# Como Montar un Nodo para BilboWireless

# Indice:

## 0.Que es un Nodo. Introducción a BilboWL

## 1.Capa Física(lo que se puede romper)

### 1.1. Dos maneras de montar un nodo para BilboWL

#### a)PC + WLAN NIC

TARJETAS:
PCI,PLX,PCMCIA,USB,CF...
Chipsets:PRISM, ATMEL, ORINOCO, TI ....

#### b)PC+Ethernet+WLAN AP

Pros y contras de cada método.

### 1.2. ANTENAS:

Direccionalidad:Omnis,Direccional,sectoriales
Ganancia.
Cables.
Calculo de enlaces.

### 1.3 HARDWARE

P60 o superior + Linux(o BSD http://www.eldemonio.org/docs/freebsd/wireless.pdf, o lo que sea)

## 2.Capa Software(nos tiramos al barro)

### Nivel 0: (palabra clave: iwconfig)

Drivers: Para Prism:Kernel(simple), wlan-ng(RFMON), HOSTAP(MASTER!)
Túneles: Modulo Tun/Tap, openvpn, ipsec....

## Nivel 1:(palabra clave: ifconfig)

Breve introducción a redes
Direccionamiento:WiFi+Túneles. Direccionamiento según Redlibre.
OSPF
DHCP
DNS(?)

YA FUNCIONA!

## Nivel 2:(palabra clave: iptables)

IPTABLES
Radius(?)

## Nivel 3:Aplicación

Bibliografía:
http://wiki.madridwireless.net/UnirNodoAMadridWireless

Perdidas en los cables RG
http://murcia.redlibre.net/docs/tablaRG.html

Muy completo, desde comparativa de tecnologias hasta tipos de redes y modulacions,...)
http://sindominio.net/suburbia/article.php3?id_article=22

Esquema de red:
http://www.raubacapeu.net/people/yves/2000/10/26-network

Linux WLAN howto:
http://linux.grmbl.be/wlan/

Linux Wireless Router HOWTO
http://www.rage.net/wireless/wireless-howto.html

Howto OpenVPN:
http://zgor.int80h.net/files/openvpnMiniHowto.txt

AP al aire libre COMO
http://www.redlibre.net/wiki/moin.cgi/PAenExterio

# 0.Que es un Nodo. Introducció na BilboWL

Bilbowireless es un (proyecto de) Red Libre Ciudadana. Red Libre Ciudadana implica que sea de libre acceso y libre tránsito. Por ello nos centramos en el uso de Software Libre y Hardware que siga los estándares. BilboWireless es también una red auto gestionada, donde son los participantes los que tienen el control sobre ella.

Un nodo de BilboWL es una vía de acceso a la red libre. Tiene varios cometidos, desde dar acceso físico (a través de la tarjeta WiFi o un AP) hasta ser capaz de encaminar el trafico. Actualmente dependemos de túneles para la interconexión de nodos. El ideal sería la interconexión inalámbrica.

# 1.Capa Física (lo que se puede romper)

En este apartado veremos todo lo que se refiere a la parte física de la red. Esto incluye tarjetas, antenas, PCs y demás material.

## 1.1. Dos maneras de montar un nodo para BilboWL

La primera duda a la hora de montar un nodo es que hardware sera necesario. Hay 2 maneras de montar un nodo. La primera sería empleando un PC, generalmente una máquina vieja, y una tarjeta inalámbrica. Esto implica que necesitaremos un cable entre la tarjeta y la antena. Esta opción es útil si pensamos mantener el conjunto dentro de un edificio y la antena va a estar separa por muy poca distancia (2 o 3 m, un nodo alado de una ventana p.j.). La segunda opción incluye el uso de un Punto de Acceso (AP) que, probablemente, se comunicará a través de Ethernet con un PC. Veamos estas opciones mas en detalle.

### a)PC + WLAN NIC

Si optamos por este camino la primera elección a realizar es la de la tarjeta inalámbrica que se va a usar. Actualmente en el mercado hay 3 "tecnologías". 2 son estándar y una no. Se las conoce como 802.11b (11Mbs, la mas extendida), 802.11b+(una mejora que algunos fabricantes han hecho, alcanzando los 22Mbs), 802.11g (de reciente aprobación).

El primer problema que se nos plantea es por que tecnología decantarnos. Una rápida solución seria 802.11b. Por que? Por que es la que mejor soporte tiene bajo GNU/Linux y otros sistemas libres, es la mas extendida y es un estándar probado.

802.11b+ es una especificación propietaria, que aunque sobre el papel ofrezca mejores resultados, implica problemas a la hora de drivers y compatibilidades entre distintos fabricantes. 802.11g acaba de ser aprobada y aparte de poco hardware para ella nuevamente nos encontramos con el problema de los drivers (se comenta que para finales de 2003 habrá driver para Linux).

Un problema serio es que 802.11b esta desapareciendo del mercado. La solución es buscar hardware descatalogado (mas barato). Se están haciendo esfuerzos para crear un driver libre para 802.11b+ (ya que el binario que hay actualmente es virtualmente inútil).

Otro motivo para decantarse por 802.11b es que es la única tecnología que nos va a permitir el poner nuestras tarjetas en modo master, necesario para que el PC con una tarjeta WiFi actúe como un AP.

Pero para poder hacer esto es necesario tener en cuenta otro factor, el chipset.

**Chipsets: PRISM, ATMEL, ORINOCO, TI ....**

Si no es poco con tener que elegir tecnología nos encontramos con que hay varios chipset (circuitería interna de cada tarjeta) que den soporte para ella. Algunos de los mas conocidos son:

**Atmel**. Muy usado en dispositivos USB. Soporte en modo cliente bajo Linux
**Hermes**. El usado en las orinoco, avaya...Soporte en modo cliente y primeras pruebas en modo Master
**Prism**. El mas idóneo para nuestros propósitos. Soporte total, modo cliente, master, monitor....
**Texas Instrument**(802.11b+). Se niegan a dar sus especificaciones técnicas. Se esta trabajando por ingeniería inversa.

El mas indicado para montar nuestro nodo es el Intersil Prism 2.5. El orinoco también se puede poner en modo master y podría ser usado, aunque no tenemos experiencias cercanas aun.

La ultima elección relativa a la tarjeta es su formato. Podemos elegir entre muchos: PCI,PLX,PCMCIA,USB,CF...

PCMCIA solo nos sería si nuestro nodo dispone de tal ranura(hay algún loco que usa portátiles viejos como nodos). Si nuestra maquina es un PC convencional la elección se restringe a PCI, PCMCIA+Adaptador (comúnmente llamado PLX) y USB.

El empleo de tarjetas USB puede presentar algunas ventajas con respecto a las PCI o PLX. La mas notables es que una tarjeta USB puede ser alejada del PC con un alargador USB. Este cable, a diferencia de un cable que una una antena con una tarjeta, no presenta perdida alguna.Como pega hay que decir que es dificil encontrar tarjetas USB con chipset Prism (usualmente emplean chipset ATMEL) y que un alargador USB puede ser algo caro (5m rondan los 30e). Es por ello que aunque una tarjeta USB no nos sea útil para funcionar en modo master si nos puede resultar muy útil como enlace entre nodos.

### b)PC+Ethernet+WLAN AP

Esta opción es la mas adecuada si queremos subir el AP al tejado. Para ello necesitaremos un PC que se encargue de las funciones mas complejas del nodo, rutas dinámicas y túneles, ya que la mayoría de los APs carecen de la posibilidad de hacerlo. Un ejemplo seria el siguiente:

Un ordenador viejillo, P100 64Mb RAM, al cual le ponemos 2 tarjetas de red,una para la red local con salida por ADSL, y otra que ira al AP. Del ordenador al AP tiremos Ethernet + PoE (alimentación incluida en el cable ethernet). El AP estará en una caja estanca(o "tuperware", reivindiquemos el "hecho en casa" una vez mas :P), a la sombra preferentemente, en cualquier lugar (terraza, tejado, fachada, hasta en una farola! :P). No suele ser necesario modificar la antena del AP (la antena del USR cubre 3km teniendo una lata en el otro extremo) a no ser que solo vallamos a cubrir una zona específica (antena de panel en una fachada) o lo vayamos a usar para un enlace punto a punto (en el futuro es lo que nos gustaría).

Este sistema es algo mas caro. Una tarjeta WiFi puede andar sobre los 60e mientras q un AP fácilmente lo triplica (la media son unos 120e). Aun así los beneficios de evitar el cable tarjeta-antena se notan, tanto en dinero (el cable coaxial de radio frecuencia bueno puede ser caro) como en potencia perdida.

Además el PoE nos permite poner el AP literalmente donde queramos. Por ejemplo para un USR por 12e del material mas los que cueste el cable ethernet (aprox. 60cnts el metro) y sin cambiar el transformador que trae podemos alejarlo hasta 15m de la cajita de PoE (esta puede estar a su vez separada del PC que haga de nodo).

Si el PC se encarga de las tareas complicadas el AP se puede poner en modo bridge y no requiere mucha complicación.

## 1.2. ANTENAS

Como siempre hay muchos tipos de antenas. La clasificación obvia es según el área que cubren. Así podríamos hacer 3 grupos:

**<u>Omnidireccionales</u>**: Cubren los 360 de un plano horizontal(el lóbulo de radiación es como un donuts). Si la ganancia es muy grande(mas de 5dB) no cubren bien las zonas de debajo y arriba(un donuts muy plano). La ideal para un nodo, ya que querremos cubrir toda la zona posible.

**Secctoriales:** Cubren un ángulo que va desde los 180 hasta los 60(o menos). Son utiles si tenemos una zona físicamente cegada(nodo en una pared) y solo vamos a dar cobertura a una zona concreta. Mayor ganancia que las omnis(por ejemplo nuestras ustrip 12dBs) por lo tanto mejor disposición a través obstáculos y grandes distancias.

**Direccionales:** Desde algunos grados hasta muy pocos. Cuanto mas cerrado mas ganacia. hasta 24dB en parabólicas. Para un nodo no muy útiles pero quizá para un cliente si. también validad para los enlaces inter-nodos. largas distancias (10Km).

Hemos nombrado la palabra ganancia y no hemos explicado que significa. La ganancia de una antena es la capacidad de amplificación que tiene la misma. Se expresa en dB, que es el resultado de 10log(ganancia). Al trabajar con unidades logarítmicas podemos sumar ganancias y restar perdidas. Por ejemplo si tenemos una tarjeta de 30mW típica tendríamos en principio 10Log(30mW)=15dBm (usamos dBm pq nos referimos a potencias). Si nuestra antena tiene 12dB (la ustrip o una lata por ejemplo) y sumamos tenemos 15+12=27dB. Si lo convertimos a potencia tendremos 10^(27/10)~500, osea, seria como si estuviéramos usando medio vatio(fuera de la norma de 100mW).

Una cosa a tener en cuenta es que una antena funciona en los 2 sentidos. Vale tanto para amplificar la emisión como la recepción. Usando una analogía sencilla nos permite "gritar mucho", como si tuviéramos un megáfono, y también escuchar muy bajito, como si tuviéramos una gran oreja.

Otro punto a tener en cuenta es el cable necesario para unir la antena y la tarjeta. Este cable introduce perdidas siempre. Por lo tanto este cable no podrá ser muy largo. Por ejemplo si usamos un cable sencillito (RG58 o equivalente) que tiene 5mm de diámetro no podremos usar mucho mas de 2 m ya que perderíamos demasiada potencia. Si necesitamos forzosamente mayor distancia hay que recurrir a cables de mayor calidad (RG213, LMR400...) que son bastante gruesos(10mm) y bastante caros (el LMR400 se puede poner por encima de los 5e/m).

El usar cables implica el usar conectores. El tipo de conector ideal es el N. Es el mas usado en antenas. En pigtails (cables que se conectan a las tarjetas) podemos encontrar muchos tipos de conectores, desde SMA(los que les ponemos a las USR) hasta conectores propietarios(llamados de polaridad inversa, y difíciles de conseguir).

Para acabar con la chapa de antenas ganancias y demás conceptos confusos vamos a ver como se calcula un enlace.

-------------------EJEMPLO DEL HIPOTETICO ENLACE SrDON - MoeBIUS---------------

### 1.3 HARDWARE

Hemos dicho que necesitamos un PC para encargarse del rutado dinámico, túneles, etc... Entonces que máquina necesitamos? Con lo raro que suenan esas cosas por lo menos un P4 con 1Gb de RAM, no? Pues no. Por ejemplo, para encargarse de rutado dinámico(OSPF), túneles con openvpn(3 en concreto), servidor de DHCP, un pequeño servidor de Web, un servidor DNS de cacheo y del rango de IPs de bilbowireless y alguna cosilla mas nos basta con un P100 y 64mb de RAM. Lo bueno de usar una maquina vieja es que nos saldrá económica (probablemente gratis) y que generara poco calor (y con un poco de maña poco ruido).

Además del PC necesitaremos una, o dos, tarjetas de red. La típica tarjeta de 6-7e nos bastara. también necesitaremos una tarjeta wireless (unos 60e) o un AP(unos 120e). Algo de cable de red(mas si usamos el AP) que ronda los 60cents metro. Si usamos un AP y vamos a hacer PoE habrá que sumar unos 12e mas(en plan chulo). Necesitaremos una antena, que si es casera andará por los 12e y si es comprada bastante mas, aunque quizá nos valga con la q incluya la tarjeta o el AP(la del AP del USR cubre 3km!). Para acabar algo de cable, un pigtail sale unos 30e, si lo construimos nosotros unos 3e por conector mas el cable que elijamos.

Y ya esta. Si sumamos nos salen unos 100e en caso de usar una tarjeta wifi y unos 190 si usamos un AP.

## 2.Capa Software(nos tiramos al barro)

En esta parte empezaremos a mancharnos las manos(bueno las llemas de los dedos al teclear). Daremos por hecho unos conocimientos básicos de Linux. Por lo menos la parte de la instalación nos la saltaremos (somos asi de chulos) ya que hay amplia documentación sobre ello (LIPP, Guía Debian). También seria recomendable tener unos conocimientos básicos de redes, aunque trataremos de explicar los mas sencillo posible.

### Nivel 0: (palabra clave: iwconfig)

Antes de empezar ha explicar como elegir el canal y modo de funcionamiento de la tarjeta hablaremos un poco de los distintos drivers para tarjetas wireless que hay para GNU/Linux. Cada chipset(de los soportados) tiene su propio driver. Los mas conocidos son:

Chipset Atmel:Atmelwlan.Soporta tarjetas USB, miniPCI, PCMCIA. Desarrollado por una persona con la documentación que le da atmel. Soporta el modo cliente y el modo adhoc.

Chipset hermes: Driver del kernel. Soporta modo cliente y modo adhoc
Driver Hermes AP. Reciente. Permite poner en modo master

Chipset Prism. Driver del kernel. Modo cliente y modo adhoc.
WLAN-NG. Muy elaborado. permite modo monitor(para escaneo de redes)
HOSTAP. Permite el modo maestro. Es el que usaremos en estas explicaciones

Otros: El kernel (y los módulos pcmcia_cs) incluyen soporte para otras tarjetas/chipsets

Explicaremos ahora un poco el caso ideal: HOSTAP. Realmente la compilación de este driver es bastante sencilla. Lo primero es que nuestro kernel disponga de soporte genérico wireless. para lograr esto deberemos acudir a la sección "Network Device Support" del kernel y entrando en "Wireless LAN (non-hamradio) " activamos "[*] Wireless LAN (non-hamradio) ".

Una vez tengamos preparado nuestro kernel con soporte para wireless nos pondremos a compilar hostap. Se puede conseguir de : URL. Una vez bajada la ultima versión y descomprimida en algún lugar entramos a su directorio y hacemos make. enseguida vemos cuales serán los pasos a seguir. tan sencillo como make plx ( si es que tenemos una tarjeta con adaptador plx) y luego un make install_plx.

Si queremos saber como funciona el driver es muy recomendable la lectura del fichero README.prism2. Realmente una documentación bastante completa de todo lo que podemos hacer con este driver.

Si todo ha ido bien, al hacer un modprobe hostap_plx debería cargarnos el modulo y tener lista nuestra tarjeta WiFi.

Una cosa que no hemos comentado aun , a pesar de que es la palabra clave de esta sección, son la wirelesstools. Estas herramienta sera necesaria para poder controlar nuestra tarjeta. Después de instalarla ( mejor hacerlo antes de intentar probar el driver) y teniendo el driver a punto si ejecutamos iwconfig, nos listara los interfaces de red y de los que sean wireless sus propiedades. Por ejemplo:

*wlan0 IEEE 802.11-b ESSID:""*
*Mode:Managed Frequency:2.472GHz Access Point: 00:90:D1:01:65:34*
*Bit Rate:11Mb/s Tx-Power:2346 dBm*
*Retry min limit:8 RTS thr:off Fragment thr:off*
*Encryption key:off*
*Link Quality:48/1 Signal level:-68 dBm Noise level:-99 dBm*
*Rx invalid nwid:0 Rx invalid crypt:0 Rx invalid frag:0*
*Tx excessive retries:0 Invalid misc:0 Missed beacon:0*

En este caso se trata de una tarjeta pcmcia en modo cliente. Veamos que quieren decir algunos de los campos mas significativos.
Mode: En este caso aparece managed(cliente de un ap) pero si estamos usando el driver hostap debería figurar Master.

Frequency: Frecuencia o canal que estamos usando. En este caso es el canal 13 que equivale a una frecuencia de 2,472GHz. Si se quiere saber la correspondencia canal-frecuencia mirar la bibliografía.

ESSID: Nombre de la red wireless.

El resto de entradas no las comentaremos ya que solo son útiles si somos clientes( y se supone que estamos montando un nodo :P)

Suponiendo que disponemos del driver hostap vemos como cambiar alguno de los parámetros:

*iwconfig wlan0 mode master*
*iwconfig wlan0 essid BilboWL nickname EuskalXI channel 1*

Con esos comandos lo que lograremos es ponernos en modo master ( si es que no lo estavamos ya) establecer el ESSID en BilboWL y elegir el canal 1.

La elección de nickname es meramente informativa( por si hay varios APs con el mismo ESSID)
La elección del canal depende de las circunstancias. Si estamos solo y con un solo dispositivo, o red, wireless podremos elegir el que queramos de entre los 13 disponibles. Pero si tenemos que compartir el espectro radiofónico con algún otro AP o red tendremos q tener en cuenta que deberán estar separadas por lo menos 4 canales. Enseguida nos damos cuenta que solo podremos tener 3 redes wireless en la misma área.

***Túneles.***

En bilbowireless hemos optado por usar los túneles a través de Internet para ir haciendo backbone de prueba mientras vamos logrando enlaces wifi entre nodos. Para estos túneles usamos OpenVPN. Un miembro de bilbowireless ha hecho un magnifico HOWTO de como ponerlos ha andar.

http://zgor.int80h.net/files/openvpnMiniHowto.txt

Solo comentar que sera necesario activar en el kernel el soprte TUN (otra vez en "Network device support" <M> Universal TUN/TAP device driver support ).

## Nivel 1:(palabra clave: ifconfig)

Breve introducción a redes

En este nivel empezaremos ha hablar de redes por fin. Antes de entrar en detalle de como configurar un nodo para formar parte de BilboWL vamos a explicar que esquema sigue la red.

La red se puede dividir en 2 partes. Por un lado esta la red que ven los clientes( y también los nodos) es la red a la que las personas daremos uso. Pero para que los distintos nodos se comuniquen necesitamos otra red. Esta red recibe el nombre de backbone(espina dorsal) y es la que permite que podamos ver a clientes y nodos a los que no tenemos acceso físico directa( que no están conectados al mismo nodo vamos).

Empezaremos explicando algo sobre la red backbone. Teóricamente esta red puede ser de cualquier tipo ( ethernet, wifi,...) pero idealmente seria inalámbrica, ya que nos permitirá unir puntos distantes de una manera económica. Pero (siempre hay un pero) tenemos un gran problema en nuestra contra: la orografía. La cantidad de montes, monticulos, edificios altos y demas estorbos no le hacen mucha gracia a la las microondas. Es por ello que es realmente difícil en este momento inicial (pocos nodos) lograr la interconexión entre ellos por wireless. Como solución hemos optado por lo túneles a través de Internet entre los nodos (que disponga de conexión de "banda ancha", por llamarla de alguna manera).

Los túneles a través de Internet no son una solución óptima, pero nos permite ir experimentado con la tecnología (es muy difícil experimentar protocolos de rutado dinámico solito en tu casa :P ).

Bueno, independientemente de la tecnología que empleemos esta red backbone necesita su rango de IPs. En BilboWL hemos optado por seguir el direccionamiento (reparto de IPs) propuesto por RedLibre. Esto supone que disponemos de 2 rangos de IP (privadas, no accesibles desde Internet) para nuestra comunidad. Un rango de IPs es como hemos dicho para el backbone. Como usamos túneles(conexiones punto a punto) usamos peque;os rangos de IPs para cada enlace entre nodos. Este rango concretamente es de 4 IPs, 1 para cada nodo, la de broadcast y la de red. Es por ello que si acudimos a la web donde se coordina el direccionamiento en RedLibre vemos que las conexiones entre nodos usan rangos del estilo ------EJEMPLO-------

Aparte de la red backbone esta la red que realmente usaremos. Esta red tiene un rango de IPs totalmente distinto del backbone. tenemos asignado el rango 10.34.LALALALALALA. cada nodo coge de este rango otro subrango (32 IPs por nodo) es por ello que si miramos nuevamente en la web de RedLibre vemos que cada nodo tiene asignado un rango del estilo 10.34.132.128/27.

***Nota. Explicación de los rangos y las mascaras.***

Es muy usual que para especificar un rango de IPs usemos IP/RANGO. Que queremos decir con esto? vemos con un ejemplo. Si tenemos 10.134.132.128/24 la IP es la IP de la red y el rango es el numero de bits que permanece inmutable dentro de la misma. Como una IP tiene 32 bits ( 4 bytes) si 27 son inalterables nos quedan 5 para combinar. Si elevamos 2 a 5 nos salen 32 IPs pornodo. En nuestro ejemplo irían desde 10.34.132.128 (IP de red) hasta 10.34.132.159 (IP de broadcast). Para este tipo de calculos es MUY recomendable el pequeño programa ipcalc que nos sacra de mas de un apuro.

***Nota2. Porque RedLibre?***

MAS CHAPA AQUI.

Todo este lío de backbone,direccionamiento y subrangos IP es necesario conocerlo para tener una idea del porque de los siguientes pasos. Vamos con ellos pues. Supongamos que vamos a montar un nodo y que ya hemos cogido un rango de IPs de RedLibre (es un wiki, cualquiera puede apuntarse). Este rango de IPs(que supondremos que es el ya conocido 10.34.132.128/24) es el que usaremos para nuestra interfaz wireless ( o la etherenet que conecte con el AP).

Vemos como se haría esto. En plan bruto podríamos hacer:

*ifconfig wlan0 10.34.132.129 netmask 255.255.255.224 broadcast 10.34.132.159*

Esto funcionara, pero hay un problema , cuando reiniciemos (si en linux también se reinicia, aunque poco) perderemos la configuración. Para evitarlo tendremos que acudir a la configuración de red. En Debian por ejemplo bastaría con editar el fichero /et/network/interfaces y añadir algo del estilo:

```
auto wlan0
iface wlan0 inet static
    address 10.34.132.129
    network 10.34.132.128
    netmask 255.255.255.224
    broadcast 10.34.132.159
    gateway 10.34.132.129
```

Una vez editado este fichero bastara con hacer ifup wlan0 para que tengamos nuestra tarjeta configurada.

## ***<u>OSPF</u>***

Ha estas alturas deberíamos tener activos varios dispositivo de red. Por un lado tendremos el dispositivo inalámbrico (o la ethernet que nos lleve hasta el AP) con un rango 10.34.1..... y por otro lado los túneles 172...... Ahora como hacemos que los paquetes pasen de unos a otros? El primer paso es permitir que nuestra maquina actúe como un enrutador. Para ellos hay que escribir un "1" en /proc/sys/net/ipv4/ip_forward, por ejemplo de la siguiente manera:

echo "1" > /proc/sys/net/ipv4/ip_forward

Una vez mas este cambio es temporal hasta el siguiente reinicio. Para hacerlo permanente solo hay que editar ( en Debian) /etc/network/options y dejar una lineal tal que:

*ip_forward=yes*

Ahora somos capaces de dejar pasar paquetes que entren por una interfase y tenga como salida otra. Pero nos falta una cosa importante, Como sabemos los caminos hasta otros nodos? Para ello la manera mas sencilla es usar OSPF. OSPF es un protocolo de rutado estándar. Cada nodo con OSPF activo se anuncia a los nodos con los que tiene conexión directa ( en nuestro caso túneles) y les comunica que rutas conoce ( empezando por las suyas propias y siguiendo con las que ha aprendido de otros nodos).

Por ejemplo. Si yo tengo el rango 10.34.132.128/27 y tengo un túneles con NodoA, lo que pasaría sería algo parecido a esto (esto es una explicación con el único fin de hacer entender porque usamos OSPF, no pretender ser una explicación técnica (ni real) de como funciona). Lo primero seria mandar un paquete Hello a los otros nodos. Por ejemplo al NodoA le mandamos un "Hola soy paquito y se como llegar hasta 10.34.132.128.27". Si este NodoA esta conectado con el NodoB cuando le diga las rutas que conoce, aparte de la suya, le dirá algo como: "Oye, si quieres llegar a 10.34.132.128/27 tendrás que hablar con paquito al que yo estoy conectado a un salto". Que es esto de los saltos? Pues una manera de calcular el coste de una camino. Para liarlo un poco mas y ver como se usa el coste imaginemos un tercer nodo el NodoC. Supongamos que este nodo esta conectado con paquito y con NodoB. Este nodo tendrá pues 2 caminos para llegar al rango 10.34.132.128/27. Pero mientras la conexión directa con paquito es de un solo salto, a través del NodoB seria 2. Por eso decimos que el coste del primer camino es menor y sera el que se use.

Como vemos es muy útil el empleo de OSPF ya que no hay que mantener una tablas de rutado que pueden crecer bastante. recordemos además que para la conexión de nodos usamos otro rango y con OSPF no nos tendremos que preocupar de ellos.

Para usar OSPF recurrimos al paquete zebra. Este paquete incluye varios protocolos de rutado (OSPF, BGP. RIP....) pero nosotros de momento solo usaremos OSPF (BGP seria útil para la interconexión de distintas comunidades wireless).

-----Explicación de la configuración de OSPF/Zebra------

***<u>DHCP</u>***

Para que los clientes de nuestra red libre no tengan que preocuparse de que rango tiene el nodo al que se conectan ni ningún otro parámetro del estilo recurrimos a proporcionar DHCP en el nodo. DHCP de lo que se encarga de de responder a una petición de un cliente asignándole una IP y otros parámetros. Los datos a proporcionar serán:

IP del cliente. En la configuración del servidor de DHCP hay que indicar cual es la primera IP disponible y la ultima.

Gateway por defecto. Este dato le indica al cliente a quien tiene que acudir si tienes paquetes para fuera de su rango(el mismo que el del nodo). Este dato sera seguramente la IP que tenga el propio nodo.

DNS. Si queremos que el cliente sea capaz de resolver dominios habrá que usar este parámetro. Si tenemos salida a Internet podemos poner 2 cosas: Una IP de un DNS publico o la IP del propio nodo si es que tenemos un servidor DNS en el nodo ( no es necesario). Si no vamos a facilitar salida a Internet este parámetro se puede a bien ignorar o bien apuntar al que sea el DNS de nuestra comunidad.

Por ejemplo si usamos el servidor de DHCP udhcpd su fichero de configuración sera parecido a:

------------------------------------/etc/udhcpd.conf----------------------------------------------------

```
start        10.34.132.131
end          10.34.132.142

interface      eth1

lease_file      /var/lib/misc/udhcpd.leases

opt     dns     10.34.132.129
option  subnet  255.255.255.240
opt     router  10.34.132.129
opt     wins    10.34.132.129
#option dns     129.219.13.81   # appened to above DNS servers for a total of 3
option  domain  bilbow
option  lease   864000         # 10 days of seconds
```

---------------------------------------------------------------------------------

DNS(?)

**YA FUNCIONA!**

En este momento nuestro nodo es funcional. Un cliente se podrá conectar a el y si tenemos túneles o enlaces con otros nodos navegar por toda la red de BilboWL. Pero como siempre las cosas se pueden mejorar un poquito.

## Nivel 2:(palabra clave: iptables)

Si queremos que nuestro nodo posea una cierta seguridad, que seamos capaces de diferenciar el trafico de la red libre del nuestro trafico personal, necesitaremos recurrir a un firewall. En Linux a partir de la versión 2.4 del núcleo se emplea el firewall iptables. Este firewall implica por un lado unos módulos o drivers a nivel del núcleo y una aplicación (o aplicaciones) con las que crearemos nuestras reglas.

La configuración de un firewall no es para nada trivial y menos aun cuando vamos a tener trafico de muy distintos tipos. Además deberemos decidir si vamos a ceder (o compartir los gastos con alguien) nuestra salida a internet.