Realizar una copia incremental diaria de los directorios más importantes según la funcionalidad del servidor, como mínimo se incluirán de los directorios **/etc, /home,** y **/var**.
- Realizar una copia de todas las particiones del sistema una vez al mes.
- Las copias de seguridad deberán ejecutarse en horas en que el sistema tenga menos carga de trabajo.
- Las copias de seguridad se almacenaran cifradas.
- Las copias de seguridad deberán almacenarse a un sistema externo o distinto del que se está realizando la copia.
- Una vez al mes se extraerá la última copia seguridad y se almacenará en una localización segura distinta de donde se encuentran los servidores, para que en caso de catástrofe, se disponga de dicha copia de seguridad.

Para la realización y gestión de nuestras copias de seguridad podríamos usar la aplicación **Amanda Community** o **Bacula** junto a un disco exterior, ambas open source. En caso de tener varios servidores deberíamos tener un solo servidor dedicado a copias de seguridad. Si se diera el caso de que falle algún disco, deberíamos reemplazarlo y volcar las copias de

seguridad realizadas, siendo primero la última copia mensual y luego todas las copias incrementales diarias a partir de la última mensual hasta la última diaria incremental que tengamos. Además estos backups deberían estar cifrados.

En concreto para nuestro servidor se haría necesario realizar backups de la carpeta **/home/owncloudddatos,** en la cual se encuentra toda los datos de nuestros usuarios de la nube, y de las carpetas correspondientes a **MySql**. Debido a lo largo del proceso en esta guía no se explica la instalación y configuración del servidor de backups.



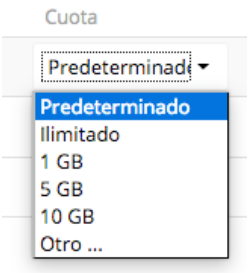
BORRADO SEGURO

Si el algún caso tenemos la necesidad de borrar datos, debido al uso que se le dará a este servidor este debe ser lo más seguro posible por tratarse en muchos casos de información confidencial o clasificada. Para ello usaremos el comando **shred** que sobrescribirá el archivo 25 veces. La forma de usarlo para un dispositivo y un archivo es como se indica en el ejemplo.

```
# shred /dev/dispositivo

#shred –n 10 –vfz --remove nombre  fichero  borrar
```

CUOTAS DE DISCO

Aunque podríamos limitar el uso del disco duro a través del paquete **quota**, al estar usando el software ownCloud nos permite hacerlo fácilmente mediante la cuenta root de este. Para ello yéndonos a usuarios podremos ver que todos tienen asignada una cuota por defecto, pues simplemente pulsando sobre esta se abrirá un desplegable y podremos indicar aquella que queramos asignarles.



4.1.11. Securización avanzada

AppArmor se define como Mandatory Access Control, es decir, se utiliza para el control de acceso obligatorio usando el módulo de seguridad de Linux LSM. El objetivo de tener este servicio instalado en nuestro servidor es dificultar cualquier acción que quiera realizar un atacante que ya se encuentre dentro de nuestro sistema y esto se consigue mediante el aislamiento de servicios respecto al control de acceso del kernel de Linux.

APPARMOR

Como es lógico lo primero será instalar los paquetes necesarios y una vez instalado procedemos a comprobar si está funcionando y su estado.

```
# apt-get install apparmor apparmor-utils --no-install-recommends

# apparmor_status
```

```
root@tfm:/# apparmor_status
apparmor module is loaded.
29 profiles are loaded.
29 profiles are in enforce mode.
   /sbin/dhclient
   /usr/bin/evince
   /usr/bin/evince-previewer
   /usr/bin/evince-previewer//sanitized_helper
   /usr/bin/evince-thumbnailer
   /usr/bin/evince-thumbnailer//sanitized_helper
   /usr/bin/evince//sanitized_helper
   /usr/bin/freshclam
   /usr/bin/lxc-start
   /usr/lib/NetworkManager/nm-dhcp-client.action
   /usr/lib/NetworkManager/nm-dhcp-helper
   /usr/lib/connman/scripts/dhclient-script
   /usr/lib/lxd/lxd-bridge-proxy
   /usr/lib/snapd/snap-confine
   /usr/lib/snapd/snap-confine//mount-namespace-capture-helper
   /usr/lib/telepathy/mission-control-5
   /usr/lib/telepathy/telepathy-*
   /usr/lib/telepathy/telepathy-*//pxgsettings
   /usr/lib/telepathy/telepathy-*//sanitized_helper
   /usr/lib/telepathy/telepathy-ofono
   /usr/sbin/clamd
   /usr/sbin/ippusbxd
   /usr/sbin/mysqld
   /usr/sbin/ntpd
   /usr/sbin/tcpdump
   lxc-container-default
   lxc-container-default-cgns
   lxc-container-default-with-mounting
   lxc-container-default-with-nesting
0 profiles are in complain mode.
2 processes have profiles defined.
2 processes are in enforce mode.
   /usr/lib/telepathy/mission-control-5 (2429)
   /usr/sbin/mysqld (1556)
0 processes are in complain mode.
0 processes are unconfined but have a profile defined.
```

Como podemos observar nos hablan de que los **profiles** y **processes** pueden encontrarse en dos estados distintos, **complain** y **enforce** mode:

- **Complain**: este estado se tiende a usar para pruebas y desarrollo de nuevos perfiles, ya que cuando es violada una política, la acción es permitida y se crea un log al respecto.
- **Enforce**: la diferencia respecto al interior es que este modo fuerza a cumplir la política pero también crea un log para registrar aquellas violaciones.

Si quisiéramos cambiar el estado de alguno de los perfiles la forma de actuar sería la siguiente, tomando como ejemplo el primer binario de la captura anterior **/sbin/dhclient** lo pasamos a modo **complain**.

```
# aa-complain /sbin/dhclient
```

Y ejecutando de nuevo el comando **apparmor_status** vemos que el cambio de modo se ha realizado.

```
1 profiles are in complain mode.
   /sbin/dhclient
```

Para volver al modo **enforce** deberíamos introducir por terminal esto.

```
# aa-enforce /sbin/dhclient
```

4.2. 660 – Securización de proxies

La instalación de un proxy en nuestro servidor nos dará un plus en la securización de este, siendo varios los aspectos que nos aporta:

- La entrada a zonas controladas será restringida.
- Antes los ataques a nivel de protocolo y/o aplicación nos ayudará a prevenirlos.
- Podremos usarlo para limitar los permisos de navegación por Internet de nuestros usuarios.
- En nuestro caso muy particular debido a la clasificación de la información que puede almacenar nuestro servidor, nos ayudará a prevenir la divulgación de esta, ya sea por accidente o intencionada.

Un proxy es un dispositivo que se instala entre una red e Internet, actuando como un filtro según las reglas que le hayan sido configuradas. Para nuestro servidor usaremos SQUID, que funciona como un proxy caché (almacena las páginas visitadas en internet por los usuarios), además lo usaremos para bloquear contenido dañino para nuestros equipos y evitando el acceso a páginas no recomendadas.

4.2.1. Instalación y configuración de SQUID

Como es común comenzaremos con la instalación de los paquetes necesarios.

```
# apt-get install squid
```

Lo siguiente será la configuración y para ello debemos conocer, sin llegar a profundizar, algunos de los parámetros más usados e importantes en el archivo de configuración de **SQUID** son los que podemos ver en las tres siguiente imágenes pertenecientes a la guía CCN-STIC 660 – Securización en proxies.

http_port. Este parámetro define en qué puerto responderá a las solicitudes **squid**. Los puertos registrados recomendados para servidores intermediarios (Proxies) pueden ser el 3128 y 8080 a través de TCP.

Si se desea incrementar la seguridad, puede vincularse el servicio a una IP a la que sólo se pueda acceder desde la red local. Considerando que el servidor utilizado posee una IP 10.140.110.1, puede hacerse lo siguiente:

```
# You may specify multiple socket addresses on multiple lines.

# Default: http port 3128

http port 10.140.110.1:3128

http_port 10.140.110.1:8080
```

icp_port. Este parámetro define el puerto en el que el servidor squid recibe solicitudes ICP (Inter-Caché Protocol).

hierarchy stoplist. Este parámetro es útil para indicar a squid qué páginas que contengan ciertos caracteres no deben almacenarse.

caché_mem. Memoria utilizada por squid para :

- Objetos en tránsito.
- Objetos frecuentemente utilizados (Hot).
- Objetos negativamente almacenados.

caché_swap_low. Indica el nivel en porcentaje de capacidad mínima aceptada por squid, es decir, los objetos se almacenarán hasta que se cope el límite mínimo.

caché_swap_high. Parámetro que especifica en porcentaje el límite máximo que utiliza squid para mantener objetos en la caché.

maximum_object_size. Este parámetro, especificado en KB, indica el tamaño máximo que se almacena en la caché.

caché dir. Directorio de ubicación de la caché, por defecto /usr/local/squid/caché. Este parámetro incluye tres parámetros numéricos adicionales.

caché_access_log. Especifica en que directorio se realizará el registro de accesos al squid.

caché_log. Define en dónde se almacenan los mensajes del sistema.

caché_store_log. Este parámetro especifica la ubicación del archivo de registro de objetos sacados de la caché. No es necesario activarlo. Es mejor desactivarlo para ahorrar espacio en disco: **caché_store_log none**

emulate_httpd_log. Este parámetro define si se desea utilizar emulación de logs del servidor web (httpd). Es importante activarlo con el fin de poder utilizar otros sistemas de análisis de estadísticas compatibles con el formato CLF (Common Log Format), el formato de registro (log) más usado por servidores web y proxies.

mime_table. Define la ubicación del archivo mime.conf, se utiliza el valor por defecto:

pid_filename. Define la ubicación del archivo squid.pid.

Figura 7. 4.2. Imagen 1 de parámetros más usados en SQUID según guía CCN-STIC 660

debug_options. Opciones de depuración.

reference_age. Este parámetro determina cuanto tiempo permanece el objeto en la caché.

quick_abort. Este parámetro define si un objeto debe almacenarse en la caché cuando el usuario ha interrumpido una solicitud.

negative ttl. Este parámetro se utiliza para definir cuanto tiempo debe esperar squid para procesar nuevamente una página que no ha sido encontrada.

positive_dns_ttl. Este parámetro especifica el tiempo que squid mantendrá la dirección de un sitio visitado exitosamente.

negative_dns_ttl. Especifica el tiempo que espera squid antes de intentar nuevamente determinar la dirección de un sitio solicitado y que no ha sido encontrado.

http access e icp access. Define una serie de permisos para acceso al servidor squid.

```
http_access deny manager
http_access allow CONNECT !SSL_ports
http access allow all
icp access allow all
miss_access allow all
```

caché_mgr. Definición del administrador del sistema.

caché_effective_user y **caché_effective_group**. Estos dos parámetros definen que usuario (user) y grupo (group) ejecuta squid. La versión de RPM utiliza el usuario squid y grupo squid. La versión de las fuentes utiliza el usuario nobody y grupo nogroup

RPM:

```
caché_effective_user squid
caché_effective_group squid
```

Fuentes:

```
caché_effective_user nobody
caché_effective_group nogroup
```

Es importante verificar que el grupo **nogroup** exista en su sistema mirando el archivo group ubicado en el directorio /etc. Este archivo debe tener una entrada como se muestra a continuación:

```
nogroup:x:500:
```

El número que aparece al final de la línea debe ser único y corresponder al siguiente número disponible en su servidor.

visible_hostname. Este parámetro define el nombre del servidor. Coloque aquí el nombre de su servidor:

```
visible_hostname miproxy
```

Figura 8. 4.2. Imagen 2 de parámetros más usados en SQUID según guía CCN-STIC 660

httpd accel uses host header. Este parámetro se utiliza para activar el proxy transparente, necesario para controlar el acceso a Internet desde las estaciones.

```
httpd_accel_uses_host_header on
```

httpd accel host y **httpd accel port**. Estos dos parámetros son necesarios para activar el proxy transparente. Al configurar squid de esta manera, no es necesario realizar la configuración de los navegadores, lo que facilita su despliegue.

```
httpd accel host virtual
httpd_accel_port 80
```

httpd_accel_with_proxy. Este parámetro es necesario activarlo para el proxy transparente:

```
httpd_accel_with_proxy on
```

dns_testnames. Este parámetro define que hosts se utilizan para chequear el servidor de nombres.

append_domain. Este parámetro indica a squid que dominio debe añadirse a solicitudes que vengan sin dominio completo.

cachémgr passwd. Squid permite ser administrado desde una página web una vez instalado

request_body_max_size. Este parámetro limita el tamaño máximo de peticiones POST que pueden realizarse a través de Squid. Se debe configurar convenientemente, por ejemplo:

```
request_body_max_size 1 MB
```

Figura 9. 4.2. Imagen 3 de parámetros más usados en SQUID según guía CCN-STIC 660

Bueno ya con unos breves conocimientos nos iremos a su archivo de configuración en **/etc/squid/squid.conf**.

```
# nano /etc/squid/squid.conf
```

Y una vez lo tenemos en modo edición comenzaremos a escribir las siguientes líneas. Comenzamos limitando el tamaño máximo de peticiones POST en 1 MB.

```
request_body_max_size 1MB
```

Ahora estableceremos la memoria caché que podrá usar SQUID y el espacio en disco duro igualmente, teniendo que ser mayor que el tamaño de la caché.

```
cache_mem 256 MB
cache_dir ufs /var/spool/squid 512 16 256
```

4.2.2. Creación de Listas de Control de Acceso

Las Listas de Control de Acceso (ACL) es una forma de autenticación de las muchas que permite SQUID. En esta usamos los usuarios validados en nuestro servidor Ubuntu para así poder controlar sus conexiones a Internet. Con estas listas controlaremos los distintos accesos que queramos permitir o denegar a través de nuestro proxy.

Dentro de la propia carpeta de squid creamos una **/etc/squid/listas_acl** donde guardar las listas que creemos.

```
# mkdir /etc/squid/listas_acl
```

Una vez posicionados dentro de ésta vamos a crear algunas listas. Haremos tres en concreto para comenzar. La primera será **hosts.acl** y contendrá la IP de nuestra red local.

```
##RANGO DE IP DE HOSTS DE MI RED LOCAL

 192.168.0.0/24
```

La segunda será **sitios_deny.acl** y contendrá aquellos sitios de Internet a los cuales se denegará el acceso a los usuarios.

```
#SITIOS QUE SERÁ DENEGADO EL ACCESO

 .yahoo.com
```

Y la tercera **palabras_deny.acl** donde incluiremos aquellas palabras que denegarán el acceso a sitios en concreto a nuestros usuarios.

```
# PALABRAS PROHIBIDAS EN INTERNET
sexo
torrent
desnudos
```

También podríamos bloquear extensiones determinadas, impidiendo así que nuestros usuarios se descarguen de Internet material que podría ser peligroso para nuestra red. Podríamos poner **\.avi?$** pero eso solo denegaría los archivos que finalizaran en esa extensión, es decir que si un archivo es película.avi1 pasaría nuestros filtros y se descargaría, pero al colocar las extensiones como se muestran, denegará cualquier url de descarga que contenga dicha extensión independientemente de cómo finalice o donde se encuentre.

```
# EXTENSIONES QUE NO SE TENDRÁ ACCESO

\.avi(\?.*)?$
\.mp4(\?.*)?$
\.mp3(\?.*)?$
\.exe(\?.*)?$
\.mov(\?.*)?$
\.wma(\?.*)?$
\.rpm(\?.*)?$
\.tar(\?.*)?$
\.bat(\?.*)?$
\.pif(\?.*)?$
\.zip(\?.*)?$
\.rar(\?.*)?$
```

También podríamos crear listas para los puertos teniendo que poner únicamente su número.

```
# PUERTOS QUE SI PERMITIMOS
443
31892
80
21
70
210
280
488
591
777
```

Como vemos podríamos crear listas que contuvieran todas aquellas palabras, sitios, extensiones que queremos denegar o permitir. Podríamos limitar el uso de Internet para unos usuarios y a otros darle total libertad, todo dependerá de nuestras necesidades y los privilegios que queramos dar. Por lo tanto tras la creación de nuestras **listas ACL** deberemos configurar **/etc/squid/squid.conf** para que haga uso de ellas, además añadiremos algunos parámetros más. Aquí tenemos un ejemplo de lo aplicado a nuestro **SQUID**.

```
request_body_max_size 1 MB
cache_mem 256 MB
cache_dir ufs /var/spool/squid 512 16 256


acl CONNECT method CONNECT
acl download method CONNECT
acl download method GET
acl all src all
acl manager proto cache_object

# PUERTOS PERMITIDOS Y DENEGADOS
acl puertos_seguros port "/etc/squid/listas_acl/puertos_seguros.acl"
acl puertos_deny port "/etc/squid/listas_acl/puertos_deny.acl"

# CONTIENE LAS DIRECCIONES IP QUE TIENEN ACCESO A INTERNET
acl localnet src "/etc/squid/listas_acl/hosts.acl"

# DIRECCIONES WEB DENEGADAS
acl sitios_deny dstdomain "/etc/squid/listas_acl/sitios_deny.acl"

# PALABRAS DENEGADAS EN LOS DOMINIOS DE INTERNET
acl palabras_deny url_regex "/etc/squid/listas_acl/palabras_deny.acl"

# LIMITAMOS DIAS Y HORAS DE CONEXION A INTERNET
acl con_internet time MTWHF 10:00-17:30

# EXTENSIONES QUE NO SE PODRAN DESCARGAR DE INTERNET
acl extensiones urlpath_regex -i "/etc/squid/listas_acl/extensiones.acl"

# INDICO LA IP Y PUERTO A USAR POR SQUID
http_port 192.168.0.20:3128

# http_access deny all
http_access allow localnet !sitios_deny !palabras_deny con_internet !extensiones
http_access deny all

# SOLICITUDES DE NEGACIÓN A CIERTOS PUERTOS NO SEGUROS
http_access deny !puertos_seguros
http_access deny puertos_deny

# CONEXIÓN DENEGADA A OTROS PUERTOS SSL DISTINTOS DE 443
http_access deny CONNECT !SSL_ports
http_access deny CONNECT puertos_deny

http_access allow manager localhost
http_access deny manager
```

Podemos observar que para permitir o denegar acceso tenemos varias formas:

- Permitimos
  - http_access ALGO
  - http_deny !ALGO (al ser una doble negación realmente damos permiso)
- Denegamos
  - http_access !ALGO
  - http_deny ALGO

A continuación comprobaremos que nuestro servidor proxy está funcionando. No debemos olvidar que debemos redireccionar nuestros navegadores a nuestro servidor 192.168.0.20 y al puerto que usa **SQUID**, el cual como no lo hemos cambiado, será por defecto 3128.

Observamos como se nos deniega el acceso al dominio de yahoo.com tal como habíamos configurado.

En este caso observamos que esta página a pesar de estar dedicada a deporte y salud, MensHealth, se nos prohíbe el acceso al artículo por contener la palabra sexo.

Podemos añadir más configuraciones a nuestro SQUID como limitar las horas y días que los usuarios pueden conectarse a Internet. Este ejemplo en concreto limita el uso de internet todos los días de la semana (están puestos de lunes a domingo y con la inicial que en SQUID hace referencia a cada uno) de 12 a 12:30 pm.

```
acl sin_internet time MTWHFAS 12:00-12:30
```

Como ya sabemos para llevar un buen control de nuestro servidor y todas sus facetas, como este proxy, es importante observar los logs. Por defecto todos los logs de SQUID son almacenados en **/var/log/squid/access.log.**

```
1504779393.704      0 192.168.0.26 TCP_DENIED/403 5317 GET http://as.com/favicon.ico - HIER
_NONE/- text/html
1504779613.847     25 192.168.0.26 TCP_DENIED/403 3986 CONNECT 192.168.0.20:443 - HIER_NONE
/- text/html
1504779619.891      0 192.168.0.26 TCP_DENIED/403 3986 CONNECT 192.168.0.20:443 - HIER_NONE
/- text/html
1504779970.222      0 192.168.0.26 TCP_DENIED/403 4019 CONNECT safebrowsing.google.com:443
- HIER_NONE/- text/html
```

Así analizándolo podremos ver los accesos denegados y desde que IP se ha intentado acceder a una dirección, recurso, puerto o en horario prohibido.

4.3. 661 – Seguridad en Firewalls de Aplicación Web Modsecurity

Para comprobar la efectividad de nuestro **Firewall de Aplicación Web** (**WAF**), instalaremos la herramienta **Mutillidae**. Esta herramienta web es un proyecto realizado cuyo objetivo es dejarnos atacarla y observar así las vulnerabilidades existentes, convirtiéndose en una excelente herramienta de instrucción para la ciberseguridad. Como veremos se basa en las reglas OWASP de 2007 a 2017.

Para comenzar nos iremos a **/var/www/html/** y una vez allí descargaremos **Mutillidae**.

```
# git clone git://git.code.sf.net/p/mutillidae/git mutillidae
```

Una vez se ha realizado la descarga deberemos de darle los permisos adecuados y cambiar su grupo y usuario a **www-data**.

```
# chown –R www-data:www-data mutillidae

# chmod –R 755 mutillidae
```

Tras esto podremos acceder a su página de inicio **/mutillidae/index.php** a través de nuestro servidor.

Ahora si comenzaremos con nuestro firewall web, instalando los paquetes necesarios para **Modsecurity**. Este es un firewall de aplicaciones Web que se ejecuta como un módulo más de nuestro servidor Apache. Nos da protección contra ataques hacia nuestras aplicaciones web pudiendo además monitorizar el tráfico HTTP y realizar análisis en tiempo real.

```
# apt-get install libapache2-mod-security2 libpache2-modsecurity libapache2-mod-evasive
```

Una vez finalizada la instalación será necesario cambiar el nombre del archivo **modsecurity.conf-recommended** por el mismo pero sin **recommended**.

```
# cp /etc/modsecurity/modsecurity.conf-recommended /etc/modsecurity/modsecurity.conf
m
```

Y editamos este fichero buscando las 4 líneas que se muestran y dejándolas tal cual la captura.

```
SecRuleEngine On
SecRequestBodyLimit 20971520
SecRequestBodyNoFilesLimit 20971520
SecResponseBodyAccess Off
```

Luego tocará descargar las reglas de **OWASP Modsecurity Core Rule Set**.

```
# wget https://github.com/spiderlabs/owasp-modsecurity-crs/archive/2.2.9.tar.gz

# gzip –d 2.2.9.tar.gz

# tar xvf 2.2.9.tar
```

Ahora movemos el fichero **modsecurity-crs** para tener una copia de seguridad y le cambiaremos su nombre.

```
# mv /usr/share/modsecurity-crs /usr/share/modsecurity-crs.bak
```

Tras esto colocaremos en el directorio anterior las reglas descargadas de OWASP ya que son las que emplearemos, asignándoles el nombre **modsecurity-crs**. Cambiaremos el nombre al fichero **/usr/share/modsecurity-crs/modsecurity_crs_10_setup.conf.example** y crearemos un enlace a todos los archivos con extensión **conf** en la ruta que se muestra.

```
# mv owasp-modsecurity-crs-2.2.9 /usr/share/modsecurity-crs

# mv /usr/share/modsecurity-crs/modsecurity_crs_10_setup.conf.example
/usr/share/modsecurity-crs/modsecurity_crs_10_setup.conf

# ln –s /usr/share/modsecurity-crs/base_rules/*.conf /usr/share/modsecurity-
crs/activated_rules/
```

Lo siguiente será comentar las líneas mostradas en el archivo **/usr/share/modsecurity-crs/activated_rules/modsecurity_crs_35_bad_robots.conf**.

```
#SecRule REQUEST_HEADERS:User-Agent "@pmFromFile modsecurity_35_scanners.data" \
        "phase:2,rev:'2',ver:'OWASP_CRS/2.2.9',maturity:'9',accuracy:'9',t:none$

#SecRule REQUEST_HEADERS:User-Agent "@pmFromFile modsecurity_35_bad_robots.data$
        "chain,phase:2,rev:'2',ver:'OWASP_CRS/2.2.9',maturity:'9',accuracy:'9',$

#SecRule REQUEST_HEADERS:User-Agent "(?i:(?:c(?:o(?:n(?:t(?:entsmartz|actbot/)|$
```

También comentaremos una línea en **/usr/share/modsecurity-crs/activated_rules/modsecurity_crs_40_generics_attacks.conf**. Es la única que contiene el nombre del fichero en ella, ya que existen multitud de líneas muy similares.

```
#SecRule REQUEST_COOKIES|!REQUEST_COOKIES:/__utm/|REQUEST_COOKIES_NAMES|ARGS_NAMES|ARGS|XML:/*
 "@pmFromFile modsecurity_40_generic_attacks.data" \
 "phase:2,id:'981133',rev:'2',ver:'OWASP_CRS/2.2.9',maturity:'9',accuracy:'9',t:none,t:htmlEntityDecode,t:
  compressWhiteSpace,t:lowercase,nolog,pass,setvar:tx.pm_score=+1"
```

E igual para **/usr/share/modsecurity-crs/activated_rules/modsecurity_crs_50_outbound.conf**. La línea también contendrá como en el caso anterior el nombre del propio fichero.

```
#SecRule RESPONSE_BODY "!@pmFromFile modsecurity_50_outbound.data" \
        "phase:4,rev:'2',ver:'OWASP_CRS/2.2.9',maturity:'9',accuracy:'9',pass,id:'981178',t:none,capture$
```

Siguiendo con la configuración ahora tocará añadir también unas líneas a **/etc/apache2/mods-enabled/security2.conf**.

```
IncludeOptional /etc/modsecurity/*.conf
IncludeOptional "/usr/share/modsecurity-crs/*.conf"
IncludeOptional "/usr/share/modsecurity-crs/activated_rules/*.conf"
```

Tras esto lo haremos con **/etc/apache2/mods-enabled/evasive.conf**.

```
<IfModule mod_evasive20.c>
    DOSHashTableSize    3097
    DOSPageCount        10
    DOSSiteCount        30
    DOSPageInterval     1
    DOSSiteInterval     3
    DOSBlockingPeriod   3600
    DOSLogDir           "/var/log/apache2/mod_evasive.log"
```

Creamos el archivo mod_evasive.log.

```
# touch /var/log/apache2/mod_evasive.log
```

Por último reiniciamos el servidor y debemos comprobar que **Modsecurity** está funcionando adecuadamente.

```
# apachectl –M | grep security2

# apachectl –M | grep evasive
```

Podría darse el caso que nos salieran estos avisos.

```
root@tfm:/etc/apache2/mods-enabled# apachectl -M | grep security2
AH00558: apache2: Could not reliably determine the server's fully
 qualified domain name, using 127.0.1.1. Set the 'ServerName' dir
ective globally to suppress this message
 security2_module (shared)
root@tfm:/etc/apache2/mods-enabled# apachectl -M | grep evasive
AH00558: apache2: Could not reliably determine the server's fully
 qualified domain name, using 127.0.1.1. Set the 'ServerName' dir
ective globally to suppress this message
 evasive20_module (shared)
```

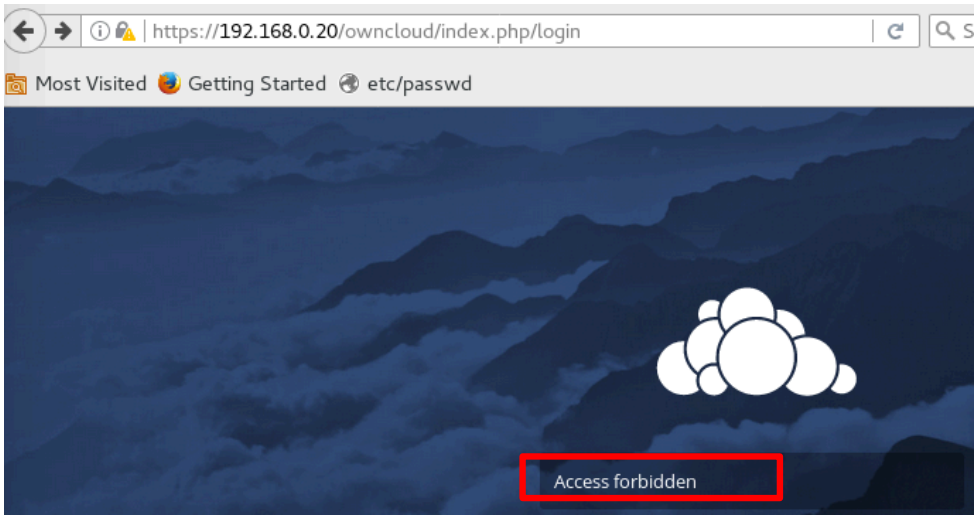
Para solucionarlos bastará con introducir en el terminal esto.

```
# echo “ServerName localhost” >> /etc/apache2/conf-available/fqdn.conf

# a2enconf fqdn

# service apache2 reload
```

Estas instrucciones nos crearán el archivo **fqdn.conf** con el texto **ServerName localhost**, luego arrancará el servicio necesario **fqdn** y tras reiniciar el servidor este se activará, de manera que al comprobar de nuevo si **Modsecurity** corre de manera óptima vemos que los avisos ya no aparecen.

```
root@tfm:/etc/apache2/mods-enabled# apachectl -M | grep security2
 security2_module (shared)
root@tfm:/etc/apache2/mods-enabled# apachectl -M | grep evasive e
vasive20_module (shared)
```

Con esto estaría instalado nuestro firewall **Modsecurity**. Así que lo siguiente será tratar de navegar y comprobar su funcionamiento. Comenzaremos accediendo a la página inicial de ownCloud. Como vemos el acceso nos ha sido prohibido.



Vamos a tratar de acceder ahora a nuestro **phpMyAdmin**. De nuevo observamos que el acceso nos ha sido vetado.

Por lo tanto necesitamos y debemos saber que regla es la que nos impide acceder a nuestro servidor a través de la IP y para ello debemos consultar en **/var/log/apache2/modsec_audit.log**

```
HTTP/1.1 403 Forbidden
Strict-Transport-Security: max-age=15552000; includeSubDomains
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Content-Security-Policy: default-src 'self'; script-src 'self' 'unsafe-eval'; style-src 'self' 'unsafe-inline'; fr
ame-src *; img-src * data: blob:; font-src 'self' data:; media-src *; connect-src *
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
X-Robots-Tag: none
X-Frame-Options: SAMEORIGIN
X-Download-Options: noopen
X-Permitted-Cross-Domain-Policies: none
Content-Length: 6636
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8

Message: Access denied with code 403 (phase 2). Pattern match "^[\\d.:]+$" at REQUEST_HEADERS:Host. [file "/usr/sh
are/modsecurity-crs/activated_rules/modsecurity_crs_21_protocol_anomalies.conf"] [line "98"] [id "960017"] [rev "2
"] [msg "Host header is a numeric IP address"] [data "192.168.0.20"] [severity "WARNING"] [ver "OWASP_CRS/2.2.9"]
[maturity "9"] [accuracy "9"] [tag "OWASP_CRS/PROTOCOL_VIOLATION/IP_HOST"] [tag "WASCTC/WASC-21"] [tag "OWASP_TOP_
10/A7"] [tag "PCI/6.5.10"] [tag "http://technet.microsoft.com/en-us/magazine/2005.01.hackerbasher.aspx"]
```

Como vemos esta regla tiene la **id 960017**, la cual usaremos para poder desactivarla accediendo a **/usr/share/modsecurity-crs/modsecurity_crs_10_setup.conf** y añadiendo al final del archivo lo que se muestra a continuación.

```
<LocationMatch .*>
        <IfModule mod_security2.c>
                SecRuleRemoveById 960017
        </IfModule>
</LocationMatch>
```

Reiniciamos el servidor y tratando de acceder de nuevo a las url anteriores veremos que no tendremos ningún problema para hacerlo.

Seguiremos probando y trataremos de acceder a otra url para seguir comprobando el funcionamiento de **Modsecurity**, en concreto a **/mutillidae/index.php?page=../../../../../etc/passwd**.

De nuevo vemos que nos restringe el acceso a la carpeta y si de nuevo fuéramos al archivo **/var/log/apache2/modsec_audit.log** podremos comprobar qué regla nos impide este tipo de acceso.

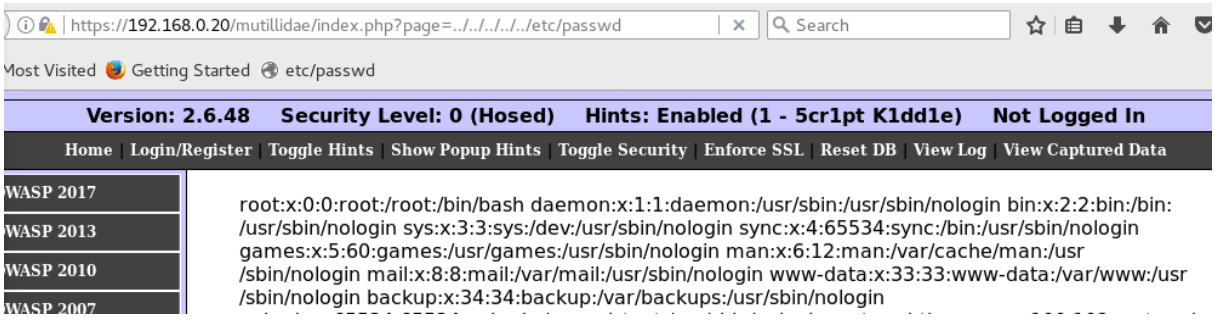
```
Message: Access denied with code 403 (phase 2). Pattern match "\\W{4,}" at ARGS$
 [id "960024"] [rev "2"] [msg "Meta-Character Anomaly Detection Alert -
 Repetative Non-Word Characters"] [data "Matched Data: ../../../../../ found
 within ARGS:page: ../../../../../etc/passwd"] [ver "OWASP_CRS/2.2.9"] [maturit$
```

```
Message: Access denied with code 403 (phase 2). Pattern match "(?:\\b(?:\\.
(?:ht(?:access|passwd|group)|www_?acl)|global\\.asa|httpd\\.conf|boot\\.ini)\\b|
\\/etc\\/)" at ARGS:page. [file "/usr/share/modsecurity-crs/activated_rules/
modsecurity_crs_40_generic_attacks.conf"][line "205"] [id "950005"] [rev "3"]
 [msg "Remote File Access Attempt"] [data "Matched Data: /etc/ found within ARGS:
page: ../../../../../etc/passwd"] [severity "CRITICAL"] [ver "OWASP_CRS/2.2.9"]
 [maturity "9"] [accuracy "9"] [tag "OWASP_CRS/WEB_ATTACK/FILE_INJECTION"]
 [tag "WASCTC/WASC-33"] [tag "OWASP_TOP_10/A4"] [tag "PCI/6.5.4"]
```

```
Message: Access denied with code 403 (phase 2). Pattern match "(?i)(?:\\x5c|(?:%
(?:2(?:5(?:2f|5c)|%46|f)|c(?:0%(?:9v|af)|1%1c)|u(?:221[56]|002f)|%32(?:%46|F)
|e0%80%af|1u|5c)|\\/))(?:%(?:2(?:(?:52)?e|%45)|(?:e0%8|c)0%ae|u(?:002e|2024)|%32
(?:%45|E))|\\.){2}(?:\\x5c|(?:%(?:2(?:5(?:2f|5c)|%46|f)|c(?:0%(?:9v|af)|1%1c)|
 ..." at REQUEST_URI. [file "/usr/share/modsecurity-crs/activated_rules/
modsecurity_crs_42_tight_security.conf"] [line "20"] [id "950103"] [rev "2"]
 [msg "Path Traversal Attack"] [data "Matched Data: /../ found within REQUEST_URI:
 /mutillidae/index.php?page=../../../../../etc/passwd"] [severity "CRITICAL"]
 [ver "OWASP_CRS/2.2.9"] [maturity "9"] [accuracy "7"] [tag "OWASP_CRS/WEB_ATTACK/$
```

En caso que desactiváramos estas tres reglas (960024, 950005 y 95013) podríamos acceder a la información que se muestra en relación a los usuarios de nuestro servidor.

```
<LocationMatch .*>
        <IfModule mod_security2.c>
                SecRuleRemoveById 960017
                SecRuleRemoveById 960024
                SecRuleRemoveById 950005
                SecRuleRemoveById 950103
        </IfModule>
</LocationMatch>
```



Por lo tanto sabiendo como desactivar reglas, deberíamos de hacerlo con aquellas que impidan el funcionamiento de nuestra nube pero sin afectar a la seguridad del servidor. Como es lógico para montar este servidor correctamente lo suyo sería tener un dominio y restringir así también el acceso por IP, pero para este trabajo si lo permitiremos.

4.4. 665 – Configuración segura de SSH

Secure Shell (**SSH**) es un protocolo que se usa para poder acceder remotamente a un equipo e interactuar con él mediante su terminal. Desde su creación ha evolucionado y mejorado muchísimo su seguridad pero sigue siendo un punto crítico en cualquier servidor debido sobre a todo a una mala configuración.

Un claro ejemplo es usar contraseñas para la identificación de los usuarios en vez de certificados SSH. El tipo de ataque favorito es el realizado por fuerza bruta, usando bibliotecas con los nombres y claves más comunes usadas por los usuarios.

Para este trabajo instalaremos la aplicación **Fail2band**, la cual está escrita en Python y cuya labor consiste en la prevención de intrusos en un sistema, penalizando o bloqueando las conexiones remotas que intentan accesos por fuerza bruta.

Instalaremos la aplicación y haremos una copia de seguridad de su archivo de configuración, llamándola **jail.local** en su propio directorio.

```
# apt-get install fail2ban

# cd /etc/fail2ban/

# cp jail.conf jail.local
```

Una vez hecho esto comenzamos a configurar nuestro **/etc/fail2ban/jail.local**. Lo que haremos será añadir todas esto texto al archivo, borrando todo el contenido anterior.

```
##To block failed login attempts use the below jail.
[apache]
enabled  = true
port     = http,https
filter   = apache-auth
logpath  = /var/log/apache2/*error.log
maxretry = 3
findtime = 300
bantime = 600
ignoreip = 192.168.0.0/24

##To block the remote host that is trying to request suspicious URLs, use the below jail.
[apache-overflows]
enabled  = true
port     = http,https
filter   = apache-overflows
logpath  = /var/log/apache2/*error.log
maxretry = 3
findtime = 300
bantime = 600
ignoreip = 192.168.0.0/24
```

```
##To block the remote host that is trying to search for scripts on the website
# to execute, use the below jail.
[apache-noscript]
enabled  = true
port     = http,https
filter   = apache-noscript
logpath  = /var/log/apache2/*error.log
maxretry = 3
bantime = 600
ignoreip = 192.168.0.0/24

##To block the remote host that is trying to request malicious bot, use below jail.
[apache-badbots]
enabled  = true
port     = http,https
filter   = apache-badbots
logpath  = /var/log/apache2/*error.log
maxretry = 3
findtime = 300
bantime = 600
ignoreip = 192.168.0.0/24


##To stop DOS attack from remote host.
[http-get-dos]
enabled = true
port = http,https
filter = http-get-dos
logpath  = /var/log/apache*/access.log
maxretry = 3
findtime = 300
bantime = 200
ignoreip = 192.168.0.0/24
action = iptables[name=HTTP, port=http, protocol=tcp]
destemail = jorgemibavaz@gmail.com
sendername = Fail2ban
mta = sendmail

##To block the failed login attempts on the SSH server, use the below jail.
[ssh]
enabled  = true
port     = 31892
filter   = sshd
logpath  = /var/log/auth.log
maxretry = 3
findtime = 300
bantime = 600
ignoreip = 192.168.0.0/24
destemail = jorgemibavaz@gmail.com
sendername = Fail2ban
mta = sendmail


[sshd]
enabled = true
port = 31892
filter = sshd
logpath = /var/log/auth.log
maxretry = 3
findtime = 300
bantime = 600
ignoreip = 192.168.0.0/24
destemail = jorgemibavaz@gmail.com
sendername = Fail2ban
mta = sendmail
#action = %(action_mwl)s
```

A continuación haré una breve explicación de algunos de los campos que hemos añadido para comprender mejor que estamos haciendo:

- **Enabled**: nos permite activarlo mediante true o apagarlo con false.
- **Port**: indicamos el puerto que usará.
- **Filter**: el filter hace referencia a los archivos que se encuentran en **/etc/fail2ban/filter.d**.
- **Logpath**: es la ruta hacia el archivo que guardará los logs.
- **Maxretry**: número de intentos que tiene el usuario para loguearse antes de bloquearle la cuenta.
- **Findtime**: periodo de tiempo en el que contaremos los intentos, es decir, tal como está ahora una cuenta será baneada si un usuario comete 3 intentos fallidos en 300 segundos.
- **Bantime**: tiempo en segundos que estará bloqueada una cuenta tras fallar un usuario todos sus intentos.
- **Ignoreip**: dirección IP que no se verá afectada por la regla.

Otros aspectos a tener en cuenta son los siguientes:

- Por seguridad al no usar **root** para conectarnos por **SSH** debemos crear un usuario nuevo que nos permita administrar el sistema, en nuestro caso es **admin_ssh** y agregarlo al grupo **root**.

```
# adduser admin_ssh root
# passwd admin_ssh
```

  Le crearemos su propia carpeta en **/home/admin_ssh** y le daremos los permisos apropiados.

```
# mkdir /home/admin_ssh
# chown admin_ssh:root /home/admin_ssh
```

- Debemos comprobar que hemos incluido los servicios **ssh**, **sshd**, **openssh** en **/etc/hosts.allow**

```
ssh, sshd, openssh-server, mysql-server-core-5.7, vsftpf : ALL
```

- Y que al mismo tiempo hemos negado todos en **/etc/hosts.deny,** de tal forma que solo los añadidos en **allow** podrán conectarse al tener preferencia sobre **deny.**

```
ALL: ALL
```

- Como se puede observar en las capturas anteriores hemos cambiado el puerto a usar para conectarnos por **SSH** de 22 al **31892**, de esta manera evitamos un ataque directo contra el puerto por defecto.

Para conectarnos a nuestro servidor a través de otro equipo que pertenece a nuestra red debemos escribir lo siguiente.

```
# ssh –v –p 31892 admin_ssh@192.168.0.20
```

Tras esto vemos que desde la terminal del equipo cliente nos hemos podido conectar sin ningún problema.

```
user@user-VirtualBox:~$ ssh -v -p 31892 admin_ssh@192.168.0.20
OpenSSH_7.2p2 Ubuntu-4ubuntu2.2, OpenSSL 1.0.2g  1 Mar 2016
debug1: Reading configuration data /etc/ssh/ssh_config
debug1: /etc/ssh/ssh_config line 19: Applying options for *
debug1: Connecting to 192.168.0.20 [192.168.0.20] port 31892.
debug1: Connection established.
```

Pero si intentamos conectarnos con algún usuario que no existe en el servidor o incluso existiendo este, fallamos varias veces al introducir su password, **iptables** se encargará de denegarlos el acceso. Para ello vemos el contenido de **/var/log/auth.log**.

```
 17:12:08 tfm sshd[1682]: pam_unix(sshd:auth): authentication failure; logname= uid=0 euid=0 tty=ssh ruser= rhos
168.0.33  user=admin_ssh
 17:12:10 tfm sshd[1680]: error: PAM: Authentication failure for admin_ssh from 192.168.0.33
 17:12:12 tfm sshd[1683]: pam_unix(ss                                                    ty=ssh ruser= rhos
168.0.33  user=admin_ssh
 17:12:13 tfm sshd[1680]: error: PAM: Authentication failure for admin_ssh from 192.168.0.33
 17:12:14 tfm sshd[1684]: pam_unix(ss                                                    ty=ssh ruser= rhos
168.0.33  user=admin_ssh
 17:12:16 tfm sshd[1680]: error: PAM: Authentication failure for admin_ssh from 192.168.0.33
 17:12:19 tfm sshd[1680]: pam_unix(sshd:auth): authentication failure; logname= uid=0 euid=0 tty=ssh ruser= rhos
168.0.33  user=admin_ssh
 17:12:21 tfm sshd[1680]: Failed password for admin_ssh from 192.168.0.33 port 43084 ssh2
 17:12:26 tfm sshd[1680]: message repeated 2 times: [ Failed password for admin_ssh from 192.168.0.33 port 43084

 17:12:26 tfm sshd[1680]: error: maximum authentication attempts exceeded for admin_ssh from 192.168.0.33 port 4
sh2 [preauth]
 17:12:26 tfm sshd[1680]: Disconnecting: Too many authentication failures [preauth]
```

Y mirando nuestro **iptables**, vemos como la IP 192.168.0.33, perteneciente al cliente que estamos usando para conectarnos en remoto al servidor, ha sido bloqueada.

```
# iptables -L
```

```
root@tfm:/# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP       all  --  192.168.0.33         anywhere
ACCEPT     all  --  anywhere             anywhere
DROP       all  --  100.64.0.0/10        anywhere
DROP       all  --  127.0.0.0/8          anywhere
DROP       all  --  link-local/16        anywhere
DROP       all  --  192.0.0.0/24         anywhere
DROP       all  --  192.0.2.0/24         anywhere
DROP       all  --  198.18.0.0/15        anywhere
DROP       all  --  198.51.100.0/24      anywhere
DROP       all  --  203.0.113.0/24       anywhere
DROP       all  --  base-address.mcast.net/4  anywhere
```

De hecho si escribimos en nuestro terminal lo siguiente para comprobar el estado de las conexiones **SSH** así podremos comprobarlo.

```
root@tfm:/# sudo fail2ban-client status ssh
Status for the jail: ssh
|- Filter
|  |- Currently failed: 0
|  |- Total failed:     19
|  `- File list:        /var/log/auth.log
`- Actions
   |- Currently banned: 1
   |- Total banned:     1
   `- Banned IP list:   192.168.0.33
```

Y si de nuevo tratásemos de conectarnos veríamos que en ningún momento se establece ésta pues nuestra IP ha sido baneada.

```
user@user-VirtualBox:~$ ssh -v -p 31892 admin_ssh@192.168.0.20
OpenSSH_7.2p2 Ubuntu-4ubuntu2.2, OpenSSL 1.0.2g  1 Mar 2016
debug1: Reading configuration data /etc/ssh/ssh_config
debug1: /etc/ssh/ssh_config line 19: Applying options for *
debug1: Connecting to 192.168.0.20 [192.168.0.20] port 31892.
debug1: connect to address 192.168.0.20 port 31892: Connection timed out
ssh: connect to host 192.168.0.20 port 31892: Connection timed out
```

Hasta aquí nos hemos estado conectando en remoto a través de la **password** del usuario pero debemos evitar esto ya que podemos ser atacados mediante fuerza bruta. Lo más seguro es la conexión mediante el uso de claves públicas y privadas y por lo tanto configuremos nuestro servidor para esto.

Yéndonos a **/etc/ssh/sshd_config** deberemos cambiar algunos parámetros que antes no se nombraron pues realmente es ahora cuando es importante la configuración de este archivo, ya que será así como lo usaremos.

PasswordAuthentication no

PubkeyAuthentication yes

RSAAuthentication yes

AllowTcpForwarding	**no**
X11Forwarding	**no**
MaxStartups	**2:30:10**
LoginGraceTime	**30**
LogLevel	**VERBOSE**
Banner	**/etc/issue.net**
AllowUsers	**admin_ssh tfm**
Port	**31892**

Reiniciamos el servicio **SSH** y comprobamos que está funcionando correctamente.

```
# ps –A | grep sshd
```

```
root@tfm:/etc/ssh# ps -A | grep sshd
14023 ?        00:00:00 sshd
```

Y que se encuentra a la escucha del puerto indicado y no de ningún otro.

```
# ss –lnp | grep sshd
```

```
root@tfm:/etc/ssh# sudo ss -lnp | grep sshd
tcp    LISTEN     0      128          *:31892
                users:(("sshd",pid=14023,fd=3))
tcp    LISTEN     0      128         :::31892
                users:(("sshd",pid=14023,fd=4))
```

Ahora en el host que usaremos para conectarnos en remoto debemos crear las claves. Primero creamos el directorio donde almacenarlas y le daremos los permisos necesarios.

```
# mkdir /.ssh

# chmod 700 /.ssh
```

Ahora sí creamos las claves, usando 4096 bits para hacerlas lo más complejas posible. Podemos introducir un directorio donde guardar las claves pero mejor usar las que vienen por defecto. Lo que si es importante es recordar la **passphrase** que introduzcamos, ya que sin ella no podremos loguearnos en nuestro servidor.

```
# ssh-keygen –t rsa –b 4096
```

```
root@user-VirtualBox:/home/user# ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:EFYs97ADaeRcjjN6GGdJGSRWt7Yj728E/At5Mzksh8g root@user-VirtualBox
```

Lo siguiente será copiar nuestra clave pública en el servidor.

```
# ssh-copy-id admin_ssh@192.168.0.20 -p 31892
```

```
root@user-VirtualBox:/home/user# ssh-copy-id admin_ssh@192.168.0.20 -p 31892
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/root/.ssh/id_rsa.pub"
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any th
at are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it
is to install the new keys

**************************AVISO LEGAL**********************
Todo intento de intrusion sin autorizacion en este sistema
es un acto ilegal y por lo tanto se procedera a perseguirlo
y denunciarla ante los organismos competentes, presentando
como prueba todo los registros que este servidor almacena
************************************************************

Password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh -p '31892' 'admin_ssh@192.168.0.20'"
and check to make sure that only the key(s) you wanted were added.
```

Como vemos ya nos avisa que intentemos el logueo por el procedimiento normal a nuestro servidor, así que lo haremos.

```
# ssh  -p 31892 admin_ssh@192.168.0.20
```

Y observando la captura vemos que nos pide la **passphrase** que usamos anteriormente para crear las claves, de forma que al introducirlas podremos acceder finalmente en remoto a nuestro servidor.

```
root@user-VirtualBox:/home/user# ssh -p 31892 admin_ssh@192.168.0.20

**************************AVISO LEGAL**********************
Todo intento de intrusion sin autorizacion en este sistema
es un acto ilegal y por lo tanto se procedera a perseguirlo
y denunciarla ante los organismos competentes, presentando
como prueba todo los registros que este servidor almacena
************************************************************

Enter passphrase for key '/root/.ssh/id_rsa':
Welcome to Ubuntu 16.04.3 LTS (GNU/Linux 4.4.0-91-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

8 packages can be updated.
8 updates are security updates.


Last login: Wed Sep  6 00:52:28 2017 from 192.168.0.12
shred: /home/tfm/.bash_history: fallo al abrir para escritura: Permiso denegado
admin_ssh@tfm:~$
```

Por último debemos recordar añadir nuestro puerto 31892 y su redireccionamiento al router, para permitir así el acceso desde el exterior a nuestra LAN vía **SSH**.

Redirección de puertos

Nombre del servicio	LAN IP	Protocolo	Puerto LAN	Puerto público
owncloud http	192.168.0.20	TCP	80	80
owncloud https	192.168.0.20	TCP	443	443
ssh	192.168.0.20	TCP	31892	31892

Y para conectarnos sería la misma instrucción simplemente que cambiando la IP privada de nuestro servidor por la pública de acceso a nuestra red.

```
# ssh  -p 31892 admin_ssh@IP_PUBLICA
```

4.5. 820 – Guía de protección contra Denegación de Servicio

Nuestro firewall de web **Modsecurity** ya es una herramienta que nos permite luchar contra ataques de denegación de servicio DDOS, pero siempre vendrá bien instalar algunas herramientas más que hagan nuestro servidor más robusto ante este tipo de ataques.

Lo primero que instalaremos será **Spamhaus**, es uno de los mayores servicios de lista negra DNS antispam conocidos. El proyecto **Spamhaus** no sólo es un editor de espacio de direcciones IP que se sabe que es la fuente de spam y otros correos electrónicos maliciosos, sino que también trabaja activamente con todas las formas de aplicación de la ley y las entidades jurídicas para hacer justicia a los afectados por el spam a gran escala. Debido a esto, **Spamhaus** ha creado el Registro de Operaciones de Spam Conocidas (ROKSO), que es utilizado por ISP y por la propia ley para rastrear y supervisar las actividades de los delincuentes que realizan sus acciones delictivas mediante spam a gran escala. Actualmente, ROKSO rastrea a los 100 principales remitentes y grupos profesionales del spam en todo el mundo.

Para su instalación pasaremos los siguientes comandos por el terminal.

```
# apt-get install libapache2-mod-spamhaus

# touch /etc/spamhaus.wl

# nano /etc/apache2/apache2.conf
```

Y añadimos al archivo que estamos editando **/etc/apache2/apache2.conf** estas líneas al final.

```
<IfModule mod_spamhaus.c>
        MS_METHODS POST,PUT,OPTIONS,CONNECT
        MS_WhiteList /etc/spamhaus.wl
        MS_CacheSize 256
</IfModule>
```

Otro ataque del que debemos protegernos es el conocido **Slowloris**, software de ataque DDoS que permite a una sola computadora tumbar un servidor web. Debido a la naturaleza simple pero elegante de este ataque, requiere un ancho de banda mínimo para implementar y afecta al servidor web del servidor de destino únicamente, sin casi efectos secundarios en otros servicios y puertos.

Así que procederemos a instalar lo necesario para prevenirlo.

```
# apt-get install libapache2-mod-qos
```

Otra cosa importante y muy sencilla de hacer es protegernos de los ataques SYN FLOOD, Es una forma de ataque de denegación de servicio en el que un atacante envía una sucesión de peticiones SYN al sistema de un objetivo en un intento de consumir suficientes recursos del servidor para hacer que el sistema no responda al tráfico legítimo. La forma de remediarlo es sencilla y para ello tenemos que habilitar las cookies SYN en el equilibrador de carga o en nuestro propio servidor

```
# sysctl –w net.ipv4.tcp_synkookies=1

# sysctl –w net.ipv4.tcp_max_syn_backlog=2048
```

Y tras esto editaremos **/etc/sysctl.conf** debiendo añadir estas 5 líneas.

```
net.ipv4.tcp_syncookies = 1
net.ipv4.tcp_max_syn_backlog = 2048
net.ipv4.tcp_synack_retries = 3
net.ipv4.netfilter.ip_conntrack_tcp_timeout_syn_recv = 45
net.ipv4.conf.all.rp_filter = 1
```

Lo que hemos hecho por orden según las líneas es activar las SYN cookies, aumentar el tamaño de la cola de espera a 2048, hemos reducido el número de conexiones a 3 para que el Kernel cierre antes las conexiones, hemos bajado el tiempo de espera a 45 segundos y por último hemos activado la protección contra IP spoofing.

Ahora instalaremos un firewall basado en iptables, llamado **Advanced Policy Firewall** (**APF**), el cual está diseñado para resolver las necesidades de los servidores actuales desplegados en Internet y de una instalación única como nuestro Linux.

```
# wget http://www.rfxn.com/downloads/apf-current.tar.gz

# tar –xzvf apf-current.tar.gz

# cd apf-*

# chmod +x install.sh

# ./install.sh
```

Lo siguiente será configurarlo y para ello accederemos a **/etc/apf/conf.apf** y deberemos modificar todas estas líneas.

```
IFACE_IN="enp0s3"
IFACE_OUT="enp0s3"
IFACE_TRUSTED="enp0s3"
SET_VERBOSE="1"
```

```
# Common inbound (ingress) TCP ports
IG_TCP_CPORTS="31892,20,21,25,26,37,43,53,80,110,113,143,443,465,873,993,
995,2077,2078,2082,2083,2086,2087,2095, 2096,3306,6666"

# Common inbound (ingress) UDP ports
IG_UDP_CPORTS="53,6277"

# Common ICMP inbound (ingress) types
# 'internals/icmp.types' for type definition; 'all' is wildcard for any
IG_ICMP_TYPES="3,5,11,30"

# Common outbound (egress) TCP ports
EG_TCP_CPORTS="21,25,37,43,53,80,110,113,123,443,873,953,2089,2703"

# Common outbound (egress) UDP ports
EG_UDP_CPORTS="20,21,53,873,953,6277"
```

```
TCP_STOP="DROP"
UDP_STOP="DROP"
ALL_STOP="DROP"
BLK_PRVNET="0"
```

Por último tocará iniciar nuestro firewall.

```
# apf -s
```

Al ejecutarlo podría darnos este aviso por la versión de nuestro kernel.

```
root@tfm:/home/tfm/apf-9.7-2# apf -s
apf(15409): {glob} status log not found, created
apf(15409): {glob} activating firewall
apf(15457): {glob} kernel version not compatible or netfilter sup
port missing, aborting.
apf(15409): {glob} firewall initialized
apf(15409): {glob} !!DEVELOPMENT MODE ENABLED!! - firewall will f
lush every 5 minutes.
```

Para solucionar este aviso por versión del kernel deberemos añadir una excepción en función de ésta. Para ello escribimos por consola **uname –r** y la obtendremos.

```
root@tfm:/proc/sys/net/netfilter# uname -r
4.4.0-91-generic
```

A continuación editaremos el fichero **/ect/apf/internals/functions.apf** añadiendo lo que mostramos en el recuadro en rojo y la versión según lo obtenido con la instrucción anterior.

```
if [ "$KREL" == "2.4" ]; then
        MEXT="o"
elif [ "$KREL" == "2.6" ]; then
        MEXT="ko"
elif [[ "$KREL" =~ "3." ]]; then
        MEXT="ko"
elif [[ "$KREL" =~ "4." ]]; then
        MEXT="ko"
elif [ ! "$KREL" == "2.4" ] && [ ! "$KREL" == "2.6" ]
```

Volvemos a iniciar nuestro firewall **APF** y esta vez todo funcionará con normalidad.

5. Conclusiones y trabajo futuro

Este proyecto permite la implantación de una nube en un servidor linux usando UBUNTU 16.04 Server securizado. Para ello se han usado multitud de paquetes y configuraciones siguiendo las guías y manuales oficiales y sobre todo las CCN-STIC, aunque debo destacar que en muchos casos me he encontrado con que estas guías ya se encontraban obsoletas, pues inevitablemente las distintas versiones de UBUNTU hacen que ciertos directorios y archivos cambien tanto su dirección como nombres, teniendo que apoyarme además en todos los conocimientos que aportan las redes sociales dedicadas a Linux y en especial en Ask Ubuntu.

En el aspecto personal tengo que reconocer que la realización de este trabajo ha sido mucho más duro de lo que apenas reflejan sus más de cien páginas, pues como todos sabemos, en la tecnología van surgiendo diversos problemas inesperados, aleatorios, que es donde realmente radica el esfuerzo de conseguir una buena configuración segura. Por supuesto todas estas “pequeñas” pegas que me han surgido durante la elaboración de esta memoria las he reflejado como solucionarlas.

En especial tengo que destacar todos los módulos y configuraciones realizadas en pos de securizar al máximo el servidor de servicio en la nube, entre ellos antivirus, proxies, firewall web y de aplicaciones, protección del kernel o ante distintos rootkits… Aunque nunca debemos olvidar que la seguridad total no existe y por lo tanto siempre se deberá de seguir revisando, mejorando e implantando nuevas medidas, que protejan, dificulten al máximo y que mitiguen todo lo posible los efectos de un probable ataque que explote alguna vulnerabilidad.

Este trabajo constituye un guía de referencia para montar un servidor seguro basado en software gratuito y apoyado por una gran comunidad. Como trabajo futuro habría que realizarle una auditoría de seguridad mediante OWAS ZAP, someterla a distintos tipos de ataques LOIC y HOIC, buffer overflow, rootkit, ataques de fuerza bruta… En general sería someter a prueba toda su estructura defensiva y si la pasara seguir con los trámites para obtener la acreditación del CNI y poder ser usado como una herramienta más para el ejercito, en especial en zona de operaciones y en maniobras nacionales e internacionales, aumentando así nuestra capacidad de Mando y Control.

Bibliografía y webgrafía

ADVANCED POLICY FIREWALL (APF)

https://www.rfxn.com/projects/advanced-policy-firewall/

ASK UBUNTU

https://askubuntu.com/

CHKROOTKIT

http://www.chkrootkit.org/

CLAMAV

https://www.clamav.net/

LINUX MALWARE DETECTOR (LMD)

https://www.rfxn.com/projects/linux-malware-detect/

OSSEC

https://ossec.github.io/

OWNCLOUD

https://owncloud.org/

SQUID

http://www.squid-cache.org/

Anexos

MANUAL SQUID

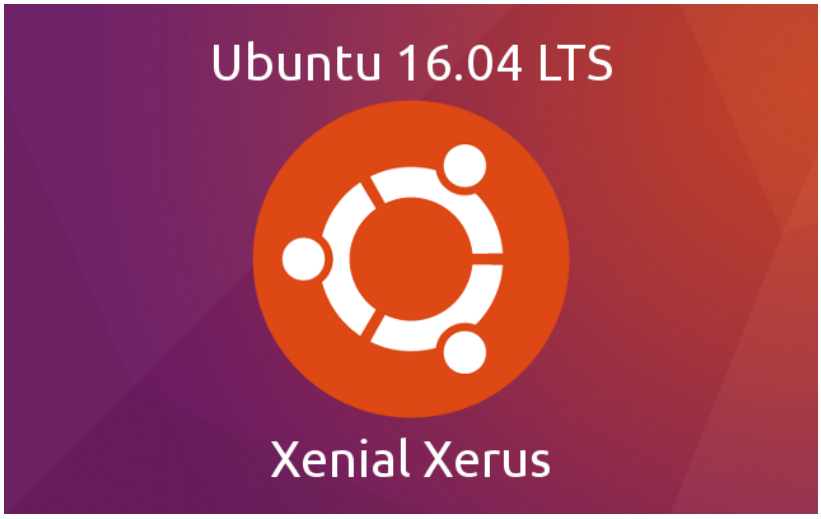
Manual de configuración de SQUID.

http://www.visolve.com/uploads/resources/squid30.pdf

MANUAL UBUNTU 16.04 SERVER

Manual instalación y configuración UBUNTU 16.04 SERVER.

https://help.ubuntu.com/lts/serverguide/serverguide.pdf



MANUAL OWNCLOUD

Manual de administración de ownCloud X.

https://doc.owncloud.org/server/latest/ownCloud_Server_Administration_Manual.pdf