MONTAR UN PUNTO DE ACCESO INALÁMBRICO 802.11 EN LINUX

Juan Hernández-Serrano, Josep Pegueroles

jserrano@entel.upc.es, josep.pegueroles@entel.upc.es

***Resumen* - La gente se mueve, las redes no. Estas dos sentencias definen más que nada la explosión de las redes inalámbricas de área local o WLAN (Wireless Local Area Network). En unos pocos años el precio de los dispositivos WLAN ha bajado lo suficiente como para que se plantee su adquisición para usos domésticos. Aun así el precio del punto de acceso o AP (Access Point) sigue siendo elevado, y en la mayoría de casos su capacidad sobrepasa los pequeños requerimientos de una red inalámbrica doméstica. En este artículo explicamos como montar un AP inalámbrico con un ordenador, una tarjeta WLAN y software de código libre.**

1 INTRODUCCIÓN

La gente se mueve, las redes no [MG02]. Con estas dos sentencias queda más que definida la explosión de las redes inalámbricas de área local o WLAN (*Wireless Local Area Network*).

Con el paso de los años el mundo es cada vez más móvil. Especialmente en el mundo laboral, la facilidad para dar cobertura a nuevas sedes y espacios, la flexibilidad para la conectividad de nuevos trabajadores, y especialmente el ahorro en infraestructura de cableado; han impulsado la implantación de WLANs.

Actualmente estamos viviendo la explosión del mundo inalámbrico. La telefonía móvil ha basado su éxito en permitir que la gente se comunique entre sí independiente de su localización, y parece que el acceso a redes de datos (p.e. Internet) siga el mismo camino. La tecnología de redes inalámbricas que más ha fructificado es 802.11, que pone la base para crear WLANs.

En unos pocos años el precio de los dispositivos WLAN ha bajado lo suficiente como para que se plantee su adquisición para usos domésticos y no sólo con carácter empresarial. Aun así el precio del punto de acceso o AP (*Access Point*) sigue siendo elevado, y en la mayoría de casos su capacidad sobrepasa los pequeños requerimientos de una red inalámbrica doméstica con más de 5 usuarios simultáneos.

En este artículo explicamos como montar un AP inalámbrico con un ordenador, una tarjeta wireless y software de código libre. En la sección 2 introducimos la tecnología de red WLAN. A continuación se introduce el concepto de puente de red o *bridge*. La sección 4 indica los requerimientos de hardware y software para implementar un AP casero funcionando como *bridge* entre una red 802.11 y una red ethernet (802.3). A continuación se comentan los pasos de configuración para la puesta a punto del AP y puente red. Y finalmente concluimos con un breve estudio de la situación actual de las WLAN y de su futuro.

2 REDES INALÁMBRICAS DE ÁREA LOCAL

La WLAN es un tipo de LAN en la que la comunicación entre nodos se transporta a través de un medio inalámbrico, normalmente radio-frecuencia, en vez de a través de cables. Este tipo de comunicación permite combinar la conectividad de los nodos con la movilidad de los mismos.

Las redes WLAN deben proporcionar al menos el mismo nivel de funcionalidad que las redes LAN, con la ventaja adicional de tener menores restricciones físicas y sobretodo de la movilidad tanto de los usuarios como de la propia red.

Si bien se han desarrollado diversos estándares para WLANs, los dos principales son la americana IEEE 802.11 y la europea HyperLAN, aunque actualmente se puede considerar que la gran mayoría de dispositivos WLAN están basados en la familia de especificaciones IEEE 802.11. Veamos pues como funciona.

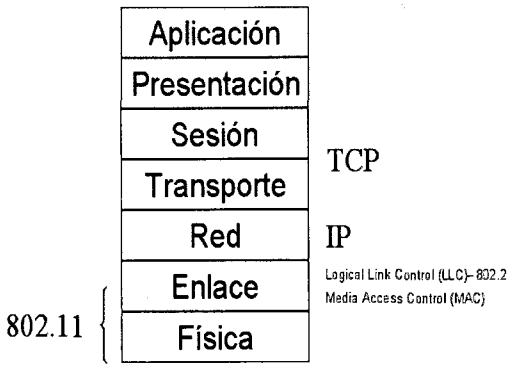

***Figura 1.** 802.11 en la pila OSI*

2.1 Estándar IEEE 802.11 para WLANs

802.11 es una familia de especificaciones desarrollada por el IEEE para tecnología WLAN. 802.11 especifica una interfaz «a través del aire» entre un cliente inalámbrico y una estación base, o entre varios clientes inalámbricos. Su objetivo principal es el de simular el comportamiento de redes ethernet (802.3) usando radio-frecuencia en vez de cables.

Tal y como se muestra en la Figura 1, 802.11 define dos capas lógicas de la pila OSI (Open System Interconnection):

- Control de Acceso al Medio o MAC (Medium Access Control)

- Física o PHY (PHYsical)

802.11 se define originalmente con una tasa de transferencia de hasta 2 Mb en la banda de 2,4 GHz y utilizando modulaciones de espectro expandido (FHSS - *Frequency-Hopping Spread Spectrum* y DSSS *Direct-Sequence Spread Spectrum*), aunque actualmente se define sobre las siguientes 3 capas físicas (véase Tabla 1):

- **802.11a**. Opera en la banda de 5 GHz y provee una tasa de transmisión de hasta 54 Mbps a una distancia máxima de unos 45 metros (150 pies). Utiliza técnicas de codificación OFDM (*Orthogonal Frequency Division Multiplexing*). Tiene peor comportamiento frente a interferencias de radio-frecuencia que 802.11b, pero sin embargo se comporta mejor para el transporte de voz y datos multimedia así como en escenarios con una alta densidad de usuarios.

- **802.11b** (también denominada 802.11 High Rate). Opera en la banda de 2,4 GHz y provee una tasa de transmisión de hasta 11 Mbps a una distancia máxima de unos 75 metros (250 pies). Utiliza sólo DSSS debido a que DSSS es capaz de manejar mejor que FHSS (también recomendado en la norma original 802.11) las señales de baja intensidad. Con DSSS pueden extraerse los datos de un fondo de interferencias sin necesidad de retransmitirlos. 802.11b es el primer estándar que provee a las redes WLAN de una funcionalidad comparable a la de las redes LAN (ethernet familia 802.3). Su ventaja principal respecto a 802.11a es que requiere un menor número de puntos de acceso para cubrir grandes superficies de terreno. Además es el primer estándar en cubrir el modo de funcionamiento Ad-Hoc.

- **802.11g**. Opera en la banda de 2,4 GHz y provee tasas de transmisión de 6 a 54 Mbps. Al igual que 802.11b, utiliza 3 canales no-superpuestos, y como en 802.11a, usa modulación OFDM, aunque para garantizar la compatibilidad «hacia atrás» con 802.11b, 802.11g usa también modulación CCK (*Complementary Code Keying*) y opcionalmente PBCC (*Packet Binary Convolutional Coding*). Esta compatibilidad «hacia atrás» viene a decir que cuando un dispositivo móvil 802.11b se adhiere a un punto de acceso 802.11g, todas las conexiones de este punto de acceso se reducen a velocidades 802.11b. 802.11b se adhiere a un punto de acceso 802.11g, todas las conexiones de este punto de acceso se reducen a velocidades 802.11b.

Además se definen:

- **802.11e**. Provee Calidad de Servicio (QoS - *Quality of Service*) para aplicaciones WLAN, como por ejemplo Voz sobre IP inalámbrica (VoWIP - *Voice over Wireless IP*).

- **802.11h**. Se trata de un suplemento a la capa MAC para que cumpla las regulaciones europeas de la banda de 5GHz, que obligan a utilizar tanto control de potencia de transmisión (TPC - *Transmission Power Control*) como selección dinámica de frecuencias (DFS - *Dynamic Frequency Selection*). TPC limita la potencia de transmisión a la mínima necesitada para llegar hasta el usuario más lejano. DFS selecciona el canal de radio al punto de acceso de manera que minimice las interferencias con otros sistemas, en concreto con el radar.

- **802.11i**. Borrador de estándar cuyo objetivo es incrementar la seguridad 802.11. Describe la transmisión de datos cifrados tanto en 802.11a como en 802.11b. Actualmente se usa WPA (*Wi-Fi Protected Access*) como solución interina hasta que se apruebe 802.11i como estándar, por lo que por el momento las mejoras en seguridad que provee son similares a las de WPA.

Variedades	Velocidad	Frecuencia	Distancia
802.11a	< 54 Mb	5 GHz	< 45 m
802.11b	5,5 y 11 Mb	2,4 GHz	< 75 m
802.11g	< 54 Mb	2,4 (y 5) GHz	< 75 m

Tabla1. *Variedades según capa física de 802.11*

2.2 Topologías 802.11

El bloque básico de construcción de una red 802.11 es el BSS (*Basic Service Set*), que es simplemente una agrupación de estaciones WLAN que se comunican entre sí. La comunicación entre estaciones se establece dentro del área de servicio básico o BSA (*Basic Service Area*).

Los BSSs son útiles para dar cobertura a pequeñas oficinas y casas, pero no sirven para dar cobertura a grandes áreas. Para poder dar una cobertura mayor, 802.11 permite unir o enlazar varios BSSs en un ESS (*Extended Service Set*). 802.11 no especifica ningún tipo de infraestructura, sino sólo qué servicios o *service set* se ha de proveer con ella.

Independientemente del tipo de *service set* utilizado, se definen también dos tipos de topología de red para WLAN:

- **Modo independiente o ad-hoc**. Los terminales se conectan unos a otros de igual a igual para establecer una red dinámica en la que todos los nodos o terminales tienen las mismas funciones de enrutamiento

- **Modo infraestructura**. Existe un punto de infraestructura fijo denominado punto de acceso (AP) al que se conectan los terminales. La conexión entre terminales se produce, por lo tanto, a través de dicho AP, que además suele ejercer funciones de puente entre dos redes, p.e. extender la red ethernet de la oficina a los dispositivos WLAN.

El escenario más común es el de una red WLAN 802.11 trabajando en modo infraestructura para extender la red local o LAN (*Local Area Network*), basada generalmente en tecnología ethernet (802.3), a los nuevos dispositivos inalámbricos. En este tipo de escenarios el AP suele funcionar como puente o *bridge* entre las dos redes físicas, con lo que a efectos del usuario inalámbrico su dispositivo se encuentra directamente conectado a la red ethernet.

En este artículo explicamos como montar un AP 802.11 funcionando como *bridge* entre las red 802.11 y una red ethernet.

3 EL PUENTE DE RED O *BRIDGE*

Un puente de red es básicamente un nexo de unión entre dos o más segmentos de red. Como la unión se produce al nivel de capa 2 (enlace), todos los protocolos pueden correr de forma transparente sobre él. Para entender como funciona un puente de red, es por lo tanto importante entender qué es eso del segmento de red.

Un segmento de red es una sección de medios de red que conecta dispositivos. Por ejemplo, supongamos que tenemos 3 ordenadores A, B y C. El ordenador A tiene 2 tarjetas de red, y los ordenadores B y C sólo una. Un cable ethernet que una A y B creará un segmento de red S_{AB}, y otro que una A y C creará otro segmento S_{AC}. Véase Figura 2. Si quisiéramos simular una conexión directa entre B y C, es decir; simular que están conectados entre ellos por un sólo cable ethernet (un sólo segmento de red) deberíamos configurar el ordenador A como puente de red que una los dos segmentos de red S_{AB} y S_{AC} para establecer un sólo segmento entre B y C S_{BC}. Véase Figura 3.

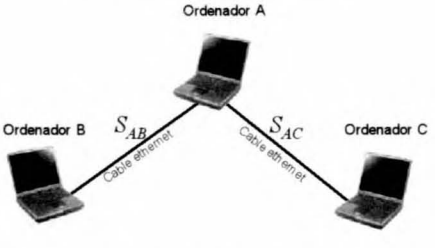

Figura 2. *Segmentos de red*

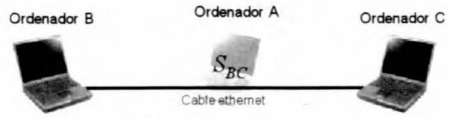

Figura 3. *Ordenador A actuando como puente de red*

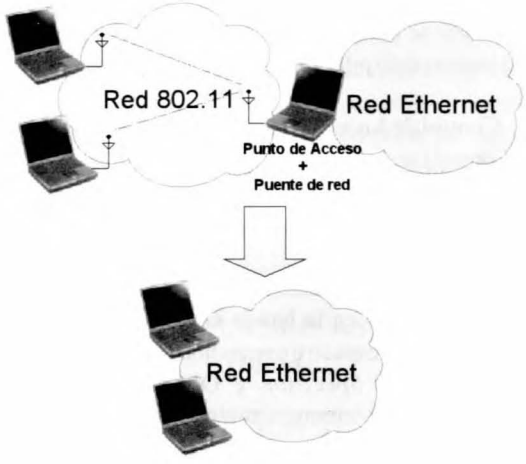

Figura 4. *Punto de Acceso 802.11 y puente de red*

Nótese que al configurarse el ordenador A como puente de red, A es ahora transparente entre B y C y por lo tanto invisible. Además, nótese que el puente de red lo que simula es una conexión a nivel de enlace (MAC), en este caso una conexión ethernet, muy diferente a la conexión IP (independiente de la red física) que lograríamos con un enrutador IP o *router*.

En el escenario de este artículo vamos a configurar un AP con *bridge* entre una red WLAN 802.11 y una red ethernet. En este caso, podríamos entender, simplificando, que el AP extiende la red ethernet a los dispositivos inalámbricos de forma que estos dispositivos parezcan conectados directamente a la red física ethernet. De esta forma nuestro escenario queda definido como en la Figura 4.

4 REQUERIMIENTOS DE HARDWARE Y SOFTWARE

El objetivo es montar un AP 802.11 que funcione como puente de red con una red ethernet 802.3. La elección de hardware y software está estrechamente ligada. La elección de un determinado software determinará el hardware a comprar y viceversa.

4.1 Hardware necesario

Sólo necesitamos un ordenador, una tarjeta de red 802.11 y una tarjeta de red ethernet.

4.1.1 Un ordenador

El AP se ha probado con éxito en un ordenador PII 366Mhz con 128Mb de RAM y debería funcionar incluso con máquinas menos potentes. Es importante que exista una distribución de Linux reciente que sea compatible con nuestro ordenador, por lo que recomendamos un ordenador con procesador Pentium X o AMD. Además es importante que el ordenador tenga soporte de PCI (ordenador) o de PCMCIA(portátil), aunque raro será el caso (sino prácticamente imposible) en el que no dispongamos de ellos.

4.1.2 Tarjeta de red 802.11

La elección de la tarjeta es crítica y determinará el éxito o el fracaso de nuestro propósito. Es fundamental que la tarjeta adquirida acepte el modo de funcionamiento de Host AP (también llamado modo Master), y sobretodo que tenga soporte de drivers para Linux. Además, hemos de tener en cuenta que tipo de variedad 802.11 deseamos implementar (802.11a, 802.11b u 802.11g). Descartada la tecnología 802.11a por obsoleta y por su predisposición única al ámbito empresarial, hemos de elegir entre la tecnología 802.11b y la 802.11g.

Lo más sencillo es optar por la tecnología 802.11b, que al ser más antigua, tiene un mayor soporte; y buscar tarjetas que soporten el software Host AP que es, desde nuestro punto de vista, la mejor implementación software de todas las funciones de un AP, incluidas muchas de las nuevas funciones criptográficas para privacidad y autenticación (Wi-Fi Protected Access - WPA). El software Host AP puede usarse con tarjetas basadas en los chipsets Intersil Prism 2, 2.5 y 3 que las proveen varios fabricantes (D-Link, LinkSys, Netgear, SMC ...). Aún así hay muchas tarjetas que inicialmente pone que están basadas en un determinado chipset, luego resulta que no es cierto y dejan de ser útiles. Recomendamos encarecidamente a todo el que quiera adquirir un tarjeta con este chipset que lea primero esta la URL en [4] a fin de asegurarse la adquisición de una tarjeta plenamente compatible con Linux y con soporte de modo Host AP.

La otra opción es optar por la tecnología 802.11g, más moderna y por lo tanto con un soporte menor para Linux. 802.11g ofrece sin embargo algunas ventajas. Además, claro está, del incremento de velocidad (54Mbs frente a 11Mbps), 802.11g es la tecnología que se está actualmente imponiendo, ya que no es más que la evolución natural de la tecnología 802.11b. Pronto toda la funcionalidad de 802.11b sobre Linux será extendida a 802.11g, sobre la que ya se soportan las funciones más básicas. Las tarjetas 802.11g adquiridas deben estar basadas en lo chipsets Intersil Prism GT o Duette y ser soportadas por el driver prism54 (véase una lista de tarjetas soportadas en [5]).
En nuestro caso hemos optado por la tecnología 802.11g y hemos adquirido la tarjeta PCMCIA SMC2835W con un resultado de éxito.

4.1.3 Tarjeta de red ethernet

Cualquier tarjeta debería funcionar. De todas formas sería bueno asegurarse de que la tarjeta tiene desarrollados drivers para Linux, aunque como ya hemos dicho es muy raro encontrar una tarjeta ethernet no soportada.

4.2 Software necesario

Tan importante como el hardware adquirido es el software que vamos a utilizar para montar tanto el AP como el puente de red.

4.2.1 Punto de Acceso (AP)

La mayoría de funciones del AP se realizan directamente a nivel de *driver* o de *firmware*. El *firmware* podría considerarse como un «mini» sistema operativo que utiliza cada dispositivo para implementar diferentes funciones; y el *driver*, como un interfaz de comunicación entre nuestro sistema operativo y el *firmware* del dispositivo.

Si hemos optado por la tecnología 802.11b, habremos adquirido una tarjeta con chipset Intersil Prism 2, 2.5 ó 3 y el driver que hemos de usar es Host AP ([6]). Este driver se distribuye por defecto con cualquier distribución reciente de Linux (SuSE, Fedora, Debian...) y por lo tanto la tarjeta debería funcionar con tan sólo insertarla. De todas formas para poder acceder a las nuevas funciones de seguridad implementadas seguramente tengamos que actualizarnos a la última versión del driver.

En caso de haber optado por la tecnología 802.11g, habremos adquirido una tarjeta con chipset Intersil Prism GT o Duette y el driver a usar es prism54 ([7]). Este driver todavía no está integrado en las distribuciones de Linux con lo que tendremos que compilarlo nosotros mismos. Para poder compilar el driver necesitamos una versión del *kernel* superior a la 2.4.23, el código fuente de dicho *kernel*, la herramientas de compilación (gcc, make, ...), el código fuente del driver ([8]) y el último *firmware* ([9]). Los pasos de compilación los tenemos en el mismo sitio web.

Una vez que tenemos el driver funcionando hemos de obtener los paquetes *wireless-tools* y *wireless-extensions*. El paquete *wireless-extensions* viene incluido como parte del *kernel*, con lo que a la práctica sólo hemos de obtener las *wireless-tools*. Normalmente tendremos este paquete precompilado dentro de nuestra distribución (si es actual), pero si no podemos descargarlo de [10].

El paquete *wireless-extensions* es una API genérica que permite al driver mostrar al usuario información sobre configuración y estadísticas comunes a las WLANs. Su belleza reside en que este paquete puede soportar todas las variaciones de WLANs independientemente de su tipo, y además los parámetros pueden cambiarse "al vuelo" sin necesidad de reiniciar el driver.

El paquete *wireless-tools* provee un paquete de herramientas para manipular las *wireless-extensions*. Utilizan un interfaz modo texto bastante tosco, pero que nos permite configurar y mostrar de forma más inteligible todas las *wireless-extensions*. De hecho, hay otras muchas herramientas para manipular las *wireless-extensions*, pero *wireless-tools* es siempre la implementación de referencia. Las 4 herramientas que provee este paquete son:

- `iwconfig` para manipular los parámetros básicos de la conexión inalámbrica

- `iwlist` nos permite iniciar el escaneo del medio y listar frecuencias, bit-rates, claves de encriptación...

- `iwspy` permite obtener la calidad del enlace para cada nodo

- `iwpriv` permite manipular parámetros específicos de cada driver / firmware (privados)

4.2.2 Puente de red

Para montar el puente de red sólo necesitamos el paquete *bridge-utils*, que viene incluido en la mayoría de distribuciones de Linux, y tener las dos tarjetas de red (la ethernet y la 802.11) funcionando.

Normalmente el código de las *bridge-utils* se compila como un módulo dentro del sistema operativo. Si el módulo está correctamente configurado e instalado, se cargará automáticamente la primera vez que llamemos al comando `brctl`. Si todo ha ido bien entonces al introducir el comando `brctl` debe aparecernos una pequeña sinopsis del comando.

```
# brctl
commands:
addbr           <bridge>                add bridge
addif           <bridge> <device>       add interface to bridge
delbr           <bridge>                delete bridge
delif           <bridge> <device>       delete interface from bridge
show                                    show a list of bridges
showmacs        <bridge>                show a list of mac addrs
showstp         <bridge>                show bridge stp info
setageing       <bridge> <time>         set ageing time
setbridgeprio   <bridge> <prio>         set bridge priority
setfd           <bridge> <time>         set bridge forward delay
setgcint        <bridge> <time>         set garbage collection interval
sethello        <bridge> <time>         set hello time
setmaxage       <bridge> <time>         set max message age
setpathcost     <bridge> <port> <cost>  set path cost
setportprio     <bridge> <port> <prio>  set port priority
stp             <bridge> <state>        turn stp on/off
```

5 CONFIGURACIÓN DEL AP Y EL PUENTE DE RED

Si el módulo del driver de la tarjeta WLAN 802.11 está cargado al ejecutar el comando `iwconfig` deberíamos ver algo como:

```
# iwconfig
lo        no wireless extensions.

eth0      no wireless extensions.

eth1      NOT READY!  ESSID:off/any
          Mode:Managed  Channel:6  Access Point: 00:00:00:00:00:00
          Tx-Power=31 dBm   Sensitivity=0/200
          Retry min limit:0   RTS thr=0 B   Fragment thr=0 B
          Encryption key:off
          Link Quality:0  Signal level:0  Noise level:0
          Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
          Tx excessive retries:0  Invalid misc:0   Missed beacon:0
```

En lo que observamos que la tarjeta 802.11 está asociada al dispositivo eth1. En caso de no ver ningún dato, puede que se deba a que no tenemos cargado el módulo de la tarjeta. Podemos cargarlo con `modprobe prism54` si se trata de una tarjeta 802.11g o con `modprobe` y el nombre del módulo en caso de trabajar con otra tarjeta.

Una vez cargado el módulo de la tarjeta de red correctamente debemos asignar los parámetros básicos de nuestra conexión inalámbrica. Estos son:

- **essid**. identifica el área en el que nuestro AP va a dar cobertura. El AP emitirá constantemente este parámetro por el medio. Cuando un cliente se quiera conectar a nuestro AP buscará el essid de nuestro AP en el medio y entonces le localiza físicamente.

- **channel**. Se refiere al canal de comunicaciones o frecuencia a la que queremos transmitir. Los canales se establecen con saltos de 5 Mhz de tal forma que el canal 1 está en 2,412 GHz, el 2 en 2,417 y así hasta el 14. Para cubrir la banda de 5 GHz (sólo 802.11 a y g) también se establecen otros 11 canales separados 10 MHz (ó 20 MHz) empezando en 5,17 GHz. Debemos procurar configurar algún canal diferente al de otras posibles WLAN del entorno.

- **mode**. denota el modo de trabajo de la tarjeta. En principio hay 4:

  - mode **master**: trabaja como un AP

  - mode **managed** (modo por defecto): trabaja como una estación cliente que se conectará al AP definido en essid o a cualquiera si no se define dicho parámetro

  - mode **ad-hoc**: para funcionar como una red ad-hoc en que todas las estaciones se tratan de igual a igual (sin infraestructura)

  - mode **monitor**: permite escuchar el medio de forma pasiva, es decir, sin asociarse o identificarse ante ningún otro dispositivo.

Es decir, que si queremos crear un AP con essid CASABLANCA en el canal 11 debemos introducir

```
# iwconfig eth1 mode master essid CASABLANCA channel 11
```

y si ahora tecleamos iwconfig deberá aparecernos algo como lo siguiente

```
# iwconfig
lo        no wireless extensions.

eth0      no wireless extensions.

eth1      NOT READY!  ESSID:"CASABLANCA"
          Mode:Master  Channel:11  Access Point: 00:00:00:00:00:00
          Tx-Power=31 dBm   Sensitivity=0/200
          Retry min limit:0   RTS thr=0 B   Fragment thr=0 B
          Encryption key:off
          Link Quality:0  Signal level:0  Noise level:0
          Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
          Tx excessive retries:0  Invalid misc:0   Missed beacon:0
```

donde se puede apreciar que todavía vemos la tarjeta como NOT READY!. Esto es debido a que debemos darla de alta como tarjeta de red en el sistema operativo, lo que se hace con el comando `ifconfig eth1 up`. Si ahora volvemos a ejecutar `iwconfig` obtenemos:

```
# iwconfig
lo        no wireless extensions.

eth0      no wireless extensions.

eth1      IEEE 802.11b/g  ESSID:"CASABLANCA"
          Mode:Master  Channel:11  Access Point: 00:04:E2:A4:B2:8A
          Bit Rate:54Mb/s   Tx-Power=31 dBm   Sensitivity=20/200
          Retry min limit:8   RTS thr:2347 B   Fragment thr:2346 B
          Encryption key:off
          Link Quality:0  Signal level:0  Noise level:0
          Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
          Tx excessive retries:0  Invalid misc:0   Missed beacon:0
```

Como se puede observar ahora si que tenemos nuestro AP activo. Está utilizando el protocolo 802.11b y g; está dado de alta en el essid CASABLANCA; trabaja como AP (mode: Master); funciona sobre el canal 11; y tiene como dirección física única 00:04:E2:A4:B2:8A. De momento, como se puede observar, no hay clave de encriptación.

En este punto tendríamos un AP funcionado y anunciándose en el medio, pero que todavía no da acceso a nada. Como nuestro objetivo es dar acceso a la red ethernet (en el dispositivo eth0), debemos configurar el puente de red entre la tarjeta 802.11 (eth1) y la ethernet (eth0).

El primer paso será activar ambas tarjetas en el sistema pero sólo a nivel físico (deshabilitando TCP/IP), lo que realizamos con:

```
# ifconfig eth0 0.0.0.0 up
# ifconfig eth1 0.0.0.0 up
```

Una vez activadas creamos un puente de red que llamaremos *wlan-bridge* con:

```
# brctl addbr wlan-bridge
```

Ahora añadimos las dos tarjetas al puente de red con:

```
# brctl addif wlan-bridge eth0
# brctl addif wlan-bridge eth1
```

Y para terminar activamos el puente de red como un dispositivo de red en el sistema con:

```
# ifconfig wlan-bridge up
```

En este punto cualquier usuario 802.11 que se conecte a nuestro AP tendrá acceso transparente a la red ethernet, luego ya hemos alcanzado nuestro objetivo inicial de dar acceso a la red ethernet a los estaciones de la WLAN.

Ahora bien, muy posiblemente no deseemos que cualquier estación 802.11 pueda conectarse a nuestra red ethernet (posiblemente conectada a Internet). Imaginemos que montamos el AP en nuestra casa y lo que hacemos es dar Internet gratuito a nuestros vecinos. Seguramente, si los que pagamos somos nosotros sólo querremos que se conecten nuestros ordenadores, es decir, querremos un mecanismo de autenticación/autorización que deniegue el acceso al resto.

Todas las tarjetas de red 802.11 con chipset Intersil Prism X (como la nuestra) permiten al menos dos tipos de autenticación:

- Autenticación basada en secreto compartido (parte del WEP - *Wired Equivalent Privacy*)

- Autenticación basada el dirección MAC única de cada dispositivo de red

La primera es estándar de cualquier dispositivo 802.11 y se basa en la compartición de un secreto entre el AP y las estaciones cliente. Este secreto debe ser de 40 bits ó 104 bits, y se usa para cifrar/descifrar la información enviada entre el AP y las estaciones. Para configurar una clave en el AP debemos escribirla en hexadecimal (10 dígitos - 40 bits, 26 dígitos - 104 bits) o en ASCII (5 caracteres - 40 bits, 13 caracteres - 104 bits). Por ejemplo, si quisiéramos establecer la contraseña en código ASCII *qwert* (*7177657274* en hexadecimal) la podemos introducir como:

```
# iwconfig eth1 enc s:"qwert"
```

o

```
# iwconfig eth1 enc 7177657274
```

Evidentemente se ha de configurar la misma clave en cada estación que vaya a conectarse al AP, y sobretodo utilizar claves seguras y actualizarlas con frecuencia. Existen diversas páginas de Internet desde las que podemos generar claves seguras para WEP, un ejemplo lo podemos encontrar en [11].

La segunda forma de autenticación no es estándar del protocolo 802.11, pero sí que viene implementada en la mayoría de dispositivos. Se trata simplemente de establecer una política de aceptación basada en la dirección MAC única de cada tarjeta de red. Al no tratarse de un estándar la debemos configurar con el comando iwpriv que nos permite manipular parámetros específicos de cada driver. Lo primero es establecer que política vamos a usar de las tres disponibles:

- MAC_POLICY_OPEN = 0 : Se acepta a cualquier cliente.

- MAC_POLICY_ACCEPT = 1 : Se acepta a cualquier cliente excepto a aquellos cuya MAC esté en la lista.

- MAC_POLICY_REJECT = 2 : Se rechaza a cualquier cliente excepto a aquellos cuya MAC esté en la lista.

Y después añadir las MACs de las tarjetas de red que no queremos dejar acceder o a las que solamente queremos dejar acceder. Por ejemplo, si en nuestra red sólo tenemos un ordenador que se va conectar vía 802.11, y cuya dirección MAC única de su tarjeta de red es 00:04:E2:A5:C3:DE, deberíamos ejecutar los siguientes comandos:

```
# iwpriv eth1 setPolicy 2
# iwpriv eth1 addMac 00:04:E2:A5:C3:DE
```

Con lo que nos aseguraríamos que sólo se utiliza nuestra red desde esta máquina.

Nótese que ambos sistemas de autenticación son perfectamente compatibles y pueden complementarse.

6 SUMARIO Y CONCLUSIÓN

Actualmente las redes inalámbricas están teniendo un desarrollo muy grande tanto a nivel corporativo como doméstico, sin embargo las soluciones domésticas suelen tener todavía un precio elevado, especialmente cuando hablamos del AP. En este artículo hemos explicado como montar un AP en un ordenador con Linux ya conectado a una red ethernet y una tarjeta de red 802.11. También hemos explicado como implementar unos mecanismos de seguridad básicos pero que, por otra parte, son perfectamente válidos para evitar ataques «casuales».

De todas formas los mecanismos de seguridad que actualmente vienen por defecto presentan muchísimas debilidades y se han presentado diversas formas de saltárselos en un periodo de tiempo muy reducido. Airsnort ([12]) es un ejemplo de herramienta de código libre que circula por Internet que nos permite recuperar la clave WEP de cifrado/descifrado en muy pocas horas. Por ello los fabricantes han implementado WPA (*Wi-Fi Protected Access*) que mejora sustancialmente el estándar actual, y que está previsto que salga como estándar de forma inminente bajo el nombre de 802.11i. Algunas de las tarjetas que podemos adquirir soportan WPA, pero en general los drivers de Linux no son los suficientemente funcionales en este aspecto (por el momento).

7 AGRADECIMIENTOS

Este trabajo ha sido soportado por el proyecto DISQET [CICYT TIC2002-00249], dentro del Plan Nacional de I+D.

REFERENCIAS

[1] Gast, M. *802.11 Wireless Networks – The Definitive Guide*. O'Reilly, Abril 2002. ISBN 0-596-00183-5

[2] Nichols, R.K.; Lekkas, P.C. *Seguridad para comunicaciones inalámbricas*. McGraw-Hill, 2003. ISBN 84-481-3782-5

[3] Geier, J. *Wireless LANs. Implementing High Performance IEEE.11 Networks*. SAMS. 2001.

REFERENCIAS A HIPERTEXTO

[4] http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/Linux.Wireless.drivers.802.11b.html#Prism2

[5] http://prism54.org/supported_cards.php

[6] http://hostap.epitest.fi/

[7] http://prism54.org/

[8] http://prism54.org/download/

[9] http://prism54.org/~mcgrof/firmware/

[10] http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/Tools.html

[11] http://www.warewolflabs.com/portfolio/programming/wepskg/wepskg.html

[12] http://airsnort.shmoo.com/

AUTORES

Josep Pegueroles *(Tortosa 1974) recibió el título de Ingeniero de Telecomunicación en 1999 y el de doctor en 2003, ambos por la UPC. En 1999 entró a formar parte del Grupo de Seguridad de la Información - ISG dentro de la Línea de Investigación de Servicios Telemáticos - SERTEL del Departamento de Ingeniería Telemática (http://sertel.upc.es). Actualmente es profesor asociado de la ETSETB y sus intereses de investigación incluyen la seguridad para servicios multimedia en red y las comunicaciones seguras de grupo.*

Juan Hernández *(Salamanca 1979) recibió el título de Ingeniero de Telecomunicaciones por la UPC en 2002. Ese mismo año pasó a formar parte del Grupo de Seguridad de la Información - ISG dentro de la Línea de Investigación de Servicios Telemáticos en el Departamento de Ingeniería Telemática de la UPC. Actualmente es estudiante de doctorado de la Universidad Politécnica de Cataluña (UPC) en el Departamento de Ingeniería Telemática y su investigación se centra en seguridad en redes ubícuas y gestión de claves de grupo para redes multicast.*