CONFIGURACIÓN SERVIDOR WEB, DNS, FTP, POP3 Y SMTP

Este texto es sacado del trabajo de fin de bachillerato que hice este año. El objetivo era reutilizar uno de los ordenadores que el instituto no utilizaba (porque se consideraban "pequeños" para el windows) y montar un servidor de red para dar cuentas de correo y espacios para webs personales al alumnado y al profesorado. Todo esto con el sistema operativo Linux, por supuesto. Este texto es una parte del trabajo, donde explico el procedimiento práctico.

El ordenador en cuestión es un Pentium I de 90MHz con 24 MB de memoria RAM y dos discos duros, uno de 460MB y el otro de 20GB adosado posteriormente, ya que resultaba necesario utilizar un disco de mayor tamaño para cumplir con las funcionalidades del proyecto. Mencionar que no fue tarea fácil hacer que la BIOS del ordenador reconociera el disco de 20 GB, dado que en aquellos tiempos (1993-1994) no se imaginaba que pudieran existir discos duros de tal capacidad.

Una vez terminado el trabajo, debería estar todo funcionando correctamente, y que tanto profesores como alumnos dispongan de una cuenta de correo electrónico del tipo *pmanils@dominio.org*, y un espacio web para colgar su página personal del estilo *www.dominio.org/alumnos/pmanils* o *www.dominio.org/professorat/mnicolau* en el caso que fuera un profesor. Sin embargo, para los perezosos, se pudiera visitar la página de cualquier usuario utilizando la dirección *www.dominio.org/~nombre-profesor-o-alumno*, escribiendo el carácter "~" (Alt+126) seguido del nombre de un profesor o alumno, por ejemplo *www.dominio.org/~brios*.

Para los usuarios independientes al centro de estudios, decidí separarlos en el grupo "users", por lo tanto, si desean visitar la página web de cualquier usuario ajeno al instituto, la URL sería con este formato:

www.dominio.org/users/knoopx

o con el "universal":

www.dominio.org/~knoopx

Se intentará explicar todo esto lo más sencillo y fluido posible, para que esté al alcance de los usuarios, aunque deben tenerse mínimas nociones sobre informática e Internet para su perfecta comprensión. Algunos de los temas que se explicarán no se desarrollarán en su plenitud ni se profundizará mucho, sino lo suficiente. Lo explicaré con mis palabras y de la manera que yo lo he entendido. Puede no ser la mejor forma, pero así creo que resultará más fácil su comprensión. Como se verá más adelante, las

configuraciones serán mínimas y básicas para no enturbiar ni hacer complejo este documento, que trata justamente la combinación de la simpleza con la eficacia.

1. Servidor DNS, WEB y FTP

En este apartado se describirán los pasos que se siguieron para poner en marcha el servidor web apache. apache es el software por excelencia utilizado para dar servicio de web en el mundo Linux (y fuera de él, ya que hay versión para otras plataformas, como FreeBSD, Solaris, Windows, IRIX, ...)

Antes de empezar con la instalación y configuración del servidor apache, necesitábamos un dominio para la web. Los dominios son los nombres (google.com, xtec.es, sorgonet.com...) a partir de los cuales se nos redirige a la IP del servidor web deseado. Decidimos registrar *dominio.org* (ficticio por supuesto). Para ello fuimos a *www.godaddy.com* (página web donde se pueden registrar dominios). En una fase del proceso de registro se nos pedía introducir dos servidores DNS. Un servidor DNS es el encargado de traducir los nombres de dominio a IPs y viceversa. Necesitábamos pues un servidor que tradujera dominio.org a nuestra IP *80.80.80.80* (también falsa) donde teníamos pensado montar el servidor web. Este fue el primer problema que encontramos. No teníamos ningún servidor DNS que hiciese eso, por lo que había dos alternativas: Primero; Pedir a un conocido que tuviera montado un servidor DNS y que nos lo configurara para que dominio.org se redirigiera a 80.80.80.80 (nuestra IP). o, Segundo; Montar nuestro propio servidor DNS.

La primera opción es la más sencilla, solo falta tener un amigo que tenga un servidor DNS y pedirle que configurara su servidor DNS para que resolviera el dominio *dominio.org* a nuestra IP. Pero tiene una gran desventaja, no seríamos nosotros quienes controlaríamos el servidor DNS, por lo que no tendríamos control sobre dicho servidor y por lo tanto, creciendo nuestra desconfianza, ¿quién nos asegura que un día, a nuestro amigo, no se le pase por la cabeza configurar su servidor DNS para que redireccione *dominio.org* a cualquier otra IP, por ejemplo la de una web de contenidos pornográficos? Sería verdaderamente divertido...

Por esta razón, por tener nosotros el control del servidor DNS, decidí montar nuestro propio servidor DNS. Y aquí muestro los pasos que seguí para hacerlo.

1.1. Servidor DNS

Bind es el servidor DNS mas popular en entornos Linux. Necesitábamos tenerlo instalado, y así lo hicimos:

dns# apt-get install bind

apt-get es un gran programa que, al indicarle que instale otro programa (*install bind*) él solo se baja de internet dicho programa y todo lo que haga falta para que funcione, lo instala y lo deja apunto para que solo tengamos que configurar lo que necesitamos.

Ahora que tenemos *bind* instalado (en mi caso la versión 8.3.3-REL-NOESW), falta configurarlo adecuadamente.

Normalmente, para configurar un programa en linux, basta editando los archivos apropiados del programa. Los archivos de configuración del *bind* se encuentran en */etc/bind/*. El primer fichero que editaremos será el *named.conf*, que es el fichero principal de *bind*. Una recomendación es hacer una copia de seguridad del archivo original antes de editarlo. Utilizaremos el editor llamado *vi* o *vim* puesto que es uno de los editores más comunes en entornos Unix/Linux:

Teclearemos lo siguiente:

```
# cp /etc/bind/named.conf /etc/bind/named.conf.old
# vi /etc/bind/named.conf
```

Nos fijamos que en la segunda página se repite bastante una parte de código parecida a esto:

```
zone "localhost" {
type master;
file "/etc/bind/db.local";
};
```

Bien, nos dirigiremos al final del archivo y añadiremos lo siguiente:

```
zone "dominio.org" {
type master;
file "/etc/bind/db.dominio.org";
};
```

Esas líneas definen una nueva zona, dominio.org, sobre la que se ejerce el control, y el fichero (file) de configuración de esta zona se encontrará en */etc/bind/db.dominio.org*. Guardaremos los cambios y saldremos (*:wq*).

Ahora falta crear el fichero *db.dominio.org*, y como que el formato del archivo es parecido al ya existente */etc/bind/db.local*, haremos una copia de éste con el nombre *db.dominio.org* sobre el cual modificaremos a nuestro parecer:

```
# cp /etc/bind/db.local /etc/bind/db.dominio.org
# vi /etc/bind/db.dominio.org
```

El archivo db.dominio.org tiene que quedar parecido a este:

```
----- Inicio del archivo (esto no debe incluirse)-----
;
; BIND data file for dominio.org
;
$TTL 604800
dominio.org. IN SOA dns.dominio.org. root.localhost. (

1 ; Serial
604800 ; Refresh
86400 ; Retry
2419200 ; Expire
604800 ) ; Negative Cache TTL

 IN NS dns.dominio.org.
 IN A 80.80.80.80
 IN MX 10 mail.dominio.org.
dns IN A 80.80.80.80
mail IN A 80.80.80.80
www IN CNAME dns
ftp IN CNAME dns
----- Fin (esto tampoco debe incluirse)-----
```

Para entender esto: Lo que sigue después de ";" es ignorado como código, se interpretan como comentario. En las siguientes líneas se entiende que dominio.org se encuentra en la máquina dns.dominio.org (host local de la máquina donde está el servidor DNS), y el encargado de este dominio es *root.localhost*. (no olvidar poner los puntos en *dominio.org*. *dns.dominio.org*. y *root.localhost*.). El *Serial*, *Refresh* y todo esto lo dejamos como está, son tiempos de expiración y otros. Luego se le indica que el servidor DNS (*NS*) se encuentra en *dns.dominio.org.* (otra vez lo del punto final) y utiliza como servidor de email (*MX*) con prioridad máxima (10) (se puede poner la prioridad que se quiera, y la máxima, caso de especificar varios registros *MX*, es aquella con un número menor) la máquina mail.dominio.org., cuya dirección IP en Internet es *80.80.80.80*, y por consiguiente, que el alias dns también está en *80.80.80.80*. En este caso, que no hemos puesto punto al final de dns, automáticamente *bind* lo interpreta de manera que le añade dominio.org al final, quedando de la manera dns.dominio.org. Lo mismo ocurre con www, ftp y mail pero la etiqueta *CNAME* matiza que se tratan de tres alias de dns, por lo que si sigues la cadena, tenemos que valen la misma IP que dns, que es lo que nos interesa: tener el servidor web, ftp, dns y mail en la misma máquina. De esta manera, *www.dominio.org*, *ftp.dominio.org*, *mail.dominio.org* y *dns.dominio.org* se refieren a la misma IP de nuestro ordenador.

Después de esta confusa explicación, debemos poner en marcha *named* (*bind*):

```
# /etc/init.d/bind restart
```

Lo de restart es porque al haber hecho antes *apt-get install bind* él solo se ejecuta después de instalarse, con la configuración por defecto que lleva.

Ahora podemos comprobar si todo ha ido bien con el comando siguiente:

```
# dig @localhost dominio.org
```

Tendría que salir algo parecido a esto:

```
; <<>> DiG 9.2.1 <<>> @localhost dominio.org
;; global options: printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 63073
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1,
ADDITIONAL: 1

;; QUESTION SECTION:
;dominio.org. IN A

;; ANSWER SECTION:
dominio.org. 604800 IN A 80.80.80.80

;; AUTHORITY SECTION:
dominio.org. 604800 IN NS dns.dominio.org.

;; ADDITIONAL SECTION:
dns.dominio.org. 604800 IN A 80.80.80.80
```

dig se encarga de mirar la configuración de *dominio.org* utilizando como servidor DNS el que nosotros le indiquemos después de la @. Ya que nos interesa que utilice el servidor DNS que hemos montado nosotros para probar si funciona correctamente, que se encuentra en nuestra máquina local, le escribiremos después de la @ *localhost*.

Podríamos decir que ya tenemos nuestro propio servidor DNS.

Pero no estamos aún, el ordenador que hace de servidor está detrás de un router, en una red interna, y deberemos abrir el puerto 53 del router (puerto que utiliza el *bind* para las peticiones DNS) y redirigirlo al puerto 53 del servidor DNS de la intranet.

Desde el servidor Linux haremos un telnet a la IP del router:

```
# telnet 192.168.0.1
Trying 192.168.0.1...
Connected to 192.168.0.1.
Escape character is '^]'.
login:
```

Introduciremos el login y la contraseña adecuada y una vez dentro del router teclearemos lo siguiente:

```
3Com-DSL>add nat tcp vc internet public_port 53 private_port 53
private_address 192.168.0.5

3Com-DSL>add nat udp vc internet public_port 53 private_port 53
private_address 192.168.0.5
```

Esto hace lo dicho anteriormente: abre el puerto 53 del router (public_port 53) y envía las peticiones que le llegan, a la máquina 192.168.0.5 que se encuentra dentro de la red (private_address 192.168.0.5) , al puerto 53 de esa máquina (private_port 53). Con lo de tcp y udp indicamos que se valide por esos dos protocolos (esta operación que hace el router se denomina NAT, Network Address Translation (Traducción de dirección de red).

Le decimos al router que guarde los cambios y reinicie para que los cambios tengan efecto:

```
3Com-DSL>save all
3Com-DSL>reboot
```

Esperamos un par de minutos más, y todo está listo para que, cuando en el registro de dominio.org se nos pregunte por un servidor DNS, poner el que hemos montado: *dns.dominio.org* (cuando nos pregunte por el servidor dns habrá que poner no "*dns.dominio.org*" sino nuestra IP de internet) como primario y cualquier otro servidor DNS como secundario.

No lo expliqué antes, pero nos piden dos servidores, uno primario y otro secundario, por si el primario dejara de funcionar alguna vez, podamos seguir visitando la web gracias al DNS secundario. En este caso, pusimos como secundario un DNS de Terra, que podría haber sido cualquier otro. Si nuestro servidor DNS se apagara por alguna razón, se cogería el secundario, pero como el de Terra no tiene ninguna información acerca de dominio.org, no podría mostrarnos la página. De todas maneras, si se apagara el servidor DNS, tampoco podríamos dar servicio de WEB, ya que se trata del mismo ordenador.

Los cambios de configuración en el DNS son lentos, y no será hasta unas horas después (incluso hasta 48 horas) que podremos comprobar los resultados.

Primer problema afrontado y superado, pasemos a la configuración del servidor apache para que funcione definitivamente www.dominio.org.

1.2. Servidor WEB

En primer lugar tendremos que tener instalado dicho programa, que estando en Debian será de lo más fácil (siendo root):

```
# apt-get install apache
```

Una vez instalado, pasemos a su configuración, que se basará en la edición de algunos archivos de configuración del apache (repito que es necesario tener privilegios de administrador).

Lo primero de todo será editar */etc/apache/httpd.conf*, su archivo principal, en el cual personalizaremos apache para ajustarlo a nuestras necesidades. Recuerdo lo de las copias de seguridad, para poder restaurar el archivo en caso de que fallemos:

```
# cp /etc/apache/httpd.conf /etc/apache/httpd.conf.old
# vi /etc/apache/httpd.conf
```

Nos aparecerá una pantalla negro con líneas. Algunas de estas líneas empiezan con el símbolo "#", eso indica que no serán leídas como código, sino como comentarios, y por lo tanto serán ignoradas. Esto resulta útil para apuntar lo que hacemos, aclarar alguna línea de código...

Es hora de modificar el archivo y para ello buscaremos la línea siguiente:

ServerType standalone

Esto significa que *ServerType* coje el valor de *standalone*. Es la manera en que se ejecutará el servidor. Lo dejaremos así.

A medida que vayas bajando, notarás que hay muchas más definiciones del tipo "Variable valor", de momento no tocaremos nada porque son propias del funcionamiento de apache y no nos interesa cambiar nada, por lo tanto dejaremos los valores por defecto que se nos indica.

Cuando llegamos a *### Section 2: 'Main' server configuration* es cuando empezaremos a indicar lo que deseemos.

Vamos bajando y encontraremos las siguientes definiciones:

```
Port 80
```

Es el puerto donde apache escuchará. Por defecto el *80*, ya que si pusiéramos otro cualquiera, deberíamos especificarlo en la petición de la URL. Si ponemos *7001* por ejemplo, para visitar la página tendríamos que poner en el navegador "*http://www.dominio.org:7001*". Por lo que lo dejamos en 80.

```
User www-data
Group www-data
```

Nombre de usuario y grupo con el cual se ejecutará el servidor. Es necesario, por razones de seguridad, no poner ni root ni cualquier usuario privilegiado. *www-data* es el que traía por defecto y me pareció el más apropiado.

```
ServerAdmin webmaster@dns
```

Se especifica la dirección de email a la que se enviarán los mensajes de error. En nuestro caso pondremos *webmaster@dns* que es la dirección de email local del administrador de la web.

```
ServerName dominio.org
```

Nombre del servidor. Este valor tiene que ser un DNS válido o la IP del servidor web. Dado que compremos *dominio.org*, será lo que pondremos.

```
DocumentRoot /home/web
```

Indica el path (ruta completa) de la carpeta donde guardaremos los archivos de la página principal. En nuestro caso */home/web*.

```
<Directory /var/www/>
```

Esta etiqueta no debe quedarse así, sino que se tiene que canbiar */var/www/* por lo que pusimos en *DocumentRoot*. Así que quedaría: *<Directory /home/web/>*

```
UserDir /home/todos/*/tu_web
```

Path del directorio que almacenará la página personal de cada usuario. */home/todos/*/tu_web* es lo que puse. Así, si alguien escribe en el navegador *www.dominio.org/~eberney* entrará en la página personal del usuario *eberney*.

```
DirectoryIndex index.html index.htm default.htm index.php
```

Será el fichero o ficheros que tomará el servidor como índice del directorio web. Se suele poner "index.html", aunque se admiten varios más separándolos con espacios (default.html, index.php, index.asp...). No sabía con qué trabajarían los usuarios, así que puse los que se me ocurrieron en ese momento.

Hay muchas más opciones que deberían comentarse, pero no es el caso, nosotros solo queremos lo mínimo para que funcione bien.

Ahora falta poner una página en el *DocumentRoot* que hayamos escogido.

Crearemos la carpeta indicada, en mi caso:

```
# mkdir /home/web
```

Y editaremos cualquier archivo html para probar si funciona:

```
# vi /home/web/index.html
```

Escribimos:

```
<html>
Hola
</html>
```

Salimos del editor *vi* (*:wq*).

Después de haber configurado el apache toca reiniciarlo con el siguiente comando:

```
# apachectl restart
```

Solo nos queda comprobar que todo funciona bien.

lynx localhost

lynx es un navegador en modo texto para visitar páginas web desde la consola. Tendría que aparecernos en pantalla "Hola" (el contenido de */home/web/index.html*).

Pero esto es en ámbito local, por lo que si desde cualquier ordenador conectado a internet que esté fuera de la LAN ponemos en el navegador *www.dominio.org* no saldrá la página, por varias razones:

- El router no tiene abierto el puerto público 80 ni redirigido a nuestro servidor apache.
- Falta aplicar el uso de *VirtualHosts* en el servidor *apache*.

La primera es lo mismo que hicimos con el servidor DNS, pero cambiando el puerto a 80, que es el de la web.

El segundo problema se puede resolver fácilmente. De nuevo tenemos que editar el archivo */etc/apache/httpd.conf* e irnos al final de todo donde hay las etiquetas llamadas *VirtualHost*.

Tenemos que buscar y sustituir lo siguiente:

```
NameVirtualHost 192.168.0.5:80
```

La dirección con el puerto 192.168.0.5:80 es la del servidor apache que tenemos en la ethernet.

Y ahora abrimos la etiqueta siguiente:

```
<VirtualHost 192.168.0.5>
DocumentRoot /home/web/
DirectoryIndex index.php index.html index.htm
ServerAdmin ntn@drdoom.net
ServerName dominio.org
ServerAlias *.dominio.org
ErrorLog /home/web/logs/logerror
CustomLog /home/web/logs/access-log common
</VirtualHost>
```

Esto crea un host virtual en *192.168.0.5* con características explicadas con anterioridad, excepto el *ErrorLog*, archivo donde se guardan los errores, y el *CustomLog*, fichero donde se almacenará un registro de todos los accesos a la web.

Finalmente llega el esperado momento de la prueba, poner en marcha apache otra vez:

```
# apachectl restart
```

Ahora, desde cualquier lugar de internet podría uno visitar *www.dominio.org*.

Hasta aquí llegaría la instalación y configuración mínima de apache, pero se me ocurrieron un par de cosas para hacerlo todo más bonito y tenerlo más ordenado.

Cuando añadimos un usuario, normalmente su home se crea en */home/nombre-de-usuario*, pero claro, entre profesores, alumnos y otros usuarios, se liaría una bien grande en el directorio /home, así que pensé que sería conveniente dividirlos según su estatus. Crearía una carpeta para cada grupo de usuarios y una para englobarlos a todos:

```
# mkdir /home/alumnos
# mkdir /home/profes
# mkdir /home/users
# mkdir /home/todos
```

De esta manera, a la hora de añadir un usuario, si se trata de un profesor, le indicaremos al programa que añade usuarios (*adduser*) que utilice como home del usuario la carpeta */home/profes*, de esta manera, si añadimos el usuario *pgarcia*, su home estaría en */home/profes/pgarcia*.

¿Y qué pasa con la carpeta todos?

Cuando un usuario es añadido, hay una serie de ficheros y directorios que se encuentran en */etc/skel*, que se copian automáticamente en el home del usuario. Entre esos ficheros y directorios está la carpeta tu_web que está predeterminada para almacenar la página web de cada usuario, es decir, el usuario que desee tener página web, deberá poner todos los archivos de su página en el directorio tu_web de su home.

Por ejemplo, el usuario *pmanils* es un alumno, pues en su home, que será */home/alumnes/pmanils*, tendrá una carpeta llamada tu_web (*/home/alumnes*/*pmanils/tu_web*) donde guardará su página web.

Recordamos que UserDir se encarga de definir el directorio de la página web de los usuarios cuando se hace una petición web del estilo www.dominio.org/~pepe .

Si no hubiésemos hecho la separación de usuarios y todos se encontrasen en */home*, definiríamos UserDir como /home/*/tu_web donde * sustituye al nombre de usuario de la petición web (siguiendo el ejemplo de arriba, se desea ver la página del usuario pepe, entonces en ese momento *UserDir* valdría /home/pepe/tu_web y nos mostraría la página de pepe).

Pero al hacer la separación de homes por tipo de usuario (*alumno/profesor/usuario*) tendríamos que añadir tres tipos de *UserDir's*, uno por el caso que se pidiera la página de un alumno, otro por si fuera la de un profesor, y otra pos si fuera un usuario. Esto no es posible (o al menos yo no lo conseguí) y tuve que buscar una alternativa. Existe un comando llamado ln que sirve para hacer enlaces entre archivos o directorios, esto es, para que fichero1 se refiera a fichero2 pudiendo estar en distintos directorios del disco duro. Con un ejemplo lo veremos mejor:

```
# ln -s /home/ies/fichero1 /root/fichero-linkado
```

Este comando crearía el fichero */root/fichero-linkado* y si accediéramos a este archivo, veríamos simplemente el contenido de */home/ies/fichero1*. Es decir, */root/fichero-linkado* no sería más que un espejo (enlace) de */home/ies/fichero1*. Con el modificador *-s* le indicamos que el enlace sea simbólico, es decir, que lo que le ocurra a */root/fichero-linkado* no tendrá efecto en el archivo original */home/ies/fichero1*, pero las modificaciones que hagamos al archivo original siempre se verán si accedemos al enlace (link).

Todo esto lo explico para que comprendan lo siguiente.

ln nos permite también enlazar directorios, de tal modo que si creamos enlaces con el nombre de cada usuario que vayan desde su home *(/home/alumnes/brios* y los demás alumnos; */home/profes/mnicolau* y los demás profesores; y finalmente */home/users/teckk* y los otros usuarios) hacia un solo directorio, tendremos la situación en que todos los usuarios del sistema tengan un enlace en un único directorio destino que queramos, de tal forma que habremos creado un "home" donde tenemos acceso a todos los homes de todos los usuarios. Como podéis suponer, yo los agrupé todos en */home/todos*.

De aquí que definimos la variable *UserDir* de esta manera:

```
UserDir /home/todos/*/tu_web
```

Así que cualquier petición a cualquier usuario, ya sea alumno, profesor o usuario, se irá a buscar en la carpeta */home/todos* donde existirá un enlace con el mismo nombre de usuario que la petición, que nos llevará al home verdadero del usuario. Ejemplo:

www.dominio.org/~llmfabrega

llmfabrega es un profesor, pero no importa, *UserDir* valdrá */home/todos/llmfabrega/tu_web* (el * del *UserDir* es sustituido por el nombre introducido en la petición, en este caso *llmfabrega*), y dado que */home/todos/llmfabrega* es un enlace que lleva al home de este usuario, */home/todos/llmfabrega/tu_web* equivaldría a ir al directorio */home/profes/llmfabrega/tu_web*.

A parte también quería que hubiera una dirección web directa para visitar la página de un alumno o profesor del estilo *www.dominio.org/alumnes/brios* o *www.dominio.org/professorat/mnicolau*. Para ello utilicé por segunda vez los enlaces simbólicos.

En la configuración del apache teníamos como directorio principal de la web, */home/web*. La idea que tuve fue crear tres carpetas más dentro de ese directorio:

```
# mkdir /home/web/alumnes
# mkdir /home/web/professorat
# mkdir /home/web/users/
```

Y luego hacer enlaces simbólicos del directorio donde tiene la web cada usuario, a la carpeta de la web principal correspondiente:

Si cgarcia fuera un alumno, entonces el enlace que haríamos sería de este tipo:

```
# ln -s /home/alumnes/cgarcia/tu_web/ /home/web/alumnes/cgarcia
```

De esta manera si en la petición web del navegador pusiéramos "*www.dominio.org/alumnes/cgarcia*" estaríamos entrando en */home/web/alumnes/cgarcia* que al ser un enlace, nos llevaría a */home/alumnes/cgarcia/tu_web* que es donde nos interesa.

Así debería hacerse con cada alumno, profesor y usuario ajeno.

1.3 Servidor FTP

El servidor de FTP es necesario para que los usuarios del sistema puedan subir los ficheros de sus páginas web. Es una tarea fácil y sencilla.

Decidí instalar *proftpd*, uno de los mejores servidores para Linux:

```
# apt-get proftpd
```

Editaremos el archivo de configuración principal, después de hacer una copia del archivo:

```
# cp /etc/proftpd.conf /etc/proftpd.conf.old
# vi /etc/proftpd.conf
```

Podremos ver que es bastante parecido al *httpd.conf* del apache. Pero de este fichero solo nos interesa modificar una pequeña cosa.

```
DefaultRoot ~
```

Esta definición debería quedar como se muestra. El *DefaultRoot* es el directorio raíz donde iremos a parar si entramos por el FTP. El símbolo "~" indica que sea el directorio home de cada usuario, de esta manera, si soy el usuario *gonssal* y conecto al servidor FTP, iré a parar a */home/users/gonssal/* que es mi directorio home y no podré acceder a los directorios superiores a éste, solo podré moverme dentro de *home/users/gonssal/*. Es por una razón de seguridad. El *proftpd* trae por defecto que *DefaultRoot* sea "/", y esto sería bastante peligroso porque los usuarios podrían moverse por todo el sistema (recordemos que / es el directorio raíz de todo el sistema) y serían capaces de subir y bajar archivos sensibles de configuraciones.

Nos quedaría hacer lo de siempre, abrir el puerto 21 (que es el del FTP) en el router y redirigirlo a nuestra máquina. Como ya sabemos cómo hacerlo, no volveré a repetirlo.

Si recordamos lo que pusimos en el archivo de la zona *dominio.org* en la configuración del servidor DNS vemos lo siguiente:

```
ftp IN CNAME dns
```

Como vemos, *ftp* no tiene un punto al final, por lo que acabaría siendo *ftp.dominio.org* que es un alias (*CNAME*) de dns (*dns.dominio.org* porque no tiene punto tampoco) y que en la configuración de más arriba definimos dns como la IP *80.80.80.80*. De esta manera *ftp.dominio.org* equivaldría la misma IP que *dns.dominio.org*.

Esto resulta útil porque si queremos acceder al FTP de *dominio.org*, lo podremos hacer indicando como *host ftp.dominio.org* que queda más estético.

De esta manera ya quedaría todo a punto para que los usuarios puedan subir sus páginas personales.

2 Servidores SMTP y POP3

Para poder dar el servicio de mail, es necesario saber que existen dos programas: el encargado de enviar los emails (*SMTP*) y el de recogida de mails (*POP*). Por lo tanto deberemos configurar dos servicios, el de *POP3* (ya que *POP2* ha pasado un poco a la historia) y el *SMTP* (esto no es del todo cierto, ya que el servicio SMTP se encarga de transportar el correo entre servidores, y sólo se habla con servidores, de tal modo que *POP* es el protocolo usado para entregar el correo a los clientes).

2.1 Servidor SMTP

Debian trae por defecto el programa *exim* que se encarga del correo saliente del servidor. Durante la instalación de Debian se ejecuta un programa llamado *eximconfig* que nos permite configurar de una manera fácil *exim*. Después de la instalación podremos ejecutar cuando nos plazca dicho programa para modificar la configuración existente.

eximconfig nos hará una serie de preguntas a las cuales deberemos responder según nos interese:

- Uso del servidor: será la primera pregunta que se nos mostrará. Aquí deberemos pulsar el número 1, puesto que queremos configurar un servidor de correo electrónico a Internet. Las otras opciones son para otras configuraciones que no nos atañen.
- Nombre visible de la máquina: es lo que queremos que aparezca como remitente en los emails que enviemos. Yo puse "*dominio.org*" que es nuestro dominio que compremos.
- ¿Tiene otros nombres nuestro sistema a parte del anterior? En nuestro caso no, pondremos "none" o le daremos al enter ya que por defecto es "none".
- En este apartado se nos informa que nuestro servidor de correo aceptará correo que venga de Internet con destino a nuestra máquina y todo el correo que se envíe de forma local con destino a Internet (correo local es el que tiene como destino un usuario de nuestra máquina; correo saliente es el que tiene como destino otras máquinas o Internet), pero no reenviará correo (hacer *relay*) para otros que se conecten desde Internet a nuestra máquina. De esta manera evitaremos que alguien se aproveche de nuestro servidor para enviar mails masivos, publicidad, etc... (*spamming*). De todas formas podemos indicarle a

exim que sí lo deje hacer para ciertos dominios. En nuestro caso no nos interesaba, así que "none".

- ¿Para qué dominios queremos actuar de servidor de mail (a parte del nuestro)? Ahora se lo podemos indicar. "none" para mi configuración.
- Ahora podemos escoger para qué máquinas haremos de servidor de correo. Podemos indicarle direcciones IP o dominios. Yo le puse "**.dominio.org*" (con el asterisco), que en realidad es una misma máquina, pero que puede ser *mail.dominio.org*, *ftp.dominio.org* o *dns.dominio.org*.
- A qué usuario se redirigirá el correo que vaya para el *superusuario* (*root*). Esto también es por razones de seguridad, de esta manera el correo nunca podrá ser leído por el administrador y por lo tanto se reducirán los posibles ataques con correos malintencionados. En mi caso pues el usuario "*ies*", que soy yo mismo.
- Nos informa que ya tenemos un */etc/aliases* y si deseamos reemplazarlo por uno nuevo que generará *exim* o si queremos mantener el nuestro. Si elegimos la primera opción no perderemos nuestro fichero, sino que será renombrado a */etc/aliases.0* . Yo le indiqué que sí, que lo reemplazara por el nuevo.
- El último paso es un resumen de todos los datos que le hemos introducido durante el proceso de configuración. Si son correctos pulsaremos "y", y si no, podemos repetir el cuestionario pulsando "n".

Una vez tengamos esta configuración, queda hacer una pequeña cosa. Durante las preguntas, respondimos que no queríamos reenviar correo de gente que venía de Internet, pero claro, es justamente lo que queremos nosotros, que los usuarios desde su casa configuren su cliente de correo electrónico para que puedan conectarse a este servidor y poder enviar sus emails con la dirección *nombre-de-usuario@dominio.org*. Entonces era necesario controlar quién se conectaba al servidor *SMTP* para dejarle o no enviar emails. Una de las maneras es hacerlo por IPs, especificar qué IPs dejamos conectar a nuestro servidor. Pero no todos los usuarios tendrían una IP fija, sino que la mayoría se conectarían por módem, y cada vez que lo hicieran se les asignaría una IP distinta y sería imposible controlarlos. Entonces no resultaba muy útil este método y decidí controlar el reenvío de correo (*relay*) por dirección de correo origen. Esto es, que solo se dejarían enviar aquellos correos cuyo remitente figurase en una lista elaborada por nosotros. Es decir, si alguien intentase enviar un correo a *miamigo@cualquier.com* con remite *pepe@otro.org* utilizando nuestro servidor de correo, se le denegaría el envío, solo podría enviar emails con nuestro servidor si pusiera como remite usuario-*valido@dominio.org*.

Para llevar a cabo este control, editaremos el */etc/exim/exim.conf* y añadiremos lo siguiente:

```
# cp /etc/exim/exim.conf /etc/exim/exim.conf.old
# vi /etc/exim/exim.conf
```

Nos dirigiremos hacia *host_accept_relay* (para tenerlo un poco ordenado) y justo debajo escribiremos lo siguiente:

```
relay_match_host_or_sender
sender_address_relay = /etc/exim/lista_relay
```

Esto le indica a *exim* que utilice el método de control del *relay* por host o por dirección de origen, y que la lista de los usuarios permitidos para hacer *relay* se encuentra en */etc/exim/lista_relay*.

Por lo tanto solo nos queda añadir qué direcciones permitiremos que hagan *relay*:

```
# vi /etc/exim/lista_relay
```

Y escribir las direcciones al estilo "*mnicolau@dominio.org*". Aquí también se hizo un pequeño script para añadir a todos los usuarios del sistema en esa lista.

Ahora abriremos el puerto *25* en el router, como hemos hecho desde siempre.

Hasta aquí finalizaría la configuración de exim, y cualquier usuario desde su casa, configurando su cliente de correo electrónico adecuadamente, podría ya enviar correo con la dirección *mnicolau@dominio.org* tratándose del usuario *mnicolau*, por ejemplo.

Pero no acaba aquí, ¿de qué nos sirve enviar correo con un remite al que luego si intentan contestar, no puedan? Acabamos de configurar un servidor de correo saliente, para enviar correo solo, pero necesitamos también un servidor de correo entrante para poder recibir emails, para que la gente pueda responder a *mnicolau@dominio.org* si lo desea y que luego *mnicolau* pueda recoger dichos mensajes.

2.2 Servidor POP3

Aquí encontramos la solución. POP3 se encarga del correo entrante, de tal manera que si alguien envía un mail a *llmfabrega@dominio.org*, *POP3* recibe este email y lo guarda al directorio home del usuario correspondiente dentro de un fichero llamado *mbox* (en este caso se guardaría en */home/profes/llmfabrega/mbox*). Luego, este usuario, desde su casa podría conectarse al servidor *POP3* y tras identificarse con su login y su contraseña, recoger los mensajes que se encuentren en su *mbox*. Sí, *mbox* es un archivo de texto que guarda todos los emails recibidos, de tal manera que si deseas guardar los mails, se van acumulando en este fichero, y si más tarde quieres leer un mail de hace un mes, tendrás que hurgar en este fichero hasta encontrarlo. Por esto que el sistema de almacenaje en *mbox* es un poco precario.

Es por eso que existe otro sistema para almacenar los mails recibidos, es el sistema de *Maildir*. Este sistema almacena los mensajes recibidos en una carpeta llamada "new" dentro del directorio *Maildir* que se encuentra en el home de cada usuario. Pero no guarda los emails en un solo fichero como *mbox*, sino que cada email es un fichero. De esta manera resulta mucho más fácil trabajar con los emails recibidos.

Para pasar al sistema de *Maildir*, tendremos que editar de nuevo el archivo de configuración de *exim*:

```
# vi /etc/exim/exim.conf
```

Nos desplazaremos hasta encontrar la línea que ponga local_delivery: y haremos lo siguiente:

- Comentaremos (añadir un # al inicio de la línea)

```
file = /var/spool/mail/${local_part}
```

- Y añadiremos estas tres líneas:

```
directory=${home}/Maildir
maildir_format = true
prefix = ""
```

Quedando pues de la siguiente manera:

```
local_delivery:
driver = appendfile
group = mail
mode = 0660
mode_fail_narrower = false
envelope_to_add = true
return_path_add = true
#Añadimos lo siguiente
directory=${home}/Maildir
maildir_format = true
prefix = ""
# file = /var/spool/mail/${local_part}
```

Terminada esta parte, nos movemos un poco más abajo hasta encontrar *address_directory*: y allí descomentaremos la línea que dice # *maildir_format*, quedando esta parte así:

```
address_directory:
driver = appendfile
```

```
no_from_hack
prefix = ""
suffix = ""
maildir_format
```

Una vez hechos estos cambios podremos guardar y salir.

Ahora pasearemos a la instalación en sí de POP3.

Existen bastantes programas que ofrecen el servicio de *POP*, así que escogimos uno que me pareció sencillo de instalar. *ipopd* es el programa en cuestión. Tras investigar un poco, descubrí que existía un versión segura de este software. Cuando nos conectamos a un servidor *POP*, necesitamos suministrar una contraseña. Esta clave se envía normalmente como texto plano, es decir, que cualquier individuo que estuviera "escuchando" nuestra línea, podría hacerse fácilmente con la clave de acceso. Por eso se desarrollan versiones seguras del software, que cifran la contraseña de tal manera que sólo el destinatario pueda descifrarla. Así nos aseguramos que nadie pueda hacerse con nuestro password. Cabe decir que esta versión segura soporta también conexiones no seguras, por lo tanto "matamos dos pájaros de un tiro".

La versión segura de ipopd se llama ipopd-ssl, así que vayamos a instalarla:

```
# apt-get install ipopd-ssl
```

Una vez hecho esto, hay que abrir el puerto *110* del router y redirigirlo al *110* de nuestro servidor. Tendremos instalado nuestro servidor de *POP3*, y todos los usuarios del sistema gozarán de una cuenta de email para poder enviar y recibir su correo electrónico desde sus casas.

3. Cerrando

Posteriormente instalé un SAI. Además aparece un hub de la red por aquí en medio y encima de la torre un AP/Router de la red wireless que estoy construyendo para dar cobertura al centro y al resto del poblado. El tema wireless se está desarrollando, aún no se ha finalizado la red.

Y aquí doy por concluido este documento. Dudas, consejos y demás:

Pere Manils
nTn@DrDoom.net