Figura 1. Licencia Reconocimiento (by) CC

Tabla de contenidos

Lista de figuras

Capítulo 1. Introducción

Si te interesa montarte tu propio servidor en casa para servirte de repositorio de música o películas, o para tener un control de versiones en tus programas y proyectos web, o para ser un servidor FTP donde almacenar copias de seguridad de datos importantes guardándolos desde cualquier parte del planeta, o si deseas poder leer el correo desde cualquier sitio y de forma segura, o si quieres montar un servidor web para tener la web de la asociación de perros dálmata de tu barrio, quizás este manual sea un buen punto de partida porque en él explico cómo montar un servidor doméstico con el sistema operativo libre OpenBSD.

En realidad, no será mucho más complicado si lo montas con OpenBSD o con cualquier otro sistema operativo, ya que los conceptos e ideas que vengo a presentar en el presente documento se pueden implementar y adaptar con otros sistemas de tipo UNIX.

Objetivo de este manual

Este manual va dirigido a todas aquellas personas que pretendan crear un servidor doméstico. En mi casa tengo una máquina con el sistema operativo OpenBSD llamada *nuvolet* que me brinda muchos servicios 24x7 y también la uso como plataforma donde guardar copias de respaldo que lanzo con Rsync. A través de estas páginas iré contando mi experiencia y las configuraciones que he realizado para el servidor de mi casa: nuvolet.strangled.net

Es posible que de vez en cuando vaya actualizando este manual a medida que añada o implemente nuevas funcionalidades y servicios al servidor de mi casa. Así pues, al mismo tiempo que comparto conocimiento e ideas, este manual me sirve para guardarme anotaciones de las configuraciones que realizo en mi servidor.

Capítulo 2. Manos a la obra

Comencemos a planificar un poco lo que pretendemos hacer conociendo los pros y los contras.

Comenzando, ¿qué es OpenBSD?

OpenBSD es un sistema operativo libre tipo Unix, multiplataforma, basado en 4.4 BSD, Es un descendiente de NetBSD con un foco especial en la seguridad y la criptografia.

Este sistema operativo se concentra en la portabilidad, cumplimiento denormas y regulaciones, corrección, seguridad proactiva y criptografía integrada. OpenBSD incluye emulación de binarios para la mayoría de los programas de los sistemas Solaris, FreeBSD, Linux, BSD/OS sunOs y Hp-UX

Se creó como una variante de la famosa NetBSD debido, según leo en la wikipedia, a diferencias filosóficas entre Theo de Raadt y el resto del equipo de NetBSD. Dejando aparte el hecho de que la seguridad sea la principal razón para que OpenBSD exista, el proyecto también tiene otras metas. Siendo un descendiente de NetBSD, es un sistema operativo muy portable, de hecho funciona en 17 plataformas distintas de hardware. Así que sin duda funcionará en tu netbook, en tu antiguo servidor o en cualquier máquina virtual que montes con Vmware o VirtualBox.

La filosofía de OpenBSD es: “Free, Functional and Secure” (Libre, Funcional y Seguro). Libre, porque su licencia mantiene el espíritu original del copyright libre original de Berkeley Unix, funcional porque siempre se publica una versión que sea absolutamente estable lista para instalar en producción y segura porque tiene una extrema revisión y supervisión del código.

A día de hoy, cuando escribo este documento, la versión actual de OpenBSD es la 5.1, que fue publicada el pasado 1 de mayo de 2012. Debes saber que desde sus inicios, OpenBSD publica una versión cada 6 meses, por lo tanto la nueva versión estará lista para hacerse pública en noviembre de 2012.

La web oficial de OpenBSD es http://www.openbsd.org y la página de OpenBSD en la Wikipedia está en http://es.wikipedia.org/wiki/OpenBSD

¿Por qué montar un servidor con OpenBSD ?

Esta es sólo una cuestión personal. Por supuesto, también puedes montar un servidor con alguna distribución de GNU/Linux o puedes instalar un XAMPP sobre Windows o Mac OS X y tener tu servidor MySQL + Apache + PHP funcionando muy fácilmente, hay muchas opciones en la actualidad, de eso no hay dudas.

Lo que pasa es que yo adoro la seguridad, la estabilidad y la libertad, y eso lo obtengo fácilmente con OpenBSD.

No pretendo crear ninguna discusión ya que lo que voy a decir es únicamente mi opinión personal basada en mi experiencia. Verás, empecé a programar pascal y Object Pascal con Delphi en Windows 3.11, luego trabajé muchos años con Windows 95 y los que vinieron. NT me gustaba mucho. Millenium fue un fiasco total. Años más tarde, me compré un powerbook con Mac OS X y disfruté mucho tiempo con él y con otro macbook que compré hace cinco años, y llevo más de 10 años con GNU/Linux como sistema de escritorio (empecé con Slackware y Suse, luego Red hat un tiempo, luego Debian y desde hace 3 años con Ubuntu), sin embargo nada es perfecto. A mi lo que no me gusta del Mac OS X, a parte de que es privativo, es que cada versión que sacan es más restrictiva que la anterior y te cuelan muchas aplicaciones que ni uso y ni estoy interesado en usarlas. Lo que no me gusta de Windows, a parte de que es privativo, es que cada versión que sacan lo cambian todo y lo renombran todo, y no sé como hacer las cosas que antes ya sabía. Ahora acaban de sacar el Windows 8 y lo instalé en un portátil para probarlo y, en serio, es espantoso. Y

ya para acabar, una cosa que no me gusta mucho de las distros GNU/Linux es que cada distribución hace las cosas de una manera, mantiene distintos repositorios y algunas veces sistemas de paquetes. E incluso ahora, en la última Ubuntu 12.10, hay un aplicativo que permite buscar y comprar en Amazon. Me parece bastante feo esto. Sería como si ahora compras Debian GNU/Linux y ves en el escritorio unos iconos de el Corte Inglés y de Mercadona. Sería ... mmm... ¿cómo se dice...?... mmmm... ¿spam?

En cambio, desde que probé OpenBSD allá por el año 2003 cuando vivía en Barcelona, veo que todo está donde tiene que estar, todo se administra con la filosofía Unix y todo funciona bien. Además está todo muy bien documentado. Me gusta como servidor seguro.

Dicho esto, debo decir que usar GNU/Linux o algún BSD (NetBSD, OpenBSD, FreeBSD u otros) es una elección personal. Para mi no hay buenos ni malos, simplemente algunos nos convencen más y otros menos.

Inconvenientes

Pues sí, el hecho de montar un servidor doméstico en tu casa te generará posiblemente algunos inconvenientes, y muchos dolores de cabeza. Estás avisado. A continuación te dejo una lista de los inconvenientes que yo puedo ver si te decides por montarte un servidor propio en tu casa para darte algunos servicios 24x7:

- Poco ancho de banda de subida, si lo comparamos con una gran empresa de Hosting.
- Es posible que algunas veces vaya lento (sobre todo java o acceso a bases de datos grandes).
- Poca capacidad de respuesta ante picos inesperados de tráfico.
- Tendremos menos disponibilidad del ancho de banda de casa, por ejemplo, para usar P2P.
- No es recomendable tener un ordenador 24x7 en casa ya que el cable puede derretirse o sobrecalentarse generando un cortocircuito o causar un incendio.
- Preocupación psicológica durante todo el día de que el servidor esté dando un servicio correcto sin fallos.
- Si se va la luz de casa nos quedamos en bragas.
- El disco duro tiende a petar al cabo de un tiempo. Comprar uno nuevo cuesta sobre los 70 euros.
- Si hay cualquier otra avería del hardware del equipo, debemos actuar rápido ya que todos los servicios estarán parados.
- Es necesario invertir tiempo en la administración del servidor.

Beneficios

Aquí también dejo los beneficios que obtengo al montar mi propio servidor.

- Aprender.
- Mejorar.
- Compartir lo aprendido con la Comunidad.

Hay un proverbio famoso que me gusta mucho del filósofo chino Confucio (551 a.C. - 479 a.C.) que dice así: “Oigo y olvido. Veo y aprendo. Hago y entiendo.”

Capítulo 3. Instalación y puesta en marcha

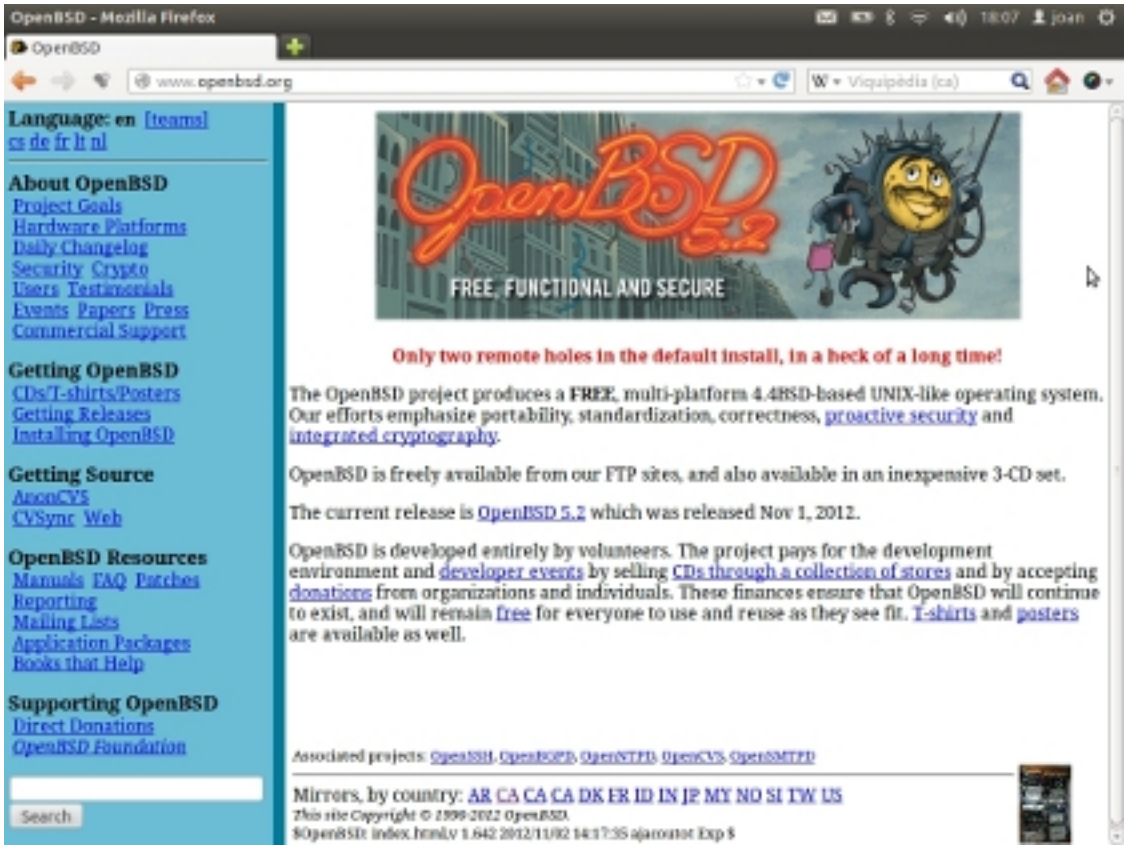
En este apartado voy a explicar, a la velocidad de la luz, cómo puedes descargar la última versión de OpenBSD y poner en funcionamiento tu servidor.

Obtención e instalación

Aquí no voy a explicar paso a paso cómo es la instalación de OpenBSD 5.1., tienes una infinidad de manuales y videotutoriales en la red. Lo único que quiero que sepas es que debes descargar o comprar un CD de OpenBSD o descargarlo libremente desde alguno de los mirrors que verás en http://www.openbsd.org/ftp.html.

Descarga install51.iso y quema algún CD que tengas por casa, y manos a la obra... ¡a instalar!

Figura 3.1. Web oficial de OpenBSD, http://openbsd.org



Una instalación de OpenBSD, al contrario de lo que puedas imaginar, es una tarea que acaba en 10 minutos. Lo que pasa es que posteriormente vas a tener que instalar Postfix - por ejemplo - y comenzar a configurarlo.

Sugerencia

Es de vital importancia que aprendas a instalar correctamente OpenBSD, a configurar el teclado, a descargar los paquetes desde FTP o HTML y que domines el particionado de los discos. Y esto no está detallado en este manual.

Puesta en marcha

Tras la instalación te dirá que ya tienes el sistema operativo instalado. Deberás reiniciar el servidor y entonces ya tienes OpenBSD totalmente operativo y ya puedes disfrutar de la experiencia OpenBSD 5.1 para ofrecer servicios desde internet. Felicidades, bébete una cerveza y disfruta del momento ;-).

Configuración de la red

Durante el proceso de instalación (no recuerdo ahora si era el tercer o cuarto paso) habrás configurado tu red, ya que el menú de instalación detecta tus interfaces de red y te pregunta si deseas configurar una ip dinámica o si deseas introducir manualmente una. Pero, si de todas formas tras tener instalado el sistema, necesitamos ambiar la configuración de la red debemos saber que la configuración de las interfaces de red se encuentra en /etc/hostname.TU_TARJETA, donde TU_TARJETA es el nombre de la interfaz, en OpenBSD dependen del fabricante de la tarjeta de red, por lo tanto en cada ordenador es diferente.

Para saber cómo se llaman nuestras tarjetas ejecutamos la orden ifconfig tal que así:

```
bash-4.2# ifconfig -a
lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> mtu 33196
        priority: 0
        groups: lo
        inet6 ::1 prefixlen 128
        inet6 fe80::1%lo0 prefixlen 64 scopeid 0x4
        inet 127.0.0.1 netmask 0xff000000
msk0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        lladdr 00:13:a9:4f:83:8f
        priority: 0
        groups: egress
        media: Ethernet autoselect (100baseTX full-duplex,txpause)
        status: active
        inet6 fe80::213:a9ff:fe4f:838f%msk0 prefixlen 64 scopeid 0x1
        inet 192.168.1.36 netmask 0xffffff00 broadcast 192.168.1.255
ath0: flags=8822<BROADCAST,NOTRAILERS,SIMPLEX,MULTICAST> mtu 1500
        lladdr 00:19:7d:9c:6d:2a
        priority: 4
        groups: wlan
        media: IEEE802.11 autoselect
        status: no network
        ieee80211: nwid ESSID wpakey 0xda5cbd95 wpaprotos wpa1,wpa2 psk tkip
enc0: flags=0<>
        priority: 0
        groups: enc
        status: active
vlan0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        lladdr 00:13:a9:4f:83:8f
        priority: 0
```

```
vlan: 1 parent interface: msk0
groups: vlan
status: active
inet6 fe80::213:a9ff:fe4f:838f%vlan0 prefixlen 64 scopeid 0x5
```

Configurar una IP estática en nuestra máquina

Como puedes ver, aquí nos salen varias interfaces: la interfaz de 'loopback' (lo0), la Ethernet (msk0) y un par de interfaces wifi.

En mi caso, como tengo el servidor 'nuvolet' conectado con un cable RJ45 directamente al router a través de la conexión Ethernet msk0, voy a configurar la red en ese dispositivo con una IP fija.

Para ello, creamos un fichero en /etc/ llamado hostname.msk0 que contendrá la dirección IP y la máscara de subred:

```
inet 192.168.1.2 255.255.255.0
```

Configurar una red dinámica en nuestra máquina

Si al contrario que en el apartado anterior, queremos que nuestra máquina OpenBSD adquiera una dirección dinámica que le dé el router u otra máquina, configuraremos nuestro fichero hostname.msk0 con la siguiente línea:

```
dhcp
```

Así, al arrancar OpenBSD le pedirá una ip aleatoria al servidor DHCP.

Configurar el nombre de tu máquina

El nombre de tu máquina lo puedes configurar simplemente añadiéndolo en el fichero */etc/myname*. No es necesario decir que te asegures de configurar bien tu dominio o subdominio y que apunte a la IP de tu casa, ya sea fija o dinámica (en mi caso es fija, por eso no comento nada de los servicios de IP's dinámicas):

```
$ cat /etc/myname
nuvolet.strangled.net
$
```

Importante

Busca en los manuales oficials de OpenBSD si tienes algún problema con la configuración de tu red, con tu pasarela, con la conexión Wifi, etc.

Capítulo 4. Ajustes iniciales

En los ajustes iniciales instalaremos BASH, emacs, wget y unzip con paquetes. Para ello, primero es fundamental explicar cómo se instalan, eliminan y actualizan paquetes en OpenBSD. Para ello, lee el siguiente apartado dónde sólo explciaré cómo gestionar paquetes pero no hablaremos de qué son los ports y cómo se instalan.

Instalación y gestión de paquetes

Para instalar paquetes, lo primero que haremos sreá añadir el “pkg_path” en el .profile de nuestro super-usuario:

```
export PKG_PATH=ftp://ftp.openbsd.org/pub/OpenBSD/5.0/packages/i386/
```

Ahora, ya podemos instalar o actualizar paquetes. Te mostraré las principales órdenes que ejecutaremos a la hora de administrar los paquetes en una máquina con OpenBSD:

- Instalación: *pkg_add*
- Instalación interactiva: *pkg_add -i PAQUETE*
- Instalación con verbose:*pkg_add -v PAQUETE*
- Actualización de un paquete: *pkg_add -u PAQUETE*

Procedemos a instalar algunos paquetes

OpenBSD lleva la shell *Ksh* por defecto, pero algunas personas que venimos del sistema GNU/Linux estamos más familiarizados con Bash y ya conocemos muchos trucos y atajos de teclado usando esta shell.

Entonces, vamos a instalar la shell Bash para comenzar a tener un entorno que se ajuste a nuestros gustos. Para ello ejecutaremos la siguiente orden:

```
pkg_add bash
```

Y a continuación, para tener la shell Bash predefinida por defecto ejecutaremos:

```
chsh -s bash
```

A continuación, instalamos el wget:

```
pkg_add wget-1.12p1.tgz
```

que también podríamos instalarlo apuntando al paquete directamente del repositorio:

```
pkg_add ftp://ftp.openbsd.org/pub/OpenBSD/4.9/packages/i386/
```

Ahora instalaremos eMacs:

```
pkg_add emacs-23.4-no_x11
```

Y por último instalaremos el unzip:

```
pkg_add unzip-6.0
```

En realidad, la experiencia hace que te hagas práctico en esta serie de tareas de administración e instalar múltiples paquetes a la vez:

```
pkg_add bash unzip emacs-23.4-no_x11 unzip-6.0
```

Y un truco que yo hago es descargarme el listado de todos los paquetes de OpenBSD y luego puedo hacer búsquedas con grep dentro de ese fichero, al igual que hacia en Debian GNU/Linux usando *apt-cache seach PROGRAMA*. Para ello, me lo descargo con:

```
wget ftp://ftp.openbsd.org/pub/OpenBSD/4.9/packages/i386/index.txt
```

Y ahora busco un programa escribiendo:

```
grep PAQUETE index.txt
```

Una vez hemos aprendido a instalar aplicaciones en OpenBSD, ya podemos instalar resto de aplicaciones que utilizaremos en nuestro servidor: mutt para leer el correo, nmap para escaneos de puertos, fetchmail para recoger el correo de otros servidores remotos, subversion para el control de versiones en nuestros scripts, el irssi para conectar a canales IRC y el genial screen:

```
pkg_add mutt-1.5.21p0v0 nmap fetchmail nano screen-4.0.3p2
       subversion-1.7.2 irssi-0.8.15p2
```

Capítulo 5. Centralización de datos en una red mediante NFS

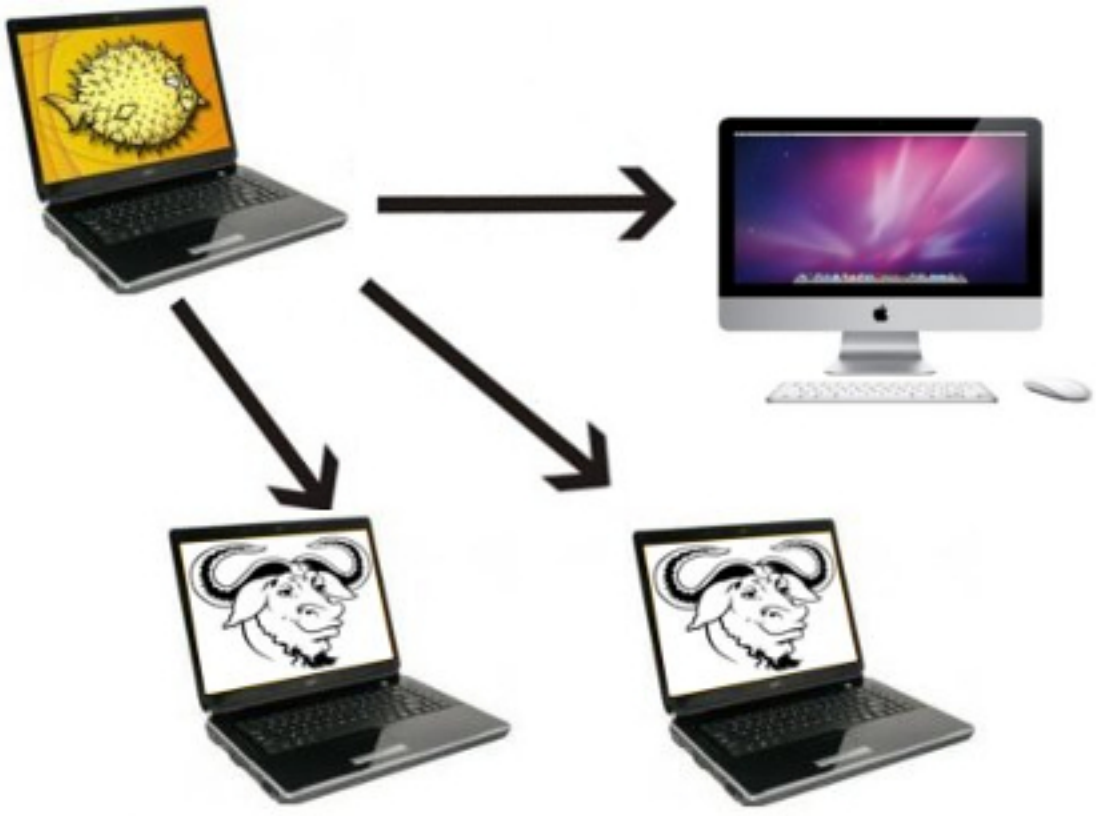
En este apartado vamos a aprender qué es NFS y cómo podemos tener todos los datos de una red (documentos, imágenes, videos, programas, etc) centralizados en un único equipo que compartirá la información con todos los ordenadores que forman la red. Al principio supone perder algo de tiempo analizando requerimientos y necesidades de tu red y de tus usuarios, configurar y realizar las pertinentes pruebas hasta que todo quede como quieres. Pero al final, tendrás menos dolores de cabeza ya que todos los datos estarán en un único directorio de un único ordenador, y las copias de seguridad serán más fáciles de gestionar.

¿Pero qué es eso de NFS?

El *«Sistema de Archivos de Red»* (NFS, Network File System, en la lengua de Shakespeare) se usa para compartir un sistema de archivos en una red. El objetivo es poder trabajar con la misma información trabajes en el ordenador que trabajes, ya que sólo van a montar los directorios mediante NFS pero en realidad los datos están en mi servidor con OpenBSD (es lo mismo que lo que hoy en día llaman "en la nube").

Imagina una oficiona con 2 trabajadores, *Fulanito* y *Menganito*. Ambos tendrán en el escritorio de su ordenador los informes de la empresa, la contabilidad en sus hojas de cálculo, manuales, etc. Y si ahora *Menganito* tiene que bajar a otra planta del eficio y accede a otro ordenador, seguirá teniendo los mismos directorios en su escritorio: los informes de la empresa, la contabilidad en sus hojas de cálculo, los manuales, etc. Esto se consigue a partir del NFS.

Figura 5.1. Centralización de archivos en una LAN con NFS



Como puedes ver en la imagen anterior, en mi casa dispongo de:

- Un servidor (nuvolet) aparcado en una estantería entre mis libros
- Mi portátil Lenovo con Ubuntu Linux
- Un portátil antiguo Dell que no uso nunca con Ubuntu Linux
- Un iMac con OS X Leopard

Desde cualquiera de estos ordenadores, puedo acceder a mis documentos importantes, a las fotos de mis viajes, a mis películas, a mi música, etc. Y no necesito tener la información duplicada ni nada parecido... el iMac usa MacOsX y si quiero escuchar música, lo normal es meterla también en el directorio del iTunes y tal, porque Apple lo quiere así. Pero a mi no me gusta esta manera, y tengo toda mi música en un único fichero centralizado para que, sea cualquiera el ordeandor, pueda añadir, eliminar o escuchar música.

Ventajas de usar NFS

Las ventajas de usar el servicio NFS para centralizar información en nuestra red son:

- Es un protocolo libre y transparente que muchos sistemas Unix y la mayoría de distribuciones Linux incluyen por defecto.
- Permite almacenar los archivos home de todos los usuarios de una red en un único servidor. Así evitamos problemas y duplicidades de archivos.

- Al unificar todas las cuentas de usuarios o todos los directorios de trabajo, reducimos costos notablemente.
- Facilita la disponibilidad de los datos, ya que facilita las copias de seguridad en el servidor al estar todo centralizado.
- La versión 4 de NFS es especialmente segura en la gestión de recursos ya que soporta Kerberos, lista de control de acceso (ACLs) y permite trabajar bajo cortafuegos. Esto nos asegura que los datos almacenados en el servidor NFS son accedidos por quien nosotros realmente queremos.
- Ahora también Windows Server (2003/2008/2012) también permite hacer tando de servidor NFS como de cliente. Esto permite que sistemas Windows puedan acceder también a datos almacenados en sistemas Unix y por supuesto, simplificar el esfuerzo de una posible migración de Windows Server a un servidor Unix y viceversa, asegurando la continuidad en la disponibilidad de recursos y aplicaciones

Configuración del servidor

Para configurar el NFS en mi servidor con OpenBSD y permitir puntos de montaje a las máquinas lenovo, dell e iMac debemos tener activados los siguientes servicios en el fichero /etc/rc.conf:

```
portmap
nfsd -u -t -n 4
mountd
```

A continuación pongo la configuración del fichero /etc/exports del servidor:

```
$ cat /etc/exports
# $OpenBSD: exports,v 1.2 2002/05/31 08:15:44 pjanzen Exp $
#
# NFS exports Database
# See exports(5) for more information.  Be very careful:  misconfiguration
# of this file can result in your filesystems being readable by the world.

/home/joan/documents -alldirs -maproot=root 192.168.1.33
/var/www/htdocs -alldirs -maproot=root 192.168.1.33
```

Como puedes ver, voy a compartir el fichero 'documents' del usuario *joan* y el directorio 'htdocs' de Apache. De tal manera, cualquier cliente podrá programar las páginas webs fácilmente, sin necesidad de conectarse por FTP o SSH, ya que dispondrán de los ficheros php, css, etc.. en sus ordenadores.

Si hacemos cambios sobre el fichero /etc/exports y queremos ejecutarlos al momento tenemos dos opciones: o reiniciar el servidor o matar mountd con *"kill -9 nº_de_proceso"* y ejecutarlo de nuevo.

Configuración de los clientes

Para montar manualmente los directorios 'documents' i 'htdocs' del servidor en los clientes, podemos ejecutar estos dos comandos:

```
sudo mount -t nfs 192.168.1.2:/var/www/htdocs /home/joan/web
sudo mount -t nfs 192.168.1.2:/home/joan/documents /home/joan/documents
```

Donde 192.168.1.2 es la ip del servidor y el último parámetro /home/joan/web, por ejemplo, es el destino en donde montaremos el directori 'htdocs' del servidor.

Si no queremos ejecutar estos comandos cada vez, sino que deseamos que se monten automáticamente tras el arranque de nuestros equipos, añadiremos la configuración en el /etc/fstab del cliente (Ubuntu Linux, en mi caso) para que estos directorios se monten automáticamente en el arranque del sistema (¡siempre que el servidor nuvolet esté arrancado!). Así que añadiremos:

```
#Muntem via NFS el directori /var/www/htdocs del servidor nuvolet
192.168.1.2:/var/www/htdocs /home/joan/web nfs rw 0 0

#Muntem via NFS el directori /home/joan/documents del servidor nuvolet
192.168.1.2:/home/joan/documents /home/joan/documents nfs rw 0 0
```

Ahora ya podemos reiniciar el cliente y veremos cómo se montan automáticamente nuestros recursos vía NFS.

Figura 5.2. Directorio montado con NFS en el GNU/Linux

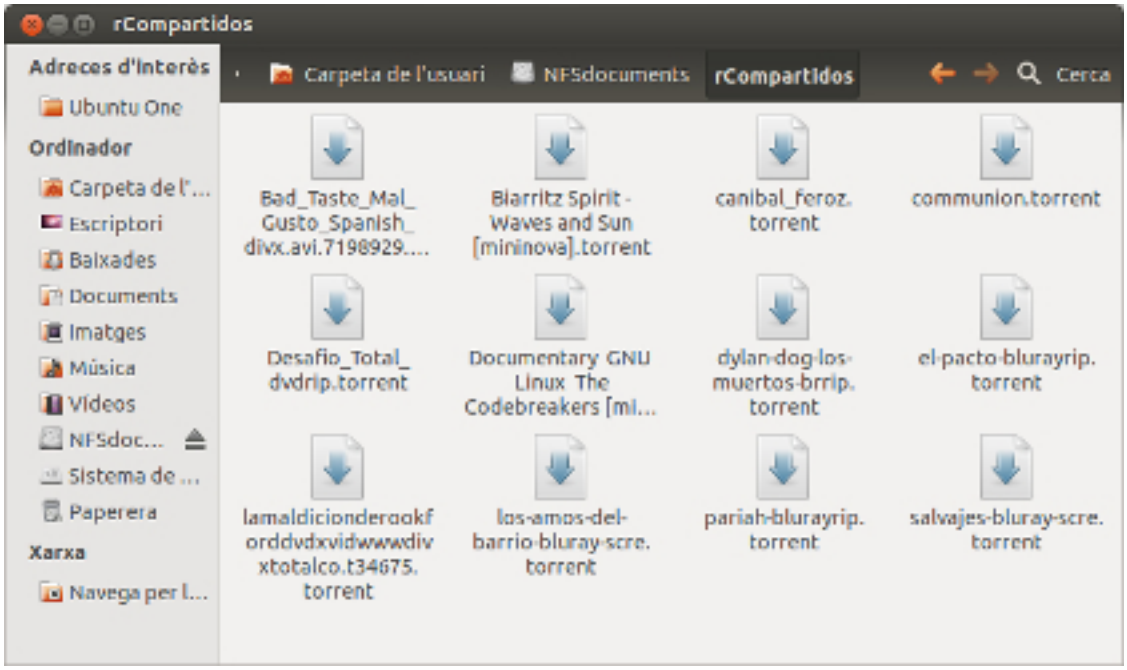
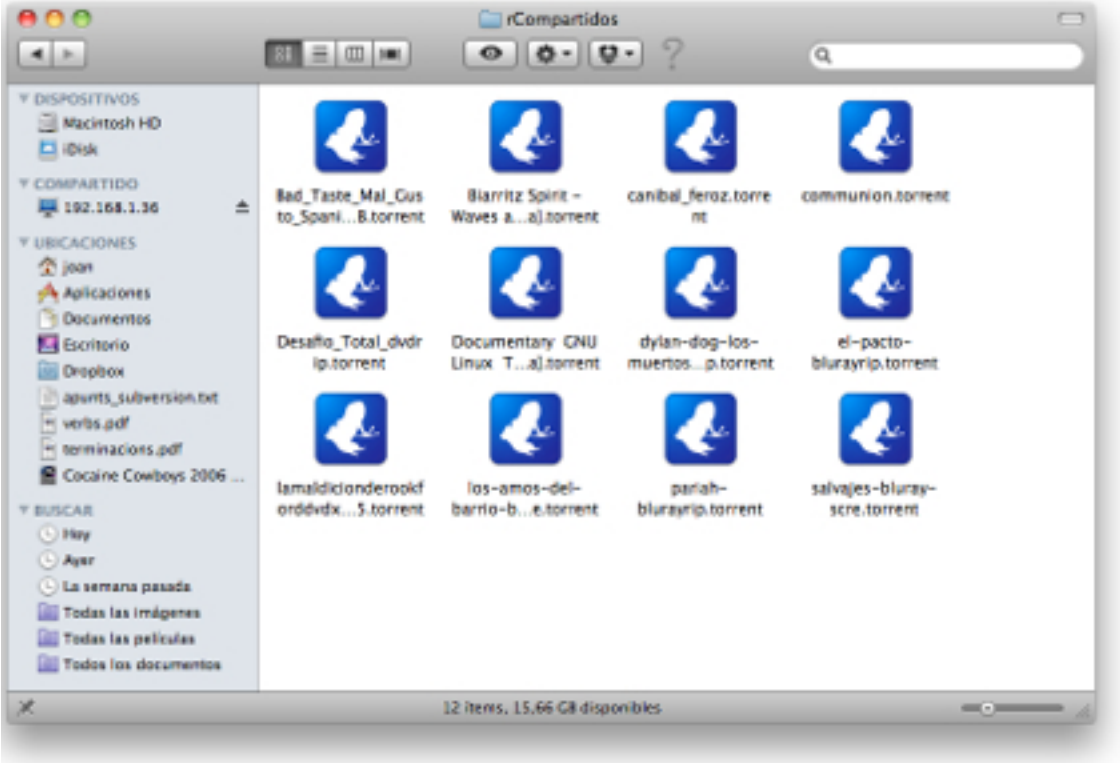


Figura 5.3. Directorio montado con NFS en el MacOS X



Finalmente, como puedes ver, ya puedo trabajar con el portàtil y parece como que los ficheros están en local, pero en realidad estoy trabajando en mi servidor doméstico nuvolet y hacer las cosas que hago siempre pero con la seguridad de que lo tengo todo centralizado en un servidor que funciona 24x7 y que tiene programadas copias de seguridad de forma regular.

¿Problemas con NFS?

Si experimentas algún problema puedes primero cerciorarte, desde tu cliente, que el servidor NFS está funcionando y compartiendo algún (algunos) recurso (s) compartido (s) con el comando showmount:

```
joan@lenovo:~$ showmount -e 192.168.1.2
Export list for 192.168.1.2:
/home/joan/documents (everyone)
/var/www/htdocs (everyone)
joan@lenovo:~$
```

Puedes comprobar que los servicios mount, portmap y NFS estén corriendo en un servidor con el comando rpcinfo como te muestro a continuación:

```
joan@lenovo:~$ rpcinfo -p nuvolet.strangled.net
   program vers proto   port  service
    100000    2   tcp    111  portmapper
    100000    2   udp    111  portmapper
    100003    2   udp   2049  nfs
```

```
    100003    3   udp   2049  nfs
    100005    1   udp    776  mountd
    100005    3   udp    776  mountd
    100005    1   tcp    763  mountd
    100005    3   tcp    763  mountd
joan@lenovo:~$
```

También podríamos hacer, antes que nada, un escaneo de puertos para saber si el servidor tiene los puertos que nos interesan abiertos:

```
nmap -O ip-del-servidor
```

Montar un servidor de datos mediante NFS para que nuestros ordenadores monten los directorios o particiones es bastante sencillo de realizar, y debemos tener en cuenta siempre la seguridad valorando qué directorios vamos a compartir y, lo más importante, a quién. Por este motivo, yo aconsejo acotar los directorios especificando redes concretas o IP's concretas de máquinas, ya que de lo contrario esos servidores estarán abiertos a todo el mundo.

Podemos comprobar los puntos demontaje NFS que tenemos en nuestro ordenador con el comando df:

```
joan@lenovo:~/NFSdocuments$ df -h
S. fitxers                          Mida En ús Lliure  %Ús Muntat a
/dev/sda1                            70G   38G    29G  57% /
udev                                2,0G  4,0K   2,0G   1% /dev
tmpfs                               791M  848K   790M   1% /run
none                                5,0M     0   5,0M   0% /run/lock
none                                2,0G  588K   2,0G   1% /run/shm
192.168.1.2:/home/joan/documents   21G  4,6G    15G  24% /home/joan/NFSdocuments
192.168.1.2:/var/www/htdocs   21G  4,6G    15G  24% /home/joan/webs
```

Si quieres más información técnica y directa sobre NFS, lee el manual http://www.openbsd.org/faq/es/faq6.html#NFS.

Capítulo 6. Instalación y configuración del servidor web

OpenBSD tiene Apache preinstalado como parte del sistema, pero no lo tiene activado. Para activar Apache y que se inicie tras el arranque (el boot time) editaremos el fichero /etc/rc.conf/ y modificamos la línea donde dice: *"http_flags=NO"* y la dejamos así:

```
use -u to disable chroot, see httpd(8)
httpd_flags=""          # for normal use: "" (or "-DSSL" after reading ssl(8))
```

Si reinicias tu servidor, ya puedes poner en un navegador http://tu-dirección-ip y verás que ya tienes el servidor Apache funcionando. Lo que pasa es que aún no entiende PHP, así que vamos a proceder a instalar y configurar el PHP.

Instalación y configuración de PHP

Vamos a instalar PHP, le daremos la orden:

```
pkg_add php-5.3.10
```

En pocos minutos ya tendremos el PHP instalado. Seguidamente tenemos que hacer el enlace *ln -sf* para que funcione el PHP con Apache.

Tras la instalación del PHP verás que el sistema OpenBSD te recomienda que pongas un enlace haciendo un enlace simbólico para tener habilitado y configurado el PHP5. Entonces, le haremos caso y ejecutaremos ese comando:

```
ln -s /var/www/conf/modules.sample/php-5.3.conf  /var/www/conf/modules/php.conf
```

Y luego nos dice que la configuración de PHP se ha instalado en /etc/php-5.3.ini, así que es aquí donde realizaréis los cambios en el PHP. Necesitamos asegurarnos de que la siguiente línea se encuentre descomentada en /var/www/conf/httpd.conf (primero le damos permisos de escritura con # chmod o+w /var/www/conf/httpd.conf):

```
AddType application/x-httpd-php .php
```

También que la línea de DirectoryIndex tenga lo siguiente:

```
DirectoryIndex index.html index.php
```

Ahora necesitamos instalar las librerías "Graphics Draw" en el PHP, que son unas librerías de código abierto desarrolladas en C para la creación dinámica de imágenes en aplicaciones. Entre otros muchos formatos, permiten manipular al vuelo imágenes PNG, JPEG y GIF.

Para instalar estas librerías ejecutamos:

```
pkg_add php-gd-5.3.10
```

Y el sistema nos responde esto:

```
pkg_add php-gd-5.3.10
php-gd-5.3.10:jpeg-8c: ok
php-gd-5.3.10:t1lib-5.1.2: ok
php-gd-5.3.10:png-1.5.6p0: ok
php-gd-5.3.10: ok
--- +php-gd-5.3.10 -------------------
You can enable this module by creating a symbolic
link from /etc/php-5.3.sample/gd.ini to
/etc/php-5.3/gd.ini.
ln -fs /etc/php-5.3.sample/gd.ini /etc/php-5.3/gd.ini
```

Por lo que debemos crear ese enlace:

```
ln -fs /etc/php-5.3.sample/gd.ini /etc/php-5.3/gd.ini
```

También instalaremos la librería Curl de PHP, que aunque no es necesaria por una instalación mínima de Drupal, puede que algunos módulos la requieran. Así pues, instalaremos la librería de la misma versión de nuestro PHP (en mi caso la 5.3.10):

```
bash-4.2# pkg_add php-curl-5.3.10
php-curl-5.3.10: ok
--- +php-curl-5.3.10 -------------------
You can enable this module by creating a symbolic
link from /etc/php-5.3.sample/curl.ini to
/etc/php-5.3/curl.ini.

ln -fs /etc/php-5.3.sample/curl.ini \
        /etc/php-5.3/curl.ini
```

Y ahora creamos el acceso directo que nos pide la libreríra:

```
bash-4.2# ln -fs /etc/php-5.3.sample/curl.ini /etc/php-5.3/curl.ini
bash-4.2#
```

Y ahora realizaremos lo mismo con la librería php-mcrypt:

```
bash-4.2# grep mcrypt index.txt
-rw-r--r--  1 276  125     176724 Feb  7 06:46:17 2012 libmcrypt-2.5.8p1.tgz
-rw-r--r--  1 276  125      54203 Feb  7 23:45:35 2012 mcrypt-2.6.8p1.tgz
```

```
-rw-r--r--  1 276  125      14273 Feb  7 17:52:09 2012 php-mcrypt-5.2.17p5.tgz
-rw-r--r--  1 276  125      15776 Feb  7 09:10:11 2012 php-mcrypt-5.3.10.tgz

bash-4.2# pkg_add php-mcrypt-5.3.10.tgz
php-mcrypt-5.3.10:libmcrypt-2.5.8p1: ok
php-mcrypt-5.3.10:libltdl-2.4.2: ok
php-mcrypt-5.3.10: ok
--- +php-mcrypt-5.3.10 -------------------
You can enable this module by creating a symbolic
link from /etc/php-5.3.sample/mcrypt.ini to
/etc/php-5.3/mcrypt.ini.

ln -fs /etc/php-5.3.sample/mcrypt.ini \
        /etc/php-5.3/mcrypt.ini

bash-4.2# ln -fs /etc/php-5.3.sample/mcrypt.ini /etc/php-5.3/mcrypt.ini
bash-4.2#
```

Al finalizar esta configuración vuelve a dejar los permisos como estaban antes, es decir, sin el permiso de escritura sobre este fichero:

```
chmod o-w /var/www/conf/httpd.conf
```

Aunque seguramente ya existe, debemos verificar que exista el directorio /var/www/tmp y que tenga permisos de escritura para todos los usuarios o que sea del grupo y del usuario "www" con todos los permisos. Este directorio /var/www/tmp puede ser usado por algunas aplicaciones que están en PHP (jaws por ejemplo).

En caso de que no existiese, haremos lo siguiente:

```
mkdir /var/www/tmp
chown www /var/www/tmp
chmod u+rwx /var/www/tmp
```

Ahora vamos a instalar la librería GD de php5:

```
pkg_add -i -v ftp://ftp.openbsd.org/pub/OpenBSD/4.9/packages/i386/php5-gd-5.2.17-
```

Arrancamos Apache y PHP: la hora de la verdad

En el capítulo anterior hemos habilitado Apache, que viene preinstalado en OpenBSD, y posteriormente hemos instalado y activado el PHP para poder tener páginas php dinámicas en nuestro servidor.

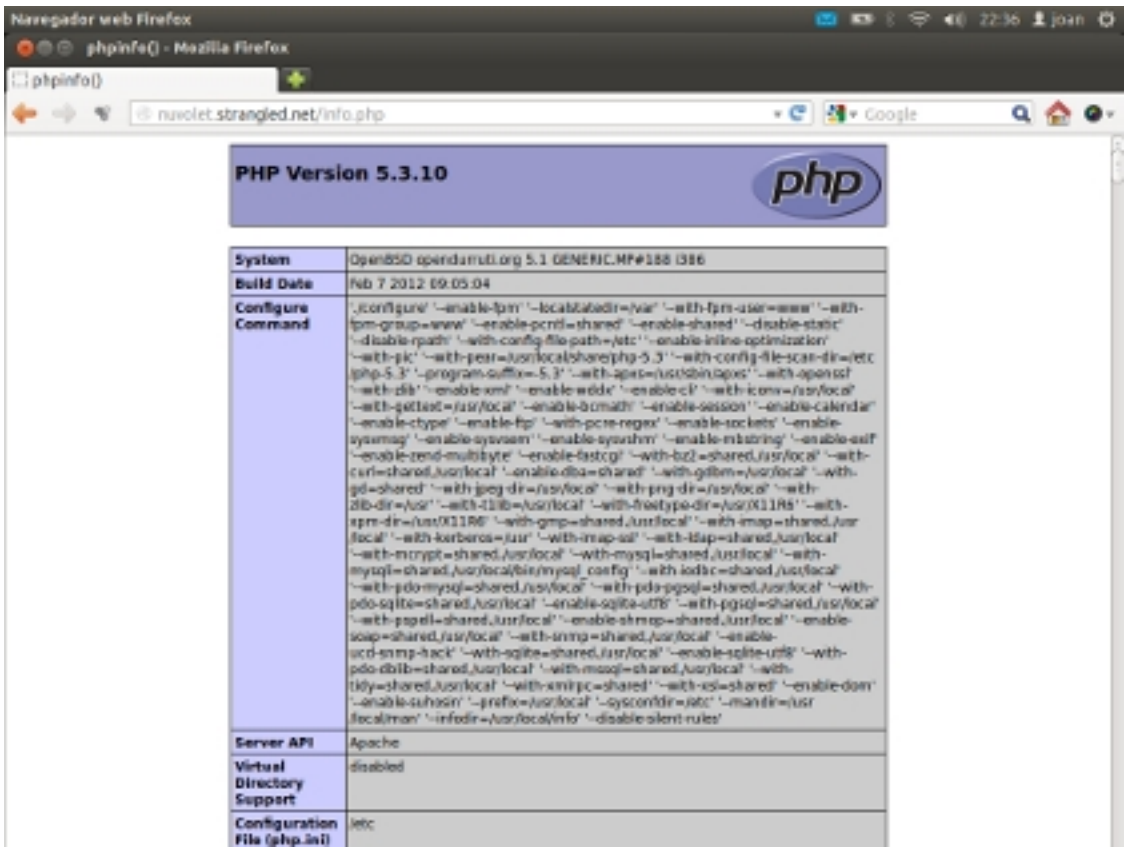
Ahora vamos a reiniciar nuestro servidor de apache con la siguiente orden:

```
apachectl stop
apachectl start
```

Podemos probar que nuestro servidor de apache ya esté funcionando, creamos un archivo test.php en el directorio de documentos de http /var/www/htdocs, y metemos en este archivo lo siguiente:

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

Figura 6.1. Ahora ya tenemos Apache y PHP configurado al 100%.



Última configuración: los Virtual Hosts

Dejo el código de ejemplo para dos dominios imaginarios: "rajoy.es" y "zapatero.es". Puedes copiarlo literalmente y copiarlo en tu *httpd.conf*:

```
<VirtualHost 192.168.1.5:80>
    ServerAdmin micorreo@correo.es
    DocumentRoot /var/www/htdocs/rajoy.es
    ServerName rajoy.est
    ErrorLog logs/rajoy.es-error_log
```

```
    CustomLog logs/rajoy.es-access_log common
</VirtualHost>

<VirtualHost 192.168.1.5:80>
    ServerAdmin micorreo@correo.es
    DocumentRoot /var/www/htdocs/zapatero.es
    ServerName zapatero.es
    ErrorLog logs/zapatero.es-error_log
    CustomLog logs/zapatero.es-access_log common
</VirtualHost>
```

Capítulo 7. Instalación y configuración de MySQL

En este capítulo aprenderemos cómo descargar MySQL y cómo configurarlo para que tengamos el servidor arrancado tras el arranque del sistema.

Instalación de MySQL

En OpenBSD 5.1, la versión de MySQL es MySQL 5.1.60. Vamos a proceder a instalar el servidor de la siguiente manera:

```
pkg_add mysql-server
```

Y a continuación tenemos que instalar un plugin para acabar de tener configurado el MySQL para que pueda conectar con nuestras páginas webs, y se trata del conector PHP-MySQL. Para ello, lo instalaremos de la siguiente manera:

```
pkg_add php-mysql-5.3.10
```

RECUERDA: cuando instales esto, recuerda que después debes hacer los enlaces “ln -s” para que funcionen correctamente. Ahora, el plugin PHP-MySQL nos dice que debemos crear el enlace simbólico:

Ahora el plugin PHP-MySQL nos dice que debemos de crear el enlace simbólico:

```
ln -fs /etc/php-5.3.sample/mysql.ini /etc/php-5.3/mysql.ini
```

Ahora lanzamos el demonio de MySQL y procederemos a la creación de la contraseña del usuario root:

```
/usr/local/bin/mysql_install_db
/usr/local/bin/mysqld_safe &
/usr/local/bin/mysqladmin -u root password 'TU-NUEVA-CONTRASEÑA'
```

Perfecto. Si has llegado hasta aquí sin problemas ya tienes el MySQL perfectamente funcionando y con la contraseña del superadministrador cambiada.

A continuación vamos a acceder a MySQL y crearemos la primera base de datos de pruebas lladama "drupal_db" para instalar un Drupal o un Wordpress para testear que tenemos Apache con PHP y el MySQL perfectamente preparados para albergar proyectos basados en gestores de contenidos:

```
mysql -u root -p
mysql> create drupal1_db;
mysql> grant all privileges
    -> on drupal1_db.*
```

```
    -> to 'usuari'@'localhost'
    -> identified by '1qazxsw2';
Query OK, 0 rows affected (0.00 sec)
mysql>
```

Ahora, para que el demonio de MySQL se inicie en el arranque del sistema OpenBSD, añadiremos la siguiente línea en el fichero /etc/rc.conf.local:

```
pkg_scripts="mysqld"
```

I ara com Apache corre com chrooted a /var/www necesitem fer uns hard links per a que puga treballar amb els sockets de MySQL. El fichero /etc/rc.local quedará así:

```
$OpenBSD: rc.local,v 1.44 2011/04/22 06:08:14 ajacoutot Exp $
Site-specific startup actions, daemons, and other things which
can be done AFTER your system goes into securemode.  For actions
which should be done BEFORE your system has gone into securemode
please see /etc/rc.securelevel.

mkdir -p /var/www/var/run/mysql
sleep 5
ln -f /var/run/mysql/mysql.sock /var/www/var/run/mysql/mysql.sock
```

Para acabar, ejecutaremos el script mysql_secure_installation el cual nos va a ir preguntando una serie de pasos para la puesta en marcha en producción de MySQL:

```
/usr/local/bin/mysql_secure_installation
```

Ahora te cuento una cosa a modo informativo: desde la versión 4.9 de OpenBSD, el iniciador de MySQL ya viene /etc/rc.d, y ahí estan los scripts de inicio, osea, que ya podemos usar "/etc/rc.d/mysqld start" o "/etc/rc.d/mysqld stop" cuando queramos para labores de mantenimiento, configuraciones en el servidor MySQL o de fallos en el rendimiento.

Y eso eso todo, ya lo tenemos. Ya tenemos el set OAM (OpenBSD + Apache + MySQL). Así que cuando quieras ya puedes reiniciar Apache y ya podemos comenzar a instalar un Wordpress, un Drupal o lo que queramos para probar que el entorno OAM está funcionando correctamente.

Capítulo 8. Saludo en una sesión remota

Siempre me gustó esos mensajios que te muestran los servidores Unix tras el inicio en una sesión FTP o SSH. Aquí te cuento cómo configurarlo en un OpenBSD.

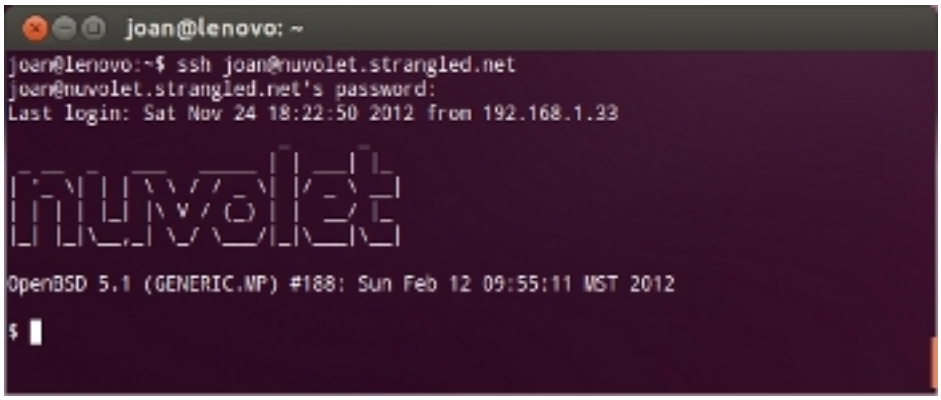
Configuración del motd

Añadiremos lo siguiente en el fichero */etc/motd*:

```
 _ __  _   ___   ______ | | ___| |_
| '_ \| | | \ \ / / _ \| |/ _ \ __|
| | | | |_| |\ V / (_) | |  __/ |_
|_| |_|\__,_| \_/ \___/|_|\___|\__|

OpenBSD 5.1 (GENERIC.MP) #188: Sun Feb 12 09:55:11 MST 2012
Bienvenido al sistema nuvolet.strangled.net
```

Figura 8.1. El saludo que nos da el servidor al acceder.



Que por cierto, obtendremos este "nuvolet" tan grande con la aplicación figlet:

```
figlet nuvolet
```

Capítulo 9. El correo electrónico en nuestro servidor

En este capítulo quiero mostrar como enviar correos desde nuestro servidor, cómo desargar el correo de otras cuentas utilizando fetchamil, ya sea con POP3 o IMAP, y cómo leer correos desde la consola con Mutt.

Habilitar sendmail

OpenBSD incluye una versión auditada de sendmail, para usarla con una configuración por defecto que permite enviar y recibir correos a otras máquinas.

Con esta configuración sendmail funcionará como MTA y esperará conexiones SMTP en el puerto 25 y en el puerto 587 (el segundo se espera que sea empleado por usuarios locales y que esté bloqueado al exterior, mientras que el primero por usuarios que deseen reenviar correo desde otros computadores).

A continuación se presenta una prueba a este servicio:

```
$ telnet localhost 25
Trying ::1...
Connected to localhost.
Escape character is '^]'.
220 amor.miescuela.edu.co ESMTP Sendmail 8.13.8/8.13.3; Mon, 16 Oct 2006
HELO localhost
250 amor.miescuela.edu.co Hello pablo@localhost [IPv6:::1], pleased to meet you
MAIL FROM: <pablo@localhost>
250 2.1.0 <pablo@localhost>... Sender ok
RCPT TO: <pablo@localhost>
250 2.1.5 <pablo@localhost>... Recipient ok
DATA
354 Enter mail, end with "." on a line by itself
1 2 3
probando
.
250 2.0.0 k9GHgf1q019958 Message accepted for delivery
quit
221 2.0.0 amor.miescuela.edu.co closing connection
Connection closed by foreign host.
```

Los logs de Sendmail los tenemos en */var/log/mailman*, tanto los de los envíos como los de recepción del correo electrónico. Hecha un vistazo aquí si necesitas analizar trazos y fallos que detectes en lo relativo a los correos electrónicos.

Fetchmail y mutt, una gran pareja

A continuación vamos a aprender cómo recibir correos con fetchmail y cómo leerlos y enviar nuevos correos desde Mutt.

Fetchmail nos permitirá tomar los correos usando POP3 (descargándolos directamente de un servidor de correo) o IMAP (los descarga pero no los elimina del servidor, así siempre tendrás la misma información tanto en el servidor como en tu Mutt).

Configurar fetchmail para que descargue los correos desde POP3 editaremos el fichero .fetchmailrc dentro de nuestro /home. Imagina que quiero descargar los correos de 2 cuentas que tengo, una en yahoo llamada "joanc@yahoo.com" y otra en gmail llamada "joanc@gmail.com". Bien, pues para ello editaremos .fetchmailrc y pondremos algo así:

```
set daemon 60
##########################################
# YAHOO mediante POP3
##########################################
poll pop.mail.yahoo.com
protocol pop3
username "joanc"
password "MI_CONTRASEÑA" is "joan" here

##########################################
# GMAIL mediante IMAP
##########################################
poll imap.gmail.com protocol IMAP
user "joanc@gmail.com" is joan here
password 'MI_CONTRASEÑA'
#fetchlimit 6
keep
ssl
```

Tras editar el fichero .fetchmailrc, cerciórate que sólo tú puedas leer dicho fichero y nadie más, para ello haz:

```
chmod g=,o= .fetchmailrc
```

Recuerda que otros usuarios pueden intentar escuadriñar tus datos y siempre debemos controlar muy bien los privilegios de lectura y escritura para vigilar que nadie vea cosas que no queremos.

Fetchmail se puede ejecutar en modo dæmon mediante el parámetro -d seguido seguido por un intervalo de tiempo (expresado en segundos) que indica cada cuánto tiempo debe fetchmail interrogar a los distintos servidores listados en .fetchmailrc.

El siguiente ejemplo hace que fetchmail interroge cada 300 segundos (5 minutos):

```
 fetchmail -d 300
```

Añadiremos este comentado en /etc/rc.local para que fetchmail comience tras cada reinicio del sistema OpenBSD, ¿fácil, no?

Capítulo 10. Transferencia de ficheros

En este capítulo aprenderemos cómo crear cuentas de FTP y configurar un servidor FTP mínimamente decente. Como sabrás, al tener habilitado el servicio SSH también tenemos transferencias SCP y de hecho, al ser cifrada, es mucho más segura que el FTP, así que tú eliges si quieres usar transferencias de ficheros FTP o SCP. Yo en este capítulo del libro explicaré como tener el servidor FTP corriendo, pero no sería necesario si eliges SCP.

Arrancar el servicio FTP

OpenBSD incluye SSH y FTP. Para activarlos o desactivarlos hay que editar el /etc/rc.conf. El FTP se activa editando el fichero /etc/rc.conf y donde pone:

```
# Set to NO if ftpd is running out of inetd
ftpd_flags="NO"           # for non-inetd use: ""
```

Lo dejamos así:

```
# Set to NO if ftpd is running out of inetd
ftpd_flags=""           # for non-inetd use: ""
```

Por defecto, durante la instalación nos pregunta si queremos activar el servicio de SSH y le dijimos que sí. En caso de que le dijésemos que no y queremos activarlo ahora, lo haremos igual que hemos hecho con el FTP.

Configuración de las cuentas FTP

Si estamos en nuestra red local, podemos transferir datos usando el protocolo FTP ya sea con un cliente de FTP ya estés en GNU/Linux, Mac OS X o Windows.

Sin embargo, te recomendamos que utilizes SCP, que es el protocolo de transferencia del SSH, ya que éste va cifrado, mientras que en un FTP las claves van en texto plano y serían muy fáciles de capturar si alguien está esnifando paquetes en la red.

Sea como sea, ya sabes ahora como activar y desactivar el FTP y el SCP (que es el SSH) en OpenBSD 5.1

Ahora, si el usuario "fulanito" tiene la página web www.fulanito.com albergada en la ruta /var/www/htdocs/fulanito.com, podemos hacer un enlace en su home para que siempre pueda subir los ficheros a su web desde su home:

```
cd /home/fulanito
$ ln -s /var/www/htdocs/servidoret.strangled.net/ servidoret.strangled.net
```

Esto lo ejecutaremos con todos los usuarios que tengan una web.

Capítulo 11. Administración de la(s) máquina(s)

En este capítulo aprenderemos una serie de utilidades para saber cómo están los procesos, para saber qué está haciendo un cierto usuario del sistema y otras utilidades que nos ayudarán a mantener nuestra máquina estable y que no se nos desborde por falta de previsión.

Ver los procesos de los usuarios.

Realizamos este chequeo con "ps" y con el atributo "-U" (de User). Por ejemplo, en este ejemplo vamos a ver los procesos del usuario "joan".

```
$ ps -U joan
  PID TT  STAT       TIME COMMAND
4227 ??  Ss      0:09.61 fetchmail -v
25568 ??  S       0:00.04 sshd: joan@ttyp0 (sshd)
15297 p0  Ss      0:00.01 -ksh (ksh)
4958 p0  R+/1    0:00.00 ps -U joan
$
```

El comando finger

El comando finger muestra información del usuario. Si somos administradores, entonces como root podemos conocer el nombre de login, el directorio, el nombre completo y otros datos de los usuarios. Supongamos que tenemos un usuario llamado “luis” y ejecutamos *finger*, obendremos:

```
$ finger luis
Login: luis                             Name: Luis de Pruebas García
Directory: /home/luis                   Shell: /bin/ksh
On since Sun Oct 28 14:34 (CET) on ttyp0 from 192.168.1.40
New mail received Sun Oct 28 18:46 2012 (CET)
Unread since Sat Oct 27 11:52 2012 (CEST)
No Plan.
$
```

Como ves, esto nos muestra el login con el que accede el usuario, la shell por defecto, el nombre de la persona, el tiempo que lleva conectado y si tiene o no mensajes de correo.

Detección de intrusos con el comando who

El comando who muestra quien está conectado al sistema. Su sintaxis es la siguiente:

```
who [-imqsuwHT] [--count] [--idle] [--heading] [--help] [--message]
   [--mesg] [--version] [--writable] [file] [am i]
```

El comando *who* por defecto muestra el nombre de la cuenta, el terminal en donde está conectado el usuario, el tiempo que dura su ingreso en el sistema y el nombre del ordenador remoto o terminal X si estuviese en las X

Comprobar los logs

Para buscar algún error o algún evento que se haya producido en nuestra máquina revisaremos los logs. Podemos monitorizar así las visitas a una determinada página web o a otra, los accesos al sistema por parte de los usuarios de FTP, los correos, etc.

Por defecto, en OpenBSD los logs se almacenan en el directorio /var/log/. A continuación, voy a listar las 15 últimas líneas del log de los correos para ver ha surgido algún problema:

```
$ tail -n 15 /var/log/maillog
```

Revisar en los logs si alguien ha intentado acceer al sistema mediante SSH

```
$ grep "Invalid user" authlog | grep "Oct 25" | wc -l
5
```

Parece ser que 5 (y hoy yo no he entrado más que una vez). En concreto, son estas entradas:

```
Oct 25 00:35:51  sshd[19510]: Invalid user test from 119.147.244.96
Oct 25 00:41:09  sshd[8207]: Invalid user direccion from 119.147.244.96
Oct 25 10:27:35  sshd[23863]: Invalid user oracle from 61.135.88.137
Oct 25 10:31:19  sshd[22022]: Invalid user oracle from 61.135.88.137
```

Hay bots o personas malintencionadas que han estado urgando en mi servidor, como puedes ver, pues yo no he creado la cuenta a ningún 'test', 'direccion' o 'oracle'... y alguien intenta acceder a mi máquina usando dichos usuarios. Así que ya sabes, ten mucho cuidado y valora la seguridad.

Ver los logs en tiempo real

Puedes mirar los logs de la máquina en tiempo real, así cuando un usuario enviará un e-mail o cuando alguna persona visite alguna de las páginas web albergadas o cuando existan errores graves, recibirás por consola la línea de lo que ocurre en tiempo real. Esto se hace con tail -f FICHERO_DEL_LOG, así que puedes elegir de entre todos los logs existentes en el sistema que por defecto en OpenBSD están en /var/log.

En el siguiente ejemplo vamos a monitorizar en tiempo real los correos entrantes y salientes de todos los usuarios del sistema:

```
$ tail -f /var/log/maillog
```

Capítulo 12. Monitorización de nuestro servidor

En este capítulo quiero ir centralizando una serie de ideas y aplicaciones que nos ayudarán a monitorizar todo aquello que sucede en nuestro servidor doméstico OpenBSD para tenerlo todo bajo control.

Controla la entrada y salida de tu tarjeta de red

Para controlar los bytes de los paquetes de entrada y de salida en nuestra máquina y cerciorarnos de que todo sea correcto, utilizaremos *netstat*:

```
netstat -b -w 1
```

He usado *-b*, que nos muestra las estadísticas de los bytes de los paquetes de entrada y salida, y también he puesto *-w* que muestra las estadísticas de la interfaz de red en intérvalos de segundos (en el ejemplo, de 2 segundos).

Monitorización y estadísticas con vnstat

Para mi *vnstat* es una aplicación por consola genial. La web del proyecto es http://humdi.net/vnstat/ y allí te explican muchos ejemplos para sacar estadísticas de red diarias, semanales, mensuales, etc.

Como ya habrás aprendido, lo instalaremos con:

```
bash-4.2# pkg_add vnstat
vnstat-1.10p6:gd-2.0.35p0: ok
useradd: Warning: home directory `/var/db/vnstat' doesn't exist
vnstat-1.10p6: ok
The following new rcscripts were installed: /etc/rc.d/vnstatd
See rc.d(8) for details.
Look in /usr/local/share/doc/pkg-readmes for extra documentation.
```

Si la primera vez lo ejecutamos, nos dirá que aún no funciona porque todavía no ha recogido los suficientes datos en la base de datos.

```
bash-4.2# vnstat
msk0: Not enough data available yet.
bash-4.2#
```

Pero tras un tiempo, ya podrás comenzar a explotar los datos de tu servidor, y en la ayuda te explican los atributos al comando de manera muy intuitiva:

```
vnStat 1.10 by Teemu Toivola <tst at iki dot fi>
```

```
-q,  --query          query database
-h,  --hours          show hours
-d,  --days           show days
-m,  --months         show months
-w,  --weeks          show weeks
-t,  --top10          show top10
-s,  --short          use short output
-u,  --update         update database
-i,  --iface          select interface (default: msk0)
-?,  --help           short help
-v,  --version        show version
-tr, --traffic        calculate traffic
-ru, --rateunit       swap configured rate unit
-l,  --live           show transfer rate in real time
```

Capítulo 13. Garantizando el servicio con copias y con un sistema de respaldo

En este capítulo aprenderemos a mantener un sistema de copias de seguridad basado en Unix y a la importancia de mantener un sistema de respaldo si pretendemos dar algún servicio serio.

Haciendo copias de seguridad

Tú eres el gran hacker onmipresente, protector del cielo y de la tierra, creador del Universo UNIX eterno que da vida a los usuarios y el que tiene el poder para salvaguardar la paz del mundo por los siglos de los siglos. Ramén.

Bueno, dejando este texto panfletario pastafari, ahora pongámonos serios: No te fíes jamás de las manazas de los usuarios, así que adelántate a los posibles problemas que pueden venir, que sin duda vendrán. Encárgate de mantener un sistema seguro, estable y que tenga copias de seguridad de todo:

- De las bases de datos.
- De los directorios de los usuarios (/home).
- De todas las webs y configuraciones (/var/www).
- Haz un listado de aplicaciones instaladas (pkg_info).
- De las configuraciones de arranque y de los distintos programas (apúntatelo todo bien y luego diseña copias).

La mejor planificación serían copias diarias y copias semanales que almacenaremos siempre en 2 sitios diferentes, por ejemplo en otro ordenador que esté en red con SCP o FTP y en un disco duro USB externo, pero esto ya depende de los dispositivos de que dispongas.

Sea como sea, planifica bien tus copias de seguridad. Sólo es necesario planificarlas una vez, ya que después lo tendremos automatizado y lo llamará el CRON la máquina, pero es importante asentar bien el método a utilizar ya que de las copias de seguridad depende que la información del servidor no se pierda jamás.

Para ficheros podemos usar tar.gz y hacer paquetes comprimidos:

```
tar -cvfz home-de-los-usuarios.tar.gz /home
```

Para exportar todas las bases de datos MySQL del sistema e incluirlas en el backup, podemos hacer un:

```
mysqldump -uUSUARIO -pCONTRASEÑA --all-databases > mi_copia.bk
```

Y lo recuperaremos con:

```
mysql -uUSUARIO -pCONTRASEÑA < mi_copia.bk
```

Creando un sistema de respaldo

Algún día, o semana o mes, te querrás ir de casa a tomarte unas pequeñas vacaciones y a beber cerveza sin control. Querrás apagar la luz o el gas de casa cerrados para prevenir algún incidente, ¿verdad? y ¿qué harás con tu servidor?, porque seguramente no quieres cerrar el servicio, ¿verdad?

Te recomiendo planificar un buen sistema de respaldo para evitar problemas e incidentes indeseados que hagan que tus servicios se vayan al garete. Para ello, puedes:

- Conseguir otro ordenador e instalar la misma versión del sistema operativo
- Hablas con algún/alguna amigo/a que tenga conexión ADSL en casa y que no le importe dar servicio de vez en cuando.
- Conseguimos un dominio o subdominio para llamar a esta segunda máquina de respaldo.
- Conseguir otro ordenador e instalar la misma versión del sistema operativo
- Creamos un sistema de sincronización con rsync y SSH de la máquina 1 a la máquina 2 y que se sincronicen un día a la semana a una hora exacta. Para ello tu amigo tiene que encender la máquina 2 dicho día de la semana.

Así pues, cuando tú te vayas a hacer surf a Hawaii, puedes apagar todo en tu casa y cambias los registros DNS de tu dominio para que apunten a la IP de tu amigo. generado allí.

Este sistema también servirá para hacer un mantenimiento bastante grande a tu máquina, formateo o reinstalación de todo el sistema.

Un script de ejemplo para nuestro sistema de respaldo

Aquí dejo un ejemplo para que tengas todos los directorios y ficheros importantes de tu servidor OpenBSD bajo control. Recuerda que puedes modificarlo y adaptarlo a tus necesidades añadiendo nuevos directorios o cambiando parámetros:

```
#####################################################################
# Backup de tots els sites del servidor OpenBSD                     #
#                                                                   #
# 1.Primer copie alguns fitxers i bases de dades                    #
# 2.Fem el tar.gz de tot el /var/www/htdocs                         #
# 3.Enviem també el backup a un servidor via FTP                    #
# 4.Enviem una notificació per e-mail.                              #
#                                                                   #
# Versió 0.2 - 26/08/2013                                           #
#####################################################################

# Entrem a /var/www/backups i creem un directori per a cada dia
#
cd /var/www/backups
```

```
NOW=$(date +"%Y%m%d")
mkdir $NOW
cd $NOW

#####################################################################
# [Copia 1] Copiem 4 bases de dades
#####################################################################
/usr/local/bin/mysqldump -u USUARI -pPASS1 | gzip > ./$NOW.BBDD1.sql.gz
/usr/local/bin/mysqldump -u USUARI -pPASS2 | gzip > ./$NOW.BBDD2.sql.gz
/usr/local/bin/mysqldump -u USUARI -pPASS3 | gzip > ./$NOW.BBDD3.sql.gz
/usr/local/bin/mysqldump -u USUARI -pPASS4 | gzip > ./$NOW.BBDD4.sql.gz

#####################################################################
# [Copia 2] Copies de fitxers importants
#####################################################################
cp /var/www .
cp /opt/programes .
cp /usr/scripts .

#####################################################################
# [Copia 3] Copiem els home dels usuaris
#####################################################################
tar cvfz ./home-root.tar.gz  /root
tar cvfz ./home-usuari1.tar.gz  /home/USUARI1
tar cvfz ./home-usuari2.tar.gz  /home/USUARI2
tar cvfz ./home-usuari3.tar.gz  /home/USUARI3
tar cvfz ./home-usuari4.tar.gz  /home/USUARI4

#####################################################################
# [Copia 4] Copiem el directori de les webs que tenim al htdocs
#####################################################################
tar cvfz ./var-www-htdocs.tar.gz /var/www/htdocs

#####################################################################
# [Copia 5] Creem un tar.gz amb tots els fitxers
#####################################################################
tar cvfz /var/www/backups/ultim-backup.tar.gz *
```

Yo lo tengo configurado para que, además, me envíe los datos a otro servidor externo vía FTP, así tendré las copias de seguridad en dos equipos distintos y en caso de que se rompa el disco duro de *nuvolet* encontraré la información en el otro servidor externo. Y también, al final de todo, me envía un correo electrónico notificándome que la copia de seguridad se ha realizado correctamente, algo muy útil para que me quede tranquilo cada mañana al recibir ese correo ;-). Pues bién, tienes la última versión del script en:

http://jcatala.net/categoria-unix/script-que-fa-copies-de-seguretat-en-local-i-un-servidor-remot

Y recuerda compartir tus modificaciones, adaptaciones y mejoras del script, para que todos nos podamos beneficiar y aprender.