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

Access Point con PrismGT - ISL3890 (51024 lectures)

Per **Luís Calero (n3cr05)**, ***n3cr05*** (http://www.vector0x00.com)

Creado el 27/11/2003 23:15 modificado el 01/12/2003 17:31

En este artículo muestro como montar un Access Point (Wireless) bajo una distro linux, utilizando un dispositivo con chip prismGT. En este caso se trata de una PCI SMC2802w, instalada en un preciado MMX 200,que con un potente linux puedes hacer virguerías...
Como novedad,esta placa soporta 802.11g y modalidad NITRO.
Que lo disfruteis !!! Ondas de info para todos.

DRIVER ISL3890 - PRISM GT en LINUX
Ver 1.2
By Luís Calero (NeCrOS)
http://www.NeCrOs.com(1)

Este artículo no es más que una aplicación práctica de la documentación proporcionada por : ruslug.rutgers.edu/~mcgrof/802.11g/Documentation/(2)

Author: Luis R. Rodriguez
Documenation Version: 0.0.5-2

!!!NOTAS:
1) La placa utilizada es la versión I, actualmente se estan vendiendo placas 2802 PCI de tipo v.2, todavía no he comprovado si esta aguanta el driver de igual modo...
v1 tiene Part No : 99-012084-178
v2 tiene Part No : 99-012084-229

2) El kernl 2.6.6 lleva incluido el driver prismgt.

(Actualmente estoy trabajando sobre el soporte DWS)

Este artículo se redacta de forma práctica, tal y como se ha seguido el manual original de instalación del driver, exponiendo los fallos encontrados y como se han solventado.
Aconsejo leer de principio a fin antes de proceder de forma física.
El autor no se hace responsable de las posibles consecuencias del uso de este mini How-to.

Hardware Utilizado:
Tarjeta Wireles :SMC2802w PCI(3)
Máquina :Intel 200 MMX (4)

(Empieza la batalla)
INSTALACIÓN SMC2802W, montar un AP (Punto de Acceso) en un 200 MMX.

Q.- ¿ Pq esta tarjeta y no otra ?
R.- Creo que la mejor forma de verlo es mediante algunas de las ventajas q me hicieron decidir:
- Tipo 802.11g , q puede proporcionar un ancho máximo de 54Mb, manteniendo la compatibilidad con el estándar 802.11b
- PCI , muchas de las tarjetas que llevan montado este tipo de chip o el famoso prism2,

estan montados sobre dispositivos PCMCIA, lo que limita en el caso de trabajar con un sobre mesa, la utilización de un adaptador PCMCIA PCI, con el coste asociado.
- Antena incorporada extraíble, tipo SMC, lo que nos permitirá adquirir un pigtail para conectar una antena externa.
- Dispone del chip PRISM GT (Subtipo Duette), existe un driver para linux que nos permite su uso como AP (tipo master),en definitiva, podemos pontar un punto de acceso bajo Linux.

Q .- ¿ Este driver sólo se limita para este chipset ?
R .- Este driver puede trabajar con los siguientes chipset:
- ISL3880 - Prism GT
- ISL3877 - Prism Indigo
- ISL3890 - Prism Duette

En nuestro caso :
SMC2802W - EZ Connect g 2.4GHz 54 Mbps Wireless PCI Card
SMC2802W PCI 1260:3890 Success

Q.- ¿Sólo funciona para este tipo de tarjeta?
R.- No, los desarrolladores del driver, mantienen un listado de los modelos testeados en :
http://ruslug.rutgers.edu/~mcgrof/802.11g/Documentation/supported_cards.php[5]
Yo por el momento sólo lo he testeado con la SMC2802w, obteniendo un resultado factible en una máquina de bajo coste.

Q.- ¿ Q hace exactamente este driver ?
R.- Este driver permite al Sistema operativo interactuar con la tarjeta Wireless,haciendo uso de ella como un dispositivo de red más.

Procedemos con la instalación :
La distro utilizada es RedHat 9 (RH9), con un kernel base 2.4.20 que actualizaremos a posteriori.

Q.- ¿ Pq RH9 y no otro?
R.- La experiencia me indica que RedHat es una de las mejores distribuciones para la detección de nuevos dispositivos,
y en este caso es una gran ventaja Tb decir que en máquinas paralelas estoy trabajando en Suse 8.2 profesional, con unos resultados satisfactorios. En los 2 casos estoy utilizando kernel >= 2.4.22 (ver estable a fecha de hoy).
Otra gran ventaja es que estoy utilizando una máquina sencilla como es un 200 MMX con 64 MB, con unos resultados realmente
satisfactorios, la RH9 corre sin ningún problema.

Histórico de movimientos:
- Instalamos RH9 a nivel base como tipo servidor.
- Rh9 en la detección de la tarjeta Wireless falla (pasa de ella totalmente).
- Una vez estamos dentro del nuevo sistema instalado verificamos que no tenemos ni rastro del uso de la nueva tarjeta:
  # ifconfig -a
  (No es visible la interficie,debemos actualizar el kernel para que se de cuenta de la existencia de esta placa y haga uso
  de ella. Recordemos que por el momento no hemos instalado el driver adecuado que se encargará de ello,
  eso no implica que en versiones de kernel futuras, este driver ya se incluya por defecto.

- Bajamos el paquete ISL3890-0.1.0 (es el driver)
wget http://ruslug.rutgers.edu/~mcgrof/802.11g/packages/ISL3890-0.1.0.tar.gz

- Bajamos el nuevo kernel:
wget http://www.kernel.org/pub/linux/kernel/v2.4/linux-2.4.22.tar.bz2

(Esta es la última versión del driver a fecha de hoy, como siempre bajar la última versión estable, en el caso del kernel

mas de lo mismo. En el ejemplo realizo el proceso sobre el 2.4.22,afirmando que funciona correctamente, en versiones de
kernel inferiores no lo he probado, si alguien lo testea, que me envie los resultados y los publicaré)

- Instalamos paquetes *estamos en /root*
tar -zxvf ISL3890-0.1.0.tar.gz && tar -jxvf linux-2.4.22.tar.bz2

- Wireless extensions : Herramientas que nos permiten hacer uso de dispositivos wireless.
En las versiones de kernel igual o superior a la 2.4.22 las wireless extensions se compilan por defecto,
en el caso de que deseemos el uso de un dispositivo wireless.

En configuración de kernel (activación wireless extensions) :
Network device support -> Wireless LAN (non-hamradio) (Seleccionar)

Mas info : http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/Tools.html#links(6)

- Parcheado de kernel.
Es necesario parchear el kernel antes de iniciar su configuración/compilación, este parcheado permitirá al nuevo kernel compilar código necesario para el uso del driver que instalaremos más tarde.
En este caso no pasa como en el driver hostap, que el parcheo permitía
la selección de diferentes opciones en el menuConfig.

(Entiendo que tenemos el paquete del driver y el kernel descompactados al mismo nivel del arbol de directorios)

cd linux-2.4.22
[root@unit0 linux-2.4.22]# patch -p1 < ../ISL3890-0.1.0/patches/kernel-intersil.patch

patching file net/core/dev.c
Hunk #1 succeeded at 2682 (offset -26 lines).
patching file include/linux/netlink.h

- Copiamos a cabecera que incluye las funciones que permiten el uso y configuración del dispositivo.

[root@unit0 linux-2.4.22]# cp ../ISL3890-0.1.0/patches/isil_netlink.h include/linux/

- Atacamos el kernel (Configuración & Compilación)

Se ha observado que para compilaciones de este tipo de drivers (modulos),
el compilador gcc 3.3 proporciona errores de compilación, esto se extiende
tb en la compilación de otros módulos de este estilo como son el hermesAP o hostAP.
Se aconseja el uso de la versión 2.95. La instalación de este paquete encuentro que no
procede en este mini-manual.
En nuestro caso hemos utilizado gcc version 3.2.2 y NO PROBLEM

Nota :Yo he trabajado en /root por defecto de forma, pero lo ideal sería
trabajar bajo el /usr/src respetando que los 2 paquetes kernel y
ISL esten descompactados al mismo nivel ...
Como podreís ver a lo largo del artículo yo hago referencia
en el tratamiento de kernel desde /usr/scr/linux , es lo correcto .
la forma de hacer es creando un Soft link al paquete descompactado
en /root :
 # cd /usr/src

 # ln -s /root/linux-2.4.22 linux
asi ya tenemos que el directorio /usr/src/linux apunta al /root/linux-2.4.22

- Procedemos
#cd /usr/src/linux
make mrproper (No se requiere si es la primera compilación del kernel)
make menuconfig

Opciones a tener en cuenta en la configuración kernel:
Básicas .-
" Code maturity level options -> o Prompt for development and/or incomplete drivers
" Loadable module support -> o Enable loadable module support
o Set version information on all module symbols o Kernel module loader
" General setup -> o Networking support
" Wireless Support (Asi instalaremos las Wireless Extensions)

Casos especiales a tener en cuenta: " Si tenemos CardBus/Tarjeta PCMCIA y tenemos activado la PCMCIA SUPPORT en nuestro kernel.
En este caso se nos solicita que no seleccionemos el soporte PCMCIA,
la razón es que necesitamos que este desabilitado el soporte PCMCIA,
para que no interfiera en la instalación posterior que debemos realizar.
Lo mismo pasa para el driver hermesAP o hostAP.

General setup ---> PCMCIA/Cardbus support (CONFIG_PCMCIA) NO

En nuestro caso, tarjeta PCI, !!! No expuesto en la documentación encontrada, el controlador de dispositivo USB, utiliza la misma IRQ que esta tarjeta, asi que para no tener conflictos, lo desactivamos
el soporte USB del kernel.Si alguien lo ha probado con éxito, que me proporcione la documentación necesaria para documentarlo en nuevas versiones del artículo.

" Si deseamos poder poner la tarjeta como Master, modo de función AP (Punto de Acceso).
- En este caso debemos seleccionar la opción de bridging support.
Networking options ---> 802.1d Ethernet Bridging (CONFIG_BRIDGE)

- Debemos editar :
ISL3890-0.1.0/pcmcia-cs-3.2.4-intersil/wireless/intersil/islpci_mgt.h
Y realizar el cambio :

#define CARD_DEFAULT_MODE INL_MODE_CLIENT
por
#define CARD_DEFAULT_MODE INL_MODE_AP

(De esta forma le indicaremos al paquete que deseamos modo AP = Master)
Si deseamos q la tarjeta trabaje en modo por defecto cliente = managed,
dejemos la entrada por defecto.

- Compilación del nuevo kernel :
#make menuconfig
#make dep
#make bzImage
#make modules
#make modules_install

" Forma simplificada si sabemos que no habrá ningun error :
#nohup make menuconfig && make dep && makebzImage && make modules && make modules_install &

" Copiamos el nuevo nucleo en la particion de boteo.

#cp /usr/src/linux/arch/i386/boot/bzImage /boot

" Copiamos el System.map nuevo.
#cp /usr/src/linux/System.map /boot

" Creamos la imagen para la carga inirtd en RAM en el booteo:
mkinirtd /boot/initrd-2.4.22.img 2.4.22

" Modificamos el fichero de configuración del programa de arranque GRUB:
#vi /boot/grub/menu.lst

splashimage=(hd0,0)/grub/splash.xpm.gz
title Red Hat Linux (2.4.22)
root (hd0,0)
kernel /bzImage root=/dev/hda2 #Particion dnde tnmos root /
initrd /initrd-2.4.22.img

title Red Hat Linux (2.4.20-8)
root (hd0,0)
kernel /vmlinuz-2.4.20-8 ro root=LABEL=/
initrd /initrd-2.4.20-8.img

- Carga del nuevo nucleo:
#reboot
(La máquina reinicia sin ningún problema)

```
[root@unit0 root]# dmesg
Linux version 2.4.22 (root@localhost.localdomain)
(gcc version 3.2.2 20030222 (Red Hat Linux 3.2.2-5))
```

Una vez haya arrancado la nueva imagen,actualizamos la dependencia de los nuevos modulos que puede cargar el nuevo núcleo :

```
[root@unit0 root]# depmod  -ae

Modulos cargados por el nuevo kernel :
[root@unit0 root]# lsmod
Module                  Size  Used by    Not tainted
iptable_filter          2316   0  (autoclean) (unused)
ip_tables              18592   1  [iptable_filter]
```

(Respirar ondo q esto todavía no ha terminado)

FIRMWARE
Actualización del FirmWare de la tarjeta :

El siguiente paso es la modificación del Firmware de la tarjeta, no es una actualización física del chip de tarjeta, si no que se realiza sobre el driver que la gestiona, de esta forma el Kernel cree que es la misma tarjeta la que contesta.

Estamos en /root .
(Recordemos que el driver lo teniamos descompactado en /root/ISL...)

- Bajamos el firmware = un fichero .arm

```
[root@unit0 root]# wget http://ruslug.rutgers.edu/~mcgrof/802.11g/firmware/1.0.4.3.arm
--17:46:25--  http://ruslug.rutgers.edu/%7Emcgrof/802.11g/firmware/1.0.4.3.arm
           => `1.0.4.3.arm'
Resolving ruslug.rutgers.edu... done.
```

```
Connecting to ruslug.rutgers.edu[128.6.24.131]:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 93,996 [text/plain]

100%[==============================================>] 93,996
26.48K/s    ETA 00:00
17:46:29 (26.48 KB/s) - `1.0.4.3.arm' saved [93996/93996]
```

- Copiamos el fichero bajado dentro del directorio del driver.

```
[root@unit0 root]# cp 1.0.4.3.arm  \
ISL3890-0.1.0/pcmcia-cs-3.2.4-intersil/wireless/intersil/isl3890.arm
```

!!!Importante: Por problemas a posteriori, bajamos el .arm de la propia tarjeta del Site de SMC Este se encuentra dentro del .zip , driver de windows, y tiene la extensión .arm . Realizamos la copia en el directorio indicado anteriormente, pq deseamos que el driver utilice este y no el otro.

```
[root@unit0 root]# cp SMC2802w.arm \
ISL3890-0.1.0/pcmcia-cs-3.2.4-intersil/wireless/intersil/isl3890.arm
```

(Asi cuando se realice la instalación del driver en el arbol de ficheros de
configuración del driver,ara uso del firmware copiado)

Localizaciones del Firmware una vez se ha terminado la instalación:
Si tienes PCMCIA/Cardbus card -> /etc/pcmica/isl3890.arm
Si tienes PCI / miniPCI card -> /etc/hotplug/isl3890.arm

En nuestro caso, se trata de una tarjeta PCI , asi que utilizamos la ubicación /etc/hotplug/isl3890.arm , esto nos indica simplemente que en cualquier momento a posteriori, si deseamos
cambiar el firmware de la tarjeta q debe cargar el módulo, simplemente debemos modificar
el fichero indicado. Esto es aconsejable en el caso de la realización de nuevas versiones del firmware
(Siempre es mejor estar a la última).

De todas forma el fichero siempre debe tener el siguiente nombre : isl3890.arm

---

ISL3890-0.1.0 . Configuración
(Estamos en /root)

```
[root@unit0 root]# cd   ISL3890-0.1.0/pcmcia-cs-3.2.4-intersil/
[root@unit0]# pcmcia-cs-3.2.4-intersil]#  ./Configure

    -------- Linux PCMCIA Configuration Script --------

The default responses for each question are correct for most users.
Consult the PCMCIA-HOWTO for additional info about each option.

PC card source directory /root/ISL3890-0.1.0/pcmcia-cs-3.2.4-intersil
Linux kernel source directory [/usr/src/linux]:
```

Apretamos Intro ya que en [] tenemos el valor por defecto y vemos que es la ubicación correcta

```
The kernel source tree is version 2.4.22.
The current kernel build date is Mon Oct 21 15:55:02 2030.

Build 'trusting' versions of card utilities (y/n) [y]: y
Include 32-bit (CardBus) card support (y/n) [y]: n
```

En este caso hemos respondido n, pq se trata de una tarjeta PCI, si fuera cardbus/Pcmcia diriamos[y]

```
Include PnP BIOS resource checking (y/n) [n]: y
```

```
Module install directory [/lib/modules/2.4.22]:
```

De nuevo el valor por defecto es correcto, se trata de la correcta ubicación de los modulos

```
Include wireless ioctls (J. Tourrilhes) (y/n) [y]: y
Include wireless events (J. Tourrilhes) (y/n) [y]: n
Include intersil events (y/n) [y]: y
Include Access Point WDS links (y/n) [n]: y

Kernel configuration options:
    Kernel-tree PCMCIA support is disabled.
    Symmetric multiprocessing support is enabled.
    Preemptive kernel support is disabled.
    High memory support is disabled.
    PCI BIOS support is enabled.
    Power management (APM) support is enabled.
    SCSI support is disabled.
    IEEE 1394 (FireWire) support is disabled.
    Networking support is enabled.
     Radio network interface support is enabled.
     Token Ring device support is disabled.
     Fast switching is disabled.
     Frame Diverter is disabled.
    Module version checking is enabled.
    Kernel debugging support is disabled.
    Preemptive kernel patch is disabled.
    /proc filesystem support is enabled.
    PAE support is disabled.

The standalone IEEE 1394 CardBus drivers are not supported with this
    kernel.  If you need them, use the kernel PCMCIA subsystem.

It looks like you have a System V init file setup.
The Forms library is not available.
The X11/Xaw libraries are not available.
The GTK+ library is not available.

Configuration successfull
```

Existe una reseña, si se tratara de una tarjeta cardbus/Pcmcia :

```
# Kernel-tree PCMCIA support is disabled
```

Si no fuera asi, es que te has olvidado de no seleccionar el soporte PCMCIA, como indicabamos en las opciones a escoger en la configuración del kernel.

ISL3890-0.1.0 compilación Instalación

```
[root@unit0 pcmcia-cs-3.2.4-intersil]# make all
[root@unit0 pcmcia-cs-3.2.4-intersil]# make install
(....)
-> Installing PCMCIA startup script as /etc/rc.d/init.d/pcmcia
-> Updating client scripts in /etc/pcmcia
->   Configuring /etc/pcmcia/network.opts for Red Hat
-> Running depmod...
make[1]: Leaving directory
```

- Ahora instalaremos algunas utilidades de soporte que nos proporciona el driver...

```
cd ../../debug-tools
root@unit0 debug-tools]# pwd
/root/ISL3890-0.1.0/pcmcia-cs-3.2.4-intersil/debug-tools
[root@unit0 debug-tools]# make all
make: Nothing to be done for `all'.
```

```
[root@unit0 debug-tools]# make install
cp -f dump_cis pack_cis /sbin
cp -f lspnp setpnp /sbin
cp -f pnp.ids /usr/share

[root@unit0 debug-tools]# iwconfig
lo        no wireless extensions.
eth0      no wireless extensions.
eth1      no wireless extensions.
```

(!!!!!!!!! Ohhhhh que desilusión, y es que falta la configuración del driver Y después se quejan de la plataforma Bill Gates.)

Configuración del driver
Configurando Cardbus/PCMCIA en tu Sistema Operativo.
!No es nuestro caso, ya que nosotros disponemos de un dispositivo PCI,
pero no esta de más.

Si deseamos configurar una tarjeta PCMCIA,
Debian -> /etc/default/pcmcia
Red Hat -> /etc/sysconfig/pcmcia

Introducimos la linea en negrita, en el caso de que no este ya :

```
 # vi /etc/sysconfig/pcmcia

PCMCIA=yes
PCIC=yenta_socket
PCIC_OPTS=
CORE_OPTS=
PCIC=i82365
```

Cargando el driver

Como hemos visto antes, la tarjeta todavía no estaba detectada, asi que
procedemos a que el sistema se de cuenta que tiene una tarjeta Wireless,
y que deseamos hacer uso de ella.
Tenemos PCMCIA/Cardbus card -> modprobe islpci_cb
Tenemos PCI /miniPCI card -> modprobe islpci

En nuestro caso como se trata de una tarjeta PCI :
[root@unit0 debug-tools]# modprobe islpci

(!!!! La hora de la verdad, cruzar los dedos .)

```
 [root@unit0 debug-tools]# iwconfig

lo        no wireless extensions.
eth0      no wireless extensions.
eth1      no wireless extensions.
eth2      PRISM Duette  Mode:Ad-Hoc  Frequency:2.437GHz  Bit Rate=0kb/s
          RTS thr=2347 B   Fragment thr=2346 B
```

(!!! Diós mio ha funcionado, lo siguiente es hacerme budista... Estaría de p.m una antena en lo alto del Tibet ;))

Detalle, fijemonos que en esta instalación se ha actualizado los módulos a cargar por el kernel, asi que por coherencia sería conveniente crear un nuevo fichero imagen de los modulos del sistema.

```
# mkinird /boot/inirtd_islpci.img  2.4.22
```

Ahora modificamos en el fichero /boot/grub/menu.lst , el fichero .img que carga por defecto el nuevo kernel...

Si se desea, se puede realizar una purga del antiguo kernel + ficheros .img que ya no se utilizaran.

Configuración de la tarjeta.
Hacemos un barrido básico del sistema a ver como esta el temilla,

```
[root@unit0 root]# /etc/rc.d/rc3.d/S10network restart
Shutting down interface eth0:                              [  OK  ]
Shutting down interface eth2:                              [  OK  ]
Shutting down loopback interface:                          [  OK  ]
Setting network parameters:                                [  OK  ]
Bringing up loopback interface:                            [  OK  ]
Bringing up interface eth0:                                [  OK  ]
Bringing up interface eth2:                                [  OK  ]

[root@unit0 root]# ifconfig eth2
eth2      Link encap:Ethernet  HWaddr 00:04:E2:64:49:80
          inet addr:192.168.1.20  Bcast:192.168.1.255 Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:34 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:4908 (4.7 Kb)  TX bytes:384 (384.0 b)
          Interrupt:12 Memory:c404a000-c404c000


Si no tiene ip, pues se la asignamos, siguiendo el ejemplo sobre la
 interficie eth2 :

 # ifconfig eth2 192.168.x.x  up



[root@unit0 root]# iwconfig
lo        no wireless extensions.
eth0      no wireless extensions.
eth1      no wireless extensions.
eth2      PRISM Duette  ESSID:"ReusWireless - NeCrOS"
          Mode:Ad-Hoc  Frequency:2.412GHz  Bit Rate=54Mb/s
          RTS thr=2347 B   Fragment thr=2346 B
```

Ahora probaremos de poner la tarjeta en modo Master = ponemos el servidor en AP.

```
[root@unit0 root]# iwconfig eth2 mode master
[root@unit0 root]# iwconfig
lo        no wireless extensions.
eth0      no wireless extensions.
eth1      no wireless extensions.
eth2      PRISM Duette  Mode:Master  Frequency:2.412GHz  Bit Rate=0kb/s
          RTS thr=2347 B   Fragment thr=2346 B
```

(!!! Observamos que esta configurado , pero nada funciona...)
Detalles a tener en cuenta :
El Ap utiliza wep por defecto, asi q se procede a crear un script de carga con todos
los parametros necesarios. Además he encontrado que el uso del fichero de configuración de devices
proporcionado por Rh9 en /etc/sysconfig/network-scripts/ifcfg-ethX da problemas
,si no lo utilizamos, nos carga todos los parámetros por
defecto de la tarjeta sin problemas.
Asi que visto lo acontecido, dejaremos q el sistema no cargue, ni configure por su
propia mano la tarjeta, se lanzará un script a posteriori de cosecha propia .
A parte, se aconseja la utilización de las herramientas propias de configuración del
driver como son **setoid** & **getoid** propocionadas por el driver.

A parte, para un buen funcionamiento, antes de configurar el SSID, cargar las keys.
En las pruebas realizadas, siempre que la tarjeta se quedaba pillada
era necesario un cierre físico de máquina.

Listado de los OID, q nos permitirá parametrizar la nueva tarjeta, los que en mi opiñón son los más importantes: (No os asusteis, más adelante proporciono scripts donde muestro el uso práctico de estos parámetros)

```
/*
 * 802.11 OIDs
 */

#define DOT11_OID_BSSTYPE               0x10000000
#define DOT11_OID_BSSID                 0x10000001
#define DOT11_OID_SSID                  0x10000002
#define DOT11_OID_STATE                 0x10000003
#define DOT11_OID_AID                   0x10000004
#define DOT11_OID_COUNTRYSTRING         0x10000005
#define DOT11_OID_SSID12OVERRIDE        0x10000006

#define DOT11_OID_MEDIUMLIMIT           0x11000000
#define DOT11_OID_BEACONPERIOD          0x11000001
#define DOT11_OID_DTIMPERIOD            0x11000002
#define DOT11_OID_ATIMWINDOW            0x11000003
#define DOT11_OID_LISTENINTERVAL        0x11000004
#define DOT11_OID_CFPPERIOD             0x11000005
#define DOT11_OID_CFPDURATION           0x11000006

#define DOT11_OID_AUTHENABLE            0x12000000
#define DOT11_OID_PRIVACYINVOKED        0x12000001
#define DOT11_OID_EXUNENCRYPTED         0x12000002
#define DOT11_OID_DEFKEYID              0x12000003
#define DOT11_OID_DEFKEY1               0x12000004
#define DOT11_OID_DEFKEY2               0x12000005
#define DOT11_OID_DEFKEY3               0x12000006
#define DOT11_OID_DEFKEY4               0x12000007
#define DOT11_OID_STAKEY                0x12000008
#define DOT11_OID_REKEYTHRESHOLD        0x12000009
#define DOT11_OID_STASC                 0x1200000a

#define DOT11_OID_PRIVTXREJECTED        0x1a000000
#define DOT11_OID_PRIVRXPLAIN           0x1a000001
#define DOT11_OID_PRIVRXFAILED          0x1a000002
#define DOT11_OID_PRIVRXNOKEY           0x1a000003

#define DOT11_OID_RTSTHRESH             0x13000000
#define DOT11_OID_FRAGTHRESH            0x13000001
#define DOT11_OID_SHORTRETRIES          0x13000002
#define DOT11_OID_LONGRETRIES           0x13000003
#define DOT11_OID_MAXTXLIFETIME         0x13000004
#define DOT11_OID_MAXRXLIFETIME         0x13000005
#define DOT11_OID_AUTHRESPTIMEOUT       0x13000006
#define DOT11_OID_ASSOCRESPTIMEOUT      0x13000007

#define DOT11_OID_ALOFT_TABLE           0x1d000000
#define DOT11_OID_ALOFT_CTRL_TABLE      0x1d000001
```

```
#define DOT11_OID_ALOFT_RETREAT         0x1d000002
#define DOT11_OID_ALOFT_PROGRESS        0x1d000003
#define DOT11_OID_ALOFT_FIXEDRATE       0x1d000004
#define DOT11_OID_ALOFT_RSSIGRAPH       0x1d000005
#define DOT11_OID_ALOFT_CONFIG          0x1d000006

#define DOT11_OID_VDCF0                 0x1b000000
#define DOT11_OID_VDCF1                 0x1b000001
#define DOT11_OID_VDCF2                 0x1b000002
#define DOT11_OID_VDCF3                 0x1b000003
#define DOT11_OID_VDCF4                 0x1b000004
#define DOT11_OID_VDCF5                 0x1b000005
#define DOT11_OID_VDCF6                 0x1b000006
#define DOT11_OID_VDCF7                 0x1b000007
#define DOT11_OID_MAXFRAMEBURST         0x1b000008

#define DOT11_OID_PSM                   0x14000000
#define DOT11_OID_CAMTIMEOUT            0x14000001
#define DOT11_OID_RECEIVEDTIMS          0x14000002
#define DOT11_OID_ROAMPREFERENCE        0x14000003

#define DOT11_OID_BRIDGELOCAL           0x15000000
#define DOT11_OID_CLIENTS               0x15000001
#define DOT11_OID_CLIENTSASSOCIATED     0x15000002
#define DOT11_OID_CLIENT1               0x15000003
```

Uso de instrucciones:

```
"mac"
Uso: setoid   mac

"long"
Uso: setoid   long

"string"
Uso: setoid   string

"ssid"
Uso: setoid   ssid

"vdcf"
Uso: setoid   vdcf

"key"
Uso: setoid   key   | 0x

"stakey"
Uso: setoid   stakey

"mlme"
Uso: setoid   mlme

"frequencies"
Uso: setoid   frequencies  []

"rssivector"
Uso: setoid   rssivector

"sta"
Uso: setoid   sta

"mt"
Uso: setoid   mt
 parameters: mode channel rate preamble length modulation scrambling filter
             antenna_rx antenna_tx power_loop key_type key_length
             key ccamode autorespond

"attachment"
Uso: setoid   attachment
```

Ejemplo : NO WEP Instrucciones:

```
setoid   ethX 12000001 long 0
setoid   ethX 12000002 long 0
```

long es el tipo de parametro que le pasamos. Práctica:

```
[root@unit0 intersil]# setoid eth2 12000001 long 0
[root@unit0 intersil]# setoid eth2 12000002 long 0
```

Listado de redes disponibles:

```
#getoid eth1 1c000043 bsslist
Nr        MAC Address                f[MHZ]     SSID
01        AA:BB:CC:DD:EE:FF          2412       ReusWireless_Necros
02        GG:HH:II:JJ:KK:LL          2457       ReusWireless_Abel
```

Caso especial de tecnologia SMC Introduce 802.11g Nitro Technology! SMC's 802.11g draft-compliant networking products offer the fastest wireless throughput possible in the 2.4GHz space. Fast enough to handle streaming video, multimedia and all other bandwidth-intensive applications, SMC's 'G' family of products provide instant, seamless high-speed network connection for wireless clients. The new PRISM Nitro technology provides up to 50% more throughput in g-only networks; up to 300% more in mixed-mode (802.11b and g) networks by eliminating collisions and employing packet bursting technology. Fully-compliant with IEEE standards, PRISM Nitro technology is backward-compatible with legacy 802.11b.

Párametro : DOT11_OID_MAXFRAMEBURST 0x1b000008

Ejemplo:setoid eth2 17000019 long 4

Recordar poner el ssid otra vez.
Valor recomendado , 1000 (pero se aconseja experimentar)
- Siempre utilizar el último firmware de la tarjeta.
- Debes conectar con otros Ap del mismo tipo 802.11g (Con la misma tecnología,claro)
- Activar como hemos indicado en la instrucción anterior, el MAXFRAMEBURST .
(No he podido probarlo por la falta de otro Ap con la misma tecnología, pero todo llega)

Mini Script básico utilizado:

```
[root@teleco root]# cat wep



modprobe islpci

ifconfig eth2 192.168.1.1 netmask 255.255.255.0 up


## -- NO WEP
## setoid eth2 12000001 long 0
## setoid eth2 12000002 long 0
##

## --  WEP --
## Cargamos las keys, en este caso 40 bits, 10 hex digits
setoid eth2 12000004 key 0 0xAAAAAAAAAA
setoid eth2 12000005 key 0 0xBBBBBBBBBB
setoid eth2 12000006 key 0 0xCCCCCCCCCC
setoid eth2 12000007 key 0 0xDDDDDDDDDD
```

```
## Autentificacion
setoid eth2 12000000 long 3

## Privacy invoked / Exunencrypted
setoid eth2 12000001 long 1
setoid eth2 12000002 long 1

## KeyId
setoid eth2 12000003 long 0

## SSID
setoid eth2 10000002 ssid  ReusWireless_NeCrOS
```

Ejecución:
#./wep

```
------- LOADING RULES ------
driver
Using /lib/modules/2.4.22/pcmcia/islpci.o
Keys
Auth
Priv / Exun
Keyd
Ssid
Red
[root@teleco root]# iwconfig
lo        no wireless extensions.

eth0      no wireless extensions.

eth1      no wireless extensions.

eth2      PRISM Duette  ESSID:"ReusWireless_NeCrOS"
          Mode:Master  Frequency:2.437GHz  Bit Rate=54Mb/s
          RTS thr=2347 B   Fragment thr=2346 B
```

- Mediante el iptables realizamos Forwarding, eso permitirá a los clientes que se conecten al nodo,acceso a Internet.
Scritp:

echo "1" > /proc/sys/net/ipv4/ip_forward #Activamos el Forwarding
route add default gw 192.168.X.X #ip de la maquina pasarela de RED
iptables --flush #borrado de rules
iptables -t nat --flush #borrado tabla NAT
iptables -t nat -A POSTROUTING -o ethx -j SNAT --to 192.168.X.X

#la ip final es la de la interficie que enruta hacia el GW "pasarela",

#es la ip con la que los paquetes de los clientes, saldrán hacia el GW
#ethx es la interficie de salida hacia el GW.
.
Resumen: Todos los paquetes que pidan una ip que no este en nuestra subred,los encaminas hacia la pasarela por defecto con ip 192.168.x.x, que esta saliendo por la interficie ethx .

Cuidado, no he impuesto ninguna rule de seguridad básica.Aconsejo mirar el artículo relacionado que trata sobre iptables...

¿ Problema con la solicitud masiva de conexiones desde un cliente conectado al AP ?.

¿ Como saberlo ? Ejemplo : Solicitud de escuchar ondacero + leer http://www.terra.es de forma

simultanea.

Resultado = volcado de pila por parte del kernel, dejando colgado completamente el sistema...

(Este ha sido mi caso, pero no significa que pueda ser el
tuyo,testealo antes de realizar el cambio,que te quedas sin soporte DWS !!!).

Solución : Debemos volver a configurar el driver, y en la opción de links DWS decirle que no

(Todavía esta en estudio del pq de todo esto, pero hasta la velocidad de transmisión
de paquetes ha mejorado)

Site de referencia actual del driver:http://www.prism54.org[7]

Espero que os sirva de ayuda este mini-howto, yo seguiré con el estudio de estos
nuevos drivers para el mundo wireliano, futuras versiones serán publicadas en bulma ;)
Siempre que me lo permiten claro, y en http://www.reuswireless.net en la zona de Downloads.

*Nota : Esta documentación es parte de un PFC que se esta elaborando para la
Universitat Rovira i Virgili (Tarragona), Ingeniería Técnica Informática de Sistemas.*

FreE WorlD <=> FreE CoDe by NeCrOS.

Lista de enlaces de este artículo:

1. http://www.NeCrOs.com
2. http://ruslug.rutgers.edu/~mcgrof/802.11g/Documentation/
3. http://www.smc-europe.com/es/products/wirel/2802W.html
4. http://www.i-legend.com/qdi/products/430vxe2.htm
5. http://ruslug.rutgers.edu/~mcgrof/802.11g/Documentation/supported_cards.php
6. http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/Tools.html#links
7. http://www.prism54.org

E-mail del autor: necrosmana _ARROBA_ hotmail.com