Bisoños Usuarios de GNU/Linux de Mallorca y Alrededores | Bergantells Usuaris de GNU/Linux de Mallorca i Afegitons

Compartir la conexión con un Cablemodem (34095 lectures)

Per **Alejandro Sobrino Beltrán**, ***Jander*** (http://tuxitos.es)

Creado el 29/08/2003 20:56 modificado el 29/08/2003 20:56

A pesar de la gran variedad de artículos sobre el tema que hay en BULMA, se repiten constantemente preguntas en la Bulmailing sobre lo mismo.

A continuación intentaré explicar como compartir la conexión a internet a través de un ejemplo concreto, mediante iptables (masquerading). Además se introducirá un pequeño firewall con unas sencillas reglas básicas.

Introducción

Hace más de un año que me encontré ante la necesidad de compartir mi conexión a internet. ONO ofrece algo similar al multipuesto de las ADSL, pero cobra un extra por ello. Así que cuando tuve mi segunda máquina en casa, me las tuve que ingeniar de alguna manera para compartir la conexión.

Me basaré en la conexión de ONO, pero no debería dar ningún problema con otra conexión de cable, ADSL, RDSI o PPP. Tan solo habrá que cambiar la interfaz conectada a internet, así como la configuración de dicha interfaz.

Lo que tengo montado en casa y que intentaré explicaros es básicamente esto:

El servidor tiene dos tarjetas de red, eth0 y eth1. eth0 será la interfaz conectada a internet a través del cablemodem (DHCP), y la eth1 (IP estática) estará conectada al hub/switch de la red de casa, en donde estarán conectados los "clientes".

La configuración de los clientes será sencilla, ya que bastará con añadir la IP del servidor como *default gateway* y listo :)

Ingredientes para la receta

Para llevar todo a cabo, necesitaremos compilar un kernel con soporte para iptables y masquerading. Así mismo, necesitaremos el paquete iptables instalado en nuestro sistema.

Para instalar iptables en Debian bastará con un:

```
apt-get install iptables
```

En gentoo:

```
emerge iptables
```

En otra distribución, siempre nos podemos bajar el código fuente y compilarlo nosotros mismos: aquí[1].

Lo siguiente que deberemos realizar, será conseguir un kernel con soporte para iptables. Para ello, nos podemos bajar cualquiera de la serie 2.4: aquí[2]. En el 2.6.0-test3 aún no he tenido tiempo de probarlo, pero supongo que no debería haber ningún inconveniente. Como siempre, si podéis, contad vuestras experiencias :)

De todas maneras, como siempre, en Debian podemos hacer un:

```
apt-get install kernel-source-2.4.21
```

Compilando el kernel

Actualmente ando sobre un kernel 2.4.21-ac4, y a continuación intentaré mostrar las opciones básicas que tengo marcadas.

En *Networking options*:
Imagen: aquí[3]

```
<*> Packet socket
[*]   Packet socket: mmapped IO
< > Netlink device emulation
[*] Network packet filtering (replaces ipchains)
[ ]   Network packet filtering debugging
[*] Socket Filtering
<*> Unix domain sockets
[*] TCP/IP networking
[*]   IP: multicasting
[ ]   IP: advanced router
[ ]   IP: kernel level autoconfiguration
< >   IP: tunneling
< >   IP: GRE tunnels over IP
[ ]   IP: multicast routing
[ ]   IP: ARP daemon support (EXPERIMENTAL)
[ ]   IP: TCP Explicit Congestion Notification support
[*]   IP: TCP syncookie support (disabled per default)
IP: Netfilter Configuration  --->
```

En *Networking options -> IP: Netfilter Configuration*:
Imagen: aquí[4] y aquí[5]

```
<M> Connection tracking (required for masq/NAT)
<M>   FTP protocol support
[...]
<M>   IRC protocol support
[...]
<M> IP tables support (required for filtering/masq/NAT)
<M>   limit match support
<M>   MAC address match support
[...]
<M>   netfilter MARK match support
<M>   Multiple port match support
[...]
<M>   tcpmss match support
[...]
<M>   Connection state match support
[...]
<M>   Unclean match support (EXPERIMENTAL)
[...]
<M>   Packet filtering
<M>     REJECT target support
<M>     MIRROR target support (EXPERIMENTAL)
<M>   Full NAT
<M>     MASQUERADE target support
[...]
<M>   Packet mangling
[...]
<M>   LOG target support
[...]
<M>   TCPMSS target support
[...]
```

Tened en cuenta que muchas de estas opciones NO son necesarias para el objetivo del artículo. Tan solo pongo parte de mi *.config*.

Bien, ahora solo faltará recompilar el kernel:

```
make dep && make clean && make bzImage modules modules_install
```

Editar el /etc/lilo.conf, y ejecutar /sbin/lilo.

Script de iptables

Para facilitarme la vida me creé un script que se ejecuta al arrancar la máquina. El script en cuestión es el siguiente:

```
#!/bin/bash

IPTABLES=/sbin/iptables
# Interfaz conectada a internet
EXT="eth0"
# Interfaz conectada a la red interna
INT="eth1"

case "$1" in
        start)
                # Cargamos los módulos necesarios
                echo "Setting firewall rules..."
                echo -n "Loading kernel modules: "
                /sbin/insmod -q -s ip_tables
                /sbin/insmod -q -s ip_conntrack
                /sbin/insmod -q -s ip_conntrack_ftp
                /sbin/insmod -q -s ip_conntrack_irc
                /sbin/insmod -q -s iptable_nat
                /sbin/insmod -q -s ip_nat_ftp
                echo "done"

                # Activamos el IP forwarding
                echo -n "Activating IP Forwarding support: "
                echo "1" > /proc/sys/net/ipv4/ip_forward
                echo "done"

                # Eliminamos las reglas anteriores
                echo -n "Deleting firewall rules: "
                $IPTABLES -F INPUT
                $IPTABLES -F OUTPUT
                $IPTABLES -F FORWARD
                $IPTABLES -F
                $IPTABLES -t nat -F
                echo "done"

                echo -n "Activating NAT: "
                $IPTABLES -t nat -A POSTROUTING -s 192.168.0.0/24 -d 0.0.0.0/0 -o $EXT -j MASQUERADE
                echo "done"

                echo -n "Activating ICMP echo request: "
                $IPTABLES -A INPUT -i $EXT -p ICMP -j ACCEPT
                echo "done"

                echo -n "Setting firewall port rules: "

                # 21: ftp
                $IPTABLES -A INPUT -i $EXT -p TCP --dport 21 -m state --state NEW -j ACCEPT
                $IPTABLES -A INPUT -i $EXT -p TCP --dport 2100 -m state --state NEW -j ACCEPT

                # 22: ssh
                $IPTABLES -A INPUT -i $EXT -p TCP --dport 22 -m state --state NEW -j ACCEPT

                # 25: smtp
                $IPTABLES -A INPUT -i $EXT -p TCP --dport 25 -m state --state NEW -j ACCEPT

                # 80: apache
                $IPTABLES -A INPUT -i $EXT -p TCP --dport 80 -m state --state NEW -j ACCEPT

                echo "done"

                echo -n "Final approach: "
                # Aceptamos paquetes de una conexión ya establecida
```

```
                $IPTABLES -A INPUT -p TCP -m state --state RELATED -j ACCEPT

                # Rechazamos los de conexiones nuevas
                $IPTABLES -A INPUT -i $EXT -m state --state NEW,INVALID -j DROP

                # Rechazamos conexiones de forwarding no establecidas
                $IPTABLES -A FORWARD -i $EXT -m state --state NEW,INVALID -j DROP

                echo "done"
                ;;

        stop)
                echo -n "Stopping firewall: "
                $IPTABLES -F INPUT
                $IPTABLES -F OUTPUT
                $IPTABLES -F FORWARD
                $IPTABLES -F
                $IPTABLES -t nat -F
                echo "done"
                ;;

        restart)
                $0 stop
                echo -n "Sleeping a few seconds before setting the rules again: "
                sleep 2
                echo "done"
                $0 start
                ;;

        status)
                $IPTABLES -L
                $IPTABLES --table nat --list --exact --verbose --numeric --line-numbers
                ;;

        *)
                echo "Usage: $0 {start|stop|restart|status}"
                exit 1
esac
exit 0
```

Debo dar las gracias a Primetime[6] por mejorar mi antiguo script, ya que lo usamos para el servidor de la Balearikus Party[7].

Bastará copiar el script (firewall.sh[8]) al directorio /etc/init.d/ y crear el enlace en /etc/rcX.d/ para que se ejecute al inicio.

Configuración

En los clientes tan solo hará falta añadir la IP del servidor (en este ejemplo la IP de la eth1) como *default gateway:*:

```
                route add default gw 192.168.0.1
```

Y ya está! Tan solo faltará adaptar el script a vuestras necesidades, así como realizar algunas pruebas desde los clientes :)

Enlaces de interés

- Artículo de carcoco:
  - Configurar un Proxy/Router/Gateway con el Kernel 2.4.x e Iptables[9]
- Artículo de Gigi:
  - Instalar un servidor con el CableModem de ONO[10]
- Artículo de Gravis:
  - Enrutado en base a marcas de paquetes. Iproute + Iptables[11]
- Artículo de carcoco:

- ♦ Ponle un Firewall a tu Linux. Iptables[12]

- Artículo de gallir:
  - ♦ iptables y NAT para vagos[13]

- Linux IP Masquerade HOWTO:
  - ♦ Linux IP Masquerade HOWTO[14]

Lista de enlaces de este artículo:

1. http://www.netfilter.org/
2. http://www.kernel.org/pub/linux/kernel/v2.4/
3. http://bulma.net/~jander/bulma/kernel1.png
4. http://bulma.net/~jander/bulma/kernel2.png
5. http://bulma.net/~jander/bulma/kernel3.png
6. http://bulma.net/todos.phtml?id_autor=123
7. http://balearikus-party.org
8. http://bulma.net/~jander/bulma/firewall.sh
9. http://bulma.net/body.phtml?nIdNoticia=1140
10. http://bulma.net/body.phtml?nIdNoticia=387
11. http://bulma.net/body.phtml?nIdNoticia=1615
12. http://bulma.net/body.phtml?nIdNoticia=861
13. http://bulma.net/body.phtml?nIdNoticia=1522
14. http://www.ibiblio.org/pub/Linux/docs/HOWTO/other-formats/html_single/IP-Masquer

---

E-mail del autor: jander _ARROBA_ mallorcaweb.net
Podrás encontrar este artículo e información adicional en: http://bulma.net/body.phtml?nIdNoticia=1860