enchufado

Buscar RSS

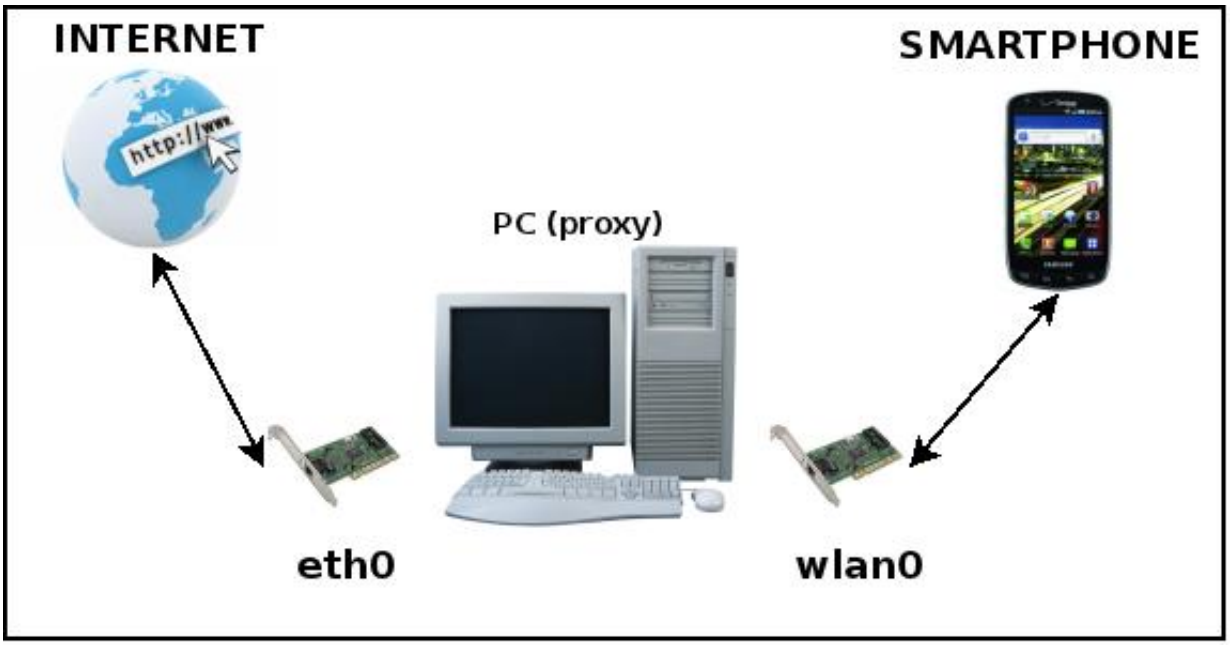
#

Crear un Punto de Acceso Wifi con Debian GNU/Linux

GNU/Linux 2011-10-05 02:48:58

Para ello usaremos la tarjeta wifi de un PC de sobremesa o portátil (que se pueda poner en modo *Master*) en lugar de un dispositivo AP (*Access Point*) dedicado. Esto está orientado a dar conexión a dispositivos con interfaz wifi.

¿En qué ocasiones puede esto sernos útil? Pues en aquellos casos en los que teniendo conexión a Internet en nuestro PC por cable, necesitamos Internet en un dispositivo que sólo pueda conectarse vía wifi (p.ej. en un smartphone Android sin 3G) y no tenemos ningún AP. Este sería el esquema de conexión:



Hay al menos 2 maneras de crear un AP. La primera y más sencilla es usar **iwconfig** para dejar la tarjeta wifi configurada para actuar como un AP (*iwconfig mode Master wlan0*). Pero si tenemos la mala suerte (aquí acostumbramos a tenerla) de que iwconfig no nos deja poner la tarjeta en modo Master, nos queda la segunda manera algo menos sencilla: usar hostap. ¿Y esto qué es? Pues llanamente, un driver + demonio que nos permitirán crear un AP con nuestra tarjeta wifi. Así pues, al grano.

Pasos:

1. Vamos a partir de una **configuración de red** con 2 interfaces en

un mismo PC: **eth0**, que supongamos que es la interfaz del PC con la que ya tenéis salida a Internet, debéis dejarla con la ip y máscara que ya tengáis actualmente (en mi caso, la ip 172.17.0.17 y la máscara de red 255.255.0.0). Y **wlan0**, que es la interfaz wifi del PC que *hará de AP* para que otros dispositivos puedan conectarse vía wifi. Esta última debemos adaptarla al mismo rango de red (o un subrango, según creamos). P.ej. en mi caso le puse la ip 172.17.30.1 y máscara de red 255.255.255.0. Aunque podría haberle puesto perfectamente una ip que estuviera en el mismo rango de red, opté por una subred porque el número de dispositivos que iban a conectarse a través de esta vía era limitado.

2. Con una Debian con kernel versión 2.6.29 o superior, **instalamos hostapd** (*apt-get install hostapd*) **y lo configuramos** como sigue:

```
interface=wlan0
driver=nl80211
ssid=Prueba
channel=1
macaddr_acl=0
auth_algs=1
ignore_broadcast_ssid=0
wpa=2
wpa_passphrase=mipassword
wpa_key_mgmt=WPA-PSK
wpa_pairwise=TKIP
rsn_pairwise=CCMP
```

   Esto configura un AP con la interfaz wlan0 con identificador "Prueba" por el canal 1. Y la segunda parte (desde el parámetro *macaddr_acl* hasta el final), en síntesis, configura una autenticación abierta protegida por contraseña y cifrada con WPA2. Esto lo pondremos en */etc/hostapd/hostapd.conf*. Además, modificaremos el script de inicio (*/etc/init.d/hostapd*) para que la variable DAEMON_CONF apunte a este archivo que acabamos de crear (*DAEMON_CONF=/etc/hostapd/hostapd.conf*).

3. **Iniciamos hostapd** (*/etc/init.d/hostapd start*). Con esto, si buscamos redes wifi ya deberíamos ver el identificador de la red ("Prueba", en nuestro caso) desde el dispositivo wifi a conectar. Pero después de autenticarnos (recordad indicarle la password que hemos definido), se quedará tratando de obtener una ip eternamente, así que continuamos.

4. Para que se lleve a cabo esta asignación de ip a nuestro dispositivo wifi, **necesitamos un servidor DHCP**. Así pues, instalamos dhcpd (*apt-get install isc-dhcp-server*), le indicamos que sirva por el interfaz wifi (*INTERFACES="wlan0"* en el archivo */etc/default/isc-dhcp-server*) y le creamos una configuración mínima. Esto sería el contenido del archivo */etc/dhcp/dhcpd.conf*:

```
default-lease-time 600;
max-lease-time 7200;
```

```
subnet 172.17.30.0 netmask 255.255.255.0 {
  # Opcional (seguridad): No queremos clientes que no tengamos
registrados.
  deny unknown-clients;
  range 172.17.30.2 172.17.30.2;
  option routers 172.17.0.17;
  option subnet-mask 255.255.255.0;
  option domain-name-servers 8.8.8.8;
  # Opcional (seguridad): El único cliente registrado.
  host htc_pepe { hardware ethernet ff:05:04:03:02:01; fixed-
address 172.17.30.2; }
}
```

Esta configuración establece una red de clase C (172.17.30.0) para servir únicamente la ip 172.17.30.2 (range) al dispositivo wifi. Si queremos servir más ips, ampliaremos el rango (p.ej. range 172.17.30.2 172.17.30.101 para cubrir 100 dispositivos). Las configuraciones de *options* son datos necesarios de configuración de red que se pasarán al dispositivo, tales como enrutador (routers), máscara de red (subnet-mask) y sevidores de nombres (domain-name-servers). En este último caso, usamos uno de los dns públicos de Google (8.8.8.8). **Es importante entender que el router será la ip de la interfaz de nuestro PC con la que éste salga a Internet** (en mi caso, la ip 172.17.0.17, la asignada a la interfaz eth0), puesto que es quien hará de *proxy*. Finalmente, observad las lineas *deny unknown-clients;* y *host htc_pepe...*, que se encargaran de denegar las peticiones de obtención de ip a las MAC's no registradas y aceptar las peticiones de obtención de ip a las MAC's registradas, respectivamente. Son opcionales de cara a incrementar la seguridad, y si se prescinde de ellas, deben quitarse ambas. Reiniciaremos el servidor (*/etc/init.d/isc-dhcp-server restart*) para que lea y aplique los cambios.

5. Ahora **necesitamos una manera de compartir la conexión a Internet**. Así pues, con iptables hacemos nat/masquerading para que la navegación salga a Internet como si fuera hecha desde el propio PC, pero de puertas para adentro vaya a quién la realiza realmente (es decir, el dispositivo wifi). Haremos un script con lo siguiente:

```
# Habilitamos el ip forwarding.
echo 1 > /proc/sys/net/ipv4/ip_forward
# Hacemos limpieza.
iptables -F
iptables -t nat -F
iptables -X
# Masquerading y forwarding.
iptables -t nat -A POSTROUTING -o eth0 -s 172.17.30.2 -j
MASQUERADE
iptables -A FORWARD -i wlan0 -j ACCEPT
# Aceptamos pings (opcional).
iptables -A INPUT -i wlan0 -p ICMP -j ACCEPT
# Aceptamos tambien paquetes de conexiones ya establecidas
(opcional: seguridad).
iptables -A INPUT -i wlan0 -p TCP -m state --state NEW,RELATED -j
ACCEPT
```

```
# Rechazamos paquetes de conexiones nuevas (opcion      uridad).
iptables -A INPUT -i ppp0 -m state --state NEW,INVALID -j DROP
# Rechazamos paquetes de forwarding de conexiones no establecidas
(opcional: seguridad).
iptables -A FORWARD -i ppp0 -m state --state NEW,INVALID -j DROP
```

Esto lo guardaremos en un script tal que *iptables_nat.sh*, le daremos permisos de ejecución y lo ejecutaremos. Como en el caso de la configuración del DHCP, las sentencias marcadas como "opcional: seguridad" se pueden sacar (todas a la vez), aunque se recomienda dejarlas.

6. Finalmente, **añadiremos en nuestro PC la ruta para sepa hacia dónde enrutar el tráfico para el rango de red de nuestros clientes wifi** (es decir, hacia el rango 172.17.30.0):

```
route add -net 172.17.30.0 netmask 255.255.255.0 gw 172.17.30.1
dev wlan0
```

Y con esto, ya lo tendríamos todo listo. Otra aplicación que se me ocurre para esto es la de dejar el AP abierto y *esnifar* el tráfico de todo aquel incauto que se conecte. Ya sabéis, con fines puramente didácticos ;)

Comentarios (10)

Volver al indice