GNU/Linux

Autores:

- Pablo Boronat Pérez (Universitat Jaume I)
- Miguel Pérez Francisco (Universitat Jaume I)
- David Rubert Viana (Universitat Jaume I)

Introducción

De la wikipedia (http://es.wikipedia.org/wiki/GNU/Linux):

«GNU/Linux es uno de los términos empleados para referirse a la combinación del núcleo o kernel libre similar a Unix denominado Linux, que es usado con herramientas de sistema GNU. Su desarrollo es uno de los ejemplos más prominentes de software libre; todo su código fuente puede ser utilizado, modificado y redistribuido libremente por cualquiera bajo los términos de la GPL (Licencia Pública General de GNU, en inglés: General Public License) y otra serie de licencias libres.

A pesar de que Linux es, en sentido estricto, el sistema operativo, parte fundamental de la interacción entre el núcleo y el usuario (o los programas de aplicación) se maneja usualmente con las herramientas del proyecto GNU o de otros proyectos como GNOME. Sin embargo, una parte significativa de la comunidad, así como muchos medios generales y especializados, prefieren utilizar el término Linux para referirse a la unión de ambos proyectos.»

GNU/Linux

Una distribución GNU/Linux se define como un conjunto de programas que permiten tanto la instalación en el ordenador del Sistema Operativo Linux como su uso posterior. Su objetivo es facilitar la instalación, la configuración y el mantenimiento de un sistema GNU/Linux. Integra un núcleo, un conjunto de aplicaciones de sistema y una colección de programas de usuario listos para instalar. Son como los helados que están todos hechos con la misma materia prima y los hay de muchos sabores. Cada sabor será una distribución GNU/Linux.

Para más información consultar ¿qué es GNU/Linux?

En este capítulo se presenta de forma práctica cómo configurar un servidor básico para conectarlo a la red guifi.net. Lo que se presenta en este capítulo se centra en la distribución Debian (también es válido para Ubuntu) por ser una de las distribuciones más estables y robustas. En otras distribuciones lo que se comenta aquí será igualmente válido aunque pueden variar algunos ficheros de configuración y su localización.

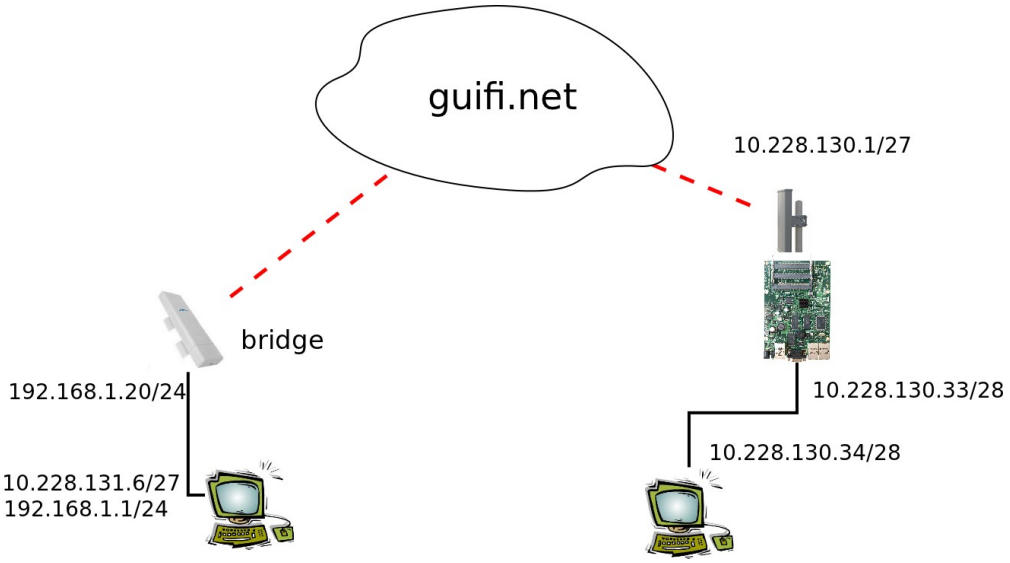
Se puede encontrar información sobre cómo instalar Debian en http://www.debian.org/distrib/ y sobre Ubuntu en http://www.ubuntu.com/download.

Un servidor puede estar conectado de muchas maneras a guifi.net, a continuación comentamos algunas de ellas:

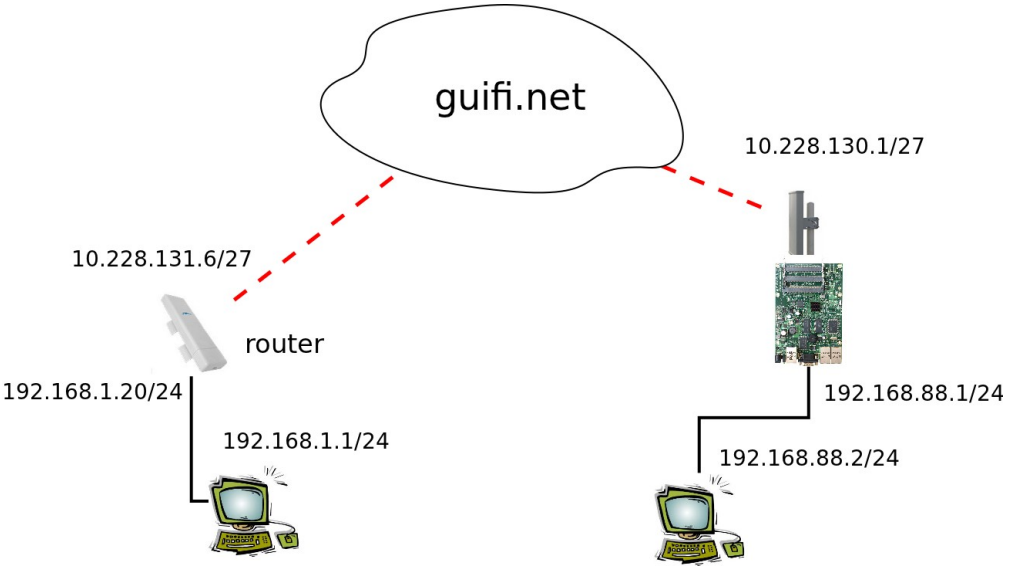
- El servidor está en un supernodo (mikrotik, ALIX, buffalo, ...)
  - conectado a una de las ethernets del supernodo con una IP pública.
  - conectado a una de las ethernets del supernodo con una IP privada y en el supernodo se redireccionan puertos al servidor (aquellos en los que se quieran ofertar servicios: 80 para la web, 22 para el ssh, ...).

- El servidor está en un nodo cliente (nanostation o similar)
  - El dispositivo está en modo bridge y la IP pública del nodo cliente la tiene el servidor. El dispositivo tiene una IP administrativa (pública o privada).
  - El dispositivo está en modo router con una IP pública, el servidor tiene una IP privada y se redireccionan puertos del nodo cliente al servidor.

La siguiente imagen muestra cómo se pondría un servidor (tanto en un supernodo como en un nodo cliente) utilizando IPs públicas.



La siguiente imagen muestra cómo se pondría un servidor (tanto en un supernodo como en un nodo cliente) utilizando IPs privadas. En ambos casos habría que redireccionar los puertos que se quieran ofrecer servicios del nodo guifi.net al servidor.



Ejercicio 5.1

Configuración de red

Interfaces de red

Un equipo Debian puede tener diversas interfaces con una o varias direcciones IP. Las interfaces pueden ser de diferentes tipos:

- Loopback: lo
- Ethernet: eth0, eth1, ...
- Wi-Fi: wlan0, wlan1, ath0, ...
- ...

El dispositivo de red *loopback* es una interfaz de red virtual. Se utiliza habitualmente la 127.0.0.1 (debe estar en la 127.0.0.0/8). Esta dirección se suele utilizar cuando una transmisión de datos tiene como destino el propio host. También en tareas de diagnóstico de conectividad y validez del protocolo de comunicación. La dirección de loopback crea un método de acceso directo para las aplicaciones y servicios TCP/IP que se ejecutan en el mismo dispositivo para comunicarse entre sí. Al utilizar la dirección de loopback en lugar de la dirección host IPv4 asignada, dos servicios en el mismo host pueden desviar las capas inferiores de la pila de protocolos TCP/IP. También es posible hacer ping a la dirección de loopback para probar la configuración de TCP/IP en el host local.

En este enlace, http://www.debian.org/doc/manuals/debian-reference/ch-gateway.es.html, está explicado con detalle todas las posibilidades de configuración de los dispositivos de red en Debian, utilizando los comandos `ifconfig` e `ip`.

Las herramientas tradicionales de configuración de red a bajo nivel en sistemas GNU/Linux son los programas `ifconfig` y `route` que vienen en el paquete `net-tools`. Estas herramientas han sido oficialmente reemplazadas por `ip` que viene en el paquete `iproute`. El programa `ip` funciona con Linux 2.2 y superior y es más potente que las herramientas anteriores.

La orden ifconfig sin parámetros muestra la configuración actual de las interfaces de red del sistema.

`/sbin/route -n`Para asignarle una IP a una interfaz de red se utiliza la siguiente orden:

```
ifconfig eth0 10.228.131.6 netmask 255.255.255.224
```

```
root
```

```
su -
```

```
sudo -i
```

Una interfaz de red puede tener más de una IP asignada, es lo que se conoce como *IP aliasing*. Para darle una segunda IP a una interfaz de red se utiliza la siguiente orden:

```
ifconfig eth0:2 192.168.1.1 netmask 255.255.255.0
```

Ejercicio 5.2

Rutas

Para conocer las rutas del ordenador se utiliza la orden `route` sin parámetros. Es conveniente utilizar la opción -n para que nos muestre la información con valores númericos, sin resolver los nombres.

```
$ route -n

Kernel IP routing table
Destination     Gateway         Genmask           Flags Metric Ref
Use Iface
10.228.131.0    0.0.0.0         255.255.255.248   U     0      0
0 eth0
192.168.1.0     0.0.0.0         255.255.255.0     U     2      0
0 eth0
192.168.2.0     0.0.0.0         255.255.255.0     U     2      0
0 eth1
0.0.0.0         10.228.131.1    0.0.0.0           UG    0      0
0 eth0
```

Muestra que cualquier ordenador en la `10.228.131.0/27` se enviará por `eth0`, igual que para `192.168.1.0/24`. Los ordenadores en la red `192.168.2.0/24` se enviará por `eth1`. La ruta por defecto (hacia donde se enviará cualquier cosa de una subred a la que el ordenador no está directamente conectado) es el router 10.228.131.1

Para añadir una ruta estática se utiliza la orden

```
route add -net 10.0.0.0 netmask 255.0.0.0 gw 10.228.131.1
```

que, en este ejemplo, añade una ruta que envía todo lo que vaya dirigido a una red que empiece por `10.` hacia el `10.228.131.1` (el ordenador debe tener acceso directo al gateway que se pone como destino).

Para borrar una ruta estática se utiliza

```
route del -net 10.0.0.0 netmask 255.0.0.0 gw 10.228.131.1
```

Para poner la ruta por defecto

```
route add default gw 192.168.2.1
```

Para borrarla

```
route del default gw 192.168.2.1
```

Ejercicio 5.3

Ficheros de configuración

Las órdenes anteriores sirven para establecer «al vuelo» direcciones IP y rutas, al apagar el ordenador se perderán los cambios que se hayan realizado. Para hacer los cambios permanentes en Debian se utiliza el fichero `/etc/network/interfaces`. A continuación se muestra un ejemplo de un ordenador con dos interfaces, una de ellas con dos IPs y con una ruta estática hacia guifi.net.

```
# The loopback network interface
auto lo
```

```
iface lo inet loopback

auto eth2
iface eth2 inet static
        address 150.128.97.38
        netmask 255.255.255.0
        network 150.128.97.0
        broadcast 150.128.97.255
        # Ruta por defecto
        gateway 150.128.97.1

auto eth0
iface eth0 inet static
     address 10.228.130.162
     netmask 255.255.255.248
     network 10.228.130.160
        broadcast 10.228.130.167

#Ip para el ADSL
auto eth0:2
iface eth0:2 inet static
     address 192.168.1.1
     netmask 255.255.255.0
     network 192.168.1.0
     broadcast 192.168.1.255
        # gateway 192.168.1.3 # Sólo se debe poner una ruta por
defecto

# Ruta para guifi.net castellon
up route add -net 10.228.0.0/16 gw 10.228.130.161
```

Al arrancar el ordenador tendremos las siguientes IPs y rutas

```
$/sbin/ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0d:56:11:70:28
          inet addr:10.228.130.162  Bcast:10.228.130.167
Mask:255.255.255.248
          inet6 addr: fe80::20d:56ff:fe11:7028/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:30082607 errors:0 dropped:0 overruns:0
frame:0
          TX packets:20592680 errors:0 dropped:0 overruns:0
carrier:0
          collisions:0 txqueuelen:100
          RX bytes:4018116822 (3.7 GiB)  TX bytes:2866353243 (2.6
GiB)

eth2      Link encap:Ethernet  HWaddr 00:a0:c9:ca:6e:a6
          inet addr:150.128.97.38  Bcast:150.128.97.255
Mask:255.255.255.0
          inet6 addr: fe80::2a0:c9ff:feca:6ea6/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:25315096 errors:0 dropped:0 overruns:0
```

```
frame:0
          TX packets:15026549 errors:0 dropped:0 overruns:0
carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1839459667 (1.7 GiB)  TX bytes:2107618688 (1.9
GiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:1588706 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1588706 errors:0 dropped:0 overruns:0
carrier:0
          collisions:0 txqueuelen:0
          RX bytes:4183812501 (3.8 GiB)  TX bytes:4183812501 (3.8
GiB)

eth0:2    Link encap:Ethernet  HWaddr 00:04:23:23:82:e6
          inet addr:192.168.1.42  Bcast:192.168.1.255
Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1


$ /sbin/route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref
Use Iface
10.228.130.160  0.0.0.0         255.255.255.248 U     0      0
0 eth0
150.128.97.0    0.0.0.0         255.255.255.0   U     0      0
0 eth2
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0
0 eth0
10.228.0.0      10.228.130.161  255.255.0.0     UG    0      0
0 eth0
0.0.0.0         150.128.97.1    0.0.0.0         UG    0      0
0 eth2
```

Ejercicio 5.4

Network manager

El Network manager es una aplicación gráfica que permite definir la configración de red a través de menús (http://wiki.debian.org/NetworkManager). Suele tener un icono en el escritorio del usuario que permite configurarlo, siempre que se tengan permisos. Utiliza ficheros de configuración independientes a los comentados en los apartados anteriores y se puede indicar qué interfaces puede y cuales no manejar.

En servidores con configuraciones de red fijas y estables se puede desinstalar y configurar como se ha explicado anteriormente.

Puertos TCP/UDP

Según la wikipedia (http://es.wikipedia.org/wiki/Puerto_de_red):

«Un puerto de red es una interfaz para comunicarse con un programa a través de una red. Un puerto suele estar numerado. La implementación del protocolo en el destino utilizará ese número para decidir a qué programa entregará los datos recibidos. Esta asignación de puertos permite a una máquina establecer simultáneamente diversas conexiones con máquinas distintas, ya que todos los paquetes que se reciben tienen la misma dirección, pero van dirigidos a puertos diferentes.

Los números de puerto se indican mediante una palabra, 2 bytes (16 bits), por lo que existen 65535»

Algunos de los puertos están asignados por la IANA (*Internet Assigned Numbers Authority*) a determinadas aplicaciones (aunque se pueden modificar). Se puede consultar los principales puertos en http://es.wikipedia.org/wiki/Anexo:N%C3%BAmeros_de_puerto

Ejercicio 5.5

Ejercicio 5.6 (a entregar)

Cortafuegos. iptables

De la wikipedia (http://es.wikipedia.org/wiki/Netfilter/iptables):

«*Netfilter* es un *framework* disponible en el núcleo Linux que permite interceptar y manipular paquetes de red. Dicho framework permite realizar el manejo de paquetes en diferentes estados del procesamiento. Netfilter es también el nombre que recibe el proyecto que se encarga de ofrecer herramientas libres para cortafuegos basados en Linux.

El componente más popular construido sobre Netfilter es *iptables*, una herramienta de cortafuegos que permite no solamente filtrar paquetes, sino también realizar traducción de direcciones de red (NAT) para IPv4 o mantener registros (*logs*). El proyecto Netfilter no sólo ofrece componentes disponibles como módulos del núcleo sino que también ofrece herramientas de espacio de usuario y bibliotecas.

iptables es el nombre de la herramienta de espacio de usuario mediante la cual el administrador puede definir políticas de filtrado del tráfico que circula por la red. El nombre iptables se utiliza frecuentemente de forma errónea para referirse a toda la infraestructura ofrecida por el proyecto Netfilter. Sin embargo, el proyecto ofrece otros subsistemas independientes de iptables tales como el *connection tracking system* o sistema de seguimiento de conexiones, que permite encolar paquetes para que sean tratados desde el espacio de usuario. iptables es un software disponible en prácticamente todas las distribuciones de Linux actuales.»

En http://www.pello.info/filez/firewall/iptables.html se explica con ejemplos qué es un firewall y el funcionamiento de iptables.

En estos apuntes sólo se pretende dar un ejemplo de cómo configurar un sencillo cortafuegos para abrir algunos puertos en un servidor conectado a guifi.net y denegar el acceso al resto. A continuación se muestra un ejemplo

con comentarios que explican cada una de las órdenes.

```
#!/bin/sh

IPTABLES=/sbin/iptables

if [ ! -x $IPTABLES ]; then
  exit 0
fi


# 1) Definir la norma por defecto de la tabla filter (la tabla por
omisión), en este caso se acepta todo
$IPTABLES -P INPUT ACCEPT
$IPTABLES -P FORWARD ACCEPT
$IPTABLES -P OUTPUT ACCEPT

# Se borran todas las reglas anteriores de iptables de la tabla
filter
$IPTABLES -F
# Se borran todas las reglas anteriores de iptables de la tabla
nat
$IPTABLES -F -t nat

#Se hace source nat de todo lo que sale por la interfaz eth0, sale
con la IP 10.228.130.162
$IPTABLES -t nat -A POSTROUTING -o eth0 -j SNAT --to
10.228.130.162

#Se acepta todo lo que llega cuyo destino es 10.228.130.162
$IPTABLES -I INPUT -d 10.228.130.162/32 -j ACCEPT

#Se acepta todo el tráfico local
$IPTABLES -A INPUT -i lo -j ACCEPT

# Se aceptan paquetes en estado establecido y relacionado
$IPTABLES -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

#Se aceptan los pings
$IPTABLES -I INPUT -p icmp -j ACCEPT

#Se aceptan conexiones nuevas en los puertos indicados
# ssh
$IPTABLES -A INPUT -m state --state NEW -p tcp --dport 22 -j
ACCEPT
# smtp
$IPTABLES -A INPUT -m state --state NEW -p tcp --dport 25 -j
ACCEPT
# DNS, observese que se abre el puerto tanto en TCP como en UDP
$IPTABLES -A INPUT -m state --state NEW -p tcp --dport 53 -j
ACCEPT
$IPTABLES -A INPUT -m state --state NEW -p udp --dport 53 -j
```

```
ACCEPT
# web, http
$IPTABLES -A INPUT -m state --state NEW -p tcp --dport 80 -j
ACCEPT
# web segura, https
$IPTABLES -A INPUT -m state --state NEW -p tcp --dport 443 -j
ACCEPT
# VNC, escritorio remoto
$IPTABLES -A INPUT -m state --state NEW -p tcp --dport 5901 -j
ACCEPT
# NTP, sincronización horaria, se abre en UDP
$IPTABLES -A INPUT -p udp --dport 123 -j ACCEPT

# Se abre el puerto 1194 de udp, se utiliza para túneles con
OpenVPN
$IPTABLES -A INPUT -p udp --dport 1194 -j ACCEPT

# Se permite el protocolo 89, el que utiliza ospf (no gasta ni TCP
ni UDP).
$IPTABLES  -A INPUT -p 89 -j ACCEPT

# Cualquier otra cosa que no se haya aceptado anteriormente se
deniega
$IPTABLES -A INPUT -j DROP
```

Este fichero puede adaptarse fácilmente a las necesidades de cualquier servidor.

Para que este fichero se ejecute al arrancar el ordenador se puede poner (en la distribución Debian o Ubuntu) en el directorio `/etc/network/if-up.d/` y darle permisos de ejecución

Existen herramientas gráficas para administrar un cortafuegos con iptables, por ejemplo GUFW desarrollado por la comunidad Ubuntu y disponible también para Debian.

Ejercicio 5.7 (a entregar, voluntario)DNS

Túneles

De la wikipedia (http://es.wikipedia.org/wiki/T%C3%BAnel_%28inform%C3%A1tica%29):

«Se conoce como túnel al efecto de la utilización de ciertos protocolos de red que encapsulan a otro protocolo. Así, el protocolo A es encapsulado dentro del protocolo B, de forma que el primero considera al segundo como si estuviera en el nivel de enlace de datos. La técnica de tunelizar se suele utilizar para trasportar un protocolo determinado a través de una red que, en condiciones normales, no lo aceptaría. Otro uso de la tunelización de protocolos es la creación de diversos tipos de redes privadas virtuales.»

A efectos prácticos, un túnel permite crear un «enlace vitual» (cifrado o no) entre dos nodos, no necesariamente conectados directamente. Una vez creado el túnel los dos nodos disponen de un enlace directo a través de él.

En GNU/Linux existen paquetes para prácticamente todos los tipos de túneles. En este apartado nos centraremos en túneles OpenVPN por tratarse de un software libre multiplataforma (disponible en Linux, Openwrt, windows, mikrotik, android, ...) que ofrece cifrado de datos y autenticación de usuarios mediante usuario y contraseña y/o certificados.

En este ejemplo se crea un túnel cifrado entre dos máquinas utilizando una clave estática. El servidor suponemos que tiene una IP fija y el cliente no. El cliente mantiene el túnel abierto, de forma que si cambia su IP lo vuelve a abrir.

En primer lugar hay que instalar el paquete openvpn:

```
# apt-get install openvpn
```

En el ordenador que actúa como cliente también hay que instalar openvpn.

Crear la clave estática en el servidor:

```
# cd /etc/openvpn/
# openvpn --genkey --secret staticVilafranca.key
# ls -l
total 3
-rw------- 1 root root  636 feb 18 19:48 staticVilafranca.key
-rwxr-xr-x 1 root root 1352 sep 18  2008 update-resolv-conf
```

y copiarla mediante ssh (u otro medio seguro) al cliente.

En el servidor, crear el fichero `/etc/openvpn/tunelVilafranca.conf` (el nombre del fichero puede ser otro, en este caso suponemos que hacemos un túnel con Vilafranca) con el siguiente contenido:

```
dev tunUJI-Vila
ifconfig 10.100.1.1 10.100.1.2
secret staticVilafranca.key
port 1194
```

Donde `10.100.1.1` es la ip del servidor en el túnel y `10.100.1.2` la ip del cliente en el túnel.

En el cliente hay que crear un fichero `/etc/openvpn/tunel.conf` (de nuevo el nombre del fichero puede ser otro) con el siguiente contenido:

```
remote 150.128.97.56
port 1194
dev tunVila-UJI
keepalive 10 60
ifconfig 10.100.1.2 10.100.1.1
secret staticVilafranca.key
```

donde `150.128.97.56` es la ip pública del servidor.

Con la instrucción `port 1194` se puede indicar otro puerto (por defecto es el 1194). Esto hay que hacerlo tanto en el cliente como en el servidor y hay que poner el mismo en ambos. Los puertos hay que abrirlos en el cortafuegos.

`keepalive 10 60` es para que automáticamente vuelva a abrir el túnel si no está funcionando. Cada 10Sg hace un ping. Si no contesta durante 60Sg, volverá a abrir el túnel.

`ifconfig 10.100.1.2 10.100.1.1` indica que `10.100.1.2` es la ip del cliente

en el túnel y `10.100.1.1` la ip del servidor en el túnel (igual que se había indicado en el servidor).

Parece que al instalar openvpn ya se pone en el arranque de la máquina. No es necesario hacer nada adicional. Si queremos probar el túnel se puede ejecutar la orden (adaptando el nombre del fichero de configuración):

```
openvpn --config /etc/openvpn/tunelVilafranca.conf
```

En la documentación de OpenVPN dicen que la gran parte de los problemas de los usuarios nuevos de OpenVPN están relacionados con el firewall:

«Bear in mind that 90% of all connection problems encountered by new OpenVPN users are firewall-related.»

Si no se quiere que el túnel esté cifrado hay que añadir en el fichero de opciones o en la linea de órdenes al ejecutar el túnel:

```
cipher none
```

Enlaces

- http://howto.landure.fr/gnu-linux/debian-4-0-etch-en/install-and-setup-openvpn-on-debian-4-0-etch
- http://blog.bodhizazen.net/linux/how-to-vpn-using-ssh/
- http://bodhizazen.net/Tutorials/VPN-Over-SSH/
- http://wiki.debian.org/HowTo/openvpn

Ejercicio 5.8

Enrutamiento. Quagga.

Si el servidor necesita realizar enrutamiento dinámico hay que instalar el paquete *quagga* que incluye el protocolo OSPF entre otros.

```
# apt-get install quagga
```

En `/etc/quagga/` se encuentran los ficheros de configuración. En primer lugar hay que editar `/etc/quagga/daemons` e indicar los *duendes* que se han de ejecutar, `zebra` es el demonio principal y que coordina al resto.

```
zebra=yes
bgpd=no
ospfd=yes
ospf6d=no
ripd=no
ripngd=no
isisd=no
```

En `/usr/share/doc/quagga/examples/` hay ejemplos de configuración de todos los duendes.

En el fichero `/etc/quagga/zebra.conf` se definen las interfaces, rutas estáticas, ...

```
! -*- zebra -*-
!
```

```
! zebra sample configuration file
!
! $Id: zebra.conf.sample,v 1.1 2002/12/13 20:15:30 paul Exp $
!
hostname castello.guifi.net
password passzebra
enable password passzebra123
!
! Interface's description.
!
!interface lo
! description test of desc.
!
!interface sit0
! multicast

interface eth0
 description interface de la freenet


! Static default route sample.
!
!ip route 0.0.0.0/0 10.228.130.161
!

log file /var/log/quagga/zebra.log
```

El fichero `/etc/quagga/ospfd.conf` contiene la configuración del duende `ospfd`, las interfaces y redes con las que hay que intercambiar rutas:

```
! -*- ospf -*-
!
! OSPFd sample configuration file
!
!
hostname castello.guifi.net
password passospf
enable password passospf127
!
interface eth0

router ospf
! network 0.0.0.0/0 area 0
 network 10.228.130.160/29 area 0
 network 10.228.130.176/29 area 0
!
!log stdout
log file /var/log/quagga/ospf.log
```

Dispositivos wireless en modo bridge

Cuando se utilizan dispositivos wireless en modo bridge pueden haber problemas con los mensajes multicast, en

http://wiki.quagga.net/index.php/Main/FAQ#QQuestions11 se explica cómo solventarlo.

En los mikrotiks trabajando con dispositivos en modo bridge también hay que especificarlo.

http://wiki.mikrotik.com/wiki/Manual:Routing/OSPF#NBMA_Neighbor
http://wiki.mikrotik.com/wiki/OSPF-examples
http://wiki.mikrotik.com/wiki/Manual:Multicast_detailed_example#Multicast_and_Wireless

Firewall

Como se ha explicado anteriormente hay que abrir el protocolo 89 (el ospf):

```
$IPTABLES  -A INPUT -p 89 -j ACCEPT
```

Posibles problemas

Si una interfaz esta definida con una máscara en `/etc/network/interfaces` y en `/etc/quagga/ospfd.conf` ponemos otra máscara, el demonio ospf no reconoce esa interfaz y no la utiliza.

Enlaces

- http://www.quagga.net/
- http://wiki.quagga.net/index.php/Main/FAQ
- http://openmaniak.com/quagga_tutorial.php
- http://infolinux.wordpress.com/2007/05/25/instalando-quagga-en-ubuntu/
- http://wiki.quagga.net/index.php/Main/TipsNTricks

Ejercicio 5.9

Logs, mensajes del sistema.

El sistema de *logs* o *registros* de GNU/Linux es el mecanismo que se encarga de almacenar los mensajes generados por las aplicaciones. En cada mensaje consta qué programa lo generó, la prioridad y la fecha y hora en que se produjo.

Los ficheros de log en un sistema linux, se encuentran habitualmente en el directorio `/var/log` o en algún directorio dentro de éste.

El sistema de logs arranca con el script `/etc/init.d/sysklogd`, el demonio que realiza los registros es `syslogd` que se configura mediante el fichero `/etc/syslog.conf`, donde se indica qué se quiere registrar y a dónde se deben enviar los logs.

Los archivos más importantes son:

- `/var/log/messages`: donde se almacenan todos los mensajes con prioridad `info` (información), `notice` (notificación) o `warn` (aviso). Es uno

de los ficheros en los que primero se mira cuando hay algún problema.
- `/var/log/kern.log`: almacena los logs del kernel.
- `/var/log/dmesg`: almacena la información que genera el kernel durante el arranque del sistema. Se puede ver su contenido con la orden `dmesg`.

Para ver el contenido (total o parcial) de alguno de estos ficheros se pueden utilizar alguna de las siguientes ordenes:

```
cat /var/log/messages
```

```
less /var/log/messages
```

```
tail /var/log/messages
```

Los archivos de log suelen crecer mucho ya que en ellos se está guardando información continuamente. Por ello, existe una aplicación, `logrotate` (configurable a través del fichero `/etc/logrotate`) que, si los ficheros de log son muy grandes, los comprime y aplica una rotación a los archivos (añadiéndoles la extensión .1.gz, .2.gz, etc.), volviendo a crear uno vacío (cuanto mayor es el número más antiguo es el log).

Existen aplicaciones gráficas para supervisar los logs, por ejemplo KSystemLog, GNOME-System-Log, Xlogmaster y Xwatch.

Enlaces

- http://www.estrellateyarde.org/so/logs-en-linux

Ejercicio 5.10`/var/log/messagesdmesg`

Copias de seguridad y restauración del sistema

Es importante mantener copias del sistema para poder restaurarlo ante una posible perdida de datos.

En barrapunto hay una interesante discusión sobre opciones de copias de seguridad para pequeñas empresas: http://preguntas.barrapunto.com/preguntas/11/04/19/0911229.shtml

En cualquier caso es importante que se realicen copias de los siguientes directorios:

- `/home` contiene las carpetas personales de los usuarios.
- `/root` contiene el directorio personal del usuario root.
- `/etc` contiene archivos de configuración del sistema.
- `/var` contiene archivos variables, tales como logs, archivos spool, bases de datos, archivos de e-mail temporales, y algunos archivos temporales. Pueden excluirse los directorios `/var/cache`, `/var/spool` y `/var/tmp`.

Para que la copia no sea excesivamente grande se pueden excluir ficheros del tipo: `*.o`, `*~`, `*.mp3`, `*.ogg`, `*.avi`, `*.mpg`, `*.mkv`, `*.iso`.

Antes de realizar la copia es conveniente parar los servicios: apache2,

postgresql, mailman, ... para que no modifiquen los ficheros mientras se está realizando la copia.

También es aconsejable guardar la lista de paquetes instalados en el sistema:

```
dpkg --get-selections | diff - /root/dpkg-selections.log >
/dev/null ||
dpkg --get-selections > /root/dpkg-selections.log
```

todo esto puede automatizarse utilizando algún paquete especializado como `backup2l`.

Las copias se deben almacenar en un lugar distinto al del servidor. Se puede también enviar por guifi.net (o internet) a un servidor remoto utilizando, por ejemplo, `rsync` y `ssh`.

Enlaces

http://www.debian.org/doc/index.es.html

http://wiki.debian.org/es/FrontPage?action=show&redirect=es

http://www.debian.org/doc/manuals/reference/

Otros enlaces interesantes

- http://www.debian.org/doc/manuals/debian-faq/
- http://wiki.debian.org/es