Administración de servidores

Remo Suppi Boldrito

PID_00238615

Universitat Oberta de Catalunya

Índice

Introducción

La interconexión entre máquinas y las comunicaciones de alta velocidad han permitido que los recursos que se utilicen no estén en el mismo sitio geográfico del usuario. UNIX (y por supuesto GNU/Linux) es probablemente el máximo exponente de esta filosofía, ya que desde su inicio ha fomentado el intercambio de recursos y la independencia de dispositivos. Esta filosofía se ha plasmado en algo común hoy en día como son los servicios. Un servicio es un recurso (que puede ser universal o no) y que permite, bajo ciertas condiciones, obtener información, compartir datos o simplemente procesar la información a distancia. Nuestro objetivo es analizar los servicios que permiten el funcionamiento de una red (además de los servicios ya mencionados en el módulo de red de Administración GNU/Linux). Generalmente, dentro de esta red existirá una máquina (o varias, según las configuraciones) que hará posible el intercambio de información entre las demás. Estas máquinas se denominan servidores y contienen un conjunto de programas que permiten que la información esté centralizada y sea fácilmente accesible. Estos servicios permiten la reducción de costes y amplían la disponibilidad de la información, pero se debe tener en cuenta que un servicio centralizado presenta inconvenientes, ya que puede quedar fuera de servicio y dejar sin atención a todos los usuarios. En este módulo se verán los principales servicios que permiten que una máquina GNU/Linux juegue un papel muy importante en una infraestructura tecnológica, tanto en centralizar y distribuir datos como en ser punto de información, acceso o comunicación. Por otro lado y con el avance de las arquitecturas (software) orientada a servicios (SOA - *Service Oriented Architecture*), y las tecnologías de desarrollo de aplicaciones que se han estandarizado en este paradigma de diseño de sistemas distribuidos, GNU/Linux se ha transformado en la infraestructura por excelencia que da soporte a la creación de sistemas de información altamente escalables. Este tipo de arquitectura (SOA) se ha transformado en una parte esencial del desarrollo de software distribuido, ya que permite la creación de sistemas distribuidos eficientes, que aprovechan toda la infraestructura subyacente, y establece una interfaz bien definida a la exposición e invocación de servicios web (de forma común pero no exclusivamente) facilitando la interacción entre los sistemas propios y externos.

Servicios replicados

Una arquitectura de servidores debe tener los servicios replicados *(mirrors)* para solventar los inconvenientes que supone.

Objetivos

En los materiales didácticos de este módulo encontraréis los contenidos y las herramientas procedimentales para conseguir los objetivos siguientes:

1. Presentar los aspectos más relevantes de los conceptos involucrados, tanto a nivel teórico como práctico, en la estructura de servidores/servicios en un sistema GNU/Linux.
2. Analizar los conceptos relativos a servicios y servidores específicos de un sistema GNU/Linux.
3. Experimentar con la configuración y adaptar la instalación de servicios a un entorno determinado.
4. Analizar y participar en discusiones sobre las posibilidades actuales y futuras de nuevos servicios y los obstáculos que existen básicamente en aspectos de seguridad en los diferentes entornos de trabajo GNU/Linux (servidor, escritorio multimedia, escritorio ofimàtica, enrutador o *router*,...).

1. Administración de servidores

Los servicios se pueden clasificar en dos tipos: de vinculación ordenador-ordenador o de relación hombre-ordenador. En el primer caso, se trata de servicios requeridos por otros ordenadores, mientras que en el segundo, son servicios requeridos por los usuarios (aunque hay servicios que pueden actuar en ambas categorías). Dentro del primer tipo se encuentran los servicios de *Active Directory* (*Domain Controller*) o los servicios de almacenamiento intermedio (*proxies*). Dentro de la segunda categoría se contemplan servicios de intercambio de información a nivel de usuario, como el correo electrónico (MTA, IMAP, POP), *news*, *World Wide Web* o *Wiki*. Para mostrar las posibilidades de GNU/Linux Debian, se describirá cada uno de estos servicios con una configuración mínima y operativa, pero sin descuidar aspectos de seguridad y estabilidad.

1.1. *Active Directory Domain Controller* con Samba4

Uno de los aspecto más importantes en la integración de sistemas, es la gestión de usuarios e identificación centralizada así como las autoridades de autentificación y los permisos. En mucho sitios basados en Windows esta tarea es realizada por un *Active Directory* (AD, servicio de directorio en una red distribuida de ordenadores –a veces también llamado PDC por *Primary Domain Controller*–) y es importante contar con herramientas de la banda de *Open Source* que permitan gestionar este tipo de entornos. Samba versión 4 es una nueva distribución de este popular software que permite la integración con sistemas Windows actuando como servidor de archivos y/o impresoras y además, en esta última versión y de forma estable, puede actuar como controlador de un dominio Windows aceptando clientes, gestionando usuarios y directorios en forma centralizada y totalmente compatible.[s40]

De los expertos de *Active Directory* (que generalmente son consultores especializados y con un alto coste) sabemos que es AD es una unión (que puede ser muy complicada de entender para los administradores que nos estén dedicados al mundo Windows) de diferentes servicios y tecnologías como DNS, Kerberos, LDAP y CIFS. Algunos incluyen DHCP también pero como veremos no es necesario en nuestra instalación. Estaríamos en un error si pensamos que configurando cada uno de estos servicios tendríamos el resultado del conjunto, pero Samba4 presenta una capa de abstracción, estable y eficiente, que las integra para ofrecer un producto complejo pero que se puede configurar y administrar siguiendo un conjunto de pasos sin grandes dificultades (aunque

no se debe pensar que es simple). El elemento crítico (base de los mayores problemas y errores) del AD es el DNS (en realidad Windows utilizará el AD como DNS) ya que este lo hará servir para 'agregar extraoficialmente' a la lista de nombres habituales en un DNS, los servidores AD para que los clientes puedan encontrarlos y tener acceso a ellos.

En relación Kerberos y LDAP los administradores de GNU/Linux saben de su potencialidad y complejidad, pero en el caso de AD están integrados en el paquete y si bien les otorga una cierta estandarización del sistema no son integrables con servidores u otros clientes, solo se utilizan para sus objetivos y particularmente cuando utilizamos Samba4, será este quien los configurará y gestionará en nuestro nombre con pequeñas modificaciones por parte del administrador. La versión actual de Samba (V4) no difiere de las anteriores en cuanto a compartición de archivos e impresoras (incluso se ha simplificado su gestión/administración) y además con la implementación de AD permitirá que aquellos administradores que utilicen Windows puedan continuar utilizando sus herramientas de gestión y administración de dominio solo con apuntar al servidor Samba. Toda la configuración de Samba pasará ahora por el archivo `smb.conf` (normalmente en */etc/samba/smb.conf*) con definiciones simplificadas pero permitiendo la gestión compleja de un dominio Windows a través de las herramientas (también complejas) de Windows como por ejemplo RSAT (*Remote Server Administration Tools*) y obviamente a través del sistema GNU/Linux y la CLI de Samba4 se tendrá acceso a toda la configuración y administración del AD (sin necesidad de utilizar Windows en ningún caso).[s40,s43]

A partir de la versión 8 (Jessie) Debian incorpora los paquetes de Samba4 (versión 4.2) y por ello no es necesario compilarlo (solamente se deberán instalar los paquetes y configurarlos). En la wiki oficial de Samba (en la dirección https://wiki.samba.org/index.php/Build_Samba_from_source) se describen todos los pasos para compilarlo desde las fuentes en caso que no se dispongan de los paquetes compilados o se desee incluir una nueva versión.

1.1.1. Configuración del Samba Active Directory Domain Controller (AD-DC)

Para hacer una prueba se considerará una máquina virtual con dos interfaces, una con NAT hacia el exterior y conexión a Internet y otra interfaz conectada a una red privada que gestionará el AD-DC para todas las máquinas de la red privada (en caso que se desee que el servidor sea para una red pública el supuesto es el mismo y solo se debe cambiar la interfaz en la configuración inicial).

En este caso la configuración de */etc/hosts* contendrá:

```
27.0.0.1        localhost
172.16.1.1      srv.nteum.org    srv
```

Y la configuración de */etc/network/interfaces*:

```
auto lo
iface lo inet loopback
allow-hotplug eth0 eth1
iface eth0 inet dhcp
iface eth1 inet static
     address 172.16.1.1
     netmask 255.255.255.0
```

A continuación, se deberán cargar todos los paquetes necesarios en Debian ejecutando:

```
apt-get install acl attr autoconf bison build-essential debhelper
dnsutils docbook-xml docbook-xsl flex gdb krb5-user libacl1-dev
libaio-dev libattr1-dev libblkid-dev libbsd-dev libcap-dev libcups2-dev
libgnutls28-dev libjson-perl libldap2-dev libncurses5-dev libpam0g-dev
libparse-yapp-perl libpopt-dev libreadline-dev perl perl-modules
pkg-config python-all-dev python-dev python-dnspython python-crypto
```

Durante la configuración nos pedirá diferentes datos (el valor por defecto lo extraerá de la configuración) y le daremos a *aceptar* (o lo cambiaremos). Entre ellos tenemos: *Default Kerberos version 5 realm:* NTEUM.ORG (respetar las mayúsculas), *Kerberos servers for your realm:* srv.nteum.org y *Administrative server for your Kerberos realm:* srv.nteum.org.

A continuación, se instalarán los paquetes de Samba necesarios:

```
apt-get install smbclient samba winbind
```

Se debe renombrar el archivo de configuración inicial con:

```
cd /etc/samba; mv smb.conf smb.conf.org
```

Y a continuación proporcionar el dominio, indicándole la interfaz (además de *lo*) donde se quiere que responda (ver detalles de los parámetros en [Sam]):

```
samba-tool domain provision --use-rfc2307 --interactive --option="interfaces=lo eth1"
--option="bind interfaces only=yes"
```

Seguidamente, el programa indicará el Realm (dominio Kerberos que se ha introducido anteriormente, NTEUM.ORG en nuestro caso) y el nombre del dominio NTEUM (si se desea se puede cambiar), luego el rol que se le desea dar [DC], el DNS que se utilizará [SAMBA_INTERNAL] y el servidor de nombres que actuará de *forwarder* para las peticiones externas donde se deberá indicar el externo de nuestro proveedor o uno público (por ejemplo, el 8.8.8.8). Finamente pedirá una clave para el dominio que debe cumplir unos criterios de seguridad entre mayúsculas, minúsculas, dígitos, longitud (por ejemplo, similar CaVeMeCat2016).

Reiniciamos la máquina y podremos probar su funcionamiento con*:

Verificaremos que los servicios *samba-ac-dc* y *winbind* están en marcha o si no lo están los pondremos en marcha desde */etc/init.d.

`smbclient -L localhost -U%` (o también con el nombre de la máquina `-L srv`)

```
Domain=[NTEUM] OS=[Windows 6.1] Server=[Samba 4.2.10-Debian]
        Sharename       Type      Comment
        ---------       ----      -------
        netlogon        Disk
        sysvol          Disk
        IPC$            IPC       IPC Service (Samba 4.2.10-Debian)
Domain= [NTEUM] OS= [Windows 6.1] Server=[Samba 4.2.10-Debian]
        Server          Comment
        ---------
        Workgroup       Master
        ---------       -------
        WORKGROUP       SRV
```

Con la ejecución de este comando se puede verificar si Samba provee los recursos compartidos por defecto *netlogon* y *sysvol*.

También se puede probar la ejecución de:

```
smbclient //localhost/netlogon -U Administrator -c 'ls'
```

Enter Administrator's password: <introducir el passwd dado durante el aprovisionamiento>
Domain=[NTEUM] OS= [Windows 6.1] Server=[Samba 4.2.10-Debian]
. D 0 Tue Jun 21 17:47:49 2016
.. D 0 Tue Jun 21 17:47:53 2016
3840152 blocks of size 1024. 1970736 blocks available

Un aspecto esencial para el funcionamiento correcto del *Active Directory* es que el DNS se encuentre bien configurado, ya que sin las entradas correctas Kerberos no validará y en AC-DC no funcionará. Para ello se deberá modificar el */etc/resolv.conf* con (para que apunte a la misma máquina y al dominio):

```
domain nteum.org
nameserver 172.16.1.1
```

Y para comprobar que todo es correcto podemos ejecutar los siguientes comandos y verificar que se recibe la respuesta que se muestra a continuación:

```
host -t SRV _ldap._tcp.nteum.org.
```

_ldap._tcp.nteum.org has SRV record 0 100 389 srv.nteum.org.

```
host -t SRV _kerberos._udp.nteum.org.
```

_kerberos._udp.nteum.org has SRV record 0 100 88 srv.nteum.org.

```
host -t A srv.nteum.org
```

srv.nteum.org has address 172.16.1.1

Finalmente se puede verificar que Kerberos funciona correctamente obteniendo un ticket:

```
kinit administrator@NTEUM.ORG
```

Password for administrator@NTEUM.ORG:
Warning: Your password will expire in 41 days on Tue 02 Aug 2016 17:47:52 BST

```
klist
```

```
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: administrator@NTEUM.ORG
Valid starting     Expires            Service principal
21/06/16 19:16:51  22/06/16 05:16:51  krbtgt/NTEUM.ORG@NTEUM.ORG
        renew until 22/06/16 19:16:46
```

A continuación, se creará un usuario en el dominio para que se pueda conectar desde Windows (y se deberá introducir el *passwd* y su repetición –con los criterios indicados anteriormente–):

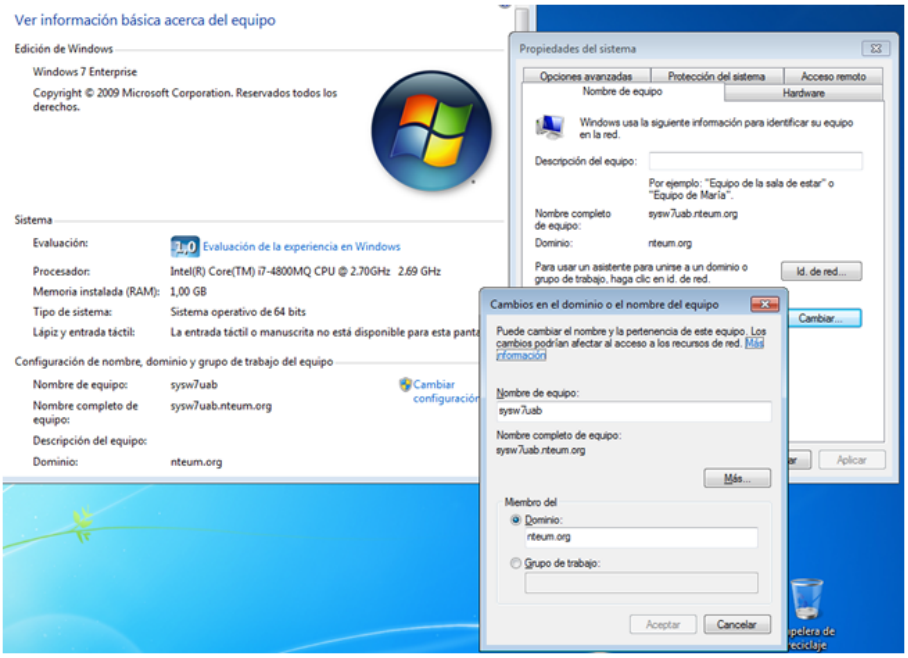
`samba-tool user add debian`	
`samba-tool user list`	Para visualizar los usuarios
`samba-tool user delete debian`	Para borrar un usuario
`samba-tool user setpassword debian`	Para cambiar el *passwd*
`samba-tool user enable|disable debian`	Para habilitar/deshabilitar una cuenta
`samba-tool group list`	Para visualizar los grupos
`samba-tool group listmembers "Domain Users"`	Para visualizar los usuarios de un grupo
`samba-tool group add|delete NteumMas`	Para agregar/borrar un grupo
`samba-tool group addmembers|removemembers NteumMas debian`	Para agregar/quitar miembros de un grupo

Finalmente nos quedaría poner una máquina (puede ser una virtual con Windows) en la misma red privada y, accediendo a ella como administrador, modificar la configuración del DNS al servidor AD-DC (172.16.1.1 en nuestro caso). Luego modificamos la configuración de esta máquina para que se conecte al dominio haciendo: *Panel de Control-> Sistema y Seguridad-> Sistema-> Configuración del nombre del equipo, dominio y grupo de trabajo*. Aquí seleccionamos *Cambiar la configuración* e introducimos el dominio **nteum.org**. A continuación nos pedirá el usuario y clave del administrador que será *Administrator* y el *passwd* introducido durante la configuración. Si se desea quitar la máquina del

dominio deberemos entrar con el usuario local (administrador) de la máquina y repetir el proceso inverso seleccionando un grupo de trabajo y quitándola del dominio.

La figura 1 muestra la ventana de configuración indicada anteriormente sobre Windows 7.

Figura 1



Reiniciamos la máquina Windows y ya se podrá conectar al usuario del dominio (definido previamente –usuario *debian* en nuestro caso–). Es muy probable que al detectar el dominio solicite que se presione Crtl-Alt-Del para introducir el usuario. Si estamos ejecutando Virtualbox sobre una máquina Windows nos dará problemas, ya que se activará esta secuencia en el *host* y no en la máquina virtual. Para evitarlo debemos presionar la tecla *Host-Key-Combination-VBox* (generalmente tecla Crlt-Derecha) + Del y, como usuario, NTEUM\debian (aunque normalmente ya indica el dominio y por lo cual será solamente debian). Si queremos volver al usuario local, haremos el *NOMBRE_DE_LA_MÁQUINA\usuario_local* (que nos servirá para quitar la máquina del dominio cuando se desee y revertir la conexión al dominio dejando la máquina con los usuarios locales).

Para administrar el sitio AD se pueden utilizar las herramientas de Windows (RSAT), las cuales se pueden obtener (sin cargo) desde el sitio del fabricante y con la guía indicada en [s41] y ejemplos de configuración en [s42]. También es posible utilizar la herramienta Swat (https://wiki.samba.org/index.php/SWAT2, última versión Noviembre de 2012) pero su instalación puede presentar algunos inconvenientes (sobre todo si Samba4 se ha instalado desde los fuentes). Por último en https://wiki.samba.org/index.php/Samba4/InitScript podremos

encontrar el *script* para iniciar y apagar el servidor AD en forma automática (si bien en Debian 8.x esta acción no es necesaria ya que el paquete de instalación configura estos dentro de */etc/init.d*).

1.2. Servicio de correo electrónico (*mail*)

1.2.1. *Mail Transport Agent*(MTA) Básico

Un MTA (*Mail Transport Agent*) se encarga de enviar y recibir los correos desde un servidor de correo electrónico hacia y desde Internet, que implementa el protocolo SMTP (*Simple Mail Transfer Protocol*). Todas las distribuciones incorporan diferentes MTA y por ejemplo las de Debian se pueden consultar en su paquete virtual *mail-transport-agent**. Una de las que se utilizan habitualmente es `exim`, ya que es más fácil de configurar que otros paquetes MTA, como son `postfix` o `sendmail` (este último es uno de los precursores). `Exim` presenta características avanzadas tales como rechazar conexiones de sitios de *spam* conocidos, posee defensas contra correo basura (*junk mails*) o bombardeo de correo (*mail bombing*) y es extremadamente eficiente en el procesamiento de grandes cantidades de correos.

***https://packages.debian.org /jessie/mail-transport-agent**

Su instalación es a través de `apt-get install exim4-daemon-heavy` (en este caso se optará por la instalación de la versión *heavy* que es la más completa y soporta lista de acceso (ACL) y características avanzadas, en instalaciones más simple se puede optar por *exim4-daemon-light*). Su configuración se realiza a través de `dpkg-reconfigure exim4-config` donde una respuesta típica a las preguntas realizadas es:

- *General type of mail configuration*: internet site; mail es enviado y recibido utilizado SMTP.
- *System mail name*: remix.world (nuestro dominio)
- *IP-addresses to listen on for incomming SMTP connections*: (dejar en blanco)
- *Other destinations for which mail is accepted*: remix.world
- *Domains to relay mail for*: (dejar en blanco)
- *Machines to relay mail for*: (dejar en blanco)
- *Keep number of DNS-queries minimal (Dial-on-Demand)*?: No
- *Delivery method for local mail: Maildir format in home directory*
- *Split configuration into small files*?: No

El servidor ya estará configurado y puede probarse utilizando la instrucción `echo test message | mail -s "test" adminp@SySDW.nteum.org` (por supuesto cambiando la dirección) y verificando que el mail ha llegado al usuario adminp (los errores se pueden encontrar en */var/log/exim4/mainlog*). La configuración será almacenada en */etc/exim4/update-exim4.conf.conf*. Para configurar autentificación por TLS, ACL y Spam Scaning consultar la web https://wiki.debian.org/Exim. Como hemos seleccionado Maildir, en el directorio *home* podemos leer los correos con un cliente que soporte formato Maildir y ejecutando por ejemplo `mutt -f $HOME/Maildir` o haciendo

`export MAILDIR=Maildir` y ejecutando `mutt` veremos el correo enviado previamente al usuario *adminp*.

1.2.2. External SMTP

Cuando instalamos un nuevo sistema como servidores o estaciones de trabajo un aspecto relevante es el servidor de correo y podemos instalar grandes paquetes como los ya mencionados Postfix, Exim o Zimbra (en su version Community http://www.zimbra.com/) haciendo que los correos hacia dominios externos utilicen los servicios externos de SMTP (por ejemplo los de Google). Para máquinas virtuales, estaciones de trabajo o portátiles es un poco más complicado ya que generalmente tienen IP privadas o en redes internas por lo cual es necesario tener un servidor que haga de receptor de los correos externo a mi dominio, es decir un servidor que haga las funciones de *smarthost* por ejemplo el Google Apps SMTP. Para detalles de su configuración se puede seguir la documentación de https://wiki.debian.org/GmailAndExim4. De acuerdo a la información de Google* y para cuentas gratuitas el número máximo de destinatarios permitido por dominio y día es de 100 mensajes y que Gmail reescribirá la dirección del remitente. Para su configuración ejecutaremos `dpkg-reconfigure exim4-config` y seleccionaremos:

***https://support.google.com /a/answer/2956491?hl=es**

- *mail sent by smarthost; received via SMTP or fetchmail.*
- *System mail name*: localhost
- *IP-addresses to listen on for incomming SMTP connections*: 127.0.0.1
- *Other destinations for which mail is accepted*: (dejar en blanco)
- *Machines to relay mail for*: (dejar en blanco)
- *IP address or host name of the outgoing smarthost*: smtp.gmail.com::587
- *Hide local mail name in outgoing mail*?: No
- *Keep number of DNS-queries minimal (Dial-on-Demand)*?: No
- *Delivery method for local mail: mbox format in /var/mail*
- *Split configuration into small files*?: Yes

Esta es la configuración más adecuada si no se tiene un IP visible externamente. El envío sobre el puerto 587 de Gmail utiliza STARTTLS para asegurar la protección del passwd y para indicar el usuario y passwd de acceso a Gmail (utilizar una cuenta solo para este objetivo, no la cuenta habitual de Gmail) se debe editar el fichero */etc/exim4/passwd.client* y agregar la siguiente línea.

```
*.google.com:SMTPAccountName@gmail.com:y0uRpaSsw0RD
```

Luego ejecutar (para evitar que otros usuarios de la máquina puedan leer su contenido:

```
chown root:Debian-exim /etc/exim4/passwd.client
chmod 640 /etc/exim4/passwd.client
```

Gmail reescribirá la dirección del remitente automáticamente pero si no lo hiciera o enviamos a un *smarthost* que no lo hace deberíamos configurar el archivo */etc/email-addresses* con todas las combinaciones de direcciones posibles a utilizar (una por línea) y las dirección que se reescribirá (por ejemplo, nteum@remix.world: SMTPAccountName@gmail.com). Luego se deberá ejecutar:

```
update-exim4.conf
invoke-rc.d exim4 restart
exim4 -qff
```

Con ello se actualiza y recarga la configuración y se fuerza a enviar todos los correos que están pendientes. Como mostramos anteriormente, en el archivo */var/log/exim4/mainlog* tendremos los errores si los hay. Si existen errores de autentificación sobre gmail verifique con el comando `host smtp.gmail.com` cuales son los *hosts* que devuelve y si estos concuerdan con el patrón incluido en */etc/exim4/passwd.client*. Si es diferente cámbielo para que coincida.

Hay que tener en cuenta, finalmente, que con las últimas actualizaciones de seguridad de Gmail la máquina/app desde la cual estamos enviado el correo será considerada no segura (ya que intenta acceder con usuario y contraseña) por lo cual recibiremos una alerta y no el correo. Para cambiar esto debemos acceder a la administración de nuestra cuenta (como se explica en https://support.google.com/accounts/answer/6010255) y en el apartado de *Sig-in & Security* debemos ir al último apartado *Allow less secure apps* y ponerlo como ON. Con esta configuración ya podremos recibir y enviar correos.

1.2.3. *Internet Message Access Protocol* (IMAP)

Este servicio permite acceder a los correos alojados en un servidor a través de un cliente de correo como por ejemplo Thunderbird del proyecto Mozilla.org. Este servicio soportado por el *daemon* `imapd` (los actuales soportan el protocolo IMAP4rev1) permite acceder a un archivo de correo electrónico (*mail file*) que se encuentra en una máquina remota. El servicio `imapd` se presta a través de los puertos 143 (imap2), 220 (imap3) o 993 (imaps) cuando soporta encriptación por SSL. Si se utiliza `inetd`, este servidor se pone en marcha a través de una línea en `/etc/inetd.conf` como:

```
imap2 stream tcp nowait root /usr/sbin/tcpd /usr/sbin/imapd
imap3 stream tcp nowait root /usr/sbin/tcpd /usr/sbin/imapd
```

En este ejemplo se llama al *wrapper* `tcpd` que funciona con `hosts.allow` y `hosts.deny` para incrementar la seguridad. Las aplicaciones más populares son `courier-imap`, `cyrus-imapd`, `dovecot-imapd` entre otros. Para probar que el servidor `imap` funciona, se podría utilizar un cliente, por ejemplo Thunderbird/Icedove (Debian), Evolution, Squirremail, o cualquier otro cliente que soporte IMAP, crear una cuenta para un usuario local, configurar-

lo adecuadamente para que se conecte sobre la máquina local y verificar el funcionamiento de `imap`.

Se puede instalar `apt-get install dovecot-imapd` como prueba de concepto, que con las opciones por defecto permite una conexión encriptada por SSL y sobre buzones mailbox (o si queremos sobre buzones maildir deberemos cambiar la configuración de `/etc/dovecot/conf.d/10-mail.conf`). Dovecot es un servidor muy potente por lo cual permite una gran cantidad de opciones y configuraciones (consultar http://wiki2.dovecot.org/). Las pruebas se pueden completar configurando Evolution o IceDove para que se conecte a nuestro servidor/usuario y leer los correos del servidor previamente configurado. Es importante notar que algunos clientes de correo/servidores de Imap solo soportan el formato MailBox y no Maildir y es por ello que se debe tener en cuenta cuando se utilicen los clientes/servidores de Imap. En las actividades que hemos realizado hasta ahora tanto `exim4` como `dovecot-imapd` soportan ambos formatos y se deben configurar durante la instalación.

1.2.4. Aspectos complementarios

Supongamos que como usuarios tenemos cuatro cuentas de correo en servidores diferentes y queremos que todos los mensajes que llegan a estas cuentas se recojan en una única, a la que podamos acceder externamente, y que haya también un filtro de correo basura (*antispam*). Primero se deben instalar `exim4 + imapd` y comprobar que funcionan.

Para recoger los mensajes de diferentes cuentas se utilizará `Fetchmail`, (que se instala con `apt-get install fetchmail`). A continuación, se debe crear el fichero `.fetchmailrc` en nuestro `$HOME` (también se puede utilizar la herramienta `fetchmailconf`) que deberá tener ser algo así como:

```
set postmaster “adminp”
set bouncemail
set no spambounce
set flush
poll pop.domain.com proto pop3
   user 'nteum' there with password 'MyPaSSwOrD' is 'nteum' here
poll mail.domain2.com
   user 'adminp' there with password 'MyPaSSwOrD' is 'adminp' here
   user 'nteum' there with password 'MyPaSSwOrD' is 'nteum' here
```

La acción `set` indica a `Fetchmail` que esta línea contiene una opción global (envío de errores, eliminación de los mensajes de los servidores, etc.). A continuación, se especifican dos servidores de correo: uno para que compruebe si hay correo con el protocolo POP3 y otro para que pruebe a usar varios protocolos con el fin de encontrar uno que funcione. Se comprueba el correo de dos usuarios con la segunda opción de servidor, pero todo el correo que se encuentre se envía al *spool* de correo de `adminp`. Esto permite comprobar varios buzones de diversos servidores como si se tratara de un único buzón.

La información específica de cada usuario comienza con la acción `user`. El `Fetchmail` se puede poner en el `cron`* para que se ejecute automáticamente o ejecutarlo en modo *daemon* (poned `set daemon 60` en `.fetchmailrc` y ejecutadlo una vez, por ejemplo, en `autostart` de Gnome/KDE o en el `.bashrc` –se ejecutará cada 60 segundos–).

***Por ejemplo, en** `/var/spool/cron/crontabs/fetchmail` **agregando** `1 * * * * /usr/bin/fetchmail -s`

Para quitar el correo basura se utilizará SpamAssassin y en esta configuración se ejecutará a traves de Procmail, que es una herramienta muy potente en la gestión del correo (permite repartir el correo, filtrarlo, reenviarlo automáticamente, etc.). Una vez instalado (`apt-get install procmail`), se debe crear un fichero llamado `.procmailrc` en el `home` de cada usuario, que llamará al SpamAssassin:

Podéis instalar SpamAssassin mediante `apt-get install spamassassin`.

```
# Poned yes para mensajes de funcionamiento o depuración
VERBOSE=no
# Consideramos que los mensajes están en "~/.Maildir"), cambiar si es otro
PATH=/usr/bin:/bin:/usr/local/bin:.
MAILDIR=$HOME/Maildir
DEFAULT=$MAILDIR/

# Directorio para almacenar los ficheros
PMDIR=$HOME/.procmail
# Comentar si no queremos log de Procmail
LOGFILE=$PMDIR/log
# filtro de Smap
INCLUDERC=$PMDIR/spam.rc
```

El archivo `~/.procmail/spam.rc` contiene:

```
# si el SpamAssassin no está en el PATH, agregar a la variable PATH el directorio
:0fw: spamassassin.lock
| spamassassin -a

# La tres líneas siguientes moverán el correo Spam a un directorio llamado
# "spam-folder". Si se quiere guardar el correo en la bandeja de entrada,
# para luego filtrarlo con el cliente, comentad las tres líneas.


:0:
* ^X-Spam-Status: Yes
spam-folder
```

El archivo `~/.spamassassin/user_prefs` contiene algunas configuraciones útiles para SpamAssassin (consultad la bibliografía).

```
# user preferences file. Ved man Mail::SpamAssassin::Conf

# Umbral para reconocer un Spam.
# Default 5, pero con 4 funciona un poco mejor
required_hits 4

# Sitios de los que consideraremos que nunca llegará Spam
whitelist_from root@debian.org
whitelist_from *@uoc.edu

# Sitios de los que siempre llega SPAM (separados por comas)
blacklist_from viagra@dominio.com

# las direcciones en Whitelist y Blacklist son patrones globales como:
```

```
# "amigo@lugar.com", "*@isp.net", o "*.domain.com".

# Insertad la palabra SPAM en el subject (facilita hacer filtros).
# Si no se desea comentar la línea.
subject_tag [SPAM]
```

Esto generará un *tag* `X-Spam-Status: Yes` en la cabecera del mensaje si se cree que el mensaje es *spam*. Luego se deberá filtrar y poner en otra carpeta o borrarlo directamente. Se puede usar el `procmail` para filtrar mensajes de dominios, usuarios, etc. Por último, se puede instalar un cliente de correo y configurar los filtros para que seleccionen todos los correos con `X-Spam-Status: Yes` y los borre o los envíe a un directorio. Después verificaremos los falsos positivos (correos identificados como basura pero que no lo son). Un aspecto complementario de esta instalación es que si se desea tener un servidor de correo a través de correo web (*webmail*, es decir poder consultar los correos del servidor a través de un navegador sin tener que instalar un cliente ni configurarlo, igual que consultar una cuenta de Gmail o Hotmail) es posible instalar Squirrelmail (`apt-get install squirrelmail`) para dar este servicio.

Enlace de interés

Para más información sobre `procmail` y el filtrado de mensajes, consultad: http://www.debian-administration.org/articles/242.

Enlace de interés

Sobre Squirrelmail en Debian, consultad: http://www.debian-administration.org/articles/200.

Enlace de interés

Hay otras posibilidades como instalar MailDrop en lugar de Procmail, Postfix en lugar de Exim, o incluir Clamav/Amavisd como antivirus (Amavisd permite vincular Postfix con SpamAssassin y Clamav). Para saber más sobre este tema podéis visitar la siguiente página web: http://www.debian-administration.org/articles/364.

1.2.5. Instalación de producción de un servidor de correo vinculado externamente

Usuarios virtuales

Los usuarios virtuales son los usuarios del servicio pero no del sistema operativo.

En este apartado se describirá cómo instalar un servidor de correo de gran capacidad sobre una distribución Debian. El objetivo es que permita alta cantidad de transacciones y servicios externos de correo y también la posibilidad de reenviarlos hacia otras MTA. También se quiere que pueda recibir y almacenar correos para los usuarios definidos del sistema o usuarios virtuales desde y hacia Internet, con IMAP para el acceso remoto y utilizando conexiones cifradas para proteger la privacidad de la información y controlando el Spam que se pueda recibir o reenviar. Para cumplir este objetivo es necesario la instalación de un servidor de correo de altas prestaciones (por ejemplo, Postfix) asociado con IMAP (por ejemplo, Dovecot), un servidor web (por ejemplo, Apache), un cliente de *webmail* (por ejemplo, SquirrelMail o RoundCube) y un conjunto de utilidades como SpamAssassin, cifrado (SSL/TLS) o antivirus (p.e Amavis/ClamAV). Y todo ello funcionando conjuntamente. El diagrama de flujo sería algo similar a lo siguiente:

1) Un correo llega por SMTP al puerto 25 y lo recibe `Postfix`, el cual realiza unas comprobaciones (listas negras / grises / etc.).

2) Luego pasa por `AMaVis`, que lo envía a `SpamAssassim` y luego a `ClamAV`.

3) Posteriormente `Postfix` realiza la expansión de alias y toma algunas decisiones.

4) Finalmente, `Dovecot` lo pone a disposición de los usuarios por IMAP, ya sea a través de un cliente (`Icedove, Evolution`) o a través de *webmail* (`Apache + RoundCube`), que acceden a ellos remotamente.

Caso 1

iRedMail

iRedMail es una unión e integración de los programas mencionados anteriormente junto con una interfaz administrativa vía web.

La primera opción es instalar iRedMail en su versión *opensource edition* que no presenta dificultades (configuración para Debian [Ired]). Para ello se debe descargar el paquete desde http://www.iredmail.org/download.html, que tendrá el formato *iRedMail-x.y.z.tar.bz2* donde *x.y.z* será la versión correspondiente. En primer lugar, hemos de verificar que se tiene un dominio configurado correctamente (en nuestro caso *srv.nteum.org*):

```
hostname -f
srv.nteum.org
```

Y que en el */etc/hosts* existe una línea con el FQDN:

172.16.1.1 srv.nteum.org srv

Para estas pruebas se utiliza una máquina virtual con eth0 en NAT y conectada a Internet, y con eth1 configurada estáticamente con IP 172.16.1.1. Una vez realizadas estas comprobaciones, iremos al directorio donde se ha descargado el paquete, descomprimiremos el archivo *bz2* y ejecutaremos el instalador:

```
tar xjf iRedMail-x.y.z.tar.bz2
cd iRedMail-x.y.z/
bash iRedMail.sh
```

Unos instantes después, el instalador preguntará:

failban2

`failban2` es una aplicación para bloquear usuarios por repetidos intentos de entrar a una cuenta para suplantar su identidad.

1) El directorio de almacenamiento de los mails: */var/vmail*

2) La base de datos para almacenar las cuentas de los usuarios: por ejemplo, `MySQL` (pero puede ser `MariaDB` u `OpenLdap`)

3) El dominio del correo: *mail.nteum.org* (no puede coincidir con el FQDN de la máquina, en nuestro caso *srv.nteum.org*)

4) También pedirá la palabra clave para el administrador (*username*: postmaster@mail.nteum.org)

5) La selección de componentes opcionales: `iRedAdmin` (aplicación web para administrar las cuentas), `RoundCube` (*webmail*), `SOGo` (*webmail*, calendario y libreta de direcciones), `failban2`.

6) La verificación final e instalación: [Y]

Luego de la instalación pedirá que se reinicie la máquina. Todos los datos importantes (URL, *passwd* y archivos de configuración) quedarán recogidos en el archivo *iRedMail.tips* dentro del directorio donde se ha iniciado la instalación.

A continuación, se pueden realizar las pruebas de funcionamiento en las siguientes URLs:

1) https://srv.nteum.org/iredadmin/, con el usuario y *passwd* del punto 4 anterior, para la interfaz de administración.

2) http://srv.nteum.org/mail/ o https://srv.nteum.org/mail/ (si se prefiere se puede acceder por SSL/TLS) para el cliente *webmail*.

Con ello se podrán enviar mails a las cuentas creadas o al mismo *postmaster* o a los usuarios locales. Para enviar hacia máquinas desde afuera debemos utilizar un *relay host* que puede ser el de nuestro proveedor de Internet configurándolo en */etc/postfix/main.cf* agregando una línea como *relayhost = nombre.dominio.ISP* y reiniciando el servicio. Para recibir correos en nuestro dominio se deben configurar los registros MX en el Servidor de DNS que tenga el registro de nuestro dominio [Ired-DNS] y también podemos configurar los clientes externos por IMAP [Ired-IMAP]. En [Ired-Relay] amplían como solucionar los problemas de *relayhost* cuando estos estén autentificados.

Desinstalación de iRedMail

Si se desea desinstalar iRedMail existe (en Internet y en los foros del propio iRedMail) un *script* (*clean_iredmail.sh*) pero que se encuentra desactualizado y que deja diferentes rastros de los paquetes instalados que se pueden acabar de desinstalar posteriormente (además de desinstalar los paquetes, debemos eliminar los directorios creados en */opt* y las entradas del *crontab*).

Caso 2

Si se desea tener control sobre los paquetes y realizar una instalación similar de las mismas características, pero teniendo el control sobre la instalación y los parámetros/configuración (opción para entornos con características particulares y administradores avanzados), se puede hacer lo siguiente:

1) `apt-get install postfix sasl2-bin`. Seleccionamos "No configuration" y hacemos lo siguiente:

```
cp /usr/lib/postfix/main.cf /etc/postfix/main.cf
```

Editamos *vi /etc/postfix/main.cf* y lo modificamos*:

```
mail_owner = postfix                        línea 59
myhostname = mail.nteum.org                 línea 76
mydomain = nteum.org                        línea 83
myorigin = $mydomain                        línea 104
inet_interfaces = all                       línea 118
mydestination = $myhostname, localhost.$mydomain, localhost, $mydomain
local_recipient_maps = unix:passwd.byname $alias_maps    línea 166
```

***Hay que ir con cuidado que algunas están definidas, pero con otros valores –se muestra la línea que hay que modificar, pero puede variar en función de la versión–.**

```
mynetworks = 127.0.0.0/8, 172.16.1.0/24          línea 265
alias_maps = hash:/etc/aliases                   línea 388
alias_database = hash:/etc/aliases               línea 399
home_mailbox = Maildir/                          línea 421
smtpd_banner = $myhostname ESMTP                 línea 559
sendmail_path = /usr/sbin/postfix                línea 632
newaliases_path = /usr/bin/newaliases            línea 637
mailq_path = /usr/bin/mailq                      línea 642
setgid_group = postdrop                          línea 648
#html_directory =                                línea 652
#manpage_directory =                             línea 656
#sample_directory =                              línea 661
#readme_directory =                              línea 665
```

Y añadimos al final:

```
message_size_limit = 10485760
mailbox_size_limit = 1073741824
smtpd_sasl_type = dovecot
smtpd_sasl_path = private/auth
smtpd_sasl_auth_enable = yes
smtpd_sasl_security_options = noanonymous
smtpd_sasl_local_domain = $myhostname
smtpd_recipient_restrictions = permit_mynetworks, permit_auth_destination,
                               permit_sasl_authenticated, reject
```

Finalmente ejecutamos `newaliases` y luego `systemctl restart postfix`.

```
apt-get install dovecot-core dovecot-imapd
```

Editamos,

```
vi /etc/dovecot/dovecot.conf
          listen = *                                  línea 30

vi /etc/dovecot/conf.d/10-auth.conf
          disable_plaintext_auth = no                 línea 10
          auth_mechanisms = plain login               línea 100

vi /etc/dovecot/conf.d/10-mail.conf
          mail_location = maildir:~/Maildir           línea 24

vi /etc/dovecot/conf.d/10-master.conf
          # Postfix smtp-auth                         línea 95
          unix_listener /var/spool/postfix/private/auth {
                    mode = 0666
                    user = postfix
                    group = postfix
               }
```

Finalmente, reiniciamos el servicio `systemctl restart dovecot`.

2) Con esto ya se puede usar un cliente IMAP, por ejemplo `Icedove` (para instalarlo, `apt-get install icedove`), y configurar una cuenta IMAP contra el servidor y un usuario definido en él.

3) Para instalar `RoundCube` previamente deberemos instalar una base de datos, por ejemplo `MySQL`, con `apt-get install mysql-server-5.5` (nos

pedirá el *passwd* para el usuario *root* de la base de datos). Para comprobar que funciona se puede ejecutar:

```
mysql --defaults-file=/etc/mysql/debian.cnf
```

o también `mysql -p` (e introducir el *passwd*). Luego ejecutamos el cliente `mysql`. Veremos el prompt *mysql>* y podremos ejecutar:

`select host, user, password from mysql.user;`	*muestra información*
`show databases;`	*muestra las DB*
`quit`	*para salir*

4) Para `Roundcube` y dado que no se encuentra en el repositorio oficial (por diferentes razones), aunque sí en *backports*, es necesario insertar la siguiente línea en */etc/apt/sources.list*:

```
deb http://http.debian.net/debian jessie-backports main
```

Primero hacemos un `apt-get update` y a continuación ejecutamos la orden `apt-get install roundcube roundcube-plugins`, haciendo la selección de *myql* como gestor de bases de datos e introduciendo el *passwd* para los usuarios de esta.

Después se debe modificar el archivo */etc/apache2/sites-available/000-default.conf* escribiendo las siguientes líneas (para incluir la configuración dentro de Apache) antes del *tag </VirtualHost>* :

```
Include /etc/roundcube/apache.conf
Alias / /var/lib/roundcube/
```

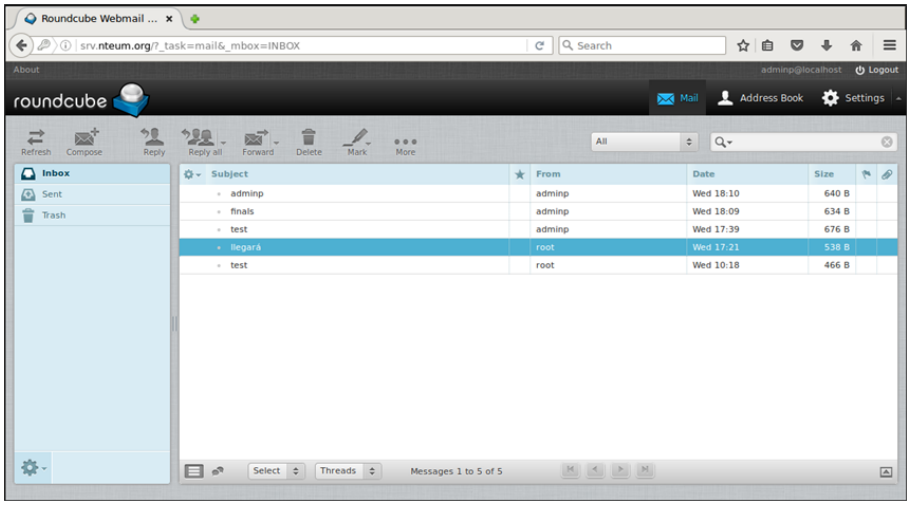
Finalmente reiniciamos el servidor con `systemctl restart apache2` y conectamos a la URL (http://srv.nteum.org) donde podremos acceder con alguno de los usuarios definidos en la máquina.

Se puede observar que además de solicitar el usuario y *passwd* también pide el servidor. Dado que es una única instalación se puede modificar el archivo */etc/roundcube/config.inc.php* para incluir '*localhost*' en la línea 35 (aproximadamente):

```
$config['default_host'] = 'localhost';
```

La figura 2 muestra una pantalla del cliente `webmail` implementado por la orden `RoundCube`.

Figura 2



5) Si se desea mantener la privacidad de la información a través de una conexión https es necesario primero crear un certificado*

***En este caso se optará por la opción más simple pero hay otras disponibles. Se puede consultar [StartSSL].**

```
openssl req -newkey rsa:4096 -nodes -sha512 -x509 -days 3650
-nodes -out /etc/ssl/certs/mail.pem -keyout /etc/ssl/private/mail.key
```

y responder a las preguntas para el certificado, teniendo en cuenta que hay que introducir como *CommonName* el FQDN del servidor (*srv.nteum.org*). A continuación para proteger la clave privada hacemos:

```
chmod 600 /etc/ssl/private/mail.key
```

6) Configuramos Apache modificando */etc/apache2/sites-available/default-ssl.conf*, en concreto las siguientes líneas:

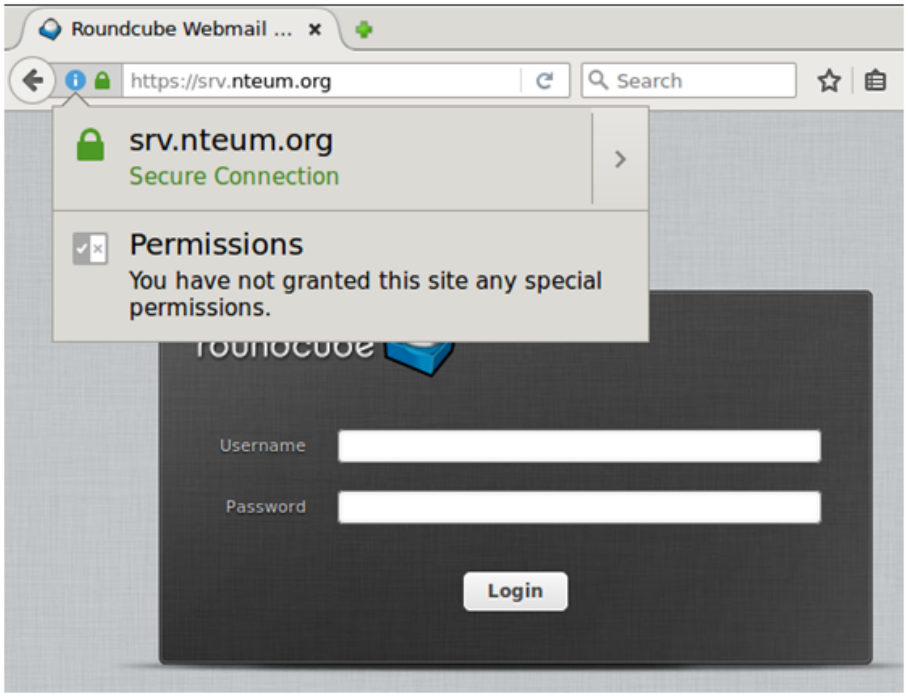
SSLCertificateFile /etc/ssl/certs/mail.pem
SSLCertificateKeyFile /etc/ssl/private/mail.key

Luego ejecutamos `a2enmod ssl` para habilitar el módulo de SSL, `a2ensite default-ssl` para habilitar el sitio web y `service apache2 reload` para reiniciar el servicio verificando que a través del navegador nos podemos conectar a la URL https://srv.nteum.org/ (después de aceptar la excepción ya que el certificado está firmado por nosotros mismos) y obtenemos la página de inicio de `RoundCube` por conexión con SSL como se puede apreciar en la figura 3.

Certificado firmado gratuito

Recordad que para evitar autofirmar el certificado y siempre que se tenga un dominio válido de Internet se puede obtener un certificado firmado gratuito desde StartSSL. Para ello se pueden seguir los pasos indicados en [StartSSL].

Figura 3



7) Si se desea habilitar la privacidad por IMAP (utilizando IMAPS), se deberá modificar Postfix y Dovecot para cifrar la comunicación con TLS. Para ello se utilizarán los certificados generados en el punto 5 y se deberá modificar

a) `vi /etc/postfix/main.cf`, y agregar al final:

```
smtpd_use_tls = yes
smtpd_tls_cert_file = /etc/ssl/certs/mail.pem
smtpd_tls_key_file = /etc/ssl/private/mail.key
smtpd_tls_session_cache_database = btree:${data_directory}/smtpd_scache
```

b) `vi /etc/postfix/master.cf`, y quitar el comentario a las líneas 28, 29 y 30:

```
smtps inet n       -       -       -       -       smtpd
 -o syslog_name=postfix/smtps
 -o smtpd_tls_wrappermode=yes
```

c) `vi /etc/dovecot/conf.d/10-ssl.conf`

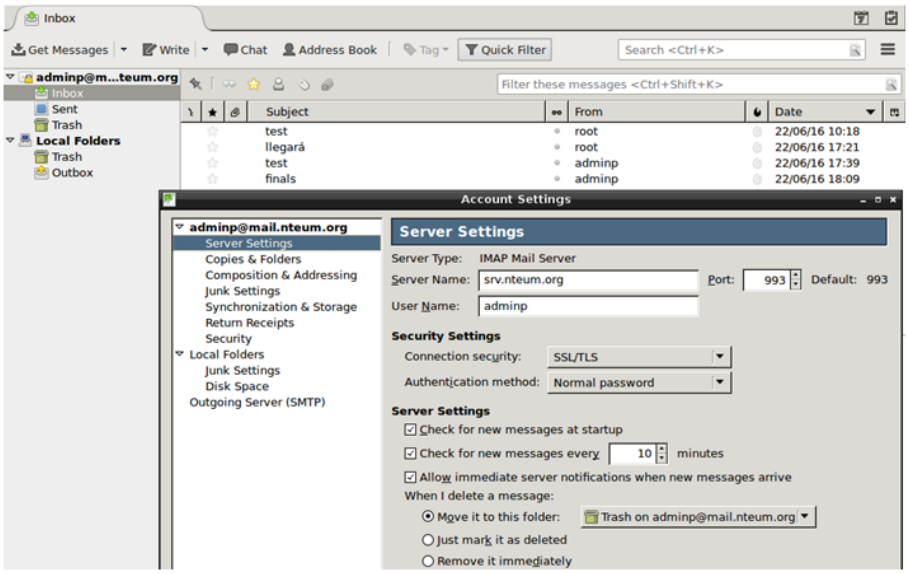
```
ssl = yes                                  cambiar línea 6
ssl_cert = </etc/ssl/certs/mail.pem        cambiar línea 12
ssl_key = </etc/ssl/private/mail.key       cambiar línea 13
```

d) Reiniciamos Postfix, `systemctl restart postfix`

e) Reiniciar Dovecot, `systemctl restart dovecot`

A continuación, en `Icedove`, cambiamos la configuración del servidor en las opciones de *security setting* a SSL/TLS y autentificamos con *normal passwd* y ya se podrá acceder por *imaps* al servidor `Postfix`, como muestra la figura 4.

Figura 4



Para instalar un antivirus asociado a `Postfix` deberemos primero instalar `ClamAV` haciendo:

```
apt-get install clamav
service clamav-freshclam stop                    parar el servicio
freshclam                                        actualizar las bases de datos
clamscan -r -i /home                             verificar su funcionamiento
```

A continuación, instalar los *daemons*:

```
apt-get install clamav-daemon clamsmtp
vi /etc/clamsmtpd.conf
```

Header: X-AV-Checked: ClamAV using ClamSMTP agregar header, línea 27
User: clamav cambiar el usuario

```
chown -R clamav. /var/spool/clamsmtp             cambiar uid y gid
chown -R clamav. /var/run/clamsmtp               cambiar uid y gid
vi /etc/postfix/main.cf                          agregar al final
```

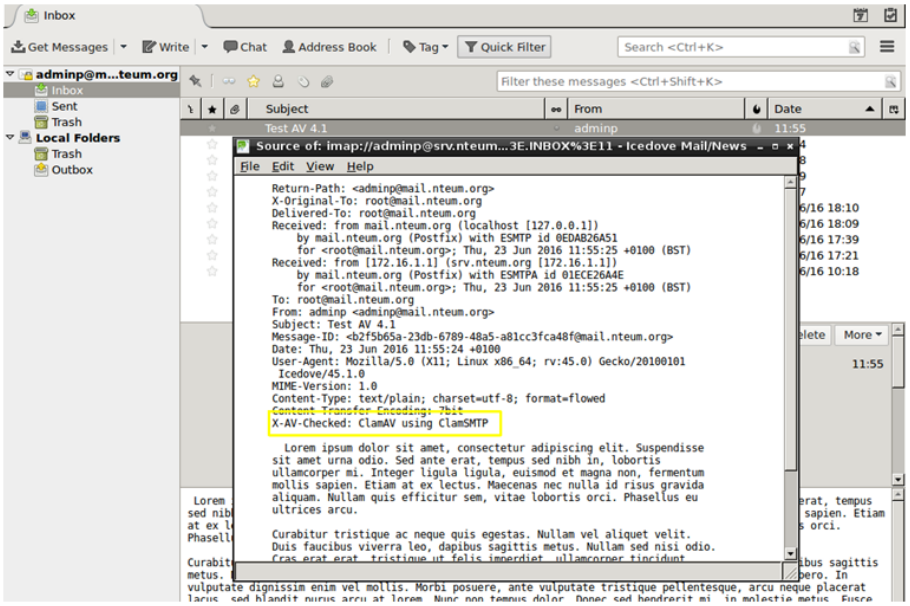
content_filter = scan:127.0.0.1:10026

```
vi /etc/postfix/master.cf                        agregar al final
```

```
scan unix  -  -  n  - 16 smtp
-o smtp_data_done_timeout=1200
-o smtp_send_xforward_command=yes
-o disable_dns_lookups=yes
127.0.0.1:10025 inet n  -  n  -  16 smtpd
-o content_filter=
-o local_recipient_maps=
-o relay_recipient_maps=
-o smtpd_restriction_classes=
-o smtpd_client_restrictions=
-o smtpd_helo_restrictions=
-o smtpd_sender_restrictions=
-o smtpd_recipient_restrictions=permit_mynetworks,reject
-o mynetworks_style=host
-o smtpd_authorized_xforward_hosts=127.0.0.0/8
```

Reiniciamos el sistema y enviamos un mail para verificar que tiene el *header*, tal y como lo hemos modificado (ver el recuadro marcado en la figura 5).

Figura 5



8) El último paquete/servicio esencial que nos falta instalar es el control de *spam* que se realizará mediante SpamAssassin. Para ello se debe ejecutar:

```
apt-get install spamassassin spamc
vi /etc/postfix/master.cf        modificar/agregar las líneas 12, 14, 33
```

```
smtp      inet  n       -       -       -       -       smtpd
   -o content_filter=spamassassin
submission inet n       -       -       -       -       smtpd
   -o content_filter=spamassassin
smtps     inet  n       -       -       -       -       smtpd
  -o syslog_name=postfix/smtps
  -o smtpd_tls_wrappermode=yes
  -o content_filter=spamassassin
```

Agregar al final de *master.cf* (todo en una línea):

```
spamassassin unix -     n       n       -       -       pipe
user=debian-spamd argv=/usr/bin/spamc -f -e /usr/sbin/sendmail -oi -f ${sender} ${recipient}
```

`vi /etc/spamassassin/local.cf` *modificar la línea 12*

```
rewrite_header Subject *****SPAM*****
```

`postfix reload` *para recargar la configuración de Postfix*

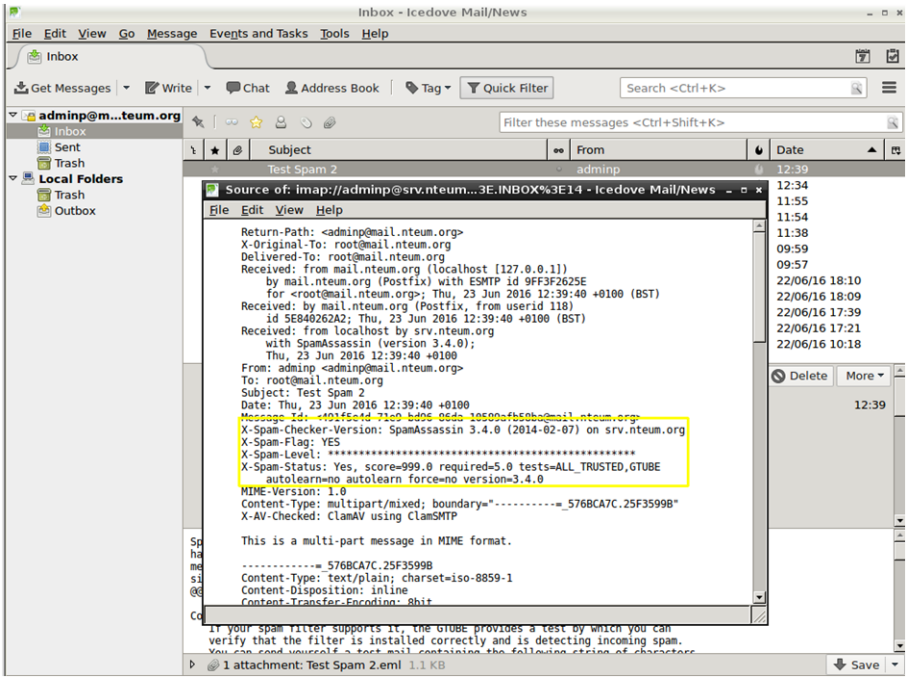
Para probar que el filtrado funciona* se puede probar con GTUBE (*Generic Test for Unsolicited Bulk Email http://spamassassin.apache.org/gtube/)* enviándonos un correo que incluya el siguiente *string*:

***Podemos enviarnos un correo nosotros mismos, pero solo veremos en la cabecera que ha filtrado sin problemas.**

XJS*C4JDBQADN1.NSBN3*2IDNEN*GTUBE-STANDARD-ANTI-UBE-TEST-EMAIL*C.34X

En la figura 6 se puede ver el resultado en el cuadrado marcado con un *status: Yes* y un *score= 999.0* sobre un *required=5*. Con ello ya podemos indicar al cliente qué debe hacer con el Spam, si eliminarlo o moverlo a la carpeta *Junk*.

Figura 6



9) Un aspecto importante es la monitorización del servidor y entre los paquetes útiles para ello se pueden instalar `pflogsumm` (simple y en modo texto), `MailGraph` (utiliza Apache y las librerías RRD) o `AWstats` (también en Apache y su configuración se puede seguir desde [SerWorld]).

Una opción interesante para el control del Spam y servidores dedicados con alto número de incidencias es Anti-Spam SMTP Proxy Server [ASSP]. EL ASSP Server es un proyecto *Open Source*, e independiente de la plataforma, que permite listas blancas autogeneradas, autoaprendizaje mediante *Bayesian filters*, *Greylisting*, DNSBL, DNSWL, URIBL, SPF, SRS, *Backscatter*, filtrado de virus, bloqueo de adjuntos y otras características avanzadas.

Lectura recomendada

Para detalles adicionales y otros aspectos interesantes se puede consultar las siguientes referencias [SerMail], [WorkMail], [DebSpam], [ASSP].

1.3. Grupos de discusión

Las *news* o grupos de discusión son soportados a través del protocolo NNTP. Instalar un servidor de grupos de discusión es necesario cuando se desea leer *news* fuera de línea, cuando se quiere tener un repetidor de los servidores centrales o se quiere un propio servidor maestro de *news*. Los servidores más comunes son INN o CNEWS, pero son paquetes complejos y destinados a grandes servidores. Leafnode es un paquete USENET que implementa el servidor TNP, especialmente indicado para sitios con grupos reducidos de usuarios, pero donde se desea acceder a gran cantidad de grupos de noticias. Este servidor se instala en la configuración básica de Debian y se pueden reconfigurar con `dpkg-reconfigure leafnode` todos parámetros como los servidores centrales, el tipo de conexión, etc. Este *daemon* se pone en marcha desde `inetd` de forma similar al `imap` (o con `xinetd`). Leafnode soporta filtros a través de expresiones regulares indicadas (del tipo `^Newsgroups:. *` [,] alt.flame$) en `/etc/news/leafnode/filters`, donde para cada mensaje se compara la cabecera con la expresión regular y, si existe coincidencia, el mensaje se rechaza.

La configuración de este servidor es simple y todos los archivos deben ser propiedad de un usuario de *news* con permiso de escritura (se debe verificar que dicho propietario existe en `/etc/passwd`). Todos los archivos de control, *news* y la configuración se encuentran en `/var/spool/news`, excepto la configuración del propio servidor que está en el fichero `/etc/news/leafnode/config`. En la configuración existen algunos parámetros obligatorios que se deben configurar (por ejemplo, para que el servidor pueda conectarse con los servidores maestros), como son `server` (servidor de *news* desde donde se obtendrán y enviarán las *news*) y `expire` (número de días a los que un hilo o sesión se borrará tras haber sido leído). Tenemos, asimismo, un conjunto de parámetros opcionales de ámbito general o específico del servidor que podrían configurarse. Para más información, consultad la documentación (`man leafnode` o `/usr/doc/leafnode/README.Debian`). Para verificar el funcionamiento del servidor, se puede hacer `telnet localhost nntp` y, si todo funciona correctamente, saldrá la identificación del servidor y se quedará esperando un comando. Como prueba, se puede introducir `help` (para abortar, haced `Ctrl+` y luego `Quit`).

1.4. *World Wide Web* (httpd)

Apache es uno de los servidores más populares y con mayores prestaciones de HTTP (*HyperText Transfer Protocol*). Apache tiene un diseño modular y soporta extensiones dinámicas de módulos durante su ejecución. Es muy configurable en cuanto al número de servidores y de módulos disponibles y soporta diversos mecanismos de autentificación, control de acceso, *metafiles*, *proxy caching*, servidores virtuales, etc. Con módulos (incluidos en Debian) es posible tener PHP3, Perl, Java Servlets, SSL y otras extensiones*.

*Podéis consultar la documentación en http://www.apache.org.

Apache está diseñado para ejecutarse como un proceso *daemon standalone*. En esta forma, crea un conjunto de procesos hijos que gestionarán las peticiones de entrada. También puede ejecutarse como *Internet daemon* a través de `inetd` o `xinetd`, por lo que se pondrá en marcha cada vez que se reciba una petición pero no es recomendado. La configuración del servidor puede ser extremadamente compleja según las necesidades (consultad la documentación); sin embargo, aquí veremos una configuración mínima aceptable. Su instalación es simple, por ejemplo en Debian,

```
apt-get install apache2 apache2-doc apache2-utils
```

La configuración del servidor estará en */etc/apache2* y por defecto el RootDirectory en */var/www/html*. Después de su instalación se pondrá en marcha y llamando a través de un navegador veremos que funciona (nos mostrará el famoso **It works!**). Existen 5 comandos que deberán están en mente de todo `apachectl` para gestionar la configuración del servidor (start|stop|restart|graceful|graceful-stop|configtest|status|fullstatus|help). Si bien todos los parámetros tienen valores funcionales por defecto, hay que destacar que en la instalación por defecto no está definida la variable `ServerName` y que se debería configurar en */etc/apache2/apache2.conf* o en los archivos de configuración de los sitios dentro del *tag* Virtualhost como `ServerName srv.nteum.org`*.

*Véanse ejemplos de *vitualhosts* en el subapartado siguiente.

La configuración de Apache2 en Debian es un poco diferente a la distribución general ya que intenta facilitar al máximo la configuración del servidor en cuanto a módulos, hosts virtuales y directivas de configuración (no obstante rápidamente se puede encontrar las equivalencias con otras distribuciones). Los principales archivos que se encuentran en el directorio */etc/apache2/* son *apache2.conf, ports.conf* y cinco directorios *mods-available*|*mods-enabled, sites-available*|*sites-enabled* y *conf.d*. Para información adicional leer */usr/share/doc/apache2.2** y en particular */usr/share/doc/apache2.2-common/README.Debian*.

1) *apache2.conf* es el archivo principal de configuración donde se define a nivel funcional las prestaciones del servidor y se llama a los archivos de configguración correspondientes (*ports, conf.d, sites-enabled*). Se recomienda poner como sufijo *.load* para los módulos que deban ser cargados y *.conf* para las configuraciones pero hay reglas más extensas en cuanto a los sufijos/nombres que

pueden ampliarse en la documentación (p. ej., se ignoran todos los archivos que no comienzan por letra o número).

2) *ports.conf* (se incluye en el archivo de configuración global) define los puertos donde se atenderán la conexiones entrantes, y cuales de estos son utilizados en los *host* virtuales.

3) Los archivos de configuración en *mods-enabled/* y *sites-enabled/* son para los sitios activos y los módulos que desean ser cargado en el servidor. Estas configuraciones son activadas creando un link simbólico desde los directorios respectivos ***-available/* utilizando los comandos `a2enmod/a2dismod`, `a2ensite/a2dissite`.

4) Los archivos de *conf.d* son para configuraciones de otros paquetes o agregados por el administrador y se recomienda que acaben con *.conf*.

5) Para que sea efectiva la configuración por defecto en estos directorios apache2 tendrá que ser gestionado a través de */etc/init.d/apache2* o `service` o `apache2ctl` (o también con `systemctl`).

6) El archivo *envvars* es el que contendrá la definición de las variables de entorno y es necesario modificar básicamente dos USER/GROUP que serán con las cuales se ejecutará y obtendrá los permisos. Por defecto se crea el usuario www-data y el grupo www-data (se pueden cambiar). Por lo cual deberá utilizarse `APACHE_RUN_USER=www-data` y `APACHE_RUN_GROUP=www-data`.

Apache también puede necesitar integrar diversos módulos en función de la tecnología que soporte y por lo cual se deberán agregar las librerías/paquetes correspondientes, por ejemplo:

1) Perl: `apt-get install libapache2-mod-perl2`

2) Rugby: `apt-get install libapache2-mod-ruby`

3) Python: `apt-get install libapache2-mod-python`

4) MySQL in Python: `apt-get install python-mysqldb`

5) PHP: `apt-get install php5 php5-cgi libapache2-mod-php5 php5-common php-pear`

6) PHP with MySQL: `apt-get install php5-mysql`

1.4.1. Servidores virtuales

Por servidores virtuales se entienden sitios aislados que serán servidos cada uno independiente del otro con sus propios archivos y configuración. En primer lugar deshabilitaremos el sitio por defecto con `a2dissite default`. Los sitios que crearemos serán remix.world y lucix.world que dispondrán de dos archivos de configuración en */etc/apache2/sites-available/* llamados como el dominio.

Contenido del archivo /etc/apache2/sites-available/remix.world.conf

```
<VirtualHost *:80>
    ServerAdmin adminpSySDW.nteum.org
    ServerName remix.world
    ServerAlias www.remix.world
    DocumentRoot /var/www/remix/
    ErrorLog /var/log/apache2/remix-error.log
    CustomLog /var/log/apache2/remix-access.log combined
    Options ExecCGI # habilitar Script en Perl
    AddHandler cgi-script .pl
</VirtualHost>
```

Contenido del archivo /etc/apache2/sites-available/lucix.world.conf

```
<VirtualHost *:80>
    ServerAdmin adminpSySDW.nteum.org
    ServerName lucix.world
    ServerAlias www.lucix.world
    DocumentRoot /var/www/lucix/
    ErrorLog /var/log/apache2/lucix-error.log
    CustomLog /var/log/apache2/lucix-access.log combined
    Options ExecCGI # habilitar Script en Perl
    AddHandler cgi-script .pl
</VirtualHost>
```

Esta configuración es básica y el estudiante deberá consultar la información detallada en [apa]. Como se puede observar los directorios raíz para cada dominio estarán en */var/www/remix|lucix* y los archivos de log en /errores/accesos en */var/log/apache2/mmmm-error.log* y *var/log/apache2/nnnn-access.log/*. Para crear los directorios `mkdir -p /var/www/remix`; `mkdir -p /var/www/lucix` y en los cuales se podría poner un *index.html* con alguna identificación que mostrara que dominio se está cargando. por ejemplo para remix.world:

```
<html><body><h1>REMIX: It works!</h1>
<p>This is the default web page for this server.</p>
<p>The web server software is running but no content has been added, yet.</p>
</body></html>
```

Y lo mismo para lucix.world pero cambiando la línea en <h1></h1>. Para los log no debemos hacer nada ya que el directorio */var/log/apache2* ya existe y los archivos los creará el servidor. Finalmente debemos activar los sitios (para ello debemos crear el enlace desde *sites-available* a *sites-enabled*) con `a2ensite remix.world.conf`; `a2ensite lucix.world.conf` y reiniciar apache2 con `service apache2 reload`. Como no disponemos de los dominios en un DNS primario podemos editar */etc/hosts* y agregar para la IP de nuestro servidor (p. ej., 192.168.1.37) dos líneas:

```
192.168.1.37 remix.world
192.168.1.37 lucix.world
```

Luego desde un navegador podremos introducir la URL remix.world y el resultado será la visualización del index.html que nos dirá: **REMIX: It works!**

Una de las ventajas de apache es que puede agregar funcionalidad a través de módulos especializados y que se encontrarán en */etc/apache2/mods-available/*.

Para obtener la lista de módulos en disponible para apache podemos hacer por ejemplo `apt-cache search libapache2*`, y para instalarlo `apt-get install [module-name]` los cuales estarán disponibles para su uso (recordad que puede ser necesario alguna configuración adicional en los archivos del sitio). Con `ls -al /etc/apache2/mods-available/` podemos mirar los disponibles e instalarlo con `a2enmod [module-name]`. Para listar los módulos cargados podemos hacer `apachectl -M` que nos listará con *shared* los cargados dinámicamente y con *static* los que se encuentran compilados con el servidor (estos se puede obtener también con `apache2 -l`). Los módulos en el directorio *mods-available* tienen extensiones *.load* (indica la librería a cargar) y *.conf* (configuración adicional del módulo) pero cuando utilizamos el comando `a2enmod` solo se debe indicar el nombre del módulo sin extensión. Para deshabilitar un módulo `a2dismod [module-name]`.

Como muestra de estas propiedades configuraremos un sitio seguro (https) bajo el dominio remix.world pero que redirigiremos al siguiente directorio: */var/www/remix.ssl*. En primer lugar crearemos un certificado (autofirmado) para nuestro sitio con `make-ssl-cert /usr/share/ssl-cert/ssleay.cnf /etc/ssl/private/remix.crt` indicándole el dominio que queremos validar (remix.world en nuestro caso) –solo introducir el dominio y dejar los alias en blanco– y si no podemos ejecutar `make-ssl-crt` asegurarnos que tenemos el paquete ssl-cert. Luego activamos el módulo SSL con `a2enmod ssl`, creamos el directorio */var/www/remix.ssl* y modificamos el index.html como hicimos con los anteriores. A continuación modificamos creamos la configuración del sitio (podemos utilizar la que viene por defecto modificándola):

```
cd /etc/apache2/sites-available; cp default-ssl remix.world.ssl.conf
```

Editamos el archivo remix.world.ssl.conf (solo mostramos las líneas principales/modificadas):

```
<IfModule mod_ssl.c>
<VirtualHost _default_:443>
    ServerAdmin adminpSySDW.nteum.org
    DocumentRoot /var/www/remix.ssl
    <Directory />
        Options FollowSymLinks
        AllowOverride None
    </Directory>
    <Directory /var/www/remix.ssl>
        Options Indexes FollowSymLinks MultiViews
        AllowOverride None
        Order allow,deny
        allow from all
    </Directory>
# líneas igual que el archivo original...
    ErrorLog $APACHE_LOG_DIR/remix.world.ssl_error.log
    CustomLog $APACHE_LOG_DIR/remix.world.ssl_access.log combined

    SSLEngine on
    SSLCertificateFile /etc/ssl/private/remix.crt
    #SSLCertificateKeyFile /etc/ssl/private/ssl-cert-snakeoil.key
# líneas igual que el archivo original...
</VirtualHost>
</IfModule>
```

Finalmente nos queda activar el sitio (`a2ensite remix.world.ssl.conf`), reiniciar apache2 (`service apache2 reload`) y desde el navegador hacer *https://remix.world* que como el certificado es autofirmado nos hará una advertencia y aceptaremos el certificado y deberemos obtener **SSL - REMIX: It works!** También se pueden configurar los certificados sin utilizar el *script* `make-ssl-cert` y utilizar los comandos como se hizo en el subapartado de la instalación de Postfix y RoundCube.

Un aspecto interesante es la función del archivo *.htaccess** en los directorios de nuestro dominio. Este archivo se puede utilizar para control de acceso al sitio (p. ej., habilitar/restringir IP), control de acceso a carpetas, listados, redirecciones (p. ej., a otra página/site, a otra carpeta, a otro dominio, a https, ...), evitar el *hotlinking* (para evitar que nos hagan enlaces a ficheros -generalmente vídeos- y consuman ancho de banda de nuestro servidor), cambiar la página por defecto, crear URL amigables, favorecer el cache de nuestro sitio, etc. Como muestra de ello para evitar por ejemplo que una carpeta sea inaccesible solo basta poner un archivo *.htaccess* en ella con el contenido `deny from all`. Para permitir que una carpeta de nuestro sitio (p. ej., del dominio remix.com/valid-user) tenga acceso con un usuario y passwd deberemos crear dentro de ella un archivo *.htaccess* con el siguiente contenido (también podemos crear un index.html modificado dentro de esta carpeta para verificar el funcionamiento):

***http://httpd.apache.org/docs/2.2/howto/htaccess.htm**

```
AuthName “Restricted Area”
AuthType Basic
AuthUserFile /etc/apache2/htpasswd
AuthGroupFile /dev/null
require valid-user
```

Para crear el usuario ejecutamos `htpasswd -c /etc/apache2/htpasswd adminp` que nos pedirá el passwd para este usuario y los almacenará en el archivo indicado. Luego ponemos como URL *http://remix.world/valid-user/* nos pedirá el usuario (adminp) y el passwd que almacenamos y entonces veremos: **REMIX->Valid-User: It works!**. En caso contrario nos continuará pidiendo el usuario/passwd y si hacemos Cancel no indicará un mensaje de Authorization Required impidiendo el acceso.

Para probar si PHP funciona después de instalarlo (véase el subapartado anterior) deberemos modificar el *Timezone* editando *vi /etc/ php5/apache2/php.ini* y modificando *date.timezone* = *“Europe/Madrid”*. Después debemos reiniciar el servidor `systemctl restart apache2`.

Para probar su funcionamiento debemos crear una página (*vi /var/www/html/index.php)* con el siguiente contenido:

```
<html>
<body>
<div style="width: 100%; font-size: 40px; font-weight: bold;
text-align:center;">
```

```
<?php
     print Date("Y/m/d");
?>
</div>
</body>
</html>
```

Llamando a la URL http://srv.nteum.org/index.php se debería de ver la fecha. En general funciona correctamente pero si presenta errores y no se visualiza la fecha, se pueden revisar los *logs* de Apache (*var/log/apache2/**) para comprobar si los módulos están habilitados (*mods-enabled/ php5.conf*). Se puede forzar agregando un *handler* en `vi /etc/apache2/mods-enabled/mime.conf` y agregar *AddHandler php5-script .php*.

Ya hemos estudiado la autentificación de usuarios a través del servidor; es muy útil para el acceso de usuarios que no forman parte del sistema operativo. Si se desea habilitar a estos usuarios es necesario vincularlos a Apache a través del sistema de autentificación PAM. Para ello se instalará

```
apt-get install libapache2-mod-authnz-external pwauth
```

Luego crearemos un archivo de configuración *vi /etc/apache2/sites-available/ auth-pam.conf* con el siguiente contenido:

```
AddExternalAuth pwauth  /usr/sbin/pwauth
SetExternalAuthMethod pwauth pipe
<Directory /var/www/html/pam>
      AuthType Basic
      AuthName "PAM Authentication"
      AuthBasicProvider external
      AuthExternal pwauth
      require valid-user
</Directory>
```

Se creará un directorio `mkdir /var/www/html/pam` y dentro de podrá un archivo *index.html* con un identificador que se está accediendo a PAM. Finalmente, se habilitará el sitio `a2ensite auth-pam`, se reiniciará el servidor `systemctl restart apache2` y se podrá acceder a http://srv.nteum.org/ *pam/* después de introducir un usuario local y el *passwd*. La autentificación se realiza con la integración del módulo *authnz-external* y el comando `pwauth`.

1.4.2. Apache + PHP + Mysql + PhpMyAdmin

Una cuestión importante para los servidores web dinámicos es aprovechar las ventajas de Apache PHP y una base de datos como MySQL incluyendo un programa administrador de MySQL como PHPMyAdmin, todo ello fun-

cionando conjuntamente. Las distribuciones han evolucionado mucho y en Debian es sumamente fácil poner en marcha este conjunto (pero tampoco representa ninguna dificultad bajarse el software fuente, compilarlo e instalarlo si se desea, por ejemplo, tener las últimas versiones de los paquetes por algún motivo pero recordad que implicará más trabajo y dedicación).

En primer lugar suponemos que tenemos PHP instalado y funcionando (véase el subapartado anterior). Como se puede observar cuando se instala PHP, se cambia del modo MPM-Worker a MPM-prefork. MPM-worker es un módulo de multi-procesamiento que puede manejar múltiples peticiones rápidamente utilizando múltiples *threads* por proceso cliente. Sin embargo este no es compatible con alguna extensiones PHP y por ello el MPM-worker es reemplazado por MPM-prefork, que permite manejar todas las peticiones PHP (en modo compatibilidad) y evitar que si una petición falla pueda afectar a otras peticiones. Existe otro módulo llamado mpm-itk (http://mpm-itk.sesse.net/) que es similar prefork pero tiene mejores prestaciones y gestión de permisos (consultar la bibliografía en apache.org). Para verificar que PHP funciona creamos un fichero por ejemplo dentro de RootDirectory de remix.world llamado *test.php* con el siguiente contenido: *<?php phpinfo() ?>* y si en la URL introducimos *http://remix.world/test.php* deberemos ver una tabla con la versión e información sobre el paquete PHP instalado.

Para instalar los paquetes MySQL y PHPMyAdmin haremos

```
apt-get install mysql-server
```

(es importante recordar la contraseña de acceso que introduzcamos pero siempre podemos hacer `dpkg-reconfigure mysql-server` que perderemos todo lo que haya en la BD pero también hay otros métodos -menos agresivos- para recuperar la contraseña del root). Luego para instalar PHPMyAdmin haremos `apt-get install phpmyadmin` y prestar atención que nos pedirá la clave de acceso para entrar en la base de datos y crear una clave de acceso para entrar en la aplicación vía un navegador. Luego podremos poner en la URL de nuestro navegador *http://localhost/phpmyadmin*, nos solicitará el usuario (root generalmente) y el passwd que hemos introducido y ya podremos gestionar el servidor de bases de datos MySQL.

1.4.3. Otros servidores httpd

Lighttpd es un servidor web (con licencia BSD) diseñado para ser rápido, seguro, flexible, que implementa la mayoria de los estándares y está optimizado para entornos donde la velocidad es muy importante (consume menos CPU/RAM que otros servidores) y es muy apropiado para cualquier servidor que tenga que dar soporte a grandes cargas. Entre sus principales características están la de Virtual hosting, redirecciones http y reescrituras de URL, dar soporte a CGI, SCGI y FastCGI, PHP, Ruby, Python entre otros y además con consumo de memoria constante.

Su instalación en Debian es `apt-get install lighttpd`, y si tenemos apache sobre el puerto 80 nos dará un error. Para ello debemos editar el archivo `/etc/lighttpd/lighttpd.conf` y cambiar la línea `server.port = 8080` y reiniciar `service lighttpd start`. Desde el navegador se puede hacer *http://localhost:8080index.lighttpd.html* y veremos la página inicial de lighttpd. Por defecto Lighttpd tiene su directorio raíz en */var/www* (en Debian) y el archivo de configuración en */etc/lighttpd/lighttpd.conf*. Configuraciones adicionales están en */etc/lighttpd/conf-available* y puede ser habilitadas con el comando `lighttpd-enable-mod` el cual crea enlaces entre *conf-enabled* y *conf-available*, las cuales se pueden deshabilitar con `lighttpd-disable-mod`.

Para habilitar el servidor de FastCGI con la intención de ejecutar PHP deberemos instalar PHP-FPM con `apt-get install php5-fpm php5` y sobre el archivo */etc/php5/fpm/php.ini* quitar el comentario a `cgi.fix_pathinfo=1`. Luego deberemos activar el servidor PHP-FPM, por lo cual haremos una copia del archivo original y lo modificaremos:

```
cd /etc/lighttpd/conf-available/
cp 15-fastcgi-php.conf 15-fastcgi-php-spawnfcgi.conf
```

Modificar *15-fastcgi-php.conf* con:

```
# -*- depends: fastcgi -*-

# Start an FastCGI server for php
fastcgi.server += ( ".php" =>
        ((
                "socket" => "/var/run/php5-fpm.sock",
                "broken-scriptfilename" => "enable"
        ))
)
```

Para habilitar fastcgi deberemos cargar los módulos `lighttpd-enable-mod fastcgi` y `lighttpd-enable-mod fastcgi-php` lo cual crea los enlaces correspondientes que podemos ver con `ls -l /etc/lighttpd/conf-enabled`. Luego podemos reiniciar `service lighttpd force-reload`. Para visualizar si el servidor y FastCGI funciona creamos un archivo */var/www/info.php* con el siguiente contenido `<?php phpinfo(); ?>` y podremos visualizar (http://localhost:8080/info.php) la página de configuración de PHP donde indica como Server API = FPM/FastCGI.

Otro servidor muy utilizado actualmente es **Nginx** (http://nginx.org/) programado en C y licencia BSD. Sus funciones principales son como servidor web/proxy inverso de muy alto rendimiento (puede soportar más de 10.000 conexiones simultáneas) y también puede funcionar como proxy para protocolos de correo electrónico (IMAP/POP3). Es un servidor utilizado por grandes

instalaciones (WordPress, Netflix, Hulu, GitHub, y partes de Facebook entre otros) y entre sus principales características están (además de servidor de archivos estáticos, índices y autoindexado y proxy inverso con opciones de caché) el balanceo de carga, tolerancia a fallos, SSL, FastCGI, servidores virtuales, streaming de archivos (FLV y MP4.8), soporte para autenticación, compatible con IPv6 y SPDY. Su instalación básica es simple y para su configuración básica* haced `apt-get install nginx`; luego ejecutad `cd /var/www/html; mv index.nginx-debian.html index.html` y cargad desde el navegador la URL *http://srv.nteum.org/*. Se verá la página básica de incio de Nginx.

Enlace de interés

Para una configuración detallada de Nginx podéis consultar su documentación en http://nginx.org/.

***Consultad la wiki de nginx en http://wiki.nginx.org /Configuration**

Enlace de interés

Una referencia interesante (aunque es un poco antigua, la mayor parte de la configuración es útil) es https://www.howtoforge.com/perfect-server-ubuntu-12.04-lts-nginx-bind-dovecot-ispconfig-3 donde sobre Ubuntu se instala y configura nginx, BIND, Dovecot para la instalación de ISPConfig 3. ISPConfig 3 es un panel de control que permite configurar diferentes servicios a través de un navegador (Apache or nginx, Postfix , Courier/Dovecot IMAP/POP3, MySQL, BIND/MyDNS, PureFTPd, SpamAssassin, ClamAV, entre otros).

1.4.4. Test de validación y prestaciones de Apache2

Una vez instalado y configurado Apache2 se puede probar la creación de algunas páginas con diferentes elementos y hacer una validación funcional, pero para un administrador es muy importante hacer ciertas investigaciones sobre el software instalado y cómo responde este ante cargas intensas y diferentes situaciones/protocolos o simplemente ante paquetes TCP. Estas pruebas proporcionarán información objetiva sobre la adaptación del servicio al entorno y la calidad del mismo permitiendo al administrador encontrar los puntos débiles y/o disfunciones, y permite también ajustar la gran cantidad de parámetros de que dispone este.

Primero y de forma simple se puede usar **ApacheBench** [ABench], que es una herramienta de evaluación comparativa para servidores web. Es una herramienta potente, instalada ya con Apache2 dentro del paquete `apache2-utils` (si no está instalado, habrá que ejecutar `apt-get install apache2-utils`) y que es fácil de utilizar. Permite extraer resultados interesantes sin experiencia previa en planes de carga ni monitorización de servicios. Un aspecto interesante es que, para no producir desviaciones en el software complementario de analíticas web, ApacheBench utiliza un *user agent* específico para que pueda ser ignorado por la mayoría de software de analíticas (aunque como se describe en la documentación puede haber algunos casos en que las estadísticas pueden verse afectadas con determinado software de analítica Web). Una ejecución simple sería:

```
ab -n 10000 -c 100 http://srv.nteum.org/
```
no olvidar poner la / final

Donde se están generando 10.000 llamadas a srv.nteum.org distribuidas en 100 *threads* (hilos) para analizar la capacidad de concurrencia y comprobar situaciones de bloqueos o condiciones de carrera que puedan dejar al servidor inutilizado. Los resultados obtenidos son (mostramos aquí las líneas más relevantes):

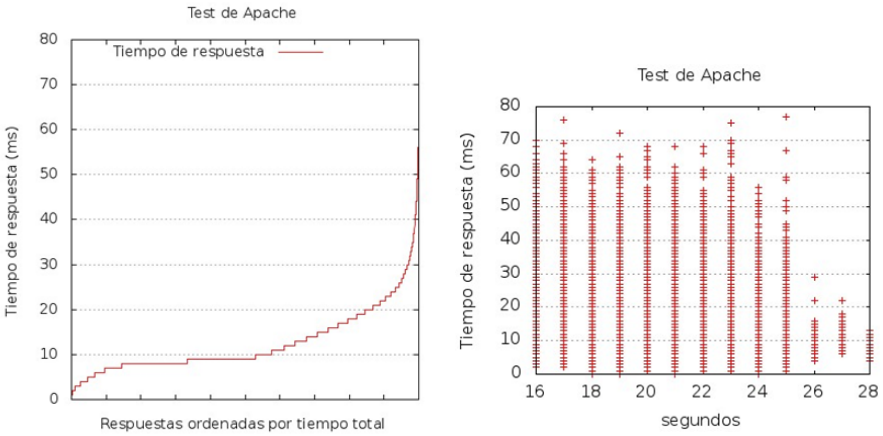
```
This is ApacheBench, Version 2.3 <$Revision: 1604373 $>
...
Document Path:          /
Document Length:        280 bytes
Concurrency Level:      100
Time taken for tests:   1.695 seconds
Complete requests:      10000
Failed requests:        0
Total transferred:      5510000 bytes
HTML transferred:       2800000 bytes
Requests per second:    5898.34 [#/sec] (mean)
Time per request:       16.954 [ms] (mean)
Time per request:       0.170 [ms] (mean, across all concurrent requests)
Transfer rate:          3173.82 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:        0    0   0.2      0       3
Processing:     1   17   6.7     15      62
Waiting:        1   15   6.7     14      62
Total:          2   17   6.7     16      63
Percentage of the requests served within a certain time (ms)
  50%     16
  66%     18
  75%     19
  80%     21
  90%     25
  95%     29
  98%     36
  99%     42
 100%     63 (longest request)
```

Los valores más interesantes son el *Requests per second* (peticiones atendidas por segundo), *Time per request* (tiempo medio en atender a un grupo de peticiones concurrentes) y *Time per request* (tiempo medio en atender una petición individual) y los valores mínimos, media, moda y máximos y si el servidor ha podido servir todas las peticiones o no (*Failed requests*). Agregando el parámetro `-g /tmp/output.txt` podremos generar los datos en formato *gnuplot* para luego visualizarlos. Para ello es interesante utilizar *scripts* como los de [BranScripts] para obtener en primer lugar el rango de tiempo de las peticiones entre las que tardan menos y las que tardan más (cuidado que la gráfica izquierda de la figura 7 no está ordenada en el orden de las peticiones sino en el valor de *ttime –total time–*). La gráfica de la derecha de la figura 7 contiene los mismos datos pero para cada segundo de la prueba (en total 12 segundos) se indica la distribución del tiempo de respuesta donde se puede observar la densidad y el máximo/mínimo de cada petición (los datos fueron obtenidos en local poniendo en el servidor una página de cierta complejidad y haciendo 100.000 peticiones en 100 *threads* concurrentes y transfiriendo 19 Gigabytes con:

`ab -n 100000 -c 100 -g /tmp/output.txt http://srv.nteum.org/test.htm`).

Figura 7

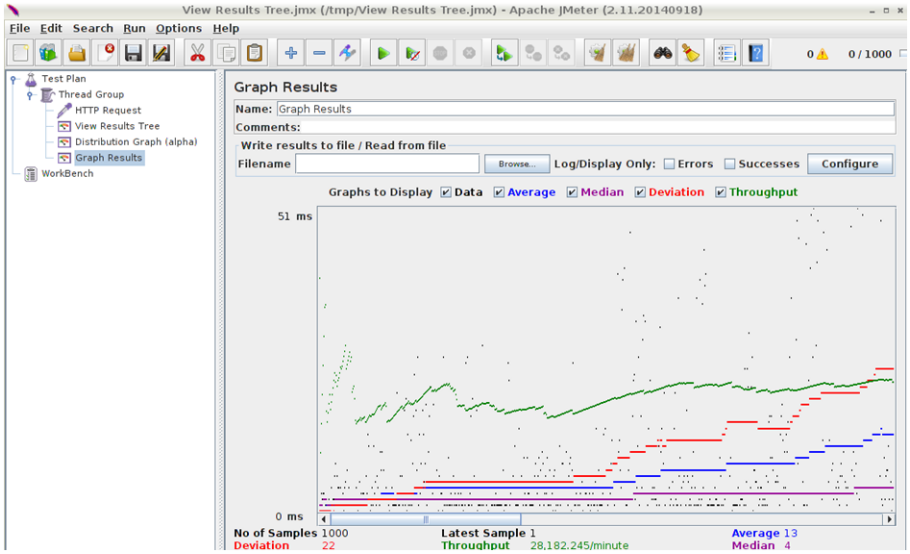


Como conclusión podemos extraer que en esta configuración de Apache el servidor responde muy bien a una gran cantidad de transacciones concurrentes y tiempos de respuesta muy aceptables. Esta herramienta admite una gran cantidad de parámetros [ABench], entre otros, que le podamos pasar una *cookie* de sesión o para evaluar páginas que tengan usuario y *passwd* (o simplemente los valores si la autentificación de básica), protocolos que hay que analizar o utilización de proxy.

Una herramienta más sofisticada y adecuada para realizar análisis complejos y detallados, bajo diferentes tipos de carga, es **JMeter** [Jmet]. Esta herramienta está escrita en Java y fue inicialmente diseñada para medir las prestaciones de aplicaciones web, pero puede ser extendida a otros tipos de prueba y es muy potente para probar el rendimiento tanto en recursos estáticos como en dinámicos (servicios web –SOAP/REST–, web con diferentes lenguajes –PHP, Java, ASP.NET, Files, etc.–, objetos Java, consultas a bases de datos, servidores FTP, etc.). Se puede utilizar para simular una carga pesada concurrente en un servidor, grupo de servidores, la red o un objeto y analizar el rendimiento general bajo diferentes perfiles y realizar un análisis gráfico de rendimiento. Su instalación en Debian es muy simple `apt-get install jmeter` y para ejecutarlo hacemos simplemente `jmeter`. Cuando se inicia se verán dos posibilidades de trabajo *Test Plan*, que permitirá configurar el test que se ha de realizar y guardarlo, y un espacio temporal de trabajo (*Workbench)* donde permitirá probar y utilizar opciones pero que no formarán parte del test de trabajo. La forma de trabajo es muy simple, ya que sobre la opción de *Test Plan* y el botón derecho se pueden agregar diferentes pruebas (p. ej., *Add->Thread Users-> Thread Group*) y luego configurarlo. Es necesario salvar el *Test Plan* desde el menú principal, pero también se puede salvar la configuración de un determinado elemento (*Save Selection*) y luego cargar solo esta parte (*Merge*). Siempre que se seleccione un elemento en la pantalla derecha se podrán ver/modificar las configuraciones específicas de este objeto.

En la figura 8 se muestra la distribución del tiempo de respuesta en una prueba contra nuestro servidor (srv.nteum.org) con 1000 *threads*. Su configuración es muy simple donde al *Test Plan* se le ha agregado un *Thread Group* y a este un *Sampler=HTTP Request* (configurado para hacer la petición a la IP de servidor) y luego tres *Listeners* para recoger los resultados (solo se muestran los resultados del tercer Listener *Graph Results*).

Figura 8



Lectura recomendada

Existen una gran cantidad de tutoriales [TutPoint] [GuTut] y también es muy útil el manual de usuario [JmUM] para obtener información sobre todos los aspectos de utilización y prueba para diferentes lenguajes, bases de datos, servicios, etc.

Finalmente, para completar el análisis de un servidor Web es necesario contar con herramientas que nos permitan analizar los registros de conexión para saber toda la información que recibe/provee nuestro servidor. Dos de las herramientas más comunes para este fin son AWStats [AW] y Webalizer [WA] (o un proyecto derivado de este AWFFull [AWFull]); a continuación, veremos algunos detalles y la instalación de la primera herramienta.

AWStats es una de las herramientas que permiten hacer análisis y obtener estadísticas de un servidor web presentando informes detallados de tablas y gráficos de barra. La forma habitual de acceder es a través del mismo servidor web protegido con usuario y contraseña (Apache en nuestro caso) para visualizar las estadísticas generadas periódicamente a través del servicio *cron*. Una de las principales ventajas es que es muy versátil, ya que soporta la mayoría de los formatos de archivos *log* de servidor web conocidos (Apache, WebStar, IIS . . .) e incorpora *plugins* para leer otros tipos de *logs* (por ejemplo, los de un servidor de correo –para su configuración consultar [SerWorld]–).

Para instalarlo en una configuración mínima haremos lo siguiente:

1) `apt-get install awstats perl` y posteriormente se deberá editar el archivo */etc/awstats/awstats.conf* y editar/verificar las siguientes líneas:

```
# Log de Apache
LogFile="/var/log/apache2/access.log"
# Formato para Apache (combined logs)
LogFormat=1
# dominio utilizado para el sitio
SiteDomain="srv.nteum.org "
# Alias y directorios de Icons
HostAliases="localhost 127.0.0.1"
DirIcons="./icon"
```

2) Crear el directorio dentro del *DocumentRoot* de Apache:

```
mkdir /var/www/html /awstats
```

3) Hacer un enlace al directorio de *Icons*:

```
ln -s /usr/share/awstats/icon /var/www/html/awstats/icon
```

4) Crear el archivo de configuración para Apache:

```
vi /etc/apache2/sites-available/awstats.conf
```

```
<VirtualHost awstats.nteum.org:80>
    ServerAdmin webmaster@example.com
    ServerName awstats.nteum.org
    DocumentRoot /var/www/html/awstats
    <Directory /var/www/html/awstats>
      #AuthGroupFile /dev/null
      AuthType Basic
      AuthUserFile /var/www/.htpasswd
      AuthName "Access Restricted"
      Require valid-user
      AuthType Basic
      Order deny,allow
      Deny from all
      Allow from 172.16.1.0/16
    </Directory>

    ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/
    <Directory "/usr/lib/cgi-bin">
      AllowOverride None
      Options +ExecCGI -MultiViews +SymLinksIfOwnerMatch
      Order deny,allow
      Deny from all
      Allow from 172.16.1.0/16
    </Directory>
</VirtualHost>
```

5) Crear un usuario y *passwd:* `htpasswd -c /var/www/.htpasswd adminp`

6) Habilitar el sitio (`a2ensite awstats.conf`) y verificar la configuración

```
apachectl configtest
```

7) Reiniciar Apache: `systemctl restart apache2`

8) Generar el contenido y los reportes:

```
/usr/lib/cgi-bin/awstats.pl -config=apache -update
```

```
/usr/lib/cgi-bin/awstats.pl -config=apache -output -staticlink
> /var/www/html/awstats/index.html
```

9) Acceder a http://awstats.nteum.org introduciendo el usuario y *passwd* generados para visualizar el contenido.

10) Editar el cron para que se actualicen los datos:

`crontab -l` e insertando:

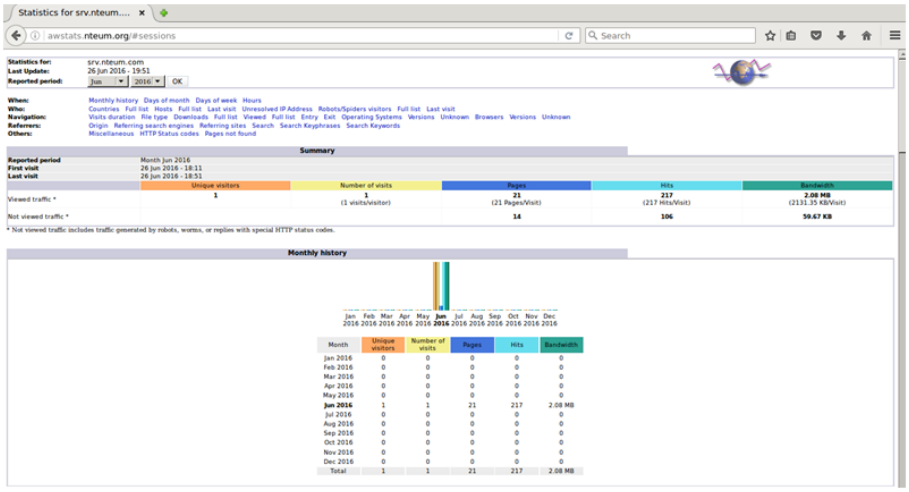
```
0 * * * * /usr/lib/cgi-bin/awstats.pl -config=apache -update
5 * * * * /usr/lib/cgi-bin/awstats.pl -config=apache -output -staticlink
          > /var/www/html/awstats/index.html
```

Es importante tener en cuenta que para algunas opciones debemos tener la ejecución de *script perl* activada por lo cual se deberá hacer

```
vi /etc/apache2/mods-enabled/mime.conf
```

y en la línea 219 se quitará el comentario y además se agregará la extensión `.pl` *AddHandler cgi-script .cgi .pl*. También se debe habilitar el módulo `a2enmod cgid` y reiniciar el servidor `systemctl restart apache2`. La figura 9 muestra una vista (parcial) de la herramienta.

Figura 9



1.5. Servidor de WebDav

El nombre webDAV son las siglas de *Web Based Distributed Authoring and Versioning* (también se refiere al grupo de trabajo de Internet Engineering Task Force) y es un protocolo que permite que la web se transforme en un medio legible y editable y proporciona funcionalidades para crear, cambiar y mover documentos en un servidor remoto (típicamente un servidor web). Esto se utiliza sobre todo para permitir la edición de los documentos que envía un servidor web, pero puede también aplicarse a sistemas de almacenamiento generales basados en la web y a los que se puede acceder desde cualquier lugar. En este subapartado instalaremos un servidor WebDav sobre Apache. El proceso es el siguiente:

Enlace de interés

Sobre la integración de WebDav con Apache podéis consultar el artículo "WebDAV on Apache2" disponible en: http://www.debian-administration.org/articles/285

1) Verificar que tenemos instalado apache2 y si no realizar su instalación como hemos visto anteriormente y verificar que funciona (`apt-get install apache2`).

2) Habilitar los módulos de Apache que son necesarios para WebDav: `a2enmod dav_fs` y `a2enmod dav`.

3) Crear el directorio para el directorio virtual (podemos hacer por ejemplo `mkdir -p /var/www/webdav`) y permitir que Apache sea el propietario del directorio `chown www-data /var/www/webdav/`.

4) Crear el archivo `/etc/apache2/sites-available/webdav.conf` para definir la funcionalidad del servidor (en esta configuración estamos configurando todo el servidor como WebDav pero podría ser un servidor virtual y en modo SSL para mayor seguridad):

```
<VirtualHost *:80>
    ServerAdmin adminpSySDW.nteum.org
    DocumentRoot /var/www/webdav/
    ErrorLog /var/log/apache2/webdav-error.log
    CustomLog /var/log/apache2/webdav-access.log combined
   <Directory /var/www/webdav>
       Options Indexes MultiViews
       AllowOverride None
       Order allow,deny
       allow from all
       DAV On
       AuthName "Restricted WeDav Area"
       AuthType Basic
       AuthUserFile /etc/apache2/htpasswd
       AuthGroupFile /dev/null
       require valid-user
   </Directory>
</VirtualHost>
```

Se puede comprobar que la configuración es correcta con la orden `apache2ctl configtest`. A continuación, se habilita el sitio con `a2ensite webdav`.

5) Como se ha hecho anteriormente, se crean los usuarios (con la orden `htpasswd [-c] /etc/apache2/htpasswd usuario`) indicando el `-c` si es el primer usuario a crear.

6) Se reinicia Apache para que lea la configuración `/etc/init.d/apache2 reload` (o con `service restart apache2` o también utilizando la orden `apache2ctl restart`) y ya nos podemos conectar a `http://localhost` (o http://srv.nteum.org/), previa autentificación.

7) Desde un GNU/Linux podemos probar la funcionalidad del servidor abriendo el `nautilus` –o equivalente, p. ej. `PCManFM`– (gestor de ficheros) y desde el menu *File->Connect to Server* podemos seleccionar Servidor WebDav introduciendo los datos (IP, directorio, usuario, passwd) y tendremos acceso como si de una carpeta local se tratara.

8) Desde MacOS podemos utilizar el mismo procedimiento que el anterior desde el gestor de archivos o instalar un cliente específico (igualmente para Windows). El más recomendado para ello es CyberDuck (http://cyberduck.io/) que tiene licencia GPL y es una excelente aplicación (soporta múltiples protocolos) y muy fácil de configurar.

9) Otra forma de probarlo es con un cliente WebDav (en modo texto), p. ej. Cadaver (http://www.webdav.org/cadaver), con `apt-get install cadaver`. Después, conectamos al servidor con `cadaver IP-nombre del servidor`

y después de autentificarnos, podemos crear un directorio (`mkdir`), editar un archivo, listar un directorio (`ls`), cambiar de directorio (`cd`), cambiar los permisos de ejecución (`chexec`), borrarlo (`rm`), etc.

En muchas ocasiones, y dado que estaremos haciendo transferencias de archivos, es importante preservar la privacidad por lo cual sería adecuado trabajar con WebDav pero sobre SSL. Su configuración no implica mayores complicaciones y veremos una forma diferente de generar los certificados (serán autofirmados –hasta que podamos tener nuestra propia entidad certificadora–) para un dominio en particular, en nuestro caso webdav.nteum.org. Para ello hacemos:

1) Nos cambiamos al directorio donde almacenaremos los certificados haciendo `cd /etc/ssl/private`. Hacemos la petición del certificado:

```
openssl req -config /etc/ssl/openssl.cnf -new -out webdav.csr
```

Este comando nos pedirá un *passwd* y una serie de información que quedará en el certificado pero la más importante es Common Name (CN) que será donde validará el certificado (en nuestro caso webdav.nteum.org). Podemos verificar la petición con:

```
openssl req -in /etc/ssl/private/webdav.csr -noout -text
```

2) Creamos la llave (*key*):

```
openssl rsa -in privkey.pem -out webdav.key
```

3) Firmamos:

```
openssl x509 -in webdav.csr -out webdav.crt -req -signkey webdav.key
-days 3650
```

Podemos verificarlo con:

```
openssl x509 -noout -in /etc/ssl/private/webdav.crt -text
```

4) Generamos en certificado en formato DER:

```
openssl x509 -in webdav.crt -out webdav.der.crt -outform DER
```

Se puede verificar con:

```
openssl x509 -in /etc/ssl/private/webdav.der.crt -inform der
-noout -text
```

5) Ahora generamos el archivo de configuración de apache a partir del anterior:

```
cd /etc/apache2/sites-available; cp webdav.conf webdav-ssl.conf
```

Modificamos para incluir las siguiente cuatro líneas al inicio y modificar el VirtualHost:

```
<VirtualHost *:443>
ServerName webdav.nteum.org
SSLEngine on
SSLCertificateFile /etc/ssl/private/webdav.crt
SSLCertificateKeyFile /etc/ssl/private/webdav.key
...
```

Solo nos resta activar el sitio (`a2ensite webdav-ssl.conf`), reiniciar apache (`service apache2 restart`, agregar una línea en el archivo */etc/hosts* con *172.16.1.1 webdav.nteum.org webdav*) y verificar que funciona en la dirección https://webdav.nteum.org, previa aceptación del certificado.

1.6. Proxies

La función de un *proxy* es jugar el papel de intermediario en las peticiones que solicita el cliente a otro servidor, es decir, el servidor *proxy* conoce ambos recursos y los pone en contacto sin que uno conozca al otro. Como actúa de punto de unión entre las peticiones y los servicios, permite llevar a cabo diferentes acciones como control de acceso, registro/control del tráfico (incluido bloqueo), mejora del rendimiento de la transacción (almacenamiento intermedio), anonimato en la comunicación, entre otras. Dado que actúa como intermediario, existen diversas opiniones/controversias sobre la utilización de *proxies* en cuanto a la seguridad y anonimato. Es por ello que es necesario cuidar bien su configuración y el servicio que presta para que no sea posible utilizarlo con otro fin que aquel para el cual ha sido concebido.

Si bien se pueden encontrar diferentes tipos de *proxies*, normalmente diferenciados por el protocolo/aplicación que gestionan (web, ftp, ARP, dns, ...), el utilizado en servicios web es probablemente el más habitual. Entre las ventajas de un *proxy* web se pueden enumerar control de tránsito, velocidad (*proxy cache*) y filtrado a nivel de aplicación/protocolo y entre las desventajas, el anonimato/abuso (el cliente nunca es responsable de la petición del servicio), carga y cuello de botella, paso adicional en la comunicación entre cliente y servidor e irregularidad (tiempo de respuesta variable en función de la carga del *proxy*).

En función del rol que cumplan tenemos diferentes definiciones de *proxies*:

1) Un servidor *proxy* que pasa las peticiones y las respuestas no modificadas generalmente se llama **puerta de enlace** (*gateway* o *tunneling proxy)*.

2) Un *proxy forward* es un servidor que conecta Internet con clientes internos que realizan peticiones a recursos externos. Generalmente se combina con un *proxy cache* para acelerar el acceso a los recursos ya que solo el primer usuario que los solicita es el que accede al recurso y los subsiguientes acceden a la copia en el servidor *proxy*.

3) Un *reverse proxy* (inverso) es, generalmente, un servidor que recibe las peticiones de Internet y las deriva a servidores internos (por ejemplo, en una red privada) protegiéndolos y permitiendo balancear la carga entre varios servidores.

Si el *proxy* está conectado desde y hacia Internet se considera un *proxy* abierto (*open proxy*) y su función es reenviar todos los paquetes que recibe permitiendo ocultar la IP del cliente al servidor lo cual es una forma de anonimato (débil). Existen extensas listas de *open proxies* (solo basta hacer la consulta https://www.google.es/search?q=open+proxies+list) pero nadie puede dar fe del anonimato que permiten. Si se desea tener anonimato real se debe utilizar redes tales como Tor (*The Onion Router*) que permite tener garantías de anonimato utilizando comunicaciones encriptadas (múltiples veces) y que pasan a través de una red mundial de servidores (voluntarios) permitiendo así obtener el anonimato de la comunicación e impidiendo que esta pueda ser vigilada o supervisada [Tor]. Otra red similar que permite el anonimato es la red I2P [I2P] del proyecto *Invisible Internet Project* con objetivos similares a Tor pero que no está tan difundida.

Una pregunta frecuente es la diferencia entre un *proxy* y un *firewall* (actuando como NAT). Generalmente cuando se especifica *proxy* se está refiriendo a una aplicación de capa 7 del modelo OSI mientras que NAT se refiere a la capa 3 de dicho modelo. En la configuración de un cliente en capa 3 (NAT) solo se debe conocer la puerta de enlace (*gateway*) y este (normalmente denominado *router*) realizará la traslación mientras que en la configuración del cliente en capa 7 se debe enviar los paquetes al servidor *proxy* y este leerá cada paquete para conocer su destino y reenviarlos.

Dado que NAT opera en capa 3 utiliza menos recursos, pero también es menos flexible que en capa 7, ya que solo actúa sobre las direcciones de paquete y no sobre el contenido como hace el *proxy* (en capa 7). Es común en los sistemas GNU/Linux que el NAT se realice con IPtables (*firewall*) mientras que como *proxy* se utilizan diferentes servidores; por ejemplo, para http/https/ftp se utilizan Apache, Squid, Nginx, Varnishm, entre otros.

1.6.1. Apache como *reverse proxy* y con balanceo de carga

Apache es un servidor http muy versátil y eficiente que posee una amplia cantidad de módulos que extienden su funcionalidad, entre ellas la de *proxy*. En este subapartado analizaremos la configuración primero como *reverse proxy* a un servidor interno. Luego estudiaremos cómo balancear la carga a más de un servidor redirigiendo las peticiones en función de diferentes políticas. Apache soporta diferentes módulos [AMod] entre los cuales para *proxy* tenemos mod_proxy (*proxy* multiprotocolo), mod_http (soporte http para *proxy*), mod_cache, mod_proxy_html (reescritura de los enlaces HTML para asegurarse que ellos funcionan fuera del *proxy*), mod_proxy_balancer (balanceo para *reverse proxy*). Comencemos:

1) En primer lugar, instalamos el paquete con los módulos correspondientes:

```
apt-get install libapache2-mod-proxy-html
```

2) Habilitamos los módulos: `a2enmod proxy proxy_http` (verificamos con `apachectl -M`).

3) Creamos un nuevo *host* en */etc/hosts* (p. ej. *172.16.1.1 proxy.nteum.org proxy*).

4) Creamos *virtualhost* en */etc/apache2/sites-avalaible/proxyr.conf*:

```
<VirtualHost proxy.nteum.org:80>
    ErrorLog "/var/log/apache2/proxy-error.log"
    CustomLog "/var/log/apache2/proxy-access.log" common
    ServerName proxy.nteum.org
    ProxyRequests Off
    ProxyPreserveHost On
    ProxyPass / http://ubub.nteum.org/
    ProxyPassReverse / http://ubub.nteum.org/
</VirtualHost>
```

Donde

- *ServerName* debe estar definido en el */etc/hosts*, e indica cómo llamaremos al servidor de entrada,

- *ProxyRequests Off* evita que este sea utilizado como *open proxy*, es decir, que los usuarios puedan ir al *proxy* y de ahí a cualquier otra dirección (por lo cual, en todo lo que hagan constará nuestra IP) y es muy importante dejarlo deshabilitado para evitar problemas de seguridad o incluso legales,

- *ProxyPreserveHost On* permite que el salto del servidor de *proxy* al de *backend* sea transparente para el usuario (si no estuviera habilitada, el usuario se dirigiría a *http://proxy.nteum.org* pero inmediatamente vería cómo la dirección cambia a *http://ubub.nteum.org*, que es el servidor interno –*backend*– y además si el servidor de *backend*) no es visible desde Internet el cliente vería un error,

- *ProxyPass* y *ProxyPassReverse* gestionan el salto y la vuelta del servidor de *frontend* al de *backend*.

5) Habilitamos la configuración del nuevo sitio (`a2ensite proxyr`) y reiniciamos el servicio (`systemctl restart apache2`). También debemos tener en el *backend* un servidor apache2 funcionando con una página diferente a la del *proxy* para verificar que se accede a él cuando ponemos en un navegador *http://proxy.nteum.org*.

Uno de los aspectos interesantes en el servicio web es poder realizar un balanceo de carga de las peticiones sobre diferentes servidores para evitar el efecto "cuello de botella" en el servicio y mejorar el tiempo de respuesta e incrementar el número de peticiones atendidas por unidad de tiempo. Esto se puede hacer mediante *hardware* específico o mediante un *reverse proxy* (*frontend*) que distribuya la carga a una granja interna de servidores (*backend*) de acuerdo a una política determinada. Apache dispone de un módulo adicional al proxy (mod_proxy_balance) que permite realizar el balanceo de carga

sobre un conjunto de servidores web y diferentes módulos para implementar las políticas (lbmethod_byrequests lbmethod_bytraffic lbmethod_bybusyness lbmethod_heartbeat).

Para configurar este módulo deberemos hacer lo siguiente:

1) Cargar los módulos:

`proxy, proxy_balancer proxy_connect proxy_html proxy_http lbmethod_byrequests lbmethod_bytraffic lbmethod_bybusyness lbmethod_heartbeat status`
(para ver los módulos cargados `apachectl -M`)

2) Crear un *virtualhost*: `vi /etc/apache2/sites-available/proxy-bal.conf`

```
<VirtualHost proxy.nteum.org:80>
ProxyRequests off
ServerName proxy.nteum.org
DocumentRoot /var/www
<Proxy balancer://mycluster>
        BalancerMember http://172.16.1.2:80
        BalancerMember http://172.16.1.3:80
        Options Indexes FollowSymlinks Multiviews
        AllowOveride None
        Order Allow, Deny
        Allow from all
        ProxySet lbmethod=bytraffic
        #ProxySet lbmethod=byrequests
</Proxy>
# Habilitar el Balancer Manager
<Location /balancer-manager>
        SetHandler balancer-manager
        Order deny,allow
        Allow from all
</Location>
ProxyPass / balancer-manager !
ProxyPass / balancer://mycluster/
ProxyPassReverse / balancer://mycluster

ProxyPass / http://cloneuno.nteum.org
ProxyPassReverse / http://cloneuno.nteum.org
ProxyPass / http://clonedos.nteum.org
ProxyPassReverse / http://clonedos.nteum.org
</VirtualHost>
```

Deberemos tener en */etc/hosts* las máquinas a las cuales se redirigirán las peticiones (en nuestro caso 2, cloneuno y clonedos):

```
172.16.1.1      proxy.nteum.org proxy
172.16.1.2      cloneuno.nteum.org      cloneuno
172.16.1.3      clonedos.nteum.org      clonedos
```

El *balancer manager* es una herramienta que integra el módulo y que permitirá ver en forma simple las estadísticas simples de la actividad del módulo y algunas modificaciones (simples también). Es por ello que las peticiones a *http://proxy.nteum.org/balancer-manager* no se deberán redirigir y ser atendidas por el *proxy*.

La configuración incluye los siguientes elementos:

- La sección *Proxy balancer*: donde se identifica el balanceador.
- *BalancerMember*: cada una de las IP del *backend*
- *ProxySet lbmethod=byrequests|bytraffic*: la política de balanceo

Lectura recomendada

Para más información se puede consultar la documentación del módulo [AModBal]

3) Habilitar la configuración del nuevo sitio (`a2ensite proxy-bal`) y reiniciar el servicio (`systemctl restart apache2`). También hay que tener en el *backend* los dos servidores (*cloneuno* y *clonedos*) apache2 funcionando con una página diferente a la del *proxy* para verificar que se accede a él cuando ponemos en un navegador *http://proxy.nteum.org* y recargamos la página repetidamente (veremos cómo va cambiando la página en función del servidor del *backend* que la sirve). Para obtener más información se puede ver las estadísticas del balanceador en *http://proxy.nteum.org/balancer-manager* y cambiar los parámetros para adecuarlos a las necesidades de carga (se puede utilizar las herramientas de carga mencionadas para analizar Apache).

1.6.2. Apache como *Forward Proxy* y *Proxy cache*

Para configurar Apache como *Forward Proxy* se deberá hacer lo siguiente:

1) Cargar el módulo (si no están cargados, verificamos con `apachectl -M`):

```
a2enmod proxy proxy_http
```

2) Agregar en la configuración de un sitio:

```
vi /etc/apache2/sites-available/proxy-f.conf
```

```
Listen 172.16.1.1:8080          #Donde escuchará las peticiones
<VirtualHost 172.16.1.1:8080>
ProxyRequests On                #Activa el Proxy Forward
<Proxy *>                       #También puede ser <Directory> ... </Directory>
    Order deny,allow            #Reglas del servicio
    Deny from all
    Allow from 172.16.1.0/24    # clientes habilitados 172.16.1.*
</Proxy>
ProxyBlock marca.es as.es       #bloquea el acceso a estos dos sitios
```

3) Habilitar el sitio (`a2ensite proxy-f`) y reiniciar el servicio (con la orden `systemctl restart apache2`). Posteriormente se deben modificar los clientes para que soliciten las peticiones al servidor *proxy*, por ejemplo, en Firefox debemos ir a *Options->Advanced->Network->Settings* y en *Proxy* poner la IP 172.16.1.1 y el puerto 8080. Para evitar un error común cuando los usuarios no configuran el puerto específico se puede poner `iptables` con una regla que redirija al puerto 8080 todo lo que viene al puerto 80 (o donde se encuentre el *proxy forward)*.

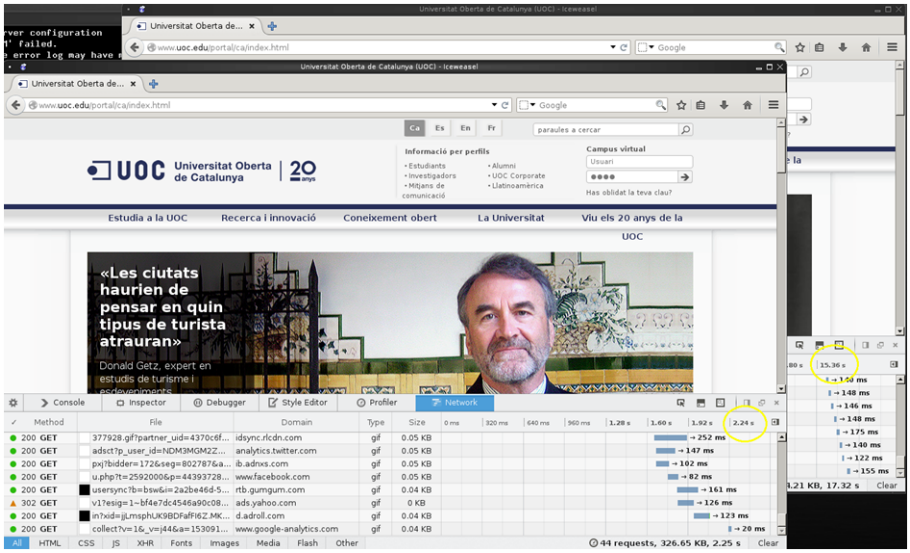
4) Para activar la capacidad de *cache del proxy* se debería agregar al *virtualhost*:

```
<IfModule mod_disk_cache.c>
        CacheRoot "/usr/local/apache/proxy"
        #CacheSize 500
         CacheEnable disk /
        CacheDirLevels 5
        CacheDirLength 3
</IfModule>
```

Después debemos agregar los módulos `a2enmod cache disk_cache`, (o en Apache 2.4 `cache_disk`) y habilitar el sitio si no lo está y reiniciar Apache.

Como se puede observar en la figura 10, que muestra la carga de la URL de la UOC, en los dos círculos amarillos podemos ver el efecto de la caché donde la primera vez tarda 15,36 segundos y en la segunda solo 2,24 segundos, es decir, el 14 % del tiempo. Se debe tener cuidado con los parámetros de caché para Apache 2.2 y 2.4, ya que difieren en su configuración [AModCach].

Figura 10



Existen otros aspectos de caché sobre Apache que no han sido tratados como *File Caching* para acelerar el acceso a los archivos que sirve Apache y *Key-Value Caching* utilizada por SSL y *authentication caching*.

Lectura recomendada

Podéis consultar [DigOcCach] sobre configuraciones y comentarios sobre *caching* y Apache.

1.6.3. Servicio de *proxy*: Squid

Squid es un *proxy caching server* para Web y da soporte a los protocolos HTTP, HTTPS, FTP, entre otros. Éste reduce el ancho de banda y mejora los tiempo de respuesta almacenado en caché y reutilizando las páginas más frecuentes. Squid tiene un extenso conjunto de reglas de control que permiten optimizar el flujo de datos ente el cliente y el servidor aportando seguridad y control y encaminando las peticiones correctamente, permitiendo su control y mejorando la utilización del ancho de banda de la red. Squid tiene diferentes modos de funcionamiento pero como los más importantes podemos mencionar: *Forward Proxy* (es el modo básico sobre el cual se configuran todo lo demás), *Transparent* o *Interception Proxy* (permite incluir un proxy en una red sin que los clientes tengan que configurar nada) y *Reverse Proxy* -o *Accelerator-mode*- (permite ejecutar squid para mejorar la respuesta de una granja de servidores web).

Para instalar Squid como *proxy-cache* (http://www.squid-cache.org/) en Debian hacemos `apt-get install squid3` y editaremos el archivo de configuración */etc/squid3/squid.conf* para realizar una configuración básica. Se debe tener en cuenta que squid es muy potente pero esto se traduce en una configu-

ración que puede ser compleja; como idea simplemente considerar que el archivo de configuración, que está muy bien explicado, tiene aproximadamente 7660 líneas (no obstante si ejecutamos

```
grep -v "^#" /etc/squid3/squid.conf | awk '$1 != "" {print $0}'
```

podremos ver que la configuración básica son unas 40 líneas mientras que el resto son comentarios y opciones comentadas).[squid][squide]

Definir la ACL (access control list) para habilitar la red/ip que deseamos hacer de proxy, en la línea 1056 agregar: `acl lan src 192.168.1.0/24`

Permitir el acceso, en la línea 1220 (aprox.) agregar: `http_access allow lan`

Cambiar el puerto de acceso (linea 1622), por defecto está `http_port 3128` y se puede dejar que es el estándar para squid o poner el que se prefiera (por ejemplo 8080). Con esto tendríamos una configuración mínima pero también se puede agregar la siguiente configuración:

Definimos el nombre visible, línea 5273 (aprox.): `visible_hostname remix.world`

Modificamos visibilidad de la IP, línea 7359 (aprox.): `forwarded_for off`

Finalmente reiniciamos el servidor: `service squid3 restart`
Nos dará un mensaje similar a:

[ok] Restarting Squid HTTP Proxy 3.x: squid3[....] Waiting.....done. (tardará unos segundos...)

Con el servidor en marcha podemos configurar los navegadores para que utilicen el proxy, esto es por ejemplo en IceWeasel/Firefox en el apartado de *Preferences/Options->Advanced->Network->Proxy* e introducir los datos de nuestro servidor (ip o nombre.dominio y puerto). Sobre Chrome p. ej. en Windows, se debe ir a *Settings->Advanced->Change proxy setting->Connections->Lan Settings* e introducir los datos de nuestro servidor (ip o nombre.dominio y puerto).

Para bloquear dominios debemos agregar lo siguiente:

1) En la línea 1058 (aprox.) pero siempre antes del http_access allow lan:

```
acl block_tld dstdomain .tv .xxx
http_access deny block_tld
deny_info TCP_RESET block_tld
```

Con esto bloqueamos los dominios *.tv* y *.xxx* y con `TCP_RESET` se reseteará la conexión y el cliente no sabrá qué ha pasado.

2) Para controlar palabras podemos hacer (siempre antes de *http_access allow lan*):

```
acl bad_keywords url_regex "/etc/squid3/denegado.txt"
http_access deny bad_keywords
```

El archivo */etc/squid3/denegado.txt* tendrá por ejemplo:

as
marca
lavanguardia

Esto indicará que la URL que tenga estas palabras/dominios no podrán ser accedidas.

Otra de las opciones que permite Squid es actuar como *Reverse Proxy* que es un tipo de *Proxy* donde los datos se recuperan de un/unos servidor/es y de devuelven a los clientes como si se originaran en el *proxy* quedando los servidores ocultos a los clientes como se muestra en la Wikipedia*. Esto permite hacer políticas de balanceo de carga y que las peticiones estén en un único dominio a pesar que internamente pueden estar distribuidas en diversos servidores. Para su configuración debemos hacer:

***http://upload.wikimedia.org /wikipedia/commons/6/67 /Reverse_proxy_h2g2bob.svg**

Especificar la dirección del servidor interno, en la línea 1626 modificar:
`http_port 80 defaultsite=192.168.1.33`
Agregar cache_peer, en la línea 2470 (aprox.) agregar:
`cache_peer 192.168.1.33 parent 80 0 no-query originserver`
Cambiar la acl para permitir cualquier conexión, en la línea 1174 (aprox.) modificar `http_access allow all`
Finalmente reiniciamos el servidor: `service squid3 restart`

Cuando nos conectemos a la IP/dominio del servidor *Proxy* en realidad veremos las páginas enviadas por el servidor 192.168.1.33. Como prueba de concepto (si queremos hacer la prueba con una única máquina) podemos poner que en lugar del servidor 192.168.1.33 poner un servidor externo (p. ej., debian.org o su IP) y cuando pongamos como URL la de nuestro dominio visualizaremos la página de debian.org.

Otras de las configuraciones ampliamente utilizadas es como *interception proxy* (o *transparent proxy*) que intercepta la comunicación normal a la capa de red sin necesidad de configuraciones específicas en el cliente y por lo cual no sabrán que están detrás de un *proxy*. Generalmente un *transparent proxy* esta normalmente localizado entre el cliente e Internet con el *proxy* haciendo las funciones de *router* o *gateway*. Es habitual en las instituciones que desean filtrar algún tráfico de sus usuarios, ISP para hacer caché y ahorrar ancho de banda o países que controlan el acceso a determinados sitios por parte de sus ciudadanos. Para implementarlo sobre una institución lo habitual es disponer de una máquina con dos interfaces de red, una conectada a la red interna y otra hacia

la red externa. Los pasos para su configuración serán básicamente los descritos en http://wiki.squid-cache.org/ConfigExamples/Intercept/LinuxDnat:

Sobre squid.conf:

Modificar la `acl/http_access` sobre las IP, IP/Mask permitidas como en el primer caso.

Configurar el puerto/modo como `http_port 3129 transparent` o en Squid 3.1+ se deberá utilizar http_port 3129 intercept para interceptar paquetes DNAT.

Sobre /etc/sysctl.conf modificar:

Permitir el packet forwarding: net.ipv4.ip_forward = 1

Controlar el source route verification: net.ipv4.conf.default.rp_filter = 0

: No aceptar source routing: net.ipv4.conf.default.accept_source_route = 0

Considerando que la IP de Proxy está en la variable SQUIDIP y el puerto esta en SQUIDPORT incluir las siguientes reglas de DNAT:

```
iptables -t nat -A PREROUTING -s $SQUIDIP -p tcp --dport 80 -j ACCEPT
iptables -t nat -A PREROUTING -p tcp --dport 80 -j DNAT --to-destination
    $SQUIDIP:$SQUIDPORT
iptables -t nat -A POSTROUTING -j MASQUERADE
iptables -t mangle -A PREROUTING -p tcp --dport $SQUIDPORT -j DROP
```

1.6.4. Proxy SOCKS

SOCKS (abreviación de SOCKetS) es un protocolo de Internet (en el modelo OSI estaría en una capa intermedia entre la de aplicación y la de transporte) que permite a las aplicaciones en modo cliente-servidor atravesar en forma transparente un *firewall* de red. Los clientes que hay detrás de un *firewall*, los cuales necesitan acceder a los servidores del exterior, pueden conectarse en su lugar a un servidor *proxy* SOCKS. Tal servidor *proxy* controla qué cliente puede acceder al servidor externo y pasa la petición al servidor. SOCKS puede ser usado también de la forma contraria, permitiendo a los clientes de fuera del *firewall* (clientes externos) conectarse a los servidores de dentro del *firewall* (servidores internos).[socks][psocks]

Se debe tener en cuenta que SOCKS solo sirve en modo cliente/servidor por o cual un usuario debe tener instalado un cliente SOCKS, ya sea en la aplicación (como Firefox, Chrome) o dentro de la pila TCP/IP desde donde el software del cliente redirige los paquetes en un túnel SOCKS. El procedimiento habitual comienza cuando el cliente SOCKS (p. ej., interno en una red privada) inicia una conexión a un servidor SOCKS (el protocolo SOCKS permite la autentificación y el registro de las solicitudes de conexión) y este (servidor SOCKS) actuará como cliente IP para la solicitud de conexión (del cliente interno en su nombre) lo cual significa que el servidor externo solo será consciente de las peticiones del servidor SOCKS (por lo cual actuará como *proxy forwarding*).

La pregunta habitual es si SOCKS es diferente a resolver el problema de acceso externo mediante NAT. La respuesta es: sí, es diferente, ya que los paquetes en NAT solo modifican las direcciones (p. ej., cambian las IP privadas por las pública del *router* como ocurre en un *router* ADSL) y el servidor recibe/contesta las peticiones. La sesión IP se establece directamente desde el cliente al servidor y el router/FW solo modifica/filtra el paquete pero no hay autenticación ni inspección del paquete/aplicación/datos (se podría hacer pero es complejo). La ventajas de SOCKS radican en que provee autentificación para protocolos que no lo permiten, puede traspasar el *routing* por defecto de una red interna, si bien HTTP y Telnet soportan autentificación por firewall (p. ej., utilizando *Authenticated Proxy* on a Cisco *firewall*) los protocolos encriptados nunca pueden ser autentificados por un FW en cambio SOCKS si puede hacerlo. No obstante existen desventajas ya que el cliente debe tener interface a SOCKS, el SO cliente debe tener interface a SOCKS (para interceptar el tráfico y reenviarlo al SOCKS *proxy*) y necesitaremos un servidor SOCKS específico para este fin.

Un caso de uso habitual es si consideramos que estamos en un punto de conexión inalámbrica abierta (p. ej., Wifi) y no se desea enviar datos de navegación sobre texto plano o se quiere acceder a una página web filtrada por router/FW perimetral. Una solución muy simple es utilizando SSH (que incluye un servidor SOCKS) que puede cifrar todo el tráfico de navegación por la web y redireccionarlo a través de un equipo de confianza cuando se está en algún otro punto de la red. Para ello deberemos disponer de un servidor SSH para que actúe como representante en un ordenador remoto (que le permita conectarse a él a través de SSH) y un cliente de SSH en el equipo que está utilizando. Lo que se hará con un proxy es la creación de un *middle-person* entre el usuario e Internet. El navegador hará las solicitudes de páginas web al servidor *proxy*, que controla la solicitud y obtiene la página desde internet y las devuelve al cliente. El sitio web en realidad piensa que la solicitud proviene del servidor *proxy*, no del equipo que la ha originado ocultando las dirección IP de origen. Además, la conexión entre el ordenador y el *proxy* que pasa a través de SSH es cifrada y esto evita que alguien pueda obtener los paquetes desde la Wifi (*sniffers* de wifi) en el sitio de la conexión.

Para su configuración desde donde nos conectamos debemos tenemos acceso a un servidor SSH y sobre el que crearemos un túnel que pasará el tráfico web entre nuestra máquina local y el proxy SOCKS sobre SSH. Para ello ejecutamos sobre nuestra máquina `ssh -ND 9999 login@remote-server.org` donde deberemos reemplazar login@remote-server.org con el usuario y nombre o IP del servidor remoto. Lo que está haciendo este comando es un *port forwarding* a través del puerto 9999 (puede ser cualquier otro pero conviene que sea superior a 1024 para evitar que solo lo pueda hacer root) y la conexión se reenvía a través de una canal seguro donde el protocolo de aplicación se utiliza para determinar dónde conectar desde la máquina remota. Actualmente OpenSSH soporta los protocolos SOCKS4-5 y por ello actuará como un servidor SOCKS. A continuación se solicitará el passwd y una vez autenticado no pasará NADA

(el -N indica que no abra un prompt interactivo pero continuará funcionando). Si por el *firewall* solo podemos salir por e puerto 443 por ejemplo deberíamos configurar el ssh server para escuchar por el puerto 443 y en su lugar ejecutar `ssh -ND 9999 login@remote-server.org -p 443`. Ahora es necesario configurar el cliente para conectarse al proxy, Por ejemplo Firefox: *Options ->Advanced ->Network ->Connection* y seleccionar SOCKS, como nombre del servidor *localhost* (o su nombre real si tiene) y el puerto (9999) y guardar los ajustes y verificar que podemos navegar sin problemas. Se puede utilizar el plugin FoxyProxy (https://addons.mozilla.org/es/firefox/addon/foxyproxy-standard/) para Firefox que permite cambiar entre el proxy y la conexión directa en función del sitio o de un control. Como medida adicional (de anonimato) se puede configurar el servidor *proxy* para resolver peticiones DNS en lugar del método habitual haciendo en Mozilla Firefox poniendo como URL: `about:config` y modificando `network.proxy.socks_remote_dns=true`. También para conexiones lentas se puede utilizar la opción -C de ssh para utilizar la compresión de SSH por gzip. En Thunderbird u otros clientes la configuración es similar.

Si el túnel deja de funcionar (suele ocurrir en redes muy ocupadas) se puede utilizar el paquete `autossh` en lugar del `ssh` para establecer la conexión que se encargará de mantener el túnel abierto reiniciando automáticamente la conexión. Otro paquete interesante es `tsocks` (http://tsocks.sourceforge.net/) que se puede utilizar cuando el cliente que deseamos utilizar no soporta el protocolo SOCKS. `tsocks` monitoriza la llamada de inicio de sesión de una aplicación (*connect*) y redirecciona la comunicación hacia el *server* SOCKS sin que la aplicación tenga ninguna información. Para ello se debe instalar tsocks y configurar el proxy SOCKS que deberá utilizar en el fichero */etc/tsocks.conf* indicándole los valores (p. ej., server = 127.0.0.1, server_type = 5, server_port = 9999). Luego bastará con llamar a la aplicación con `tsocks aplicación` o simplemente el comando que abrirá una nueva shell redirigida al proxy y por lo cual todo lo que se ejecute allí será enviado al proxy SOCKS.

1.7. Seguridad en Apache

Si bien el tema de la seguridad será tratado posteriormente, en este subapartado, y continuando con la función de Apache como proveedor de diferentes servicios, haremos un avance sobre la seguridad de las aplicaciones web, que es conocido como *Web Application Firewalls* (WAF) por la importancia que tiene y las necesidades cada vez mayores de evitar intrusiones y reducir los riesgos. Es importante indicar que además comienzan a surgir estándares, como por ejemplo PCI DSS - Payment Card Industry Data Security Standard v.1.1, en los cuales las revisiones regulares del código y el uso de un WAF es uno de los criterios que tiene que cumplir el entorno de producción.

Las aplicaciones web (tiendas en línea, portales corporativos/privados/personales, etc.) hoy en día son uno de los puntos de ataque habituales por la

información que gestionan. Es frecuente ver en los medios de comunicación incidentes de acceso a la información (o divulgación de información privada) y que por lo general comienza con la intrusión a través de un portal web utilizando la explotación de los puntos débiles en la aplicación web y que no se detectan (o no se detectan con suficiente precisión) con los sistemas tradicionales de seguridad (cortafuegos o IDS/IPS). Es por ello que el administrador debe disponer de herramientas específicas para conocer las vulnerabilidades de sus servidores, así como detectar y prevenir este tipo de ataques.

Dado su interés, existen gran cantidad de aplicaciones (complejas y de un coste elevado) que permiten las funciones de WAF, pero en el ámbito de *open source* se pueden encontrar dos proyectos que por su calidad y potencialidad son muy interesantes: OWASP (*Open Web Application Security Project)* y ModSecurity.

OWASP [Owasp] es un proyecto que tiene por objetivo desarrollar herramientas y buenas prácticas para apoyar a los desarrolladores, administradores de proyectos y analistas de seguridad en el desarrollo y funcionamiento de las aplicaciones web seguras. Es importante destacar que OWASP no solamente es un conjunto de herramientas (`WebGoat` [WGoat], `Zed Attack Proxy` [ZAP], `JBroFuzz` [JBF], o `LabRat` [LabRat]) sino todo un conjunto de proyectos, guías y distribuciones vinculadas al tema de WAF [OProj]. Hay que destacar que OWASP es en sí una fundación sin ánimo de lucro que gestiona los proyectos e infraestructura OWASP y está formada por empresas, instituciones educativas y particulares constituyendo una comunidad de seguridad informática que trabaja para crear metodologías, documentación, herramientas y tecnologías bajo las premisas del *open source*.

ModSecurity [MSec] es un módulo de código abierto y multiplataforma que actúa como *firewall* de aplicaciones Web (WAF). Para muchos administradores es conocida como la "navaja suiza" de los WAF, ya que permite aumentar la visibilidad del tráfico HTTP(S), proporciona un potente lenguaje de reglas y además una API para implementar protecciones avanzadas. ModSecurity incluye reglas desde OWASP (ModSecurity Core Rule Set) que le permite configurar reglas de detección para ataques en general (como *HTTP Protocol Protection, Real-time Blacklist Lookups, HTTP Denial of Service Protections, Generic Web Attack Protection, Error Detection and Hiding*). También permite incluir reglas de otros proveedores comerciales (con coste económico) como Trustwave SpiderLabs que están basadas sobre la inteligencia recogida de ataques, pruebas de penetración y de casos reales.

ModSecurity permite la monitorización, registro y control de acceso en tiempo real de aplicaciones web. Según su desarrollador, puede ser considerado como un "facilitador", ya que no indica qué reglas ha de poner, sino qué posibilidades tiene el administrador entre las funciones disponibles de acuerdo al camino que se desee seguir. Una lista de los escenarios de uso más importantes sería:

- **Monitorización y control de acceso.** Monitorización de seguridad de las aplicaciones en tiempo real y control de acceso permitiendo analizar el flujo de tráfico HTTP, en tiempo real, junto con la capacidad para inspeccionarlo y almacenarlo para un posterior seguimiento y correlación de eventos a través del tiempo.

- ***Virtual Patching***. Es un concepto de mitigación de las vulnerabilidades en una capa separada, donde se llega a solucionar problemas en aplicaciones sin tener que tocar las propias aplicaciones. ModSecurity permite realizar parches virtuales debido a sus capacidades de bloqueo fiables y al lenguaje de reglas flexible adaptables a cualquier necesidad.

- **Registro completo de tráfico HTTP.** Tradicionalmente, los servidores web no tienen registros, por motivos de seguridad, de los inicios de sesión. ModSecurity permite registrar cualquier cosa que necesite, incluyendo datos de la transacción en bruto, que es esencial para el análisis forense permitiendo escoger qué transacciones se registran, qué partes de una transacción se registran o qué partes no son necesarias.

- **Evaluación de la seguridad pasiva continua.** Funciona como sistema de alerta temprana que permite detectar rastros de anomalías y fallos de seguridad antes de que sean explotados.

- **Endurecimiento de aplicaciones Web.** Reduce la "superficie de ataque" ayudando a la ejecución de numerosas restricciones similares, directamente o a través de la colaboración con otros módulos de Apache (por ejemplo, solucionar los problemas de gestión de sesiones o las vulnerabilidades de *cross-site request forgery*). Entre los principios de diseño se pueden enumerar la flexibilidad, la pasividad (no interacción si no está indicado), la previsibilidad y la calidad. Admite dos opciones de funcionamiento: incrustado (como módulo) o como *reverse proxy*. Como incrustado se adapta a la infraestructura ya desplegada y escala con ella, no introduce nuevos puntos de fallos, pero tiene como reto que los recursos son compartidos con el servidor web. Como *reverse proxy* juega el papel de *router HTTP* colocado entre los servidores web y sus clientes. En este caso, ModSecurity protegerá todo el tráfico globalmente independientemente de adónde vaya, permitiendo separar la capa seguridad y aislando totalmente a los sistemas que está protegiendo. Además, al contar con recursos dedicados a ello específicamente permitirá tener reglas más complejas y adecuadas al global de la información que recibe. La principal desventaja de este enfoque es que agrega un nuevo punto de fallo y se deberá tener en cuenta con una configuración de alta disponibilidad de dos o más servidores *proxy* inversos.

La instalación y configuración (básica) de ModSecurity ha de seguir los siguientes pasos:

1) Instalamos Modsecurity: `apt-get install libapache2-modsecurity` (hay que verificar con `apachectl -M` que aparece el módulo *security2_module (shared)*).

2) Instalaremos también PHP y MySQL para hacer las pruebas (hay que recordar el *passwd* introducido en el usuario *root* de MySQL):

```
apt-get install php5 php5-cgi libapache2-mod-php5 php5-common
php-pear mysql-server php5-mysql
```

3) Activamos la configuración básica:

```
cd /etc/modsecurity

mv modsecurity.conf-recommened modsecurity.conf
```

4) Editamos este archivo `vi /etc/modsecurity/modsecurity.conf` y cambiamos las siguientes líneas:

```
SecRuleEngine On                #activación de ModSecurity
SecRequestBodyAccess On #habilita a leer la respuesta -deshabilitar si no es
       #necesario ya que reduce prestaciones y gasta espació de log
SecRequestBodyLimit 13107200                    #Ajustar de acuerdo a las necesidades.
SecRequestBodyNoFilesLimit 131072          #ya que implican recursos
SecRequestBodyInMemoryLimit 131072
```

5) Reiniciamos Apache (`apachectl restart` o también se puede verificar la configuración primero con `apachectl configtest` y luego reiniciar).

Como prueba de concepto detectaremos una inyección y una introducción de Spam. Para el primer caso crearemos una página con una consulta simple a Mysql (véase [DigOcMS]):

```
vi /var/www/html/login.php
```

```
<html><body>
<?php
    if(isset($_POST['login']))
    {
        $username = $_POST['username'];
        $password = $_POST['password'];
        $con = mysqli_connect('localhost','root','psswd_de_la_BD','test');
        $result = mysqli_query($con, "SELECT * FROM `users` WHERE username='$username'
        AND password='$password'");
        if(mysqli_num_rows($result) == 0)
            echo 'Usuario o Passwd INCORRECTOS';
        else
            echo '<h1>Logged in</h1><p> For Your Eyes Only!!</p>';
    }
    else
    {
?>
        <form action="" method="post">
            Usuario: <input type="text" name="username"/><br />
            Contraseña: <input type="password" name="password"/><br />
            <input type="submit" name="login" value="Login"/>
        </form>
<?php
    }
?>
</body></html>
```

A continuación creamos la base de datos con `mysql -u root -p` y después de introducir la contraseña ejecutamos:

```
create database test;
connect test;
create table users(username VARCHAR(100),password VARCHAR(100));
insert into users values('pirulo','123456');
quit;
```

Si ahora nos conectamos al servidor http://srv.nteum.org/login.php nos saldrá la página y podremos hacer el proceso de *Login* (usuario *pirulo* y *passwd* 123456) mostrándonos el texto de ***Logged in For Your Eyes Only!!***

Si ahora hacemos una inyección introduciendo como usuario `' or true --` (hay que tener en cuenta que debe haber un espacio después de `--` sino no funcionará) podremos ver que la inyección funciona y que nos hemos saltado el procedimiento de *login*.

Para detectar este tipo de inyección activaremos las reglas en el directorio */usr/share/modsecurity-crs/activated_rules/* haciendo un enlace desde las múltiples reglas que tiene (véase la documentación en [ModSRef]) haciendo:

```
cd /usr/share/modsecurity-crs/activated_rules/
ln -s ../base_rules/modsecurity_crs_41_sql_injection_attacks.conf .
```

Agregamos además en */etc/apache2/mods-enabled/security2.conf* dentro del bloque *<IfModule security2_module>*

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

Reiniciamos Apache (`apachectl restart`) y volvemos a cargar la página y realizamos la inyección, pero ahora veremos *Forbidden. You don't have permission to access /login.php on this server* y en */var/log/apache2/modsec_audit.log* algo como:

```
Message: Access denied with code 403 (phase 2). Pattern match "([\\~\\!\\@\\#\\$\\%\\^\\&\\*\\(\\)\\
-\\+\\=\\{\\}\\[\\]\\|\\:\\;\"\\'\\xc2\xb4\\xe2\x80\x99\\xe2\x80\x98\\`\\<\\>].*?){8,}" at REQUES
T_COOKIES:__ar_v4. [file "/usr/share/modsecurity-crs/activated_rules/
modsecurity_crs_41_sql_injection_attacks.conf"]
[line "157"] [id "981172"] [rev "2"] [msg "Restricted SQL Character Anomaly Detection Alert - Total #
of special characters exceeded"] [data "Matched Data: : found within REQUEST_COOK
IES:__ar_v4: 5SKFSKF2FJD2TP2KUCUHQ4:20160626:19|GTWKLEK3HFHKPMLGC4VUAQ:20160626:19|5Q2SJSWEVZDJTDUCO
M4QNW:20160626:19"] [ver "OWASP_CRS/2.2.9"] [maturity "9"] [accuracy "8"] [tag "OWASP_CRS/WEB_ATTACK
/SQL_INJECTION"]
```

Donde detecta claramente la inyección y provee toda la información necesaria. Para evitar que introduzcan *Spam* (véase información adicional en [DigOcModS]) creamos una página web y una caja de texto en php:

```
vi /var/www/html/test.php

<html>
    <body>
        <?php
            if(isset($_POST['data']))
                echo $_POST['data'];
            else
            {
        ?>
                <form method="post" action="">
                        Introducir algún texto:<textarea name="data"></textarea>
                        <input type="submit"/>
                </form>
        <?php
            }
        ?>
    </body>
</html>
```

Esto permitirá introducir cualquier texto sin control (en este caso solo se visualiza el texto introducido, pero podría almacenarse con el consiguiente espacio perdido si es *spam*). Para ello creamos el archivo

```
vi /etc/modsecurity/custom.conf

SecRule REQUEST_FILENAME "form.php" "id:'400001',chain,deny,log,msg:'Spam detected'"
SecRule REQUEST_METHOD "POST" chain
SecRule REQUEST_BODY "@rx (?i:(casino|lottery|rolex))"
```

Es importante verificar que tenemos en */etc/modsecurity/modsecurity.conf* el parámetro *SecRequestBodyAccess On*.

Con esto ya podemos llamar a http://srv.nteum.org/test.php e introducir cualquier palabra, que las mostrará a continuación. Pero si en el texto introducimos una de las palabras filtradas (*casino|lottery|rolex*), por ejemplo *alskjhfalksdrolexhggakjsfagsdav*, veremos que obtenemos un *Forbidden* y en el *log* el mensaje que hemos indicado (msg "Spam detected"):

```
Message: Access denied with code 403 (phase 2). Pattern match "(?i:(casino|lottery|rolex))" at
REQUEST_BODY. [file "/etc/modsecurity/custom.conf"] [line "1"] [id "400001"] [msg "Spam detected"]
```

1.8. Servidor de wiki

Lectura recomendada

Dada la potencialidad de ModSecurity es recomiendable ampliar este subapartado con [ModSRef] [ModSBook] [ASec] para controlar otros tipos de ataques.

Un (o una) **wiki** (del hawaiano *wiki wiki*, "rápido") es un sitio web colaborativo que puede ser editado por varios usuarios que pueden crear, editar, borrar o modificar el contenido de una página web, de forma interactiva, fácil y rápida; dichas facilidades hacen de una wiki una herramienta eficaz para la escritura colaborativa. La tecnología wiki permite que páginas web alojadas en un servidor público (las páginas wiki) sean escritas de forma colaborativa a través de un navegador, utilizando una notación sencilla para dar formato, crear enlaces, etc. y conservando un historial de cambios que permite recuperar de manera sencilla cualquier estado anterior de la página. Cuando alguien edita una página wiki, sus cambios aparecen inmediatamente en la web, sin pasar por ningún tipo de revisión previa. Wiki también se puede referir a una co-

lección de páginas hipertexto, que cualquier persona puede visitar y editar (definición de Wikipedia). Debian tiene su wiki en http://wiki.debian.org/ o también Apache en http://wiki.apache.org/general/ y ambas están basadas en **MoinMoin**. MoinMoin es una *Python WikiClone* que permite inicializar rápidamente su propia wiki y solo se necesitan un servidor de web y el lenguaje Python instalado. En la web de MoinMoin se encuentran las instrucciones detalladas para instalar MoinMoin, pero hay dos formas principales de hacerlo: instalación rápida e instalación de servidor.

Enlace de interés

Para saber más sobre MoinMoin podéis visitar su página web en: http://moinmo.in.
En concreto encontraréis las instrucciones detalladas para instalar MoinMoin en: http://master19.moinmo.in/InstallDocs.

1.8.1. Instalación rápida

1) Descargar el paquete desde http://moinmo.in/MoinMoinDownload que será, por ejemplo, para la versión 1.9 `moin-1.9.8.tar.gz`. Si se quiere verificar la integridad del paquete, se puede hacer `md5sum moin-x.x.x.tar.gz` y verificar que coincidan el *hash* generado con el que existe en la página de descarga.

2) Desempaquetar MoinMoin `tar xvzf moin-x.x.x.tar.gz`. Esto creará un directorio `moin-x.x.x` en el directorio actual con los archivos en su interior.

3) Dado que MoinMoin está escrita en Python, es necesario utilizar el intérprete de Python:

```
cd moin-x.x.x; python wikiserver.py
```

Esta orden mostrará por pantalla los mensajes de ejecución del servidor. Entre esta información se mostrará la dirección IP sobre la cual está corriendo el servidor, que podrá ser algo como `http://127.0.0.1:8080`. Esta opción usa un servidor web interno, será accesible desde `http://localhost:8080/` y funcionará hasta que se presione `Ctrl-C` en el terminal.

1.8.2. Instalación de servidor

MoinMoin es una aplicación WSGI (*Web Server Gateway Interface*), por lo tanto, el mejor entorno para ejecutar Moin Moin es uno que permita WSGI como, por ejemplo, Apache con mod_wsgi. En Debian podemos instalar el módulo instalando `apt-get install libapache2-mod-wsgi`.

Enlace de interés

Las instrucciones para instalar WSGI para Apache y configurar MoinMoin en este caso se pueden encontrar en la siguiente dirección: http://moinmo.in/HowTo/ApacheWithModWSGI.

Instalación de MoinMoin

Para instalar MoinMoin descargar la última versión y descompactar el archivo (p. ej., `tar xvzf moin-1.9.7.tar.gz`) y luego hacer una `cd moin-1.9.7/` y a continuación ejecutar:

```
python setup.py install --force --record=install.log --prefix='/usr/local'
```

Para hacer un test simple:
`cd /usr/local/share/moin/server`
`python test.wsgi`
En el navegador introducir como URL localhost:8000 y veremos la página de test de WSGI.

Copiar la configuración: `cd /usr/local/share/moin`
`cp server/moin.wsgi .`
`cp config/wikiconfig.py .`

Agregar un archivo en */etc/apache2/conf.d/moin.conf* con el siguiente contenido:

```
  # MoinMoin WSGI configuration
# you will invoke your moin wiki at the root url, like http://servername/FrontPage:
  WSGIScriptAlias / /usr/local/share/moin/moin.wsgi
  # create some wsgi daemons - use these parameters for a simple setup
   WSGIDaemonProcess moin user=www-data group=www-data processes=5 \
threads=10 maximum-requests=1000 umask=0007
  # use the daemons we defined above to process requests!
  WSGIProcessGroup moin
```

Modificar el archivo */usr/local/share/moin/moin.wsgi* agregando al final del párrafo a2: sys.path.insert(0, '/usr/local/share/moin')

Modificar los permisos de los directorios/páginas: `cd /usr/local/share; chown -R www-data:www-data moin; chmod -R ug+rwx moin; chmod -R o-rwx moin`
Verificar que tengamos un site por defecto y que dentro tengamos también un directorio que permita la ejecución de los *scripts*. Así por ejemplo, ejecutamos `vi /etc/apache2/sites-available/000-default.conf` y escribimos dentro de <*VirtualHost*>:

```
<Directory /usr/local/share/moin>
Require all granted
</Directory>
```

Luego hacemos `a2ensite 000-default` y reiniciamos apache (con la orden `service apache2 restart`).

Si nos conectamos a la URL localhost tendremos la página inicial de MoinMoin. Para configurar el nombre de la wiki y el usuario administrador podemos editar el archivo */usr/local/share/moin/wikiconfig.py*, quitamos el comentario de `page_front_page = u"FrontPage"` e indicamos el nombre del administrador, por ejemplo, `superuser = [u"WikiAdmin", ]` reiniciando apache nuevamente. Para configurar el lenguaje, debemos entrar como administrador (WikiAdmin) (si no tenemos un usuario seleccionamos *login*, seleccionamos 'you can create one now' y lo creamos -debe coincidir con el que introducimos como superuser-). Luego podremos configurar el idioma desde

`http://localhost/LanguageSetup?action=language_setup` y a partir de esta acción ya podremos comenzar a crear nuestra primera wiki (podéis acceder a información adicional en http://moinmo.in/HowTo y particularmente en https://moinmo.in/HowTo/Debian8 tienen más información sobre MoinMoin y Debian).

Para configurar múltiples wikis, primero debéis copiar `config/wikifarm/*` de la distribución en el directorio `moin/config/`. Luego se deben seguir las instrucciones anteriores para cada una de las wikis de la colección (*farm*) teniendo en cuenta que:

1) es necesario tener `data_dir` y `data_underlay_dir` separados para cada wiki,

2) si buscáis que compartan alguna configuración, entonces esta debe estar en `farmconfig.py` y las específicas deben estar en `mywiki.py`.

1.9. Gestión de copias de respaldo (*backups*)

Las copias de seguridad o copia de respaldo (*backup*) se refiere a una copia de los datos originales que se realiza con el fin de disponer de un medio de recuperarlos en caso de pérdida total o parcial debido a fallos en los dispositivos físicos, borrados por accidente, ataques informáticos, infectados por virus u otras causas que hacen que la información no existe o no es la deseada. El proceso de copia de seguridad se complementa con un proceso de restauración de los datos (*restore*) que puede ser total o parcial/selectivo, que permite devolver el sistema informático al punto en en cual fueron los datos almacenados. Esto puede significar la pérdida de información entre el momento que se realiza la copia de seguridad y el momento que se detecta que los datos no existen o están corrompidos por lo cual la política de planificación de copias de seguridad debe ser una de las actividades importantes en todo administrador de sistemas.

Se debe tener en cuenta que la pérdida de datos es habitual (y no por ello sin consecuencia y en algunos casos fatales) ya que de acuerdo a estadísticas recientes más del 60 % de los usuarios de Internet declaran haber sufrido una seria pérdida de datos en alguna ocasión y según un estudio de la Universidad de Texas, solo el 6 % de empresas con pérdida catastrófica de datos sobrevivirá, frente a un 43 % que nunca reabrirá su negocio y un 51 % que tendrá que cerrar en un plazo de 2 años. Para reafirmar más aún la necesidad de copias de seguridad, aquellos sistemas que contengan datos de carácter personal y estén sujetos a la legislación del país (p. ej., en España la LOPD -Ley Orgánica de Protección de Datos-) entre una de las obligaciones que deben cumplir las empresas/instituciones/individuos, es tener copias de seguridad para preservar los datos que tienen almacenados y que están sujetos a esta normativa.

1.9.1. Programas habituales de copias de respaldo

Existen diversas opciones para hacer copias de respaldo con diferentes objetivos, prestaciones e interfaces en todas las distribuciones GNU/Linux (p. ej., amanda, bareos, backintime, bacula, backup2l, backuppc, bup, chiark, dejadup, dirvish, flexbackup, lucky, rdiff, vbackup entre otras). Una de las más usadas es **Bacula*** (http://blog.bacula.org/) o Bareos (https://www.bareos.org/en/, bifurcación de Bacula con mejoras), que es una colección de herramientas para realizar copias de seguridad en una red. Bacula se basa en una arquitectura cliente/servidor que resulta muy eficaz y fácil de manejar, ya que presenta un conjunto muy amplio de características y es eficiente tanto para un conjunto de ordenadores personales como para grandes instalaciones. El paquete está formado por diferentes componentes, entre los más importantes se pueden encontrar:

***http://www.bacula.org**

- **Bacula-director**, *daemon* que gestiona la lógica de los procesos de *backup*;
- **Bacula-storage**, *daemon* encargado de manejar los dispositivos de almacenamiento;
- **Bacula-file**, *daemon* por medio del cual Bacula obtiene los ficheros que necesita para hacer el respaldo y que se deberán instalar en las máquinas fuente de los ficheros a respaldar, y
- **Bacula-console**, que permite interactuar con el servicio de *backup*.

Bacula soporta discos duros, cintas, DVD, USB y también diferentes bases de datos (MySQL, PostgreSQL y SQLite) pero como contrapartida, es necesario disponer de todos los paquetes instalados y su instalación y puesta a punto pueden resultar complejas.

Otro paquete interesante es **BackupPC***, que permite hacer copias de seguridad de disco a disco con una interfaz basada en la web. El servidor se ejecuta en cualquier sistema Gnu/Linux y admite diferentes protocolos para que los clientes puedan escoger la forma de conectarse al servidor. Este programa no es adecuado como sistema de copia de seguridad de imágenes de disco o particiones ya que no soporta copias de seguridad a nivel de bloque de disco; sin embargo, es muy simple de configurar y la posible intrusión sobre la red de ordenadores en la cual se desea hacer el respaldo es mínima. Este servidor incorpora un cliente *Server Message Block* (SMB) que se puede utilizar para hacer copia de seguridad de recursos compartidos de red de equipos que ejecutan Windows.

***http://backuppc.sourceforge.net/info.html**

Su instalación es sumamente sencilla ejecutanto, en la distribución Debian, `apt-get install backuppc`. Nos indicará que seleccionemos el servidor web (apache2), y también el usuario y *passwd* que ha creado, no obstante este *passwd*/usuario se pueden cambiar con `htpasswd /etc/backuppc/htpasswd backuppc`. Luego en nuestro navegador ponemos http://localhost/backuppc y con el usuario/*passwd* accederemos a la página principal de la aplicación donde nos dará el estado del servidor y las opciones.

Hay diversas formas de configurar los clientes para hacer las copias de respaldo y dependerá del método para hacerlo y del sistema operativo. Para ello consultar el apartado de documentación como se configurará cada una de las transferencias (http://backuppc.sourceforge.net/faq/BackupPC.html).

En el caso de un sistema (remoto) Gnu/Linux, desde la interfaz de administración editar la configuración del mismo (*host* y *xfer*) y confirmar que se ha definido como método `rsync` y el directorio directorio a realizar la copia. Como los respaldos se realizan a través de `rsync` en combinación con `ssh` es necesario que el usuario *backuppc* del servidor pueda acceder como root sin clave a la máquina remota. Para ello adoptar la entidad del usuario *backuppc* (`su - backuppc`) y generar las llaves (sin *passwd*) `ssh-keygen -t dsa` y luego utilizar el comando `ssh-copy-id root@cliente` para copiar la llave. Verificar que se puede acceder al usuario root del cliente a través de ssh y sin *passwd*. A partir de este punto, solo bastará seleccionar al equipo remoto a realizar el *backup* desde la interfaz de administración e iniciar una primera copia seleccionando el botón Comenzar copia de seguridad completa.

En sistemas Windows, la forma más simple de realizar *backups* es a través del protocolo SMB por lo cual se deberá sobre el sistema Windows ingresar como administrador y configurar el sistema para compartir carpetas que se deseen hacer la copia de seguridad o bien el disco duro completo (por ejemplo C:) definiendo un usuario/*passwd* para compartir este recurso. Desde la interfaz de administración editar la configuración del host remoto con Windows a respaldar (por su IP por ejemplo), el usuario y el método smb (no olvidar de hacer Save después de haber modificado estos datos). Defina en la pestaña *Xfer* el nombre del recurso compartido a respaldar en el equipo remoto y el nombre del usuario y clave de acceso de recurso compartido del equipo Windows remoto. A partir de este punto, solo bastará seleccionar al equipo remoto a respaldar desde la interfaz de administración e iniciar un primer respaldo seleccionando el botón Comenzar copia de seguridad completa.

Con Backuppc se puede definir la frecuencia de los respaldos totales y respaldos incrementales. De modo predeterminado el valor para los respaldos totales es cada 7 días y para los incrementales es cada día. Se recomienda utilizar un valor ligeramente inferior a los días. Por ejemplo, 6.97 en lugar de 7 días y 0.97 en lugar de 1 día para mejorar la granularidad del respaldo (recordar siempre de hacer *Save* después de modificar cada opción).

Si por las necesidades del entorno, la empresa o institución, necesita un sistema de respaldo con opciones y funcionalidades equivalentes a las de paquetes comerciales (ARCserveIT, Tivoli Storage Manager o PerfectBackup), Bareos será la opción apropiada [Bareos]. Bareos es una bifurcación (*fork*) del proyecto Bacula con notables mejoras y que está soportado por un grupo activo de desarrolladores. Al igual que Bacula, está formado por un conjunto de programas que funcionan en una estructura cliente-servidor, y que permite la gestión de respaldo, recuperación y verificación de los datos a través de una red de

ordenadores de diferentes tipos. Bareos también puede funcionar en un solo ordenador y hacer copias de seguridad en cinta y/o discos. Es relativamente fácil de usar y eficiente, pero también ofrece características avanzadas de gestión de almacenamiento que hacen que sea fácil de encontrar y recuperar archivos perdidos o dañados.

La estructura de servicios es similar a Bacula donde se cuenta con *Director, Consola, Files, Storage, Catalog, Monitor*:

- **Director** (*daemon*) es el que supervisa las copias de seguridad, la restauración, la verificación y el almacenamiento y se utiliza para programar copias de seguridad y recuperar archivos.

- **Consola** sirve de interfaz de comunicación con Director (puede ser texto o gráfico) y para planificar o gestionar las acciones que hay que realizar.

- **Files** (cliente, se ejecuta como *daemon*) se instala en la máquina de la cual se desea hacer copias de seguridad. Es específico para el sistema operativo en el que se ejecuta y es responsable de proporcionar los archivos (datos y atributos) cuando sea solicitado por Director.

- **Storage** (*daemon*) es el que realiza el almacenamiento/recuperación específica de archivos a/desde los medios de almacenamiento físicas o volúmenes.

- **Catalog** mantiene los índices de los archivos y bases de datos de volumen para todos los archivos de copia de seguridad permitiendo localizar y restaurar cualquier archivo deseado rápidamente.

- **Monitor** controla y supervisa el estado actual de la administración Bareos, los *daemons* de *File* y *Storage*.

Para realizar la instalación y una primera prueba de concepto con Bareos (en este caso lo haremos con PostgreSQL como base de datos ya que según algunos administradores tiene mejor comportamiento sobre Debian que con otras bases de datos) haremos:

1) Instalar PostgreSQL: `apt-get install postgresql`

2) Instalar Bareos: `apt-get install bareos bareos-bat`

3) Actualizar las entradas en la base de datos:

```
su postgres -c /usr/lib/bareos/scripts/update_bareos_tables
su postgres -c /usr/lib/bareos/scripts/grant_bareos_privileges
```

4) Reiniciar los *daemons*:

```
service bareos-dir start  service bareos-sd start  service bareos-fd start
```

5) Se puede probar el servicio mirando desde la interfaz texto con `bconsole` o desde la interfaz gráfica `bat`. Allí nos indicará si ha podido conectarse con el servidor y el estado. A partir de esto ya tenemos el sistema funcionando y podemos ver los trabajos planificados (definidos en */etc/bareos*) y por ejemplo forzar un respaldo en ese momento. Para un primer contacto se recomienda seguir la interfaz gráfica con el tutorial [BareosTut] de los propios desarrolladores para conocer los parámetros y funcionalidades que presta el servicio, así como es el procedimiento para recuperar o agregar un segundo cliente.

Bareos incluye una interfaz web (Bareos-webui) que puede ser instalada desde el sitio de Bareos [Bareos-WebUI] y algunas restricciones en cuanto a la utilización de las librerías TLS (GnuTLS) que permiten el cifrado durante la transmisión, pero no en el almacenamiento (consultar el manual para mayor detalle).

1.9.2. rdiff-backup y rdiff-backups-fs

Para administradores o usuarios avanzados existe la opción de `rediff-backup` que es un comando para la copia de seguridad de un directorio a otro y que también puede ser a través de una red. El directorio de destino poseerá una copia del directorio fuente pero también información adicional (diffs) para gestionar mejor las copias incrementales (aún de archivos antiguos). La aplicación también preserva subdirectorios, enlaces no simbólicos (hard links), archivos dev, permisos, propiedad (uid/gid), fechas de modificación, atributos extendidos y ACL y puede operar de forma eficiente a través de un *pipe* a *rsync* por ejemplo o utilizar rdiff-backup + ssh para realizar backups incrementales en lugar remoto transmitiendo solo las diferencias. También es habitual (y muy simple) tener un proveedor de servicios (Gdrive, Dropbox, etc) con un directorio sobre el ordenador local que será sincronizado sobre el *cloud* del proveedor y hacer la copia rdiff-backup sobre este directorio para que luego se transmita al *cloud* del proveedor. La forma habitual de trabajo (después de instalado el comando) es muy simple `rdiff-backup fuente destino` y en el caso remoto /algun/dir-local a /algun/dir-remoto sobre la máquina hostname.org será

```
rdiff-backup /algun/dir-local hostname.org::/algun/dir-remoto
```

pero puede ser a la inversa

```
rdiff-backup user@hostname.org::/remote-dir local-dir
```

y también podrían ser sobre dos máquinas remotas

```
rdiff-backup -v5 -print-statistics user1@host1::/source-dir user2@host2::/dest-dir
```

Para recuperar un directorio local es simplemente a través de la copia y si es re-

moto (más ejemplos en http://www.nongnu.org/rdiff-backup/examples.html)

```
rdiff-backup -r now hostname.org::/remote-dir/file local-dir/file
```

Uno de los problemas de recuperar la información previamente guardada es que acceder a los archivos de copia de seguridad más reciente es fácil (solo se debe introducir el directorio de copia de seguridad) pero es complicado si deseamos acceder y navegar a través de las versiones anteriores. `rdiff-backup` permite acceder a ellos por un conjunto de instrucciones, que requieren un conocimiento preciso de los nombres de archivo y los tiempos de copia de seguridad y que puede ser complicado de recordar o seleccionar. `rdiff-backup-fs` (https://code.google.com/p/rdiff-backup-fs/) permite crear un sistema de archivo en espacio de usuario basado en la información de rdiff. Para ello se utiliza FUSE (*Filesystem in userspac*e - http://fuse.sourceforge.net/) que permite que cualquier usuario pueda montar el sistema de archivo guardado por rdiff y navegar por cada incremento y copia realizada.

Una alternativa para las copias de resguardo de un sistema remoto por ssh es utilizar `sshfs` (http://fuse.sourceforge.net/sshfs.html) que permite acceso en el espacio de usuario a un directorio remoto mostrándolo localmente por lo cual luego aplicando rdiff se puede hacer copias incrementales de este recurso remoto.

1.10. *Public Key Infrastructure* (PKI)

Por PKI (*Public Key Infrastructure*) se entiende un conjunto de hardware y software, políticas y procedimientos de seguridad que permiten la ejecución con garantías de operaciones como el cifrado, la firma digital o el no repudio de transacciones electrónicas. El término PKI se utiliza para referirse tanto a la autoridad de certificación y al resto de componentes si bien a veces, de forma errónea, se utiliza para referirse al uso de algoritmos de clave pública. En una operación que se utilice PKI intervienen, además de quien inicia la acción y su destinatario un conjunto de servicios que dan validez a la operación y garantizan que los certificados implicados son válidos. Entre ellas podemos contar con la autoridad de certificación, la autoridad de registro y el sistema de sellado de tiempo. La infraestructura PKI utiliza procedimientos basados en operaciones criptográficas de clave pública, con algoritmos de cifrado bien conocidos, accesibles y seguros y es por ello que la seguridad está fuertemente ligada a la privacidad de la clave privada y la políticas de seguridad aplicadas para protegerla. Los principales usos de la PKI son entre otros: autenticación de usuarios y sistemas (*login*), identificación, cifrado, firma digital, comunicaciones seguras, garantía de no repudio.[pki][pkicry]

Como se vio anteriormente (para Apache+SSL) la generación de certificados digitales es extremadamente necesario dentro de las necesidades habituales de

los administradores y usuarios de los sistemas de información, por ejemplo, para acceder a una página SSL o para firmar un correo electrónico. Estos certificados se pueden obtener de las entidades certificadoras privadas como StartComm (https://www.startssl.com/) o Verisign (http://www.verisign.es/) que tienen algunos productos gratuitos (por ejemplo para firmar correos) pero, en su mayoría, los productos tiene un costo elevado. También podemos recurrir a utilizar GNUPG (https://www.gnupg.org/) que es *Open-Source* y para determinados fines es correcto pero no para otros o bien considerar entidades de certificación institucionales, por ejemplo en Cataluña IdCat (http://idcat.cat/) que nos permitirán obtener certificados de ciudadano (gratuitos) para firmado, encriptación, autentificación, no repudio pero no para servidores (Idcat si que puede expedir otro tipo de certificados pero solo es para la administración pública y universidades del país). En el presente apartado instalaremos y configuraremos una entidad certificadora raíz (y sub-entidades CA para los diferentes dominios de control de esta CA) basada en la aplicación TinyCA (si bien existen otros paquetes como OpenCA –https://www.openca.org/– que son más potentes y escalables pero son más complejos en su configuración) que se adapta muy bien para pequeñas y medianas instituciones/empresas. Una entidad certificadora es la base de la infraestructura PKI y emite certificados para dar garantías de autenticidad de una comunicación, un sitio o una información. Cuando instalamos Apache hemos auto-firmado el certificado digital (es decir nosotros hemos hecho todos los pasos: petición, generación y firma) que codifican la comunicación pero ello no da garantía si no se verifica la firma del certificado auto-firmado. Para solucionar este problema, y sin recurrir a un proveedor público/privado, crearemos nuestra propia estructura de CA y distribuiremos por canales seguros a nuestros usuarios/clientes el certificado personal/de servidores y el certificado raíz de la CA para que los instalen en sus navegadores (como ya están los de StartComm, Verisign, Catcert/Idcat u otras entidades de certificación). De esta forma que cuando un sitio web, por ejemplo, le presente al navegador un certificado digital para codificar la comunicación SSL, el navegador reconozca el sitio por el certificado raíz que tiene instalado y confíe en él (y no salga la típica ventana de advertencia de sitio inseguro) y manteniendo la privacidad de las comunicaciones. La utilización de estos certificados creado por esta CA puede ser varios: para firmar/encriptar un mail, para validar nuestros servidores SSL o para configurar la VPN entre otros.

La práctica habitual es tener una CA y crear Sub-CA (una por cada dominio) para que el sistema sea escalable y por lo cual la CA firmará la creación de nuevas Sub-CA y luego estas (su responsable) tendrán capacidad para firmar sus certificados y todas tendrán el mismo certificado raíz de la RootCA. (guía muy detallada en [tinyca]) Una vez instalada (`apt-get install tinyca`), ejecutamos `tinyca2` y nos presentará la pantalla principal y indicaciones para crear una nueva CA que será la rootCA. Completamos la pantalla con los datos, por ejemplo, *Name=ROOTCA-nteum.org, Data for CA=ROOTCA-nteum.org, SP, passwd, BCN, BCN, NTEUM, NTEUM, adminp@sysdw.nteum.org, 7300, 4096, SHA-1* para todos los campos respectivamente. Cuando le damos a OK, apa-

recerá una nueva pantalla para configurar la rootCA. Seleccionar *"Certificate Signing/CRL Signing"*, "*critical*" y en Netscape *Certificate Type="SSL CA, S/MIME CA, Object Signing CA*.", si se desea poner una URL para revocar los certificados (lo cual puede ser una buena idea) se pueden rellenar los campos correspondientes, luego finalmente OK. Con ello se crearán los certificados y aparecerán la pantalla principal de tinyCA con 4 tabuladores donde indica *CA* (información sobre la CA activa), *Certificates* (muestra los certificados creados por la CA) *Keys* (muestra las llaves de los certificados) y *Requests* (peticiones de certificados que esperan ser firmados por la CA). Por encima aparecen una serie de iconos para acceder a funciones directas pero Tinyca2 tiene un error y no muestra los nombres de los iconos.

El siguiente paso es crear una nueva sub-CA y para hacerlo verificar que estamos en el tab de la CA y haciendo click en el tercer icono desde la derecha que nos mostrará una ventana que como subtítulo tiene "*Create a new Sub CA*". Deberemos introducir el *passwd* que pusimos en la rootCA, darle un nombre (subca-sysdw.nteum.org), *Data-for-CA* (sysdw.nteum.org) y el resto de datos como hicimos anteriormente (el *passwd* no debe ser necesariamente el mismo de la rootCA) y cuando hacemos OK nos saldrá la ventana principal pero con la Sub-CA seleccionada, si queremos volver a la CA deberemos ir al menu *File->Open* y abrir la rootCA. Recordad que la Sub-CA será quien gestionará las peticiones y firmará los certificados para ese dominio por lo cual debemos seleccionar la adecuada en cada momento. La root-CA solo la utilizaremos para crear/revocar nuevas sub-CA y para exportar el certificado raíz de la rootCA (necesario para enviarles a nuestros clientes para que se lo instalen en sus navegadores).

Para crear un certificado que permita certificar nuestro web-server cuando utiliza SSL (https://sysdw.nteum.org), con nuestra Sub-CA seleccionada vamos al tab de *Request* y con el botón derecho seleccionamos "*New request*" en la cual se abrirá una ventana donde deberemos introducir la URL de nuestro servidor (sysdw.nteum.org) como *CommonName* y rellenar el resto de los datos. Una vez creado, lo seleccionamos y con el botón derecho indicamos "*Sign request*" y seleccionamos "*Server request*" que nos mostrará una ventana con el *password* de la CA y la validez. Este procedimiento es el que genera confianza ya que estamos validando la información aportada en la petición y firmando el certificado (los cuales ya podremos ver en los tabs correspondientes.

Ahora deberemos exportar los certificados (del web y la rootCA), la *key* y configurar Apache para que los incorpore. Lo primero será exportar el rootCA e introducirlo en Firefox/Iceweasel. Para ello en la ventana principal *File->OpenCA* seleccionamos nuestra RootCA y seleccionando el segundo icono de la derecha que corresponde a "*Export CA Certificate*" indicamos un nombre de archivo y el formato (PEM es el adecuado) y salvamos el certificado en nuestro sistema de archivo (p. ej., /tmp/ROOTCA-nteum.org.pem). Es importante tener en cuenta de hacer un `chmod 644 /tmp/ROOTCA-nteum.org.pem` ya que se salvará como 600 y por lo cual otros usuarios no lo podrán importar. Desde Firefox/Iceweasel seleccionamos *Preferences/Setting ->Advanced ->Certificates*

->*View Certificates* ->*Authorities* ->*Import* y seleccionamos el archivo antes creado marcando todas las "*trust settings*" que nos presenta en la ventana siguiente (3). Luego podremos ver con el nombre de que le dimos a "*Organization*" el certificado correspondiente.

A continuación en TinyCA2 abrimos nuestra sub-CA, seleccionamos el tab *Certificates* y sobre el certificado seleccionamos el botón derecho e indicamos *"Export certificate"* dando un nombre de archivo, p. ej, sysdw.nteum.org-cert.pem, luego repetimos el proceso con la *key* en el *Key* tab y haciendo *"Export key"* salvándolo como p. ej., sysdw.nteum.org-key.pem. Es importante decidir si le ponemos *passwd* o no ya que si le ponemos cada vez que arranque el servidor nos solicitará el *passwd*. Sobre el tab CA deberemos exportar ahora el *"certificate chain"* que es el primer icono desde la derecha y salvarlo como sysdw.nteum.org-chain.pem. Moveremos estos tres archivos al directorio */etc/ssl/private/* y pasaremos a configurar Apache modificando el archivo que configure nuestro sitio SSL, por ejemplo nosotros hemos utilizado */etc/apache2/sites-available/default-ssl* en el cual hemos modificado (solo se muestra las líneas principales):

```
<IfModule mod_ssl.c>
<VirtualHost _default_:443>
    ServerName sysdw.nteum.org
...
    SSLEngine on
    SSLCertificateFile /etc/ssl/private/sysdw.nteum.org-cert.pem
    SSLCertificateKeyFile /etc/ssl/private/sysdw.nteum.org-key.pem
    SSLCertificateChainFile /etc/ssl/private/sysdw.nteum.org-chain.pem
...
</VirtualHost>
</IfModule>
```

Solo nos resta habilitar el sitio (*a2ensite default-ssl*), e reiniciar Apache (con la orden `service apache2 restart`) -nos pedirá el *passwd* si hemos lo hemos puesto en la *key*- y probar (en el navegador que tenemos instalado el certificado raíz de la rootCA) la URL https://sysdw.nteum.org que si todo está correcto cargará la página sin la típica ventana que el sitio no es seguro.[tinyca]

Para crear certificados para una dirección de correo y distribuirlos a nuestros usuarios junto con el certificado de la rootCA podemos hacer en el tab *Certificates* de nuestra sub-CA, seleccionar "*New - Create Key and Certificate (Client)*" y entrar el nombre y la dirección de correo para la cual queremos validar así como el *passwd* para protegerlo hasta que llegue a su nuevo destinatario (y que luego el podrá/deberá cambiar). A continuación debemos exportarlo teniendo en cuenta de utilizar el formato PKCD#12 que incluye el certificado y la llave. Después de enviarle al usuario el archivo con el certificado más el de la root-CA, el podrá agregarlo a su gestor de correo de forma similar a root-CA pero como *personal certificates*. Luego podrá enviar correo firmados digitalmente y el destinatario (que deberá tener instalado el certificado de la rootCA) podrá verificar la firma del correo.

Otra opción interesante para configurar una CA es el paquete XCA, *X Certificate and key management* [XCA], que permite crear y gestionar de forma muy simple certificados X.509, peticiones de certificados, llaves privadas RSA, DSA y EC, *Smartcards* y listas de revocaciones (CRLs), es decir, dispone de todo lo necesario para implementar una CA y se pueden crear sub-CA recursivamente y además incluye plantillas para extender las necesidades de las peticiones que se tengan.

Su instalación es muy fácil, `apt-get install xca` y luego se deberá ejecutar como `xca`. La primera acción que se debe hacer es crear/seleccionar una base de datos desde el menú *File* (o abrirla si ya se ha creado en un paso anterior), la cual está protegida por una contraseña utilizada para cifrarla y mantener la seguridad de la CA. La aplicación presenta 5 pestañas (*Keys, Requests, Certificates, Templates and Revocation lists*), y mediante menús, botones a la derecha de la aplicación o menús contextuales (botón derecho del ratón) se puede acceder a todas las opciones de configuración.

Como caso de uso generaremos un certificado para SSL para que lo podamos luego cargar en Apache y en el navegador para utilizarlo en HTTPS y que no nos dé el aviso típico de seguridad de "Certificado autofirmado".

Creación del certificado de la entidad certificadora (RootCA): este certificado es el que deberemos instalar en el navegador para que valide la cadena de confianza cuando el servidor web le presente el certificado SSL de servidor. Para crearlo seguiremos los siguientes pasos:

1) Iniciamos XCA, y creamos o abrimos la base de datos introduciendo la contraseña.

2) Vamos a la pestaña *Certificates ->New Certificate* y se abrirá una nueva ventana llamada *Create X509 Certificate*.

3) Introducimos la información de identificación y vamos a la pestaña *Subject*, configuramos los valores de la sección *Distinguished name*, luego vamos a *Generate a new key* y en la ventana que se abre seleccionamos el *Name, Type (RSA), Size* y hacemos un *Create*.

4) Configuramos las extensiones X.509: vamos a la pestaña *Extensions* y desde *Type list* seleccionamos *Certification Authority* y modificamos *Validity dates*. Generalmente RootCA son válidos por 5 años.

5) Configuramos el uso: vamos a la pestaña *Key usage* y desde el panel izquierdo seleccionamos *Digital Signature, Key Agreement, Certificate Sign*. Hay que tener cuidado con escoger otras opciones ya que podría ser que algunos sistemas operativos rechazasen el certificado.

6) Finalmente hacemos *OK* para crear el certificado y nos aparecerá en la pestaña *Certificates*.

7) Para exportar el certificado, dentro de la pestaña *Certificates*, seleccionamos el certificado a exportar y hacemos *Export*. En la nueva ventana seleccionamos el nombre del archivo y el formato PEM desde la lista *Export Format* y luego hacemos *OK*.

8) Para instalar el certificado en Firefox/Iceweasel, por ejemplo, vamos a *Preferences->Advanced->Certificates->View Certificates->Authorities->Import* y cargamos el archivo que hemos salvado previamente. Luego de realizada esta operación lo veremos en la lista con el nombre introducido en la sección *Distinguished name* (punto 3) y como *OrganizationalName* e indexado por el *CommonName* de esta sección también.

Creación del certificado SSL y la llave privada para el servidor: se creará la llave privada y el certificado que deberemos instalar en Apache:

1) En XCA vamos a la pestaña *Certificate signing requests*, seleccionamos *New Request* y se abrirá una ventana de *Create Certificate Signing Request*.

2) Vamos a la pestaña *Source*, a continuación desde *Template* seleccionamos *[default] HTTPS_Server*, y hacemos un clic en *Apply extensions*.

3) Vamos a la pestaña *Subject*, rellenamos los campos de la sección *Distinguished name section* (con cuidado de poner el FDQN de nuestro servidor en el *CommonName*), generamos una nueva llave con *Generate a new key* y en la nueva ventana seleccionamos el *name* para la llave privada y el tamaño (*key size*), luego hacer un clic en *Create*.

4) Terminamos la petición haciendo un OK.

5) Firmamos el certificado: vamos a la pestaña *Certificate signing requests*, seleccionamos el certificado que se ha de firmar y haciendo clic en el botón derecho seleccionamos *Sign*; se abrirá una ventana *Create x509 Certificate*. En la sección *Source* en el apartado *Signing* seleccionamos *Use this Certificate for signing* y después el certificado de la RootCA desde el menú. Finalmente hacemos *OK*.

6) El certificado ahora aparecerá como firmado y en la ventana *Certificate* bajo la entidad que ha firmado el certificado (tendremos que hacer un clic en el + de la RootCA).

7) Para exportarlo vamos a la pestaña *Certificates*, seleccionamos el certificado previamente firmado y exportamos en un archivo (por ejemplo, server.crt) con formato PEM. Luego vamos a la pestaña de *Private Keys*, seleccionamos la llave privada correspondiente y con el botón derecho hacemos un *Exportar*: seleccionamos el archivo (por ejemplo, serverkey.pem) sin contraseña (para evitar que cada vez que reiniciemos Apache se tenga que introducir la contraseña) y escogemos el formato PEM.

Finalmente, en Apache, editamos

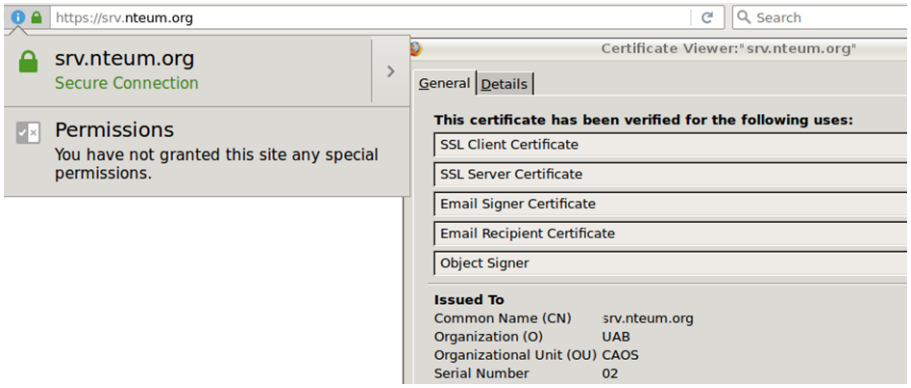
```
vi /etc/apache2/sites-available/default-ssl.conf
```

y modificamos:

SSLCertificateFile /etc/ssl/private/server.crt
SSLCertificateKeyFile /etc/ssl/private/serverkey.pem

Copiamos el certificado y la llave privada antes generada en */etc/ssl/private*, habilitamos el sitio (`a2ensite default-ssl`), habilitamos el módulo (`a2enmod ssl`) y reiniciamos Apache (`apachectl restart`). Finamente verificamos en el navegador que con la URL del *site* (y con la RootCA instalada sobre este) no nos da errores de SSL (el certificado es validado y sale en verde, como se ve en la figura 11).

Figura 11



Como apunte final con relación a la PKI, todos nuestros usuarios/clientes/visitantes de nuestros servicios deberán tener instalado el certificado de la root-CA en sus navegadores/clientes para así validar los sitios/servicios que están bajo nuestro dominio/sub-dominios ya que los navegadores/clientes de correo no incorporan estos certificados raíz por defecto. Si nuestro dominio/servicios lo requiere podemos gestionar con Mozilla la inclusión de nuestro certificado en http://www.mozilla.org/en-US/about/governance/policies/security-group/certs/ pero no es un trámite fácil y es necesario cumplir con una serie de requisitos ya que las garantías del sistema radica en la inclusión de estos certificados.

1.11. Open Computer and Software Inventory Next Generation (OCS)

OCS [OCS] es un servicio que permite gestionar los inventarios de los activos de una infraestructura tecnológica. Este funciona mediante un servidor y un conjunto de agentes (que funcionan en cada una de las máquinas que ha de ser inventariada) que recopila la información sobre el hardware y software de

equipos. Dispone de una interfaz web con la posibilidad de agregar diferentes *plugins* y diversos criterios de búsquedas para facilitar la localización de activos, buscar en la red por medio del *IPDiscovery*, o instalar aplicaciones remotamente a través de la creación de *Builds*. La información intercambiada entre los agentes y el servidor está en formato XML y el servidor utiliza Apache, MySQL y Perl para gestionar y visualizar el repositorio necesitando muy pocos recursos y siendo posible su instalación en diferentes plataformas.

Para su instalación debemos instalar algunos paquetes previos:

1) `apt-get install apache2`

2) `apt-get install php5 libapache2-mod-php5 php5-cli php5-common php5-cgi php5-gd`

3) `apt-get install mysql-client mysql-server mysql-common php5-mysql`

4) `apt-get install libxml-simple-perl libio-compress-perl libdbi-perl libdbd-mysql-perl libapache-dbi-perl libnet-ip-perl libsoap-lite-perl`

5) Ejecutamos: `cpan -i XML::Entities`

6) `apt-get install ocsinventory-server ocsinventory-agent`

7) Cargamos en el navegador http://localhost/ocsreports/install.php.

8) Nos solicitará el usuario *root* de MySQL, su contraseña y el nombre de la base de datos, que será ocsweb, y la ubicación que es *localhost*.

9) A continuación, podremos entrar en el sitio (http://localhost/ocsreports/) con el usuario *admin* y contraseña *admin*. En la primera página veremos que sale un "Aviso de Seguridad", ya que deberemos hacer unos cambios para finalizar la instalación.

10) Ejecutamos: `cd /usr/share/ocsinventory-reports`

```
mv install.php install.php.org
```

```
mysql -u root -p
```

```
SET PASSWORD FOR 'ocs'@'localhost' = PASSWORD('psswd_user_ocs');
FLUSH PRIVILEGES;
```

`vi dbconfig.inc.php` *Cambiar el passwd*

```
define("COMPTE_BASE","ocs");
define("PSWD_BASE","psswd_user_ocs");
```

```
cd /etc/apache2/conf-enabled/
```

`vi ocsinventory-server.conf` *Cambiar el passwd*

```
PerlSetVar OCS_DB_PWD psswd_user_ocs
```

```
service apache2 reload
```

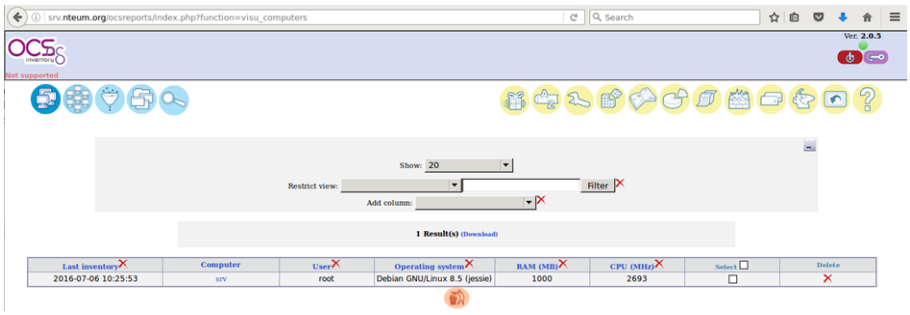
11) Accedemos a http://localhost/ocsreports/ y podremos ver que solo hay una advertencia de la contraseña por defecto del usuario administrador. Seleccionamos en los iconos el de usuarios y cambiamos o creamos un nuevo usuario con el rol de *SuperAdministrator*. Hay que tener en cuenta que si tenemos *mod_security* habilitado en Apache nos dará el error de que no puede acceder a *localhost/ocs-reports/index.php*. Esto es debido a que *mod_security* lo bloquea por lo cual se debe agregar la regla adecuada (o si solo se desea hacer unas pruebas se puede deshabilitar temporalmente desde */etc/modsecurity/modsecurity.conf* y reiniciar Apache).

12) Cuando hemos instalado el agente para la propia máquina (o para cualquier otra máquina) le debemos indicar la IP/nombre del servidor (o también en */etc/ocsinventory/ocsinventory-agent.cfg*) y este se almacenará en el archivo */etc/cron.daily* para ejecutarse. También podemos forzar la ejecución con la orden `/usr/bin/ocsinventory-agent` y ya podremos verlo en OCS y navegar por todos los parámetros. Se pueden encontrar los agentes para Windows, MacOS, Android y otros Linux desde el sitio de OCS-Dev*.

*https://launchpad.net/ocsinventoryx

Es importante tener en cuenta que es un paquete muy extenso con un gran conjunto de opciones y posibilidades y el administrador deberá analizar y evaluar cada una de ellas y configurarlas/adaptarlas al entorno [OCS].

Figura 12



1.12. GLPi

GLPi (del francés, *Gestionnaire Libre de Parc Informatique*) es un paquete que permite la gestión de activos y fallos de un sistema informático (*IT Asset Management and issue tracking system)* o también conocido como *service desk*. Es un paquete que tiene una alta integración con otros (por ejemplo, OCS). Es una aplicación web y está escrito en PHP y existe una comunidad muy activa de desarrolladores y usuarios.

Entre sus características hay que destacar que permite construir un inventario de todos los recursos de la organización y gestionar las tareas administrati-

vas (e incluso financieras). Además, ayuda a los administradores a tener y a gestionar una base de datos de activos tecnológicos, así como a almacenar un historial de las intervenciones de mantenimiento y también asistir a los usuarios en la comunicación de incidencias (*help Desk*). [GLPi]

Su instalación (p. ej., después de haber instalado OCS ya que necesita Apache, PHP y MyQSL) es sumamente fácil (`apt-get install glpi`) y nos pedirá las contraseñas del *root* de la base de datos y la del usuario *glpi* para gestionarla. A partir de ello podemos conectar a página web (http://srv.nteum.org/glpi/) donde solicitará unos usuarios/*passwd* que por defecto son:

- *glpi*/*glpi* para la cuenta de administrador,
- *tech*/*tech* para la cuenta de técnico,
- *normal*/*normal* para una cuenta normal y
- *post-only*/*postonly* para una cuenta de envío solamente.

Con ello, y entrando como usuario *glpi*, veremos una serie de tareas que deberemos hacer:

1) Cambiar los *passwds* por defecto (pestaña *Administration->Users*)

2) Renombrar el archivo de instalación:

```
mv /usr/share/glpi/install/install.php /usr/share/glpi/install/install.php.org
```

3) Para conectar GPLi y OCS deberemos buscar el *plugin* apropiado a nuestra versión (por ejemplo de Debian 8.5 disponemos de GLPi V0.84 –véase en el pie de página de GPLi–) y en la página https://forge.glpi-project.org/projects/ocsinventoryng/files el adecuado es el glpi-ocsinventoryng-1.0.3.tar.gz, que descargaremos e instalaremos en */usr/share/glpi/plugins* con

```
tar xzvf /sitio-donde-se-haya-descargado/glpi-ocsinventoryng-1.0.3.tar.gz
```

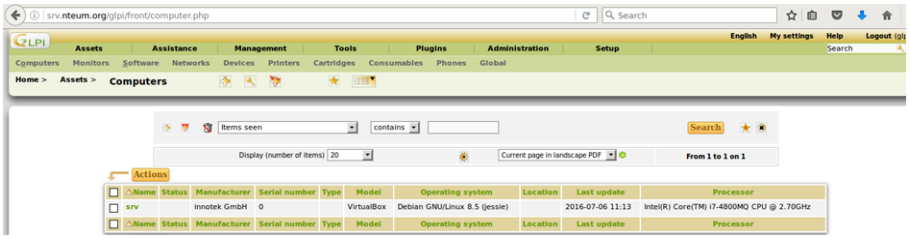
4) Ir a la pestaña de *Setup->Plugins* y veremos el *plugin OCS Inventory NG*, el cual deberemos actualizar (*Update*) y activar (*Enable*). Haremos un clic en este (columna *Name*) y accederemos a su configuración.

5) Agregar un nuevo servidor (signo +), indicando *Name* deseado, *Host for the database = localhost*, *Synchronisation method = Standard*, *Database = ocsweb*, *User=ocs*, *Passwd=el-que-tenga-la-base de datos-OCS* (si no lo recordamos podemos consultar el archivo */etc/ocsinventory/dbconfig.inc.php*) y hacer *Save*. Veremos, en un mensaje en la parte inferior de la ventana, si ha podido conectarse y acceder.

6) En la pestaña *Import* (sugerencia: poner como *global import*) y en la pestaña *General* seleccionar los parámetros deseados haciendo *Save* nuevamente.

7) A continuación, se puede esperar que se actualice automáticamente o se puede ir a *Setup->Automatic Actions->ocsng* y cambiar el *Status* y ejecutarlo en ese momento (*Execute*). Con ello veremos que el inventario se actualiza y vemos la máquina que tenemos en OCS (figura 13).

Figura 13



Plugins de GPLi

Son muy interesantes las posibilidades de extensión que tiene GLPi mediante la incorporación de *plugins*. Consultar la página http://plugins.glpi-project.org/ donde se encuentran ordenados por categorías y tipos.

1.13. Supervisor (Process Control System)

Supervisor es una aplicación (cliente/servidor) que permite a los usuarios/administradores monitorizar y controlar un conjunto de procesos [Sup].

Su función es simplificar la escritura de *scripts* en *rc.d* (o en *systemd*) para cada proceso de usuario o administrador y que tiene como función su inicio/reinicio/gestión; estos *scripts* son complicados de escribir (más en *systemd*). Además, en el caso de que un proceso/servicio "caiga", permite definir cómo ponerlo en marcha automáticamente. También permite saber de forma simple (mediante una página web) el estado de los procesos y ver rápidamente sus *logs*. Además permite que un usuario (no administrador) pueda gestionar los procesos sin tener que darle permisos de `sudo` o acceso a la consola, agrupar procesos y enviar órdenes al grupo (inicio, parada, reinicio). Y todo ello mediante una interfaz sencilla y centralizada, con una configuración simplificada, una ejecución eficiente y totalmente extensible.

Los componentes de Supervisor son **supervisord** que es el servidor, responsable de iniciar los procesos (subprocesos) hijos registrando la salida y la salida de error estándar y la generación y gestión de eventos para estos procesos/subprocesos. Este servidor se configura con */etc/supervisor/supervisord.conf* el cual se deberá proteger con los permisos adecuados, ya que contiene contraseñas en texto sin cifrar. **supervisorctl** es el cliente (CLI) que proporciona la interacción con el servidor a través de un conjunto de subcomando (o puede trabajar también interactivamente). **Servidor web** es una interfaz simplificada a la cual se accede a través del puerto 9001 (p.ej. en *http://localhost:9001/*) y permite ver el estado del proceso de control y ejecutar operaciones colectivas o individuales sobre los procesos.

Su instalación es simple:

1) `apt-get install supervisor`

2) A continuación, renombraremos la configuración inicial:

```
cd /etc/supervisor
mv supervisord.conf supervisord.conf.org
```

3) Generamos una nueva configuración:

```
echo_supervisord_conf > supervisord.conf
```

4) Modificamos esta para una similar a:

```
[unix_http_server]
file=/var/run/supervisor.sock                   ; (the path to the socket file)
chmod=0700                                      ; socket file mode (default 0700)

[inet_http_server]                              ; inet (TCP) server disabled by default
port=127.0.0.1:9001                     ; (ip_address:port specifier, *:port for all iface)
username=admin                          ; (default is no username (open server))
password=nuestro_passwd                 ; (default is no password (open server))

[supervisord]
logfile=/var/log/supervisor/supervisord.log ; (main log file;default $CWD/supervisord.log)
logfile_maxbytes=50MB                   ; (max main logfile bytes b4 rotation;default 50MB)
logfile_backups=10                      ; (num of main logfile rotation backups;default 10)
loglevel=info                                   ; (log level;default info; others: debug,warn,trace)
pidfile=/var/run/supervisord.pid        ; (supervisord pidfile;default supervisord.pid)
nodaemon=false                          ; (start in foreground if true;default false)
minfds=1024                                     ; (min. avail startup file descriptors;default 1024)
minprocs=200                                    ; (min. avail process descriptors;default 200)

[rpcinterface:supervisor] supervisor.rpcinterface_factory=supervisor.rpcinterface:make_main_rpcinterface

[supervisorctl]
serverurl=unix:///var/run/supervisor.sock ; use a unix:// URL  for a unix socket

[include]
files = /etc/supervisor/conf.d/*.conf
```

5) Reiniciaremos el servicio: `systemctl restart supervisor` (o también `service supervisor restart`) y ya podremos acceder via `supervisorctl` o por http://localhost:9001. Hay que indicar que no veremos ningún proceso ya que los deberemos configurar.

6) Para gestionar procesos deberemos escribir un archivo que contendrá las órdenes y las indicaciones para poder gestionarlos. En nuestro caso, como prueba de concepto gestionaremos los servidores de Apache y SSH. Para ellos escribiremos en */etc/supervisor/conf.d* con archivos con lo siguiente[PVan]:

```
vi /etc/supervisor/conf.d/http.conf
```

```
[program:apache2]
command=apachectl -DFOREGROUND
autostart=true
autorestart=true
startretries=1
startsecs=1
stderr_logfile=/var/log/apache2/supervisor.error.log
stdout_logfile=/var/log/apache2/supervisor.access.log
user=root
```

```
vi /etc/supervisor/conf.d/ssh.conf
```

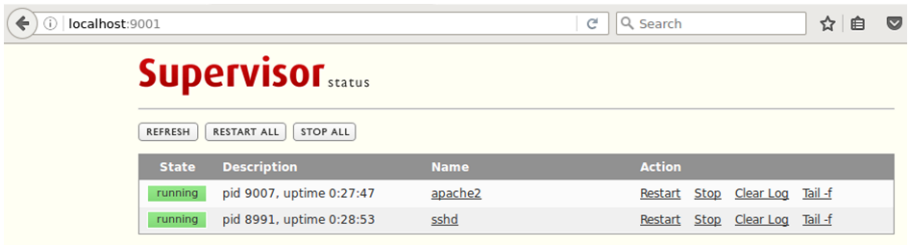
```
[program:sshd]
command=/usr/sbin/sshd -D -e -f /etc/ssh/sshd_config
autostart=true
autorestart=true
startretries=1
startsecs=1
stderr_logfile=/var/log/supervisor/ssh.error.log
stdout_logfile=/var/log/supervisor/ssh.access.log
user=root
```

7) Ya que estos procesos pueden estar ejecutándose por los archivos de */etc/init.d* los deberemos parar primero para poderlo poner bajo el control de supervisor. Luego hacemos que se relea la configuración y se actualice con:

```
supervisorctl reread
supervisorctl reload
```

8) Ahora podemos acceder a la página web y veremos algo como lo que se muestra en la figura 14, donde ya podremos gestionarlos y ver la información correspondiente.

Figura 14



Lectura recomendada

Consultar la documentación [Sup] para ver todas las posibilidades de Supervisor en la gestión de eventos, *logs* y API que permiten obtener mucha más funcionalidad de la mostrada.

Es importante tener en cuenta que cuando ponemos *autostart=true* deberemos asegurarnos de quitar el *script* de inicio habitual (*/etc/init.d* o *systemd*), ya que este está bajo el control de *supervisord* . En caso que no lo hagamos dependerá del orden de arranque pero lo habitual es que supervisord dé un error ya que no podrá poner en marcha el servicio porque ya existe o los puertos están ocupados).

1.14. OwnCloud. File Sync & Share Server

OwnCloud [OC] es un proyecto que permite el acceso, compartición y sincronización de archivos sobre un servidor (equivalente a opciones comerciales como Dropbox, Google Drive o OneDrive). Este provee el acceso a los archivos a través de una interfaz web muy intuitiva y por WebDav pero también hay clientes para Windows, Linux, MacOS y sistemas operativos móviles que permiten la sincronización fácil entre el dispositivo y el servidor. También permite gran cantidad de plugins tales como visualizador de PDF, un cliente de correo, calendario y gestor de tareas, etc. La lista de aplicaciones completa se puede obtener de https://apps.owncloud.com/.

Para su instalación seguiremos los siguientes pasos:

1) `apt-get install owncloud`

2) `mysql -u root -p` *Introducir el passwd de root para MySQL y ejecutar:*

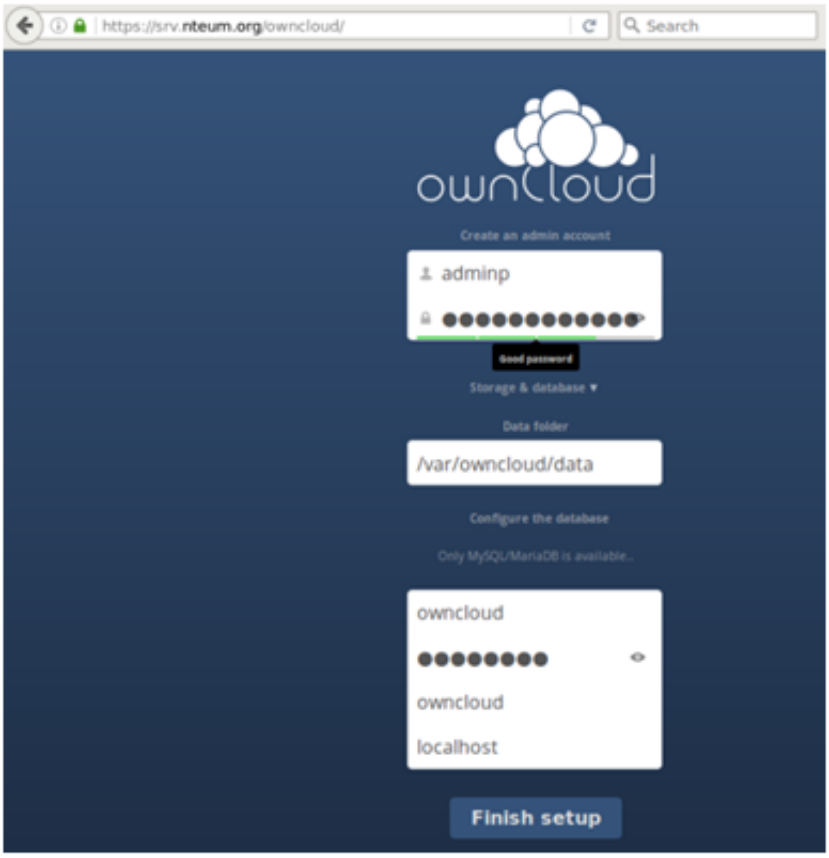
```
CREATE DATABASE owncloud;
CREATE USER owncloud@localhost IDENTIFIED BY 'passwd-DB-OwnCloud';
GRANT ALL PRIVILEGES ON owncloud.* TO owncloud@localhost;
FLUSH PRIVILEGES;
quit
```

3) Creamos el espacio para el repositorio (en un sitio que tengamos espacio disponible):

```
mkdir /var/owncloud
chown www-data:www-data /var/owncloud
chmod 750 /var/owncloud
```

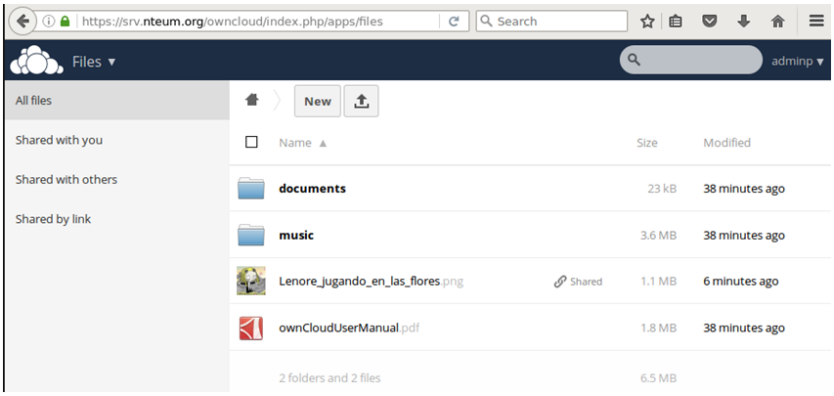
4) Finalmente vamos a la página web para terminar de hacer la configuración: http://srv.nteum.org/owncloud/. Escogemos el nombre del administrador y un *passwd* robusto, el repositorio */var/owncloud/, Username=owncloud, Password=passwd-DB-OwnCloud* (introducido en el punto 2), *Database-name= owncloud; Hostname=localhost* y hacemos clic en finalizar la instalación.

Figura 15



A partir de aquí saldrá la pantalla de OwnCloud y el aviso para descargar las *apps* para dispositivos móviles y ya podremos interactuar con la interfaz web subiendo archivos, visualizándolos, gestionando el calendar/tareas/contactos, etc. (figura 16).

Figura 16



Es importante activar la conexión vía https para mantener la privacidad y evitar problemas de administración. También se recomienda acceder a la interfaz de administración para configurar opciones como compartir documentos entre diferentes servidores, visualizar documentos OpenOffice/Office, permitir la compartición mediante un enlace, forzar https, o también habilitar/deshabilitar las apps desde la pestaña correspondiente (esquina superior izquierda).

Actividades

1. Configurad un servidor Apache+ SSL+ PHP+ MySQL+ PHPAdmin para visualizar las hojas personales de los usuarios.

2. Configurad un servidor Apache + un Reverse Proxy para acceder al servidor web (que se encuentra en una máquina interna) desde una máquina externa a través del Proxy. Ampliad la actividad añadiendo otro servidor *web* interno y haced que el *proxy* balancee entre los dos servidores internos. Probad diferentes políticas de balanceo.

3. Cread y configurad un sistema de correo electrónico a través de Exim, Fetchmail, SpamAssassin y un servidor IMAP para recibir correos desde el exterior y poder leerlos desde una máquina remota con el cliente Mozilla (Thunderbird).

4. Instalad la Wiki MoinMoin y cread un conjunto de páginas para verificar su funcionamiento.

5. Instalad el servidor de *backups* BackupPC y generad una copia de respaldo desde una máquina Windows y otra desde una máquina Linux. Escoged el método de comunicación con los clientes y justificar la respuesta. Realizad la misma experiencia con Bareos y obtened conclusiones sobre eficiencia, productividad y simplicidad.

6. Configurad una CA con TinyCA y generar/probar los certificados para una página web con SSL y para enviar correos firmados y verificarlos desde otra cuenta.

7. Configurad un sistema de correo (corporativo) de altas prestaciones con Postfix, ClamAV, SpamAssassim, Imap y RoundCube. Verificad que todas las opciones y posibilidades funcionan así como el acceso vía web y a través de un cliente (Icedove, por ejemplo).

8. Configurad un servidor de archivos con OwnCloud. Comprobad la funcionalidad a través de la interfaz web y verificad la sincronización desde un dispositivo móvil con la app correspondiente.

9. Cread un Samba Active Directory Domain Controller (AD DC) y verificad su funcionamiento desde una máquina Windows.

10. Sobre Apache insertad Mod_security y probad las reglas habituales (spam, inyecciones, repetición de usuarios, etc.)

11. Utilizando tres máquinas interconectadas (con diferentes distribuciones) instalad un repositorio OCS+GLPi y haced un inventario de las tres máquinas y gestionad las actualizaciones sobre OCS+GLPi.

Bibliografía

Todas las URLs han sido visitadas por última vez en junio de 2016.

[ABench] *Apache HTTP server benchmarking tool*.
<https://httpd.apache.org/docs/2.4/programs/ab.html>

[AMod] *Apache Module Index*.
<http://httpd.apache.org/docs/current/mod/>

[AModBal] *Apache Module mod_proxy_balancer*.
<https://httpd.apache.org/docs/2.4/mod/mod_proxy_balancer.html>

[AModCach] *Apache Caching Guide*.
<https://httpd.apache.org/docs/2.4/caching.html>

[apa] **Apache HTTP** *Server Version 2.2*.
<http://httpd.apache.org/docs/2.2/>

[Apab] *Apache2 + WebDav*.
<http://www.debian-administration.org/articles/285>

[ASec] *Apache Security HandBook*.
<https://www.feistyduck.com/library/apache%2dsecurity/online/>

[ASSP] *Anti-Spam SMTP Proxy Server*.
<https://sourceforge.net/projects/assp/>

[AW] *Awstats*. <http://www.awstats.org/>

[AWFull] *AWFull*. <https://launchpad.net/awffull>

[Bareos] *Bareos Reference Manual*.
<http://doc.bareos.org/master/html/bareos-manual-main-reference.html>

[BareosTut] *Bareos: Tutorial*.
<http://doc.bareos.org/master/html/bareos-manual-main-reference.html#x1-730006>

[Bareos-WebUI] *Interfaz web para Bareos*.
<http://doc.bareos.org/master/html/bareos-manual-main-reference.html#x1-440003>

[BranScripts] *Apache Bench and Gnuplot: you're probably doing it wrong. Brad Landers*.
<http://www.bradlanders.com/2013/04/15/apache-bench-and-gnuplot-youre-probably-doing-it-wrong/>

[BXCA] *How to Create SSL Certificates*
<https://campus.barracuda.com/product/campus/article/display/CP/30114587/>

[Deb] **Debian.org**. *Debian Home*.
<http://www.debian.org>

[DebSpam] *DebianSpamAssassin*.
<https://wiki.debian.org/DebianSpamAssassin>

[DigOcCach] **Justin Ellingwood**. *How To Configure Apache Content Caching on Ubuntu 14.04*.
<https://www.digitalocean.com/community/tutorials/how-to-configure-apache-content-caching-on-ubuntu-14-04>

[DigOcModS] **Jesin A**. *How To Set Up mod_security with Apache on Debian/Ubuntu*.
<https://www.digitalocean.com/community/tutorials/how-to-set-up-mod_security-with-apache-on-debian-ubuntu>

[exim] *Exim*.
<http://www.exim.org/docs.html>

[GLPi] *Information Resource-Manager and Administration-Interface*.
<http://www.glpi-project.org/spip.php?lang=en>

[GurJm] *Free Jmeter Tutorial*.
<http://www.guru99.com/jmeter-tutorials.html>

[HandB] *Servicios de red: Postfix, Apache, NFS, Samba, Squid, LDAP*
<http://debian-handbook.info/browse/es-ES/stable/network-services.html>

[I2P] *The Invisible Internet Project*. <https://geti2p.net/en/>

[IET] **IETF**. Repositorio de Request For Comment desarrollados por Internet Engineering Task Force (IETF) en el Network Information Center (NIC). <http://www.ietf.org/rfc.html>

[Ired] *Install iRedMail on Debian*.
<http://www.iredmail.org/docs/install.iredmail.on.debian.ubuntu.html>

[Ired-DNS] *Setup DNS records for your iRedMail server*.
<http://www.iredmail.org/docs/setup.dns.html>

[Ired-IMAP] *Configure mail client applications*.
<http://www.iredmail.org/docs/index.html#configure-mail-client-applications>

[Ired-Relay] *How-to: iRedmail with optional per-user freemail-addresses and relay*. <http://www.iredmail.org/forum/topic3474-iredmail-support-howto-iredma>

[ITE] **Instituto de Tecnologías Educativas**. *Redes de área local: Aplicaciones y Servicios Linux*.
<http://www.ite.educacion.es/formacion/materiales/85/cd/linux/indice.htm>

[JBF] *JBroFuzz web application fuzzer*.
<https://www.owasp.org/index.php/JBroFuzz>

[Jmet] *Jmeter*. <http://jmeter.apache.org/index.html>

[JmUM] *Jmeter User's Manual*.
<http://jmeter.apache.org/usermanual/index.html>

[LabRat] *OWASP Live CD Project*.
<https://www.owasp.org/index.php/Category:OWASP_Live_CD_Project/es>

[ModSec] *Mod_Security*. <https://www.modsecurity.org/about.html>

[ModSRef] *Mod_security Reference manual*.
<https://github.com/SpiderLabs/ModSecurity/wiki/Reference-Manual>

[ModSBook] *ModSecurity HandBook*.
<https://www.feistyduck.com/library/modsecurity%2dhandbook%2dfree/online/>

[Mou] **Mourani, G.** (2001). *Securing and Optimizing Linux: The Ultimate Solution*. Open Network Architecture, Inc.
<http://www.tldp.org/LDP/solrhe/Securing-Optimizing-Linux-The-Ultimate-Solution-v2.0.pdf>

[OC] Access, Sync and Share Your Data, Under Your Control.
<https://owncloud.org/features/>

[OCS] *OCS Inventory NG Documentation Project*.
<http://wiki.ocsinventory-ng.org/index.php?title=Main_Page>

[OCS-Dev] *OCS Inventory Developers*.
<https://launchpad.net/ocsinventory>

[OProj] *OWASP LiveCD Education Project*.
<https://www.owasp.org/index.php/Category:OWASP_LiveCD_Education_Project>

[Owasp] *Use of Web Application Firewalls*.
<https://www.owasp.org/index.php/Category:OWASP_Best_Practices:_Use_of_Web_Application_Firewalls>

[pki] **PKI** *Public-key cryptography*.
<http://en.wikipedia.org/wiki/Public_key_cryptography >

[pkicry] **Christof Paar, Jan Pelzl** *Introduction to Public-Key Cryptography*.
<http://wiki.crypto.rub.de/Buch/movies.php >

[procmail] *ProcMail*.
<http://www.debian-administration.org/articles/242>

[psocks] *Proxy SOCKS*.
<http://en.flossmanuals.net/bypassing-es/proxis-socks/ >

[PVan] *Patrick van Kouteren. Monitoring Apache with Supervisord.* <http://www.vankouteren.eu/blog/2014/09/monitoring-apache-with-supervisord/>

[s40] *Wiki - Samba*.
<https://wiki.samba.org >

[s41] *RSAT. Remote Server Administration Tools on a Windows workstation*.
<https://wiki.samba.org/index.php/Installing_RSAT_on_Windows_for_AD_Management>

[s42] **M. López, C. Alonso***Samba 4: Controlador Active Directory*.
<http://waytoit.wordpress.com/2013/05/12/samba-4-controlador-active-directory-parte-1-de-3/ >

[s43] **M. Rushing**. *Compiling Samba 4 on Debian Wheezy - Active Directory Domain Controllers.*.
<http://mark.orbum.net/2014/02/22/compiling-samba-4-on-debian-wheezy-active-directory-domain-controllers-ho/ >

[s44] *Guía Samba4 como Controlador de Dominio y Directorio Activo*.
<http://fraterneo.wordpress.com/2013/08/19/guia-samba4-como-controlador-de-dominio-y-directorio-activo-actualizacion/ >

[Sam] *Samba Active Directory Domain Controller*.
<https://wiki.samba.org/index.php/Setup_a_Samba_Active_Directory_Domain_Controller>

[SerMail] *Install Postfix*.
<http://www.server-world.info/en/note?os=Debian_8&p=mail>

[SerWorld] *Mail Log Analyzer : AWstats*.
<http://www.server-world.info/en/note?os=Debian_8&p=mail&f=9>

[socks] **Greg Ferro** *Fast Introduction to SOCKS Proxy*.
<http://etherealmind.com/fast-introduction-to-socks-proxy/ >

[squid] **Squid** *Proxy Server*.
<http://www.squid-cache.org/ >

[squide] **Squid** *Configuration Examples*.
<http://wiki.squid-cache.org/ConfigExamples >

[StartSSL] *Creating a TLS encryption key and certificate. Christoph Haas*.
<https://workaround.org/ispmail/jessie/create-certificate>

[Sup] *Supervisor. A Process Control System*.
<http://supervisord.org/index.html>

[SW] *Server-World*.
<http://www.server-world.info/en/note?os=Debian_8>

[tinyca] **Magnus Runesson** (2007). *Create your own CA with TinyCA2*.
<http://theworldofapenguin.blogspot.com.es/2007/06/create-your-own-ca-with-tinyca2-part-1.html>

[Tor] *Tor Project*. <https://www.torproject.org/>

[tproxy] **Kiracofe, D.** *Transparent Proxy with Linux and Squid mini-HOWTO -obs:EOL pero interesante en conceptos-*.
<http://tldp.org/HOWTO/TransparentProxy.html#toc1>

[TutPointJm] *jMeter Tutorial*.
<http://www.tutorialspoint.com/jmeter/index.htm>

[WA] *WebAlizer*. <http://www.webalizer.org/>

[WGoat] *WebGoat 7.0.1 Release*.
<https://github.com/WebGoat/WebGoat/releases>

[WorkMail] *ISPmail guide for Debian Jessie. Christoph Haas*.
<https://workaround.org/ispmail/jessie

[XCA] *X Certificate and key management*.
<http://xca.sourceforge.net/xca.html#toc15>

[ZAP] *Zed Attack Proxy*.
<https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project>