

WebServer sobre ADSL Mini-Como

Paco Brufal <2:346/3.68> pbrufal@admisspotting.org

Versión: 0.1r1 Mayo 2001

Este pequeño documento explica cómo montar uno o varios servidores web en casa, usando una línea ADSL y GNU/Linux como Sistema Operativo. Este documento se distribuye SIN NINGUNA GARANTIA bajo la licencia GPL (<http://www.gnu.org>). No me responsabilizo de los posibles daños que pueda causar la ejecución de los pasos descritos en este documento.

1. Introducción

Ahora que hay bastante gente con líneas ADSL en España, se ha despertado la fiebre de 'montar un servidor', aprovechando que están conectados a Internet 24 horas al día. En este documento voy a explicar cómo montarte un servidor de páginas web con el mejor ;) sistema operativo: GNU/Linux. Para explicarlo me basaré en 2 cosas: La conexión ADSL es la que ofrece Infonegocio a través del router 3com 812 OfficeConnect, y la distribución de Linux usada es la GNU/Debian 2.2. Intentaré ir paso por paso, si alguien se pierde que levante la mano y pregunte :)

2. Requisitos

Para entender al 100% este documento se necesita tener conocimientos básicos de NAT (Network Address Translation, traducción de direcciones de red), para saber cómo funciona el NAT del router. Si alguna vez has configurado el masquerading de Linux, entonces sabrás lo que hablo :)

Otro requisito indispensable es tener un sistema Linux ejecutando Apache Web Server, y debe estar configurado correctamente. En la parte de configuración del servidor web, me referiré a una serie de ficheros. Posiblemente en otras distribuciones que no sean Debian el fichero puede estar en un directorio diferente, pero siempre se llamará de la misma manera.

Por último, tener un nombre de dominio asociado a la IP del router. Existen montones de servicios gratis de este tipo, <http://www.justlinux.com>, <http://www.dyndns.org>, etc... Teniendo un servicio de estos, ya no tendrás que recordar la IP del router, y el servidor web será más sencillo de configurar :) Si eres poseedor de un dominio, deberás configurar (o pedir que configuren) un DNS (cosa que no entra dentro de este doc) para que apunte a la IP pública del router. Otro dato a tener en cuenta es que nuestra IP es estática, no dinámica, así que no hará falta instalar ningún programa en nuestro servidor, simplemente pondremos en la web la IP y yastá :)

3. Configurando el router

Antes de empezar, vamos a entrar en la configuración del router y cambiaremos el login y el password. Con un navegador entramos en la dirección IP del router, p.ej. <http://192.168.0.1>. Como login y password introducimos **adminttd**. En estos últimos meses, Telefónica ha cambiado estos passwords, pero el login sigue siendo el mismo. Si no te funciona con **adminttd**, puedes probar con **ttdadmin** o con **infertilidAD** (ojo a las 2 mayúsculas).

Vamos al menú **Tools**, y pinchamos en **Login/password**. Añadimos (Add) un nuevo login y una nueva clave, volvemos al menú anterior y borramos los que trae por defecto el router. Pulsamos el botón de **Save configuration** una vez, esperamos a la nueva pantalla, y volvemos a pulsar.

Bien, ya tenemos el router protegido con nuestra clave. Ahora lo que necesitamos es configurar el router para que abra el puerto 80 y lo redirija a nuestra máquina. Es decir, el router tiene una IP pública (válida en Internet), cuando una persona decida ver nuestra página, pondrá la IP del router. Cuando a éste le llegue una

petición por el puerto 80, lo redireccionará al puerto 80 de nuestra máquina, que tiene una IP privada (no válida en Internet). Esto lo haremos entrando por telnet al router mediante el login y password que pusimos. Seguidamente borramos los filtros que instala la gente de Infonegocio. Todavía no sé muy bien qué hacen esos filtros, pero lo mejor es borrarlos :)

```
23:35:truck:gabber$ telnet 192.168.0.1 2323
Trying 192.168.0.1...
Connected to 192.168.0.1.
Escape character is '^]'.
login: milogin
Password: *****
3Com-DSL>list filters
```

Debe aparecer el nombre de los filtros (posiblemente sea **filtro** y **filtro2**. los borramos con la orden **del filter nombre**. Ahora vamos a cambiar el puerto de configuración por web del router para poder poner el nuestro.

```
3Com-DSL>del network service HTTPD
3Com-DSL>add network service http server_type HTTPD socket 8080
```

Con esto lo que hacemos es poner el puerto de configuración por web en el 8080. Al dejar el puerto 80 libre, ya podemos poner la redirección.

Volvemos a entrar con el navegador en la configuración, <http://192.168.0.1:8080>, ponemos nuestro login y password, y vamos siguiendo estos pasos:

```
Configuration
-> Remote Site Profiles
  -> seleccionar Internet
    -> Modify
      -> Next
        -> TCP
          -> Add
            * puerto público (80),
            * ip privada (la IP de nuestra máquina),
            * puerto privado (80)
          -> Add
            -> Save configuration
              -> Save configuration
```

Bien, ahora entraremos por telnet y abriremos el puerto 80 para que la gente se pueda conectar al router (recuerda que la gente de fuera ve nuestro servidor con la IP del router) ejecutaremos una serie de comandos:

```
3Com-DSL>add network service web1 server_type ClearTCPD socket 80
3Com-DSL>save all
3Com-DSL>reboot
```

Vamos a añadirle una pequeña **feature** a nuestro router. Vamos a decirle que envíe todos los logs a una máquina Linux, así podemos ver todo lo que ocurra. Para esto solo tendremos que ejecutar la siguiente orden:

```
3Com-DSL>set syslog 192.168.0.3 loglevel unusual
3Com-DSL>save all
3Com-DSL>reboot
```

Si te da algun error, prueba el siguiente comando:

```
3Com-DSL>add syslog 192.168.0.3 loglevel unusual
3Com-DSL>save all
3Com-DSL>reboot
```

Y en la máquina Linux que recibirá los logs arrancaremos el demonio `syslogd` con el parametro **-r**. En Debian esto se configura en el fichero `/etc/init.d/syslogd`:

```
SYSLOGD="-r"
```

Y reiniciamos el demonio: `/etc/init.d/syslogd restart`.

4. Configurando Apache Web Server

Ahora toca la parte del servidor web. Pondré 2 posibles configuraciones, una con un servidor simple, y otra con un servidor que alberga varios dominios sobre una misma IP (NameBased Virtual Hosts). Te recomiendo la página de Apache Web Server (<http://www.apache.org>) si deseas saber más sobre el tema.

4.1 Configuración básica: 1 IP y 1 nombre de dominio

Supongamos que hemos escogido un subdominio gratuito de estos de Internet, pongamos por caso truck.dyndns.org. En el DNS (Domain Name Server) de DynDns.org el host **truck** apunta a nuestra IP. Por lo tanto, cuando una persona ponga en su navegador **http://truck.dyndns.org**, se dirigirá al puerto 80 de nuestro router y el router redirige la conexión al puerto 80 del servidor web. Así de simple.

Para que todo funcione como es debido, buscaremos el fichero httpd.conf (en Debian está en /etc/apache), y en la línea **ServerName** pondremos el nombre que escogimos para el servidor: **truck.dyndns.org**. Como ves, el nombre del servidor web NO tiene por qué ser el mismo que el nombre del host. Esto es todo, no hay que hacer nada más :)

4.2 Configuración 'avanzada': 1 IP y varios nombre de dominio

Como sabrás, Apache soporta varios nombres de dominio bajo una misma IP, lo que se llama Servidores Virtuales basados en Nombre (NameBased virtual hosts). Lo explico de manera rápida. Apache es capaz de detectar qué dominio fue llamado, y mostrará la página correspondiente que tenga asignada ese dominio en la configuración. Vamos a suponer que en nuestro servidor queremos alojar 2 dominios: uno el gratuito y otro de pago, es decir, uno que hayamos comprado, p.ej, dominio2.com.

Para que www.dominio2.com funcione, necesitamos meterlo en un DNS. Para esto hay 2 soluciones, o eres administrador de un DNS y puedes añadir los dominios que quieras, o conoces a algún administrador que pueda hacerlo por ti.

Configurar el Apache no tiene ningún misterio. En el fichero **httpd.conf**, se añaden estas líneas y ya está :)

```
NameVirtualHost 192.168.0.3

<VirtualHost 192.168.0.3>
    ServerAdmin email@deladmm.com
    DocumentRoot /www/dominio1
    ServerName dominio1.dyndns.org
    ErrorLog /var/log/apache/dominio1.error
    CustomLog /var/log/apache/dominio2.access full
</VirtualHost>

<VirtualHost 192.168.0.3>
    ServerAdmin email@deladmin.com
    DocumentRoot /www/dominio2
    ServerName www.dominio2.com
    ErrorLog /var/log/apache/dominio2.error
    CustomLog /var/log/apache/dominio2.access full
</VirtualHost>
```

Explicación.

- La línea **NameVirtualHost** especifica que en la IP **192.168.0.3** (la IP interna de nuestro host, NO del router), va a haber dominios virtuales basados en nombre.
- Dentro de las directivas **VirtualHost** pondremos la configuración específica de cada dominio. Lo que ves arriba es la configuración mínima requerida para que funcione, luego tu ya le puedes poner una configuración más adecuada a tus necesidades, como añadir directorios para CGI, etc...
- La línea **ServerAdmin** especifica la dirección de correo del administrador
- El directorio indicado en **DocumentRoot** define el directorio donde se alojarán los fichero correspondientes al dominio en cuestión.
- la línea **ServerName** indica el nombre del servidor, esta línea es importante, ya que Apache identifica el dominio solicitado a través de esta directiva.
- En el directorio **ErrorLog** se guarda el log de errores.
- y en **CustomLog** se guarda el log de acceso con el formato **full**.

- Por último, la línea `</VirtualHost>` cierra la configuración del servidor virtual.

Y esto es todo, a partir de ahora puedes añadir tantos dominios como quieras. El orden en el que los pongas no influye para nada, excepto si se accede a tu servidor a través de la IP, en cuyo caso aparecerá el primer VirtualHost que tengas definido.

5. Histórico

15 Mayo 2001

- añadidas más passwords del router
- corregido el error de HTTPD (ziri@grupoandalus.com)
- añadida la opción de syslog

18 Septiembre 2001

- Corregido el tema del syslog (puede ser con **set** o con **add**)

6. Agradecimientos

A todos los que me conocen, y a toda la peña de Fidonet, que son los mejores :)