Interconexión IRDA con Linux

Vicente D. Fernandez, quasar@undersec.com Última revisión: 01 de Marzo de 2002

Cómo conectar nuestro Linux vía IRDA a dispositivos IPAQ, PALM, GSM...

Contents

Chapter 1

PRÓLOGO

Bueno, pues después de unos 6 meses en la era de nuestro señor sin publicar nada, ya volvemos otra vez por aquí.

Y, para continuar fiel a mi linea, vuelvo a escribir sobre un tema completamente innovador y del que poco se ha dicho.
Personalmente, encuentro un buen aliciente el poder escribir algo con importantes puntos novedosos o poco documentados en este habla que el azar ha querido que tengamos. Sin ánimos de desmerecer trabajos de traducción verdaderamente importantes y dignos de admirar, prefiero seguir en una linea de 'pegarse hostias hasta sacarlo' y conseguir que otras personas tengan una base sobre la que trabajar y ahorrar así, por lo menos, un 20% del tiempo y las leches que me he tenido que dar. Veo el tema bastante mas productivo que una mera traducción de los textos que se puede ir encontrando en otros idiomas en la red. Mas adelante ya lo recalcare pero, por supuesto, el articulo está completamente abierto y con porcentajes de posibles errores. Sin embargo, para eso está escrito: para que ahora todo el resto de personas metidas en el mismo 'proyectillo' compartamos ideas e ir ampliando toda la información que el texto puede ir aportando.

La base de escribir el articulo viene a raíz de varias cosas. Empezando por la interconexión de Palm vía "cradle" (ya veréis cómo con IRDA también se puede), pasando por unas pruebas de wavelan que nada tienen que ver con esto y llegando incluso a estar motivado por cierta escena de no recuerdo qué película donde hackean un portátil a través del infrarrojo, que cosas XD.

En esta revisión, hay que recalcar que algunas de las pruebas ya tienen algunos meses. Espero que aún sean útiles con nuevas versiones de SOs como Pocketpc y PalmOS.

Existen versiones para Microsoft Reader (.LIT) en la pagina de Undersec pero que no sufren un mantenimiento tan seguido de esta documentación.

> Lo de elegir M$ Reader es porque viene en la ROM del PDA cuando lo compras

No me enrollo mas, os dejo pues con todas las pruebas que he ido realizando a los largo de dos semanas mas que largas sobre: Yo, mi Linux y el IRDA XDDDD. Sobre todo espero que os sirva.... es lo importante.

Chapter 2

INTRODUCCIÓN AL IRDA

No me voy a introducir en la parte mas técnica del stack IRDA. Me decanto directamente por la parte mas practica.

Si que es verdad, sin embargo, que para una mejor comprensión es necesaria una breve introducción. Empecemos pues a medio traducir de varios documentos. En el año 93, 50 empresas se juntaron en lo que se llamó la Infrared Data Associaton (IRDA) para definir los estándares para la transmisión de datos usando infrarrojos de corto alcance. A partir de ahí, cientos de empresas se unieron a la susodicha asociación. Desde entonces una mejora tras otra, pasando de la versión 1.0 a la 1.1 etc etc.

Como todos los modelos, el del IRDA también se basa en capas. De esta forma se reparte de la siguiente manera:

```
Tiny TP - IAS

IrLMP

Frame/Driver

Physical Layer
```

2.1 Physical Layer

La Physical Layer o nivel físico aporta todas las características físicas. Las transmisiones se realizan en difusión con un cono de 30 grados desde el punto intermedio, eso es según las especificaciones, porque yo he probado desde unos 65 grados y sigue funcionando.

El tipo de conexión requiere un nivel intermedio en la transmisión de la luz. Es como todo, mucha luz cegaría el receptor, mientras que poca no es suficiente para consolidar la transmisión. Es curioso, porque aquí no hay un modelo concreto, o eso creo. Se pueden apreciar en el mercado dispositivos que alcanzan los 5 metros, mientras que otros no alcanzan ni uno, véase móviles y algunas pda's. Una buena manera de medir esto, si posees una palm, es con el IRMonitor (*http://www.palmgear.com* <http://www.palmgear.com>). Es capaz de mostrar al instante en un S-Meter (signal meter, medidor de señal) la potencia de la señal IR recibida.

La conexión IR es semidúplex, mas que nada porque no tiene sentido recibir mientras se esta transmitiendo. La razón es sencilla, si sueltas el haz de luz el receptor queda cegado por este y no es capaz de ver el que le llega al mismo tiempo que tu transmites.

Encontramos además tres tipos, por decirlo así, de transmisión:

SIR: Comprende velocidades iguales a las de un puerto serie (max. 115200). De hecho es el tipo de conexión establecida para dispositivos conectados a un puerto serie Serial InfraRed.

MIR: Parece ser que está en desuso. Medium Infrared. Está orientado a transmisión entre 0.5Mbps y 1.152Mbps.

FIR: Esto es propio de dispositivos integrados. Lógicamente, al no estar conectado al serie, con la consiguiente limitación de la velocidad, y estar mejorado en algunos puntos, se permite velocidades de transmisión de hasta 4Mbps.

```
Existe un cuarto tipo, el VFIR, que pretende alcanzar velocidades de hasta 16Mbps.
Ni idea de cómo andará este proyecto.
```

2.2 Frame/Driver

En principio tiene dos funciones, lo que ocurre es que tienen tantas cosas en común, o eso dicen XD, que se engloba en una sola. Así que en ocasiones solo se hace referencia a este nivel como Frame.

La parte Driver inicializa lo que es el hardware, las velocidades de transmisión, e intercambia datos desde el controlador hasta el transceptor.

La parte Frame digamos que convierte el formato de datos a un formato que el hardware entienda. Esto podría incluir la comprobación de CRC, bits de inicio/parada y de transparencia.

2.3 IrLAP

Hasta aquí todo trabaja a un nivel relativamente bajo. Empezamos pues a subir un nivel mas. El IrLAP se encarga de preservar la comunicación entre puertos IR. Así que es aquí donde se detectan los errores de transmisión y por tanto se encarga también de la retransmisión de los paquetes perdidos. Tiene algo de control de flujo, pero es demasiado poco evidente. Así que lo dejaremos para un nivel mas.

Está basado en HDLC, que en la familia OSI es un protocolo de enlace de datos de alto nivel. Para mas info, buscarlo en algún buscador, que seguro que aparece.
El caso es que nuestro IrLAP está basado en HDLC, pero se mejora el aspecto de los reenlaces (reconexiones mas o menos), ya que en IR es muy fácil perder conexión y volver a enlazar.

El IrLAP supone, por tanto, una buena base para construir diferentes protocolos sobre él. Nuestro querido linux ya nos avisa de su existencia:

```
irda0     Link encap:IrLAP  HWaddr 21:18:d8:ab
          UP RUNNING NOARP  MTU:2048  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:8
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
```

2.4 IrLMP

El IrDA Link Management Protocol (IrLMP) nos permite tener uno o mas servicios corriendo en una única conexión sobre IrLAP.
Con la ayuda de IAS las aplicaciones pueden acceder directamente a este nivel para enviar sus datos.
Un ejemplo de cual dispositivo podría utilizar una conexión en este nivel es, por ejemplo, una impresora conectada por puerto IR.

2.5 IAS

Justo he nombrado este nivel en el punto anterior. Y es porque el IAS requiere de IrLMP para todo. En este nivel es donde se buscan y encuentran los diferentes dispositivos IR. Por supuesto, me muestro escueto porque no creo que sirva de algo detenerme mas.

2.6 Tiny TP

Aquí tenemos lo que seria el protocolillo a nivel de transporte y por lo tanto aquí tendríamos todo lo que englobaría el control de flujo. Es aquí pues, donde se segmentan ¿fragmentan? y reensamblan los paquetes.

2.7 Otros

Luego tenemos lo que en no se que texto de referencia llaman "Others Layers". Un punto interesante este. Es precisamente done vamos a centrar, como veréis después, todas nuestras prácticas.

Se trata de protocolos del nivel mas alto. En este punto encontramos:

2.7.1 IrOBEX: IrDA Object Exchange

Es un protocolo diseñado para que un objeto pueda ser movido de un dispositivo a otro como tal. Me explico, no se requiere un tipo de comunicación estable como podría ser, por ejemplo, la conexión de una Palm y un PC para sincronizarse. Esta mas en la linea de un "cut and paste". ¿Quien no ha visto nunca dos móviles pasándose vcards (tarjeta de visita, que es lo que pone en la pantallita) del uno al otro?

Nos detendremos mas adelante para algún caso practico con IrOBEX (Apéndice C).

2.7.2 IrCOMM

IrCOMM fué diseñado para dar soporte a aquellas aplicaciones que ya funcionaban sobre el puerto COM. Por ejemplo, es posible emular la conexión con una Palm con el crandle como si este estuviera en el serie, cuando no lo está. IR rlz!. Lo bueno está en que no se requiere ninguna modificación del software para realizar esta operación.

2.7.3 IrLPT

Si, supongo que ya lo estáis oliendo. Efectivamente su uso está destinado a la conexión con dispositivos conectados al IR, como por ejemplo impresoras.

Chapter 3

PREPARANDO NUESTRO LINUX

Uno de los principales problemas que nos encontramos es lo que voy a llamar 'nodocumentación'. En ultima estancia, es una consecuencia del paso del tiempo y entiendo que lo provoca, entre muchas cosas, la documentación obsoleta, los textos a medias, el desconocimiento mayoritario y la escasa documentación de los propios desarrolladores. Por supuesto, ni que decir que en castellano prácticamente nada de nada. Menos mal que hay buenos amigos para realizar todas las pruebas.

Las pruebas han estado realizadas sobre Debian Woody con núcleo 2.4.8 sobre un portátil clónico pIII 1000 con un dispositivo FIR integrado. Por lo tanto, es muy muy probable que surjan todo tipo de problemas con IR sobre serie con SIR. Toda esa modalidad no la he probado, aun así espero que el documento pueda servir de base de la que partir para las diferentes pruebas. Por supuesto, si conseguís hacer funcionar este modo y la manera de proceder es diferente a lo que aquí va a aparecer, estaría muy bien que facilitarais las pruebas para completar el documento.

3.1 Configuración del Núcleo

Bueno, lo primero es preparar el nícleo. He decidido no poner la opción básica, sino mas bien la completa, por si el dispositivo que usáis utiliza un driver que no es el estándar FIR o SIR. He dejado fuera también los controladores para Toshiba y para IR sobre USB. No me acuerdo ahora muy bien por qué lo hice :(.
La configuración quedaría pues:

```
 #
CONFIG_PACKET=y                 # Para poder usar irdadump
#
# IrDA (infrared) support
#
CONFIG_IRDA=m
CONFIG_IRLAN=m
CONFIG_IRNET=m
CONFIG_IRCOMM=m
CONFIG_IRDA_ULTRA=y
CONFIG_IRDA_OPTIONS=y
CONFIG_IRDA_CACHE_LAST_LSAP=y
```

```
CONFIG_IRDA_FAST_RR=y
CONFIG_IRDA_DEBUG=y

#
# Infrared-port device drivers
#
CONFIG_IRTTY_SIR=m
CONFIG_IRPORT_SIR=m
CONFIG_DONGLE=y
CONFIG_ESI_DONGLE=m
CONFIG_ACTISYS_DONGLE=m
CONFIG_TEKRAM_DONGLE=m
CONFIG_GIRBIL_DONGLE=m
CONFIG_LITELINK_DONGLE=m
CONFIG_OLD_BELKIN_DONGLE=m
# CONFIG_USB_IRDA is not set
     CONFIG_NSC_FIR=m
# CONFIG_WINBOND_FIR is not set
# CONFIG_TOSHIBA_FIR is not set
CONFIG_SMC_IRCC_FIR=m
# CONFIG_ALI_FIR is not set
```

Encontraremos en /dev/:

```
crw-r-----    1 root     root     161,   0 Nov  4 18:03 /dev/ircomm0
crw-rw----    1 root     root     161,   1 Oct  5 11:20 /dev/ircomm1
crw-rw----    1 root     root     161,  16 Oct  5 11:20 /dev/irlpt0
crw-rw----    1 root     root     161,  17 Oct  5 11:20 /dev/irlpt1
```

En caso de que no estén creados:

```
mknod /dev/ircomm0 c 161 0
mknod /dev/ircomm1 c 161 1
mknod /dev/irlpt0 c 161 16
mknod /dev/irlpt1 c 161 17
mknod /dev/irnet c 10 187
chmod 666 /dev/ir*
```

En los antiguos núcleos olvidaros del ircomm0, que he leído en diferentes listas que no venían con esos nombres.

Los módulos los tenemos situados en dos directorios, uno corresponde a lo que sería la base con los protocolos, mientras que el otro directorio son los controladores específicos.

En /lib/modules/2.4.8/kernel/net/irda encontramos el paquete básico:

```
drwxr-xr-x    2 root     root         4096 Oct 18 00:28 ircomm
-rw-r--r--    1 root     root       228921 Oct 18 00:28 irda.o
```

```
drwxr-xr-x    2 root     root         4096 Oct 18 00:28 irlan
-rw-r--r--    1 root     root        57134 Oct 18 00:28 irlan.o
drwxr-xr-x    2 root     root         4096 Oct 18 00:28 irnet
```

Expandimos todo:

```
-rw-r--r--    1 root     root       228921 Oct 18 00:28 irda.o
-rw-r--r--    1 root     root        57134 Oct 18 00:28 irlan.o

ircomm:
total 76
drwxr-xr-x    2 root     root         4096 Oct 18 00:28 .
drwxr-xr-x    5 root     root         4096 Oct 18 00:28 ..
-rw-r--r--    1 root     root        43645 Oct 18 00:28 ircomm-tty.o
-rw-r--r--    1 root     root        21539 Oct 18 00:28 ircomm.o

irlan:
total 68
drwxr-xr-x    2 root     root         4096 Oct 18 00:28 .
drwxr-xr-x    5 root     root         4096 Oct 18 00:28 ..
-rw-r--r--    1 root     root        57134 Oct 18 00:28 irlan.o
irnet:
total 44
drwxr-xr-x    2 root     root         4096 Oct 18 00:28 .
drwxr-xr-x    5 root     root         4096 Oct 18 00:28 ..
-rw-r--r--    1 root     root        35539 Oct 18 00:28 irnet.o
```

`irnet.o es para poder cargar una pila TCP/IP y así poder conectar dos Linux.`

Mientras, por otro lado, en /lib/modules/2.4.8/kernel/drivers/net/irda tenemos:

```
-rw-r--r--    1 root     root         3768 Oct 18 00:27 actisys.o
-rw-r--r--    1 root     root         2612 Oct 18 00:27 esi.o
-rw-r--r--    1 root     root         3456 Oct 18 00:27 girbil.o
-rw-r--r--    1 root     root        14164 Oct 18 00:27 irport.o
-rw-r--r--    1 root     root        13484 Oct 18 00:27 irtty.o
-rw-r--r--    1 root     root         2980 Oct 18 00:27 litelink.o
-rw-r--r--    1 root     root        21820 Oct 18 00:27 nsc-ircc.o
-rw-r--r--    1 root     root         2648 Oct 18 00:27 old_belkin.o
-rw-r--r--    1 root     root        12160 Oct 18 00:27 smc-ircc.o
-rw-r--r--    1 root     root         4596 Oct 18 00:27 tekram.o
```

3.2 Instalando el software

Ahora hacen falta los paquetes necesarios. Necesitamos instalar unos paquetes[1] básicos imprescindibles:

[1] En Debian se denominan así

irda-common - IrDA management utilities

irda-tools - IrDA handling tools

El primero (irda-common) nos permite instalar y configurar lo que es el stack irda, encontraremos pues en el paquete:

/usr/sbin/irattach

/usr/sbin/findchip

y los respectivos ficheros en el /etc: [2]

/etc/modutils

/etc/modutils/irda

/etc/init.d

/etc/init.d/irda

/etc/irda.conf *(-DEBIAN: este es generado después del config)*

En el segundo (irda-tools) se instalan todas las herramientas:

/usr/bin/irdadump

/usr/bin/irdaping

/usr/bin/irpsion5

CONFIG_PACKET=y (activado) en el núcleo o no funcionarán.

Una vez instalados los paquetes hay que empezar a retocar por todos los sitios. Esto me llevó bastante. Vamos a ver..

3.3 Configurando el sistema

Lo primero es llamar la atención sobre el /etc/irda.conf. No sé si en otras distribuciones o bien para otros controladores pasará igual, pero sobre NSC/FIR en Debian, ese archivo es gilipollo. No sé para qué sirve. De hecho, os pego el fichero para que veáis que mis últimas pruebas eran ya descaradas y absolutamente todo, después de realizar las operaciones que os indico mas adelante, funcionaba perfectamente.

```
annapurna:~# cat /etc/irda.conf
#irda.conf Version: 1.0
IRDADEV=/deV/mierda
DONGLE=none
DISCOVERY=-s
ENABLE=no # if you don't need to start irattach, set "no"
annapurna:~#
```

[2] ¡ojo!, sobre Debian solo. En otras distribuciones no se si se llaman de esa forma

Después de esto, el irda ha seguido funcionando ?¿?¿?

En el modules.conf (o conf.modules, según distribuciones) hay que añadir, lo pongo todo y lo que no os haga falta lo quitáis:

```
alias tty-ldisc-11 irtty
alias char-major-161 ircomm-tty
alias char-major-60 ircomm_tty
alias char-major-61 lirc_sir

alias irda-dongle-0 tekram
alias irda-dongle-1 esi
alias irda-dongle-2 actisys
alias irda-dongle-3 actisys
alias irda-dongle-4 girbil
alias irda-dongle-5 litelink
alias irda-dongle-6 airport
alias irda-dongle-7 old_belkin
# Esto por si usais un dispositivo serie y hay que meterle la configuración
# options smc-ircc ircc_irq= ircc_dma=
# Esto es para FIR.
# options nsc-ircc dongle_id=0x09

alias irda0 nsc-ircc

# Esto creo que es para los Toshiba
# options toshoboe max_baud=
# alias irda0 toshoboe
# options w83977af_ir io= io2= irq= qos_mtt_bits=
# alias irda0 w83977af_ir

#IRNET Module
alias char-major-10-187 irnet
```

> *DEBIAN: En Debian (Woody?) tendréis que modificarlo en el /etc/modutils y añadir un fichero "irda" (por ejemplo) que contenga estas lineas. Recomendado: man update-modules.*

Seguimos. Lo importante ahora es determinar si el núcleo sabe qué dispositivo IR tenemos. Una de las herramientas instaladas es "findchip", que nos viene como anillo al dedo:

```
annapurna:~# findchip -v
Found NSC PC87338 Controller at 0x398, DevID=0x0b, Rev. 2
SIR Base 0x2f8, FIR Base 0x2f8
IRQ = 3, DMA = 0
Enabled: yes, Suspended: no
UART compatible: yes
Half duplex delay = 0 us
annapurna:~#
```

¡Ole!.. ahí está, tengo un "NSC". Pero antes de instalar nada viene el quebradero de cabeza por excelencia.
Esto es **IMPORTANTE:**

Como vemos, el NSC nos lo detecta en:

FIR Base 0x2f8 IRQ = 3, DMA = 0

Si caemos en la cuenta, ejecutamos 'setserial /dev/ttyS1' y:

/dev/ttyS1, UART: 16550?, Port: 0x02f8, IRQ: 3

¡Coincide!. Pues bien:

¡¡¡¡NO ES POSIBLE TENER EL SERIE Y EL NSCIR INTEGRADO UTILIZANDO LO MISMO!!!!

La jugada es cambiar el uart a "none" para evitar conflictos y que funcione. En algunas listas incluso recomiendan poner todo a "none". No sé, la verdad, con el uart a mí me bastó. La forma es la siguiente:

setserial /dev/ttyS1 uart none `(en nuestro caso)`

DEBIAN: Para tenerlo definitivo, lo que hacemos es que en el /etc/serial.conf comentamos la línea original y modificamos por la nuestra:

```
#/dev/ttyS1 uart 16550A port 0x02f8 irq 3 baud_base 115200 spd_normal skip_test
/dev/ttyS1 uart none port 0x02f8 irq 3 baud_base 115200
```

Una vez modificado el modules.conf y el serial.conf, entonces nos preparamos para la instalación de los módulos. Lo mas rápido es hacer lo siguiente:

La utilidad irattach (man irattach) permitía asignar la stack IR a un puerto serie. En dispositivos conectados al serie debería ser algo como (no lo he probado):

irattach /dev/ttyS1 -s 1 `Que se supone carga la pila IR sobre SIR (serial).`

Sin embargo, para nuestro NSC de tipo FIR vamos a probar con:

irattach irda0 -s 1 `(jejejejejeje!)`

NOTA: En el howto (Ingles) dice que no hace falta usar irattach para dispositivos FIR. Hacerme caso y usar irattach si podéis.

Si el truco del almendruco funciona a la primera, tenemos en /var/log/syslog:

```
annapurna irattach: executing: /sbin/modprobe irda0
annapurna kernel: irda_init()
annapurna kernel: irlmp_init()
annapurna kernel: nsc-ircc, Found chip at base=0x398
```

```
annapurna kernel: nsc-ircc, driver loaded (Dag Brattli)
annapurna kernel: IrDA: Registered device irda0
annapurna kernel: nsc-ircc, Found dongle: Sharp RY5HD01
annapurna irattach: executing: 'echo 1 > /proc/sys/net/irda/discovery'
annapurna irattach: Starting device irda0
annapurna irattach: executing: 'echo annapurna > /proc/sys/net/irda/devname'
annapurna kernel: irlmp_register_client_R17f18bfb()
annapurna kernel: irlap_change_speed(), setting speed to 9600
```

---

y desde donde se ejecutó el irattach nos aparece:

```
annapurna:~# irattach irda0 -s 1
1.1 Tue Nov  9 15:30:55 1999 Dag Brattli
annapurna:/etc# nsc-ircc, Found chip at base=0x398
nsc-ircc, driver loaded (Dag Brattli)
IrDA: Registered device irda0
nsc-ircc, Found dongle: Sharp RY5HD01
annapurna:~#
```

Esta línea: *annapurna kernel: irlap_change_speed(), setting speed to 9600*

parece que vaya a darnos problemas. Sin embargo los programas de la Palm reajustan la velocidad. No queda registrado en ningún lado, pero he probado a sincronizar a 9600 y luego he cambiado el pilotrate a 57600 y la diferencia es considerable. Y en segundo lugar, la transmisión a móviles parece mucho mas firme y efectiva si no modifico nada que si toco de aquí y de allá.

Comentar que en el /proc, justo donde nos dice el syslog, tenemos diferentes parámetros que nos dan información o se pueden variar.

```
-rw-r--r--    1 root     root            0 Nov  6 01:45 debug
-rw-r--r--    1 root     root            0 Nov  6 01:45 devname
-rw-r--r--    1 root     root            0 Nov  6 01:45 discovery
-rw-r--r--    1 root     root            0 Nov  6 01:45 discovery_slots
-rw-r--r--    1 root     root            0 Nov  6 01:45 discovery_timeout
-rw-r--r--    1 root     root            0 Nov  6 01:45 fast_poll_increase
-rw-r--r--    1 root     root            0 Nov  6 01:45 max_baud_rate
-rw-r--r--    1 root     root            0 Nov  6 01:45 max_inactive_time
-rw-r--r--    1 root     root            0 Nov  6 01:45 slot_timeout
```

En el caso de que no vaya el irattach a la primera, puede ser por un millón y medio de motivos XD. Un punto desde donde empezar a probar combinaciones es justo los parámetros de configuración en el modules.conf. De esta forma, ejecutamos el findchip, vemos los parámetros y rellenamos las líneas:

para SMC (SIR):

options smc-ircc ircc_irq= ircc_dma=

para NSC (FIR:)

options nsc-ircc dongle_id=0x09

La otra opcion es no tirar de irattach hasta que no consolidemos cada uno de los lkms que se deberían cargar. Una posible respuesta la encontramos probando los módulos uno por uno con el orden adecuado.

El primero a cargar es el *irda.o*. Eso seguro.

El segundo es el controlador para el dispositivo. Normalmente los módulos admiten opciones que no tenemos porque ir adivinando. Lógicamente, las del smc.ircc y el nsc-ircc son intuitivas porque os he pegado las opciones que incorporan en el modules.conf de arriba.

En caso de que tengáis que tirar de otro controlador/módulo/lkm (todo es lo mismo) os recomiendo el comando 'modinfo', con el cual sabréis exactamente por donde podréis jugar con el modulo. Si aplicamos lo dicho tenemos:

```
annapurna:~# modinfo nsc-ircc
filename:    /lib/modules/2.4.8/kernel/drivers/net/irda/nsc-ircc.o
description: "NSC IrDA Device Driver"
author:      "Dag Brattli <dagb@cs.uit.no>
license:     <none>
parm:        qos_mtt_bits int, description "Minimum Turn Time"
parm:        io int array (min = 1, max = 4), description "Base I/O addresses"
parm:        irq int array (min = 1, max = 4), description "IRQ lines"
parm:        dma int array (min = 1, max = 4), description "DMA channels"
parm:        dongle_id int, description "Type-id of used dongle"

annapurna:~# modinfo smc-ircc
filename:    /lib/modules/2.4.8/kernel/drivers/net/irda/smc-ircc.o
description: "SMC IrCC controller driver"
author:      "Thomas Davis <tadavis@jps.net>
license:     <none>
parm:        ircc_dma int, description "DMA channel"
parm:        ircc_irq int, description "IRQ line"
```

En el nsc-ircc el tema del dongle no sé muy bien de donde lo sacaba. Se supone que *Found dongle: Sharp RY5HD01* tiene un id que es un decimal entero, pero no sé donde se mira.

En el caso de haberlo hecho funcionar, enhorabuena. Ya tienes el IrLAP instalado sobre el que cargar el resto de protocolos de nivel superior, importantísimos de cara a comunicar in dispositivo IR con otro.

A modo de confirmación, una vez hayáis configurado todo (aquellos que hayáis tenido que elegir el camino de 'ir a mano' acordaros de un *"ifconfig irda0 up"*) deberíais ver lo siguiente:

```
annapurna:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:40:D0:1A:F0:41
          inet addr:192.168.2.5  Bcast:192.168.2.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:3262 errors:0 dropped:0 overruns:0 frame:0
          TX packets:868 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
```

```
          RX bytes:822625 (803.3 Kb)  TX bytes:66494 (64.9 Kb)
          Interrupt:11 Base address:0x1800

irda0     Link encap:IrLAP  HWaddr 10:1e:45:b7
          UP RUNNING NOARP  MTU:2048  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:4151 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:8
          RX bytes:0 (0.0 b)  TX bytes:131053 (127.9 Kb)


lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:425 errors:0 dropped:0 overruns:0 frame:0
          TX packets:425 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:80135 (78.2 Kb)  TX bytes:80135 (78.2 Kb)
```

¡Ahí esta nuestro amigo!. Nuestro Linux esta casi listo.

Es una pena que de los mil caminos (configuraciones) que hay hasta llegar aquí, solo haya podido probar uno. Para el resto de configuraciones es necesario un trabajo que debéis de realizar. Por supuesto, si conseguís algo me lo mandáis documentado para añadirlo al documento. Mi intención es siempre aportar el máximo, pero para este pobre administrador de turno no le es factible adquirir mas de lo que ahora mismo posee. Por no tener no tengo ni móvil con IR.

Para terminar con lo que sería la "parte común a todos los dispositivos", vamos a preparar el sistema para interconectarse a los diferentes periféricos. Para ello a todo lo anterior hay que añadir:

modprobe ircomm-tty (que nos cargará además el ircomm.o)
modprobe irport

En definitiva, tenemos cargado:

```
nsc-ircc               14036   1
irport                  7240   0  (unused)
ircomm-tty             31680   0
ircomm                 14396   0  [ircomm-tty]
irda                  150848   1  [nsc-ircc irport ircomm-tty ircomm]
```

El ircomm nos carga un protocolo de nivel superior sobre el IrLAP, que se utiliza para la comunicación de dispositivos y emular el puerto COM. Me remito a mi descripción del protocolo 2.7.2 (en el capitulo anterior)

El ircomm-tty es una versión mas actualizada (mas o menos) del irtty. La diferencia parece estar en que uno está implementado sobre IrCOMM y el otro es mas genérico. De hecho, si nos fijamos en la info proporcionada por el módulo, apenas encontramos diferencias:

```
   annapurna:~# modinfo ircomm-tty |grep description:
description: "IrCOMM serial TTY driver"
annapurna:~# modinfo irtty |grep description:
description: "IrDA TTY device driver"
```

Sobre el irport..mmm:

```
   annapurna:~# modinfo irport |grep description:
description: "Half duplex serial driver for IrDA SIR mode"
```

Bien, ya tenemos todo preparado.

Chapter 4

INTERCONEXIÓN CON PALM

Bueno, ya nos vamos centrando en cada una de las distintas posibilidades que nos ofrece el IR. Queridos usuarios de Palm, welcome a la octava maravilla.

En un anterior documento escrito por este pobre pringao "Interconexion de Palm con Linux/FreeBSD" [1] vimos que era posible sincronizar la Palm con el crandle (como ya lo hacia win) y también cómo establecer una conexión ppp con la Palm para que esta pudiera conectar directamente con Internet a través de nuestro Linux o FreeBSD (esto para win, ejem, programita de turno, véase Mochappp). Bien, ahora vamos a dar un paso mas y vamos a realizar exactamente lo mismo pero sin usar el crandle. Todo gracias a nuestro querido IRDA. ¿Coste? cero euros. XD.

Empezamos.

Lo primero, asegurarnos de que tenemos todo lo necesario.

1. Hemos conseguido hacer ir el IR/IRDA.
2. Usamos un Palm OS 3.3 o superior.

4.1 Sincronización de Palm con Linux vía IR

Vamos a buscar el paquete pilot-link (0.9.5 creo que es el que uso). La instalación del paquete nos va a facilitar una barbaridad de utilidades mas que interesantes. No me voy a detener comentando cada una porque no es nuestro caso. La principal para sincronizar es el pilot-xfer.

Ahora procedemos a preparar el sistema. Por supuesto, esto solo es laborioso la primera vez. Una vez realizados todos los cambios, tanto ahora como anteriormente, se pueden dejar definitivos y ya no tener que realizar cada vez la misma operación. Ver los rc.local etc etc de cada distribución.

Creamos un enlace simbólico desde /dev/pilot a nuestro dispositivo de IR. O sea, el /dev/ircomm0.

```
annapurna:~# ls -al /dev/pilot
lrwxrwxrwx    1 root     root  12 Oct 16 16:35 /dev/pilot -> /dev/ircomm0
```

Recalcar que el /dev/ircomm0 va a ser nuestro dispositivo de IR en todo momento.
Es recomendable añadir la variable del sistema PILOTRATE que nos permite aumentar la velocidad de sincronización de 9600 hasta los 115200:

[1] También se encuentra en http://www.undersec.com

export PILOTRATE=115200

Es solo útil de cara a las pilot-link, no sirve de nada para el resto de herramientas (gsm-utils etc).

Creamos un directorio, o cogemos el directorio donde tenemos los prc para Palm, bueno, o lo que sea y le damos al pilot-xfer.

Por ejemplo:

```
   annapurna:~# pilot-xfer -s
Port: /dev/pilot
Please press the HotSync button now...
```

Ahora nos vamos a la Palm, la enfocamos hacia el IR. Buscamos en la Palm el HostSync. Elegimos "Local" y cambiamos lo de "Serie Directa" a "IR a PC/Portatil". Pulsamos sobre el icono de las flechitas (el de siempre) y ale...a funcionar:

```
annapurna:~/palm/test# pilot-xfer -s .
Port: /dev/pilot
Please press the HotSync button now...
Connected...
Backing up './AvGoPref.pdb'... OK
Backing up './AvGoVersion.pdb'... OK
Backing up './TSheetCat-IAMBIC.pdb'... OK
Backing up './AvGoChanMQ.pdb'... OK
Backing up './BlockParty Prefs.pdb'... OK
[...]
```

Es curioso comprobar toda la negociación a nivel de IrLAP.

Al principio, cuando acercamos la Palm al IR del portátil, es posible que la Palm nos anuncie un mensaje por pantalla que ponga: *"Esperando al remitente..."* El mensaje es consecuencia de los mensajes de difusión del IR del portátil para ver si detecta el dispositivo. La palm los detecta y contesta, pero el IR del portátil sigue con sus difusiones. Existe en el A (APENDICE A) una explicación detallada sobre el trafico de IR. En el siguiente ejemplo no me paro pues a explicar los detalles del trafico IR. El Ejemplo:

```
annapurna:~# irdadump -x
IR del PC:
08:19:46.801086 xid:cmd d8a82a1b > ffffffff S=6 s=1 (14)
ff3f011b2aa8d8ffffffff010100
         . ? . . * . . . . . . . . .
08:19:46.891093 xid:cmd d8a82a1b > ffffffff S=6 s=2 (14)
ff3f011b2aa8d8ffffffff010200
         . ? . . * . . . . . . . . .
08:19:46.981084 xid:cmd d8a82a1b > ffffffff S=6 s=3 (14)
ff3f011b2aa8d8ffffffff010300
         . ? . . * . . . . . . . . .
08:19:47.071089 xid:cmd d8a82a1b > ffffffff S=6 s=4 (14)
ff3f011b2aa8d8ffffffff010400
         . ? . . * . . . . . . . . .
08:19:47.161086 xid:cmd d8a82a1b > ffffffff S=6 s=5 (14)
```

```
ff3f011b2aa8d8ffffffff010500
         . ? . . * . . . . . . . . . .
08:19:47.251084 xid:cmd d8a82a1b > ffffffff S=6 s=* annapurna hint=0400 [
Computer ] (25)
ff3f011b2aa8d8ffffffff01ff000400616e6e617075726e61
. ? . . * . . . . . . . . . . . a n n a p u r n a
```

Lo recibe la Palm y contesta y en el paquete nos anuncia el protocolo que entiende:

```
08:19:48.147739 xid:rsp d8a82a1b < 7e56dd54 S=6 s=4 quasar hint=8220 [
PDA/Palmtop IrOBEX ] (23)
febf0154dd567e1b2aa8d80104008220007175617361726172
. . . T . V ~ . * . . . . . .   . q u a s a r
```

El *"q u a s a r"* que se lee en el cuerpo es el nombre que se le puso a la Palm. Como veis, la Palm ya nos anuncia que trabaja con IrOBEX (vease explicaciones dadas en el apartado anterior sobre IrOBEX). Lo que ocurre es que sobre el IR del PC no hay ninguna implementacion que interprete IrOBEX. Es un paquete externo que no he instalado aún. Por lo que no hace ni caso y sigue enviando avisos.

Otra cosa curiosa es ver la direccion de respuesta de la Palm *(7e56dd54)* Vamos a ver si es real....

```
annapurna:~/palm/test# irdaping 0x7e56dd54
IrDA ping (0xfa300b5a): 32 bytes
32 bytes from 0x7e56dd54: irda_seq=0 time=103.98 ms.
32 bytes from 0x7e56dd54: irda_seq=1 time=103.94 ms.
32 bytes from 0x7e56dd54: irda_seq=2 time=103.89 ms.
32 bytes from 0x7e56dd54: irda_seq=3 time=104.06 ms.

4 packets received by filter
```

El tema de irdaping no es mas que un paquete test. Vease A (APENDICE A). El tema está en que he descubierto que las direcciones varían cada vez. No se ahora mismo muy bien por qué. Si existe algun problema o quereis descubrir cuantos dispositivos están a la escucha basta con:

```
annapurna:~/palm/test# irdaping 0xffffffff
IrDA ping (0xffffffff): 32 bytes
32 bytes from 0xfa300b5a: irda_seq=0 time=107.92 ms.
32 bytes from 0xfa300b5a: irda_seq=1 time=107.85 ms.
32 bytes from 0xfa300b5a: irda_seq=2 time=107.83 ms.
32 bytes from 0xfa300b5a: irda_seq=3 time=107.79 ms.

4 packets received by filter
```

Veis? ha variado. El caso es que esa direccion sigue siendo la de la Palm. Aunque me da en la vena que igual es la del IR de nuestro potatil, no lo se.

Bueno, ahora ponemos el programa en marcha. Atención a la nueva solicitud de difusión:

NOTA: Voy a contar toda la negociacion como un cuentecito. Algo mas detallado lo vais a encontrar en el A (APENDICE A) y el B (APENDICE B)

```
08:19:52.711102 xid:cmd d8a82a1b > ffffffff S=6 s=0 (14)
ff3f011b2aa8d8ffffffff010000
         . ? . . * . . . . . . . . .
08:19:52.801087 xid:cmd d8a82a1b > ffffffff S=6 s=1 (14)
ff3f011b2aa8d8ffffffff010100
         . ? . . * . . . . . . . . .
08:19:52.891089 xid:cmd d8a82a1b > ffffffff S=6 s=2 (14)
ff3f011b2aa8d8ffffffff010200
         . ? . . * . . . . . . . . .
08:19:52.981084 xid:cmd d8a82a1b > ffffffff S=6 s=3 (14)
ff3f011b2aa8d8ffffffff010300
         . ? . . * . . . . . . . . .

08:19:53.071088 xid:cmd d8a82a1b > ffffffff S=6 s=4 (14)
ff3f011b2aa8d8ffffffff010400
         . ? . . * . . . . . . . . .
08:19:53.161085 xid:cmd d8a82a1b > ffffffff S=6 s=5 (14)
ff3f011b2aa8d8ffffffff010500
         . ? . . * . . . . . . . . .
08:19:53.251089 xid:cmd d8a82a1b > ffffffff S=6 s=* annapurna hint=8404 [
Computer IrCOMM ] (26)
ff3f011b2aa8d8ffffffff01ff00840400616e6e617075726e61
. ? . . * . . . . . . . . . . . . a n n a p u r n a
```

Ahora anunciamos nosotros el protocolo de negociación, algo en plan:

"...Para los que estáis por ahí escuchando: uso IrCOMM!.."

Lógicamente la Palm, que es la leche de lista, nos contesta:

```
08:19:54.057961 xid:rsp d8a82a1b < dd0fcf21 S=6 s=3 IrCOMM hint=8204 [
PDA/Palmtop IrCOMM ] (23)
febf0121cf0fdd1b2aa8d80103008204004972434f4d4d
. . . ! . . . . * . . . . . . . . I r C O M M
```

Ole....nuestro IR envía petición de conexión que es contestada en un plis por la Palm:

```
08:19:56.266869 snrm:cmd ca=fe pf=1 d8a82a1b > dd0fcf21 new-ca=e8 (33)
ff931b2aa8d821cf0fdde80102fe0182010183013f84017f8501ff8601070801
         . . . * . . ! . . . . . . . . . . . . . ? . . . . . . . . . . . .
08:19:56.402451 ua:rsp ca=e8 pf=1 d8a82a1b < dd0fcf21 (31)
e87321cf0fdd1b2aa8d801013e82010183010f840101850108860107080107 . s ! . . . . * .
. . . > . . . . . . . . . . . . . . . . . . . .
```

Ahora para confirmar conexión y por control de estado, etc..., se envía:

```
08:19:56.402519 rr:cmd > ca=e8 pf=1 nr=0 (2)
e911
         . .
08:19:56.409062 rr:rsp < ca=e8 pf=1 nr=0 (2)
e811
```

Estos paquetes los veremos multitud de veces, los utiliza para ver si aún está vigente la conexión etc etc....

Como veis, las respuestas de la Palm son inmediatas. Una vez confirmada la conexión y que todo va bien, comienza la negociación:

```
08:19:56.409080 i:cmd  > ca=e8 pf=1 nr=0 ns=0 LM slsap=12 dlsap=00 CONN_CMD (6)
e91080120100
         . . . . . .
08:19:56.416553 i:rsp  < ca=e8 pf=1 nr=1 ns=0 LM slsap=00 dlsap=12 CONN_RSP (6)
e83092008100
. 0 . . . .
8:19:56.416575 i:cmd  > ca=e8 pf=1 nr=1 ns=1 LM slsap=12 dlsap=00
GET_VALUE_BY_CLASS: "IrDA:IrCOMM" "Parameters" (28)
e9320012840b497244413a4972434f4d4d0a506172616d6574657273
. 2 . . . . I r D A : I r C O M M . P a r a m e t e r s
08:19:56.428363 i:rsp  < ca=e8 pf=1 nr=2 ns=1 LM slsap=00 dlsap=12
GET_VALUE_BY_CLASS: Success N/A (16)
e85212008400000100010200030001O6
. R . . . . . .
08:19:56.428378 i:cmd  > ca=e8 pf=1 nr=2 ns=2 LM slsap=12 dlsap=00 DISC (6)
e95480120201
. T . . . .
08:19:56.435267 rr:rsp < ca=e8 pf=1 nr=3 (2)
e871
. q
```

Ya tenemos todo el tema del IrCOMM negociado...ahora un nivel mas. La elección de tinyTP como protocolo de transporte.

```
08:19:56.435278 i:cmd  > ca=e8 pf=1 nr=2 ns=3 LM slsap=13 dlsap=00 CONN_CMD (6)
e95680130100
. V . . . .
08:19:56.442599 i:rsp  < ca=e8 pf=1 nr=4 ns=2 LM slsap=00 dlsap=13 CONN_RSP (6)
e89493008100
         . . . . . .
08:19:56.442613 i:cmd  > ca=e8 pf=1 nr=3 ns=4 LM slsap=13 dlsap=00
GET_VALUE_BY_CLASS: "IrDA:IrCOMM" "IrDA:TinyTP:LsapSel" (37)
e9780013840b497244413a4972434f4d4d13497244413a54696e7954503a4c73
. x . . . . I r D A : I r C O M M . I r D A : T i n y T P : L s
08:19:56.455230 i:rsp  < ca=e8 pf=1 nr=5 ns=3 LM slsap=00 dlsap=13
GET_VALUE_BY_CLASS: Success Integer: 02 (15)
e8b6130084000001000101000000002
         . . . . . .
08:19:56.455241 i:cmd  > ca=e8 pf=1 nr=4 ns=5 LM slsap=13 dlsap=00 DISC (6)
e99a80130201
         . . . . . .
08:19:56.462100 rr:rsp < ca=e8 pf=1 nr=6 (2)
e8d1
```

```
        . .

08:19:56.462108 i:cmd  > ca=e8 pf=1 nr=4 ns=6 LM slsap=11 dlsap=02 CONN_CMD TTP
credits=0(7)
e99c821101000e
        . . . . . .
08:19:56.470099 i:rsp  < ca=e8 pf=1 nr=7 ns=4 LM slsap=02 dlsap=11 CONN_RSP TTP
credits=0(7)
e8f89102810001
        . . . . . .
08:19:56.470121 rr:cmd > ca=e8 pf=1 nr=5 (2)
e9b1
        . .
08:19:56.476544 rr:rsp < ca=e8 pf=1 nr=7 (2)
e8f1
        . .
```

Empieza pues la sincronización:

```
08:19:56.476553 i:cmd  > ca=e8 pf=1 nr=5 ns=7 LM slsap=11 dlsap=02 TTP credits=0
(24)
e9be021100120001041004000025801101131201002001 0c
        . . . . . . . . . . . . . . % . . . . . .   . .
08:19:56.486517 i:rsp  < ca=e8 pf=1 nr=0 ns=5 LM slsap=02 dlsap=11 TTP credits=2
(5)
e81a110202
        . . . . .
08:19:56.486527 rr:cmd > ca=e8 pf=1 nr=6 (2)
e9d1
        . .
08:19:56.492879 rr:rsp < ca=e8 pf=1 nr=0 (2)
e811
        . .
08:19:56.541090 rr:cmd > ca=e8 pf=1 nr=6 (2)
e9d1
        . .
08:19:56.550883 i:rsp  < ca=e8 pf=1 nr=0 ns=6 LM slsap=02 dlsap=11 TTP credits=0
(29)
e81c1102001710040000258011010312010013021113140213112 0 0 1 0 c
        . . . . . . . . . . . % . . . . . . . . . . . . . . . .   . .
08:19:56.550924 rr:cmd > ca=e8 pf=1 nr=7 (2)
e9f1
        . .
08:19:56.557610 rr:rsp < ca=e8 pf=1 nr=0 (2)
e811
        . .
08:19:56.601092 rr:cmd > ca=e8 pf=1 nr=7 (2)
e9f1
        . .
08:19:56.609661 i:rsp  < ca=e8 pf=1 nr=0 ns=7 LM slsap=02 dlsap=11 TTP credits=0
(18)
```

```
e81e11020000beefed030303000029ccd5cb
        . . . . . . . . . . . . . . . ) . . .
[..............]


08:20:06.067169 i:cmd  > ca=e8 pf=1 nr=3 ns=5 LM slsap=11 dlsap=02 TTP credits=2
(22)
e97a02110200beefed03030200042ad002c00004222f
. z . . . . . . . . . . . . . * . . . . . " /
08:20:06.077126 i:rsp  < ca=e8 pf=1 nr=6 ns=3 LM slsap=02 dlsap=11 TTP credits=1
(5)
e8d6110201
        . . . . .
08:20:06.077179 i:cmd  > ca=e8 pf=1 nr=4 ns=6 LM slsap=11 dlsap=02 DISC (6)
e99c82110201
        . . . . . .
08:20:06.084171 rr:rsp < ca=e8 pf=1 nr=7 (2)
e8f1
        . .
08:20:06.084209 rr:cmd > ca=e8 pf=1 nr=4 (2)
e991
        . .
rd:rsp < ca=0xe8 pf=1 (2)
e853
. S
08:20:06.090113 disc:cmd > ca=0xe8 pf=1 (2)
e953
. S
08:20:06.094935 ua:rsp ca=e8 pf=1 03edefbe < 01021182 (2)
e873
. s
```

Esto último es la desconexión. Consecuencia de, o por abortar la transmisión o porque ha finalizado la sincronización.

4.2 Conexión de Palm a Internet vía IR a través de Linux (PPP)

Ahora otro pasito mas. Contectaremos la Palm con ppp via IR al Linux para que dé salida a Internet a la Palm.

Como siempre, prepararemos el núcleo por si no tenemos soporte. Vamos a utilizar enmascaramiento, así que deben de estar seleccionadas estas opciones del núcleo, entre otras muchas:

```
CONFIG_IP_NF_IPTABLES=y
CONFIG_IP_NF_FILTER=y
CONFIG_IP_NF_TARGET_REJECT=y
CONFIG_IP_NF_TARGET_MIRROR=y
CONFIG_IP_NF_NAT=y
CONFIG_IP_NF_NAT_NEEDED=y
```

```
CONFIG_IP_NF_TARGET_MASQUERADE=y
CONFIG_IP_NF_TARGET_REDIRECT=y
CONFIG_IP_NF_NAT_FTP=y
```

Iptables debe estar instalado. Procedemos:

```
echo 1 > /proc/sys/net/ipv4/ip_forward
iptables -t nat -A POSTROUTING -o eth0 -d 192.168.0.0/16 -j MASQUERADE
```

NOTA: No vengais con cuestiones de que así se enmascara todo y entonces es menos seguro porque blablabla, estoy siendo practico.

```
pppd /dev/ircomm0 115200 192.168.1.149:192.168.1.246 proxyarp passive
silent nopersist noauth local nodetach ms-dns 192.168.2.1
```

Recomiendo un man pppd. Pero vamos, así a bote pronto, que sepáis que 192.168.1.149 es el ordenador (Linux), la 192.168.1.246 la que se le asigna a la Palm y el ms-dns es la dns. Sigamos:

```
annapurna:~# cat /proc/sys/net/ipv4/ip_forward
1
annapurna:~# pppd /dev/ircomm0 115200 192.168.1.149:192.168.1.246 proxyarp
passive  silent nopersist noauth local nodetach ms-dns 192.168.1.122

PPP generic driver version 2.4.1
Using interface ppp0
Connect: ppp0 <--> /dev/ircomm0
```

Ahora nos vamos a la Palm. Elegimos "Preferencias", "Menú de red". Allí elegimos:

```
Servicio: Linux o el que sea, lo podemos crear.
Usuario: da igual mientras en el pppd este el noauth.
Contraseña: preguntar
Conexion: IR a PC/Portatil
```

Lo de "Detalles" puede estar vacio, no hace falta ningún guión.

Le damos a conectar.
Cuando conectas, en el terminal del Linux aparece:

```
PPP BSD Compression module registered
PPP Deflate Compression module registered
Cannot determine ethernet address for proxy ARP
local  IP address 192.168.1.149
remote IP address 192.168.1.246
```

Vía syslog leemos:

```
Nov  7 11:36:36 annapurna kernel: PPP generic driver version 2.4.1
Nov  7 11:36:36 annapurna pppd[1704]: pppd 2.4.1 started by root, uid 0
Nov  7 11:36:36 annapurna kernel: ircomm_tty_attach_cable()
```

```
Nov  7 11:36:36 annapurna kernel: ircomm_tty_ias_register()
Nov  7 11:36:36 annapurna kernel: irlmp_register_client_R17f18bfb()
Nov  7 11:36:36 annapurna pppd[1704]: Using interface ppp0
Nov  7 11:36:36 annapurna pppd[1704]: Connect: ppp0 <--> /dev/ircomm0
Nov  7 11:37:48 annapurna kernel: irlap_change_speed(), setting speed to 115200
Nov  7 11:37:48 annapurna kernel: iriap_connect_indication()
Nov  7 11:37:48 annapurna kernel: irlmp_state_dtr(), Unknown event
LM_LAP_CONNECT_CONFIRM Nov  7 11:37:48 annapurna kernel:
ircomm_param_service_type(), services in common=04
Nov  7 11:37:48 annapurna kernel: ircomm_param_service_type(), resulting service
type=0x04 Nov  7 11:37:48 annapurna kernel: ircomm_param_xon_xoff(), XON/XOFF =
0x11,0x13
Nov  7 11:37:48 annapurna kernel: ircomm_param_enq_ack(), ENQ/ACK = 0x13,0x11
Nov  7 11:37:48 annapurna kernel: ircomm_tty_check_modem_status()
Nov  7 11:37:48 annapurna kernel: irlmp_state_active(), Unknown event
LM_LAP_DISCOVERY_CONFIRM
Nov  7 11:37:49 annapurna kernel: PPP BSD Compression module registered
Nov  7 11:37:49 annapurna kernel: PPP Deflate Compression module registered
Nov  7 11:37:49 annapurna kernel: irlmp_state_dtr(), Unknown event
LM_WATCHDOG_TIMEOUT
Nov  7 11:37:49 annapurna pppd[1704]: Cannot determine ethernet address for
proxy ARP
Nov  7 11:37:49 annapurna pppd[1704]: local  IP address 192.168.1.149
Nov  7 11:37:49 annapurna pppd[1704]: remote IP address 192.168.1.246
```

Aparecen algunos fallitos, pero funciona.

Bueno, ya estamos conectados.

Para desconectar le damos a *Desconectar* en la Palm o matamos al pppd, y nos devuelve en la terminal lo de siempre cuando estamos con un modem normal.

```
IPCP terminated by peer
LCP terminated by peer
Modem hangup
Connection terminated.
Connect time 0.3 minutes.
Sent 73 bytes, received 62 bytes.
```

La conexión es, a nivel de IrLAP, calcada a cuando realizamos la sincronización. Sin embargo hay una parte que pertenece a la negociacion del PPP. Pego directamente los cachos del registro mas interesantes.

Conexión.....

```
10:43:31.711102 xid:cmd d8a82a1b > ffffffff S=6 s=0 (14)
ff3f011b2aa8d8ffffffff010000
         . ? . . * . . . . . . . . .
10:43:31.801086 xid:cmd d8a82a1b > ffffffff S=6 s=1 (14)
ff3f011b2aa8d8ffffffff010100
         . ? . . * . . . . . . . . .
10:43:31.891087 xid:cmd d8a82a1b > ffffffff S=6 s=2 (14)
ff3f011b2aa8d8ffffffff010200
```

```
         . ? . . * . . . . . . . . .
10:43:31.981083 xid:cmd d8a82a1b > ffffffff S=6 s=3 (14)
ff3f011b2aa8d8ffffffff010300
         . ? . . * . . . . . . . . .
10:43:32.071088 xid:cmd d8a82a1b > ffffffff S=6 s=4 (14)
ff3f011b2aa8d8ffffffff010400
         . ? . . * . . . . . . . . .
10:43:32.161083 xid:cmd d8a82a1b > ffffffff S=6 s=5 (14)
ff3f011b2aa8d8ffffffff010500
         . ? . . * . . . . . . . . .
10:43:32.251084 xid:cmd d8a82a1b > ffffffff S=6 s=* annapurna hint=8404 [
Computer IrCOMM ] (26)
ff3f011b2aa8d8ffffffff01ff00840400616e6e617075726e61
. ? . . * . . . . . . . . . . . . a n n a p u r n a
10:43:33.101644 xid:cmd ffffffff < 2542fb13 S=6 s=0 (14)
ff3f0113fb4225ffffffff010000
. ? . . . B % . . . . . . .
10:43:33.191479 xid:cmd ffffffff < 2542fb13 S=6 s=1 (14)
ff3f0113fb4225ffffffff010100
. ? . . . B % . . . . . . .
10:43:33.191502 xid:rsp d8a82a1b > 2542fb13 S=6 s=1 annapurna hint=8404 [
Computer IrCOMM ] (26)
febf011b2aa8d813fb4225010100840400616e6e617075726e61
. . . . * . . . . B % . . . . . a n n a p u r n a
10:43:33.271729 xid:cmd ffffffff < 2542fb13 S=6 s=2 (14)
ff3f0113fb4225ffffffff010200
. ? . . . B % . . . . . . .
10:43:33.421490 xid:cmd ffffffff < 2542fb13 S=6 s=3 (14)
ff3f0113fb4225ffffffff010300
. ? . . . B % . . . . . . .
10:43:33.511434 xid:cmd ffffffff < 2542fb13 S=6 s=4 (14)
ff3f0113fb4225ffffffff010400
. ? . . . B % . . . . . . .
10:43:33.601481 xid:cmd ffffffff < 2542fb13 S=6 s=5 (14)
ff3f0113fb4225ffffffff010500
. ? . . . B % . . . . . . .
10:43:33.701342 xid:cmd ffffffff < 2542fb13 S=6 s=* IrCOMM hint=8204 [
PDA/Palmtop IrCOMM ] (23)
ff3f0113fb4225ffffffff01ff008204004972434f4d4d
. ? . . . B % . . . . . . . . . I r C O M M
10:43:33.750610 snrm:cmd ca=fe pf=1 d8a82a1b < 2542fb13 new-ca=da (32)
ff9313fb42251b2aa8d8da01013f82010183010f840101850108601070801ff . . . . B % . *
. . . . . ? . . . . . . . . . . . . . . . . . .
10:43:33.750666 ua:rsp ca=da pf=1 d8a82a1b > 2542fb13 (31)
da731b2aa8d813fb422501013e82010183013f84017f8501ff860107080107 . s . * . . . . B
% . . > . . . . . ? . . . . . . . . . . . . .
10:43:33.806403 rr:cmd < ca=da pf=1 nr=0 (2)
db11
         . .
10:43:33.806419 rr:rsp > ca=da pf=1 nr=0 (2)
```

```
da11
        . .
10:43:33.813486 i:cmd  < ca=da pf=1 nr=0 ns=0 LM slsap=01 dlsap=00 CONN_CMD (6)
db1080010100
        . . . . . .
10:43:33.813522 i:rsp  > ca=da pf=1 nr=1 ns=0 LM slsap=00 dlsap=01 CONN_RSP (6)
da3081008100
. 0 . . . .
10:43:33.824685 i:cmd  < ca=da pf=1 nr=1 ns=1 LM slsap=01 dlsap=00
GET_VALUE_BY_CLASS: "IrDA:IrCOMM" "IrDA:TinyTP:LsapSel" (37)
db320001840b497244413a4972434f4d4d13497244413a54696e7954503a4c73
. 2 . . . . I r D A : I r C O M M . I r D A : T i n y T P : L s
10:43:33.824711 i:rsp  > ca=da pf=1 nr=2 ns=1 LM slsap=00 dlsap=01
GET_VALUE_BY_CLASS: Success Integer: 15 (15)
da52010084000001234301000000015
. R . . . . . # C . . . . .
10:43:33.834302 i:cmd  < ca=da pf=1 nr=2 ns=2 LM slsap=02 dlsap=15 CONN_CMD TTP
credits=0(11)
db54950201000103000104
. T . . . . . .
10:43:33.834358 i:rsp  > ca=da pf=1 nr=3 ns=2 LM slsap=15 dlsap=02 CONN_RSP TTP
credits=0(7)
da74821581000e
. t . . . . .
10:43:33.841561 rr:cmd < ca=da pf=1 nr=3 (2)
db71
. q
10:43:33.841572 i:rsp  > ca=da pf=1 nr=3 ns=3 LM slsap=15 dlsap=02 TTP credits=0
(21)
da760215000f1004000025801101031201002001 0c
. v . . . . . % . . . . . . .
10:43:33.851185 i:cmd  < ca=da pf=1 nr=4 ns=3 LM slsap=02 dlsap=15 TTP credits=2
(5)
db96150202
        . . . . .
10:43:33.851197 rr:rsp > ca=da pf=1 nr=4 (2)
da91
        . .
10:43:33.857643 rr:cmd < ca=da pf=1 nr=4 (2)
db91
        . .

[..............]
```

NOTA: ¡Atentos! entra el PPPD

```
10:43:33.933213 i:cmd  < ca=da pf=1 nr=4 ns=5 LM slsap=02 dlsap=15 TTP credits=0
(43)
db9a150200007eff7d23c0217d217d267d207d2e7d227d267d207d207d207d20
        . . . . . ~ . } # . ! } ! } & }   } . } " } & }   }   }   }
10:43:33.933238 rr:rsp > ca=da pf=1 nr=6 (2)
```

```
dad1
        . .
10:43:33.939934 rr:cmd < ca=da pf=1 nr=4 (2)
db91
        . .
10:43:33.939946 i:rsp  > ca=da pf=1 nr=6 ns=4 LM slsap=15 dlsap=02 TTP credits=2
(51)
dad8021502007eff7d23c0217d217d217d207d347d227d267d207d207d207d20
. . . . . ~ . } # . ! } ! } ! }   } 4 } " } & }   }   }   }
10:43:33.955593 rr:cmd < ca=da pf=1 nr=5 (2)
dbb1
        . .
10:43:33.955605 i:rsp  > ca=da pf=1 nr=6 ns=5 LM slsap=15 dlsap=02 TTP credits=0
(44)
dada021500007eff7d23c0217d227d267d207d2e7d227d267d207d207d207d20
        . . . . . ~ . } # . ! } " } & }   } . } " } & }   }   }   }
10:43:33.970152 i:cmd  < ca=da pf=1 nr=6 ns=6 LM slsap=02 dlsap=15 TTP credits=1
(5)
dbdc150201
        . . . . .
10:43:33.970163 rr:rsp > ca=da pf=1 nr=7 (2)
daf1
        . .

[..............]

10:43:34.041139 i:cmd  < ca=da pf=1 nr=6 ns=7 LM slsap=02 dlsap=15 TTP credits=1
(31)
dbde150201007eff7d23c0217d247d217d207d2a7d257d26e6378f95a9c27e
. . . . . ~ . } # . ! } $ } ! }   } * } % } & . 7 . . . . ~
10:43:34.041155 rr:rsp > ca=da pf=1 nr=0 (2)
da11
        . .
10:43:34.047801 rr:cmd < ca=da pf=1 nr=6 (2)
dbd1
        . .
10:43:34.047809 i:rsp  > ca=da pf=1 nr=0 ns=6 LM slsap=15 dlsap=02 TTP credits=1
(43)
da1c021501007eff7d23c0217d217d227d207d2e7d227d267d207d207d207d20
        . . . . . ~ . } # . ! } ! } " }   } . } " } & }   }   }   }

[..............]

10:43:34.141818 i:cmd  < ca=da pf=1 nr=7 ns=0 LM slsap=02 dlsap=15 TTP credits=1
(43)
dbf0150201007eff7d23c0217d227d227d207d2e7d227d267d207d207d207d20
        . . . . . ~ . } # . ! } " } " }   } . } " } & }   }   }   }
10:43:34.141842 rr:rsp > ca=da pf=1 nr=1 (2)
da31
. 1
```

```
10:43:34.148556 rr:cmd < ca=da pf=1 nr=7 (2)
dbf1
        . .
10:43:34.148567 i:rsp  > ca=da pf=1 nr=1 ns=7 LM slsap=15 dlsap=02 TTP credits=1
(27)
da3e021501008021010100100306c0a801950206002d0f0146e97e
. > . . . . . ! . . . . . . . . . . . . . . - . . F . ~
10:43:34.158795 rr:cmd < ca=da pf=1 nr=0 (2)
db11
        . .
10:43:34.158803 i:rsp  > ca=da pf=1 nr=1 ns=0 LM slsap=15 dlsap=02 TTP credits=0
(26)
da300215000080fd0101000f1a047800180478001503 2f32a37e
. 0 . . . . . . . . . . . . . x . . . x . . . / 2 . ~
10:43:34.169307 i:cmd  < ca=da pf=1 nr=1 ns=1 LM slsap=02 dlsap=15 TTP credits=1
(5)
db32150201
. 2 . . .
10:43:34.169321 rr:rsp > ca=da pf=1 nr=2 (2)
da51
. Q
[..............]
```

Solicitamos desconexión DESDE la Palm. Primero el PPP y luego el IrCOMM:

```
10:43:40.739173 i:cmd  < ca=da pf=1 nr=5 ns=0 LM slsap=02 dlsap=15 TTP credits=1
(23)
dbb0150201007eff7d23c0217d257d287d207d24235b7e
        . . . . . ~ . } # . ! } % } ( }   } $ # [ ~
10:43:40.739194 rr:rsp > ca=da pf=1 nr=1 (2)
da31
. 1
10:43:40.745815 rr:cmd < ca=da pf=1 nr=5 (2)
dbb1
        . .
10:43:40.745828 i:rsp  > ca=da pf=1 nr=1 ns=5 LM slsap=15 dlsap=02 TTP credits=1
(24)
da3a021501007eff7d23c0217d267d287d207d24ee7d5e7e
        . : . . . . ~ . } # . ! } & } ( }   } $ . } ^ ~
10:43:40.757012 rr:cmd < ca=da pf=1 nr=6 (2)
dbd1
        . .
10:43:40.757028 rr:rsp > ca=da pf=1 nr=1 (2)
da31
. 1
10:43:40.763324 rr:cmd < ca=da pf=1 nr=6 (2)
dbd1
        . .
10:43:40.763334 rr:rsp > ca=da pf=1 nr=1 (2)
da31
. 1
```

```
10:43:40.769749 rr:cmd < ca=da pf=1 nr=6 (2)
dbd1
         . .
10:43:40.769757 rr:rsp > ca=da pf=1 nr=1 (2)
da31
. 1
10:43:41.239588 disc:cmd < ca=0xda pf=1 (2)
db53
. S
10:43:41.239617 ua:rsp ca=da pf=1 d8a82a1b > 2542fb13 (10)
da731b2aa8d813fb4225
. s . * . . . . B %
10:43:46.711103 xid:cmd d8a82a1b > ffffffff S=6 s=0 (14)
ff3f011b2aa8d8ffffffff010000
         . ? . . * . . . . . . . . .
```

A modo de curiosidad, aquí teneis un fingerprint a una palm realizado con un nmap. Es curioso, la misma operación desde el crandle me cuelga la Palm.

```
SInfo(V=2.54BETA28%P=i586-pc-linux-gnu%D=10/15%Time=3BCB32E9%O=-1%C=1)
T5(Resp=Y%DF=N%W=0%ACK=S++%Flags=AR%Ops=)
T6(Resp=Y%DF=N%W=0%ACK=O%Flags=R%Ops=)
T7(Resp=Y%DF=N%W=0%ACK=S++%Flags=AR%Ops=)
PU(Resp=Y%DF=N%TOS=0%IPLEN=38%RIPTL=148%RID=E%RIPCK=E%UCK=E%ULEN=134%DAT=E)
```

Bueno, y con esto ponemos final a este punto. ¿Que os ha parecido?.

¿Mas?.

Chapter 5

INTERCONEXIÓN CON WINDOWS CE (IPAQ, JORNADA...) (PPP)

5.1 Interconexión con IPAQ (sirve para el resto también)

Bueno, una vez ya tenemos el soporte de IRDA compilado en el núcleo, los pasos a seguir tienen cierta similitud a los de Palm. Aunque hay ciertos matices a tener en cuenta con el pppd.

La pruebas se realizaron con una Compaq Ipaq 3660 con WindowsCE (PocketPC2000) vía IRDA.

El soporte de la interconexión vía USB, para aquellos que se lo estén preguntando todavía, no lo he conseguido hacer ir con la ipaq. El USB en Linux está fresquito aún.

Otra aclaración, si en vez de WindowsCE estamos utilizando algún tipo de Linux en la ipaq, la conexion IRDA es como si fuera entre PC's. Dicho esto, vamos allá.

En la secuencia de reconocimiento encontramos...

```
07:55:51.439835 xid:cmd ffffffff < 00000a33 S=6 s=* Pocket_PC hint=8204 [
PDA/Palmtop IrCOMM ] (26)
ff3f01330a0000ffffffff01ff00820400506f636b65745f5043
. ? . 3 . . . . . . . . . . . . . . P o c k e t _ P C
```

El Windows CE no es como la Palm, que marcando el 0 como número de teléfono actua en plan *modo red*. Lo que necesitamos con la Ipaq es emular todo lo que se pueda la conexión como si se tratara de un módem. Esto incluye pues, toda la secuencia de comandos AT. Interesantísimos aquí los RFCs que aparecen en la parte final del man pppd.

A la Ipaq pues, le vamos a dar un tlf que no exista, da igual la cantidad de cifras, yo por costumbre con la Palm uso también el cero, y le vamos a desmarcar la opción de esperar tono en la llamada. Lógicamente podemos emular muchas cosas, pero el tono de llamada, pues como que no.

Me voy a centrar a partir de aquí en un caso concreto. Se trata de la Ipaq 3660 con WindowsCE 3.0. Para el resto es tan sencillo como espiar la conexión e ir mirando los comandos que va mandando. Como ejemplo ahi va el caso este concreto, lógicamente me como del registro lo que no interese....

Alla vá, la primero el churro que envía el pppd:

```
07:55:54.946139 i:cmd  > ca=de pf=1 nr=4 ns=6 LM slsap=2f dlsap=01 TTP credits=0
(52)
df9c012f00007eff7d23c0217d217d257d207d347d227d267d207d207d207d20
. . . / . . ~ . } # . ! } ! } % }   } 4 } " } & }   }   }   }
Ahora el WinCE quiere reinicializar "nuestro" modem:
07:55:59.387185 i:rsp  < ca=de pf=1 nr=0 ns=5 LM slsap=01 dlsap=2f TTP credits=0
(13)
de1a2f01000041545a0d2b2b2b
. . / . . . A T Z . + + +
```

Mirando en la documentación tecnica, deberíamos mandar un *OK*. Lo apuntamos. Configuramos el pppd para que envíe el *OK* (luego digo cómo) y empezamos la secuencia otra vez. Pasamos por el *ATZ*, mandamos el *OK* y ahora nos encontramos con:

```
09:00:09.016508 i:rsp  < ca=de pf=1 nr=7 ns=3 LM slsap=02 dlsap=48 (19)
def648020000415445305631264331264432 0d
. . H . . . A T E 0 V 1 & C 1 & D 2 .
```

ATE0V1 y parámetros.... bien...miramos la documentación de turno y vemos que va otro *OK*. Volvemos a apuntar y a empezar secuencia. Mandamos este *OK* y ahora el bichito nos envía:

```
09:00:09.095605 i:rsp  < ca=de pf=1 nr=2 ns=5 LM slsap=02 dlsap=48 (14)
de5a48020000415453373d33300d
. Z H . . . A T S 7 = 3 0 .
```

ATS7.... y otro *OK* de turno. Seguimos y ahora nos envía:

```
09:00:09.225783 i:rsp  < ca=de pf=1 nr=5 ns=7 LM slsap=02 dlsap=48 (11)
debe48020000415458330d
. . H . . . A T X 3 .
```

ATX3... pos ala, otro *OK* y continuamos....

```
09:00:09.306238 i:rsp  < ca=de pf=1 nr=0 ns=0 LM slsap=02 dlsap=48 (18)
de1048020000415444543 92c30302c30300d
. . H . . . A T D T 9 , 0 0 , 0 0 .
```

---

yeyeyeyeyeyeye.... esto es el famoso *ATDT*, aquí está marcando.... para dar la conformidad de marcado, aquí va un *CONNECT*. Y a partir de este momento empieza ya toda la negociación ppp.

¿Cómo enviamos pues toda la secuencia?. Sencillo:

Vamos al directorio /etc/ppp/peers.
Allí creamos el fichero "ipaq" y en su interior metemos (para este caso concreto):

```
/dev/ircomm0 115200 crtscts
connect '/usr/sbin/chat -v ATZ OK AT OK ATE0V1 OK ATS7 OK ATX3 OK ATDT CONNECT'
noauth
local
passive
debug
ms-dns 192.168.0.122 #ip de las dns
:192.168.0.148 #ip a asignar
```

Los comandos están en el man del pppd. Una vez tenemos el fichero creado, los pasos a seguir son sencillos.

```
pppd call ipaq
```

y en la ipaq le damos a conectar, a partir de ahí... ya estamos.

Recordaros meterle el MASQ al ipchains-iptables y el *"1"* al *ip_forward* para enmascararos la salida.

Bueno, y ahora, a por la telefonía.

Chapter 6

INTERCONEXIÓN TELEFONÍA MÓVIL

6.1 ¿Qué necesitamos?

No dispongo de teléfono móvil con IR así que, posteriormente, en la sección de agradecimientos me estenderé bastante XD. Las pruebas han sido bastante limitadas por la escasez de teléfonos con IR a mi alcance. El material estaba compuesto por:

Annapurna, mi portátil clónico de siempre.

Nokia's 6150, 7110, 8210.

Cable de datos. (si modem en 6150, caso especial)

En esta sección necesitaremos:(Nomenclatura Debian, en otras distribuciones no sé si se conserva el nombre del paquete)

```
--- Package: wvdial --------
Priority: optional
Section: comm
Installed-Size: 240
Maintainer: Baruch Even <baruch@debian.org>
Architecture: i386
Version: 1.42-2
Depends: ppp (>= 2.3.0), libc6 (>= 2.2.3-7), libstdc++2.10-glibc2.2 (>=
1:2.95.4-0.010810)
Filename: pool/main/w/wvdial/wvdial_1.42-2_i386.deb
Size: 77196
MD5sum: 472934e1d8c4571842bf6b0a0d09a2fe
Description: PPP dialer with built-in intelligence.
WvDial sacrifices some of the flexibility of programs like "chat" in order to
make your dialup configuration easier.  When you install this package, your
modem will be detected automatically and you need to specify just three
parameters: the phone number, username, and password.  WvDial knows enough to
dial with most modems and log in to most servers without any other help.

In particular, you no longer need a "chat script" to handle the most common
```

```
situations.
Task: dialup

--- Package: gnokii --------
Status: install ok installed
Priority: optional
Section: comm
Installed-Size: 3249
Maintainer: Erik Rossen (Linux consultant) <rossen@freesurf.ch> Version: 0.3.3
Depends: adduser, libc6 (>= 2.2.3-7), libglib1.2 (>= 1.2.0), libgtk1.2 (>=
1.2.10-1), xlibs (>> 4.1.0)
Conffiles:
/etc/gnokiirc 5520f34285ec9e9934bccfdae51617cc
/etc/gnokiirc newconffile
Description: Linux/Unix tool suite for Nokia mobile phones
Gnokii is a Linux/Unix tool suite and (eventually) modem/fax driver
for Nokia mobile phones, released under the GPL.
http://www.gnokii.org


--- Package: gsm-utils --------
Priority: extra
Section: comm

Installed-Size: 372
Maintainer: Mikael Hedin <micce@debian.org>
Architecture: i386
Source: gsmlib
Version: 1.6-5
Depends: libgsmme1 (= 1.6-5)
Filename: pool/main/g/gsmlib/gsm-utils_1.6-5_i386.deb
Size: 118136
MD5sum: ac27d320f8898776dd3a21307ea0055a
Description: Application to access and control a GSM mobile phone.  Some simple
command line programs to access a GSM mobile phone via GSM modem or IrDA.
Functions include: modification of phonebooks and reading, writing, sending and
receiving SMS messages.  Uses the GSM standards ETSI GSM 07.07, ETSI GSM 07.05,
and others.
```

6.2 Gnokii

Bueno, esta sección está dedicada a todos los fanatiquillos de los móviles. Yo, al contrario que ellos, no tengo ni puta idea, pero les voy a intentar dar la base por si se quieren poner con ello desde Linux. A ver que tal me sale.

Tenemos dos paquetes (por supuesto con sus fuentes y documentacion, e ahí una de las ventajas) que trabajan con los protocolos de Nokia para hacerle pirulillas al móvil.

El primero es **gnokii**. Por mucho que anuncien su compatibilidad con el IR de momento no lo he conseguido hacer ir. Eso si, con el cable va de putísima madre.

Aunque en un principio dicen que es posible la conexión con:

```
(README del gnokii)
3810 Series
```

```
3110/Sonera (Finland)
3110/Telefonica (Spain)
3810/Telstra (Australia)
3810/Optus [Actually Hutchison] (Australia)
3810/Mobile One (Singapore)
3810/SingTel Mobile (Singapore)
8110/Proximus (Belgium)
8110/D2 Privat (Germany)
8110i/MTN (South Africa)
8110/Beeline Vympelkom (Russia)
8110i/A1 Mobilkom (Austria)

6110 Series
6190/Microcel (Canada) (GSM 1900)
6150/Viag Interkom (Germany) (GSM 1800)
6150/Omnitel (Italy)
6150/Beeline (Vympelkom) (Moscow, Russia)
6150/MTS (Moscow, Russia)
6150/NWGSM (St. Petersburg, Russia)
6150/NetCom GSM (Norway)
6130/IDEA Centertel (Poland) (GSM 1800)
6110/Paegas (Czech Republic)
6110/Panafon (Greece)
6110/Telstra Mobilenet (Australia)
5110/Omnitel (Italy)
5110/Paegas (Czech Republic)
5110/ERA GSM (Poland)
5110/Plus GSM (Poland)
5130/IDEA Centertel (Poland)
5190/Microcel (Canada) (GSM 1900)

3210 (some functions does not not work yet)
2110 Series
6160 Series
640 Series
```

El caso es que ni con un 7110 ni con el 8210 ni con el 6150 he conseguido que vía IR el programa funcione. Me reitero, con cable sin problemas. De las cosas curiosas es la posibilidad de activar el netmonitor y alguna chorradilla mas.

Sobre X han desarrollado xgnokii que va bastante bien.

El tema de documentación también está muy bien. Para los entendidos tenéis unos enlaces bastante atractivos:

-files (patches ?) for MIDI support

-compiling gnokii for Win32

-gnokii 0.3.3_pre6 in rpm file

-site of Jan Derfinak (ja@mail.upjs.sk) -> main author of XGNOKII

-site of (Pavel Janik Pavel.Janik@linux.cz). Here is gnokii 0.3.3_pre5

-site of Hugh Blemings (hugh.blemings@vsb.com.au)

-protocol for Nokia 2110

-protocol for calendar for N7110

-protocol for Nokia 2110 and old phones

-protocol for NHx models

-protocol for NSx models or (better ;-)))) info/fbus.txt

-protocol for TDMA phones

Ante la incompatibilidad de gnokii con el IR (o eso me parece) buscando he encontrado otra rama de desarrollo basada en gnokii. Al parecer que hay un tio que se ha cansado de que los de gnokii no le hicieran caso con los parches y se ha creado su propia rama de desarrollo. Se trata de mygnokii.

Todavía estoy pendiente de probar este paquete.

6.3 GSM-utils

Encontramos otra linea de herramientas para trabajar con el movil-gsm. El paquete son las gsm-utils.
Además de trabajar con el smsstore (tb lo hace gnokii) tiene una implementación sobre comandos AT para Nokia que creo que está muy chula también.

El paquete soporta:

Nokia 6150/Xircom REM56G.100

Nokia 6150/Options "GSM-Ready® Cellular-Only" modem from Option International

Nokia 6210/- (Linux IrDA serial device)

Nokia 8810/- (Linux IrDA serial device)

Siemens S10D/Dr Neuhaus Gipsy Card GSM

Siemens S25/- (Linux IrDA serial device)

Siemens S35i/- (Linux IrDA serial device)

Ericcson SH888/- (Linux IrDA serial device)

Ericsson 6050102/GM 12 GSM module - -/Siemens M20T (stand-alone GSM module) - -/Wavecom WM02 GSM (stand-alone GSM module)

Nokia 7110 (firware rev 4.80)/- (Linux IrDA serial device)

Nokia 8290 (USA GSM 1900MHz)/- (Linux IrDA serial device)

Como veis no es compatible con el 6150 via IR. Con el 7110 y el 8210 se supone que sin problemas.

Entre todas las utilidades que encontramos en el paquete, llamar la atención de gsmctl. Al parecer implementa una serie de comandos AT para ver las especificaciones del teléfono. Según leo en la documentación de mygnokii (nada que ver con este paquete) es posible optener información del móvil desde los comandos AT del módem.Sería de la siguiente manera:

```
ATDT number;
dial voice
AT+CGSN
get IMEI
AT+CGMI
get phone manufacter
AT+CGMM
get phone model
AT+CGMR get hardware and firmware version AT+VTS=a,b,c,d; send "abcd" DTMF
sequence
```

De esta forma, os pego una salida real del comando gsmctl. Y os lo explico al final.

```
annapurna:~# gsmctl device /dev/ircomm0 ALL
<ME0>  Manufacturer: Nokia Mobile Phones
<ME1>  Model: Nokia 8210
<ME2>  Revision: SW5.26
<ME3>  Serial Number: 350112107500xxx
<OP0>  Status: current  Long name: "   Short name: "   Numeric name: 21401
<OP1>  Status: forbidden  Long name: "   Short name: "   Numeric name: 21403
<OP2>  Status: forbidden  Long name: "   Short name: "   Numeric name: 21407
<CURROP0>  Long name: "   Short name: "   Numeric name: 21401  Mode: automatic
<FLSTAT0>  'PS'
<FLSTAT1>  'SC'
<FLSTAT2>  'AO'
<FLSTAT3>  'OI'
<FLSTAT4>  'OX'
<FLSTAT5>  'AI'
<FLSTAT6>  'IR'
<FLSTAT7>  'FD'
<FLCAP0>  'PS' 'SC' 'AO' 'OI' 'OX' 'AI' 'IR' 'AB' 'AG' 'AC' 'FD'
<PW0>  'PS' 5
<PW1>  'SC' 8
<PW2>  'AB' 4
<PW3>  'P2' 8
<CLIP0>  off
<FORW0.0>  UnconditionalReason  Voice  active number:   subaddr:   time:
134576720
<FORW0.1>  UnconditionalReason  Data  active number:   subaddr:   time:
1073785150
<FORW0.2>  UnconditionalReason  Fax  active number:   subaddr:   time: 3
<FORW1.0>  MobileBusyReason  Voice  active number:   subaddr:   time: 134576720
<FORW1.1>  MobileBusyReason  Data  active number:   subaddr:   time: 1073785150
<FORW1.2>  MobileBusyReason  Fax  active number:   subaddr:   time: 3
<FORW2.0>  NoReplyReason  Voice  active number: ++34637670xxxxxx  subaddr:
time: -1
<FORW2.1>  NoReplyReason  Data  active number:   subaddr:   time: 1073785150
<FORW2.2>  NoReplyReason  Fax  active number:   subaddr:   time: 3
<FORW3.0>  NotReachableReason  Voice  active number: ++34637670xxxxxx  subaddr:
 time: -1
```

```
<FORW3.1>  NotReachableReason  Data  active number:   subaddr:   time:
1073785150
<FORW3.2>  NotReachableReason  Fax  active number:   subaddr:   time: 3
<BATT0>  0 ME is powered by the battery
<BATT1>  50
<BITERR0>  99
<SCA0>
<CSET0>  'GSM' 'HEX' 'IRA' 'PCCP437' 'PCDN' '8859-1'
<CSET1>  'GSM'
<SIG0>  31
```

Ahora la explicacion por partes:

ME son las líneas de Mobile Equipment, de esta forma tenemos...

```
<ME0>  Manufacturer: Nokia Mobile Phones
<ME1>  Model: Nokia 8210
<ME2>  Revision: SW5.26
<ME3>  Serial Number: 350112107500xxx
```

El ME3 es el IMEI. Ni que decir que las xxx las he puesto yo por pura paranoia.

```
<OP0>  Status: current  Long name:    Short name:    Numeric name: 21401
<OP1>  Status: forbidden  Long name: "   Short name: "   Numeric name: 21403
<OP2>  Status: forbidden  Long name: "   Short name: "   Numeric name: 21407
```

El comando OP nos dice qué operadoras estan en la red disponibles. Cual es la que está en uso y cuales están disponibles o prohibidas.
Hay que tener en cuenta que:

21401 Airtel

21407 Movistar

21403 Amena

Las opciones del estado son:

unknown Deconocido

current Usado actualmente

available Disponible

forbidden Están operativos pero prohibidos para usarlos.

```
<CURROP0>  Long name:    Short name:    Numeric name: 21401  Mode: automatic
```

Pues además de decir lo mismo que el OP nos aporta si la selección del operador es automática o manual.

```
<FLSTAT0>  'PS'
<FLSTAT1>  'SC'
<FLSTAT2>  'AO'
<FLSTAT3>  'OI'
<FLSTAT4>  'OX'
<FLSTAT5>  'AI'
<FLSTAT6>  'IR'
<FLSTAT7>  'FD'
<FLCAP0>  'PS' 'SC' 'AO' 'OI' 'OX' 'AI' 'IR' 'AB' 'AG' 'AC' 'FD'
```

Nos muestra el tipo de bloqueos sobre el teléfono que hay disponible.

(man gsmctl)

PS - Código de seguridad del teléfono, en ocasiones por defecto es 12345

SC - PIN

AO - Llamadas salientes restringidas

OI - Llamadas salientes internaciones restringidas

OX - Llamadas salientes internaciones restringidas excepto el pais de casa.

AI - Llamadas entrantes restringidas.

IR - Pirula con el roadmin, no se...

AB - PUK???

AG, AC, FD mirar man...

Luego tenemos:

```
<PW0>  'PS' 5
<PW1>  'SC' 8
<PW2>  'AB' 4
<PW3>  'P2' 8
```

Se supone que son los tamaños de almacenamiento para cada una de las llaves. Lo del 8 no lo tengo muy claro. Parece ser que, como es un móvil de empresa, dispone de dos pins, uno el de la empresa y otro el normal...

```
<FORW0.0>  UnconditionalReason  Voice  active number:   subaddr:   time:
134576720
<FORW0.1>  UnconditionalReason  Data  active number:   subaddr:   time:
1073785150
<FORW0.2>  UnconditionalReason  Fax  active number:   subaddr:   time: 3
<FORW1.0>  MobileBusyReason  Voice  active number:   subaddr:   time: 134576720
<FORW1.1>  MobileBusyReason  Data  active number:   subaddr:   time: 1073785150
<FORW1.2>  MobileBusyReason  Fax  active number:   subaddr:   time: 3
<FORW2.0>  NoReplyReason  Voice  active number: ++34637670xxxxxx  subaddr:
time: -1
<FORW2.1>  NoReplyReason  Data  active number:   subaddr:   time: 1073785150
<FORW2.2>  NoReplyReason  Fax  active number:   subaddr:   time: 3
```

```
<FORW3.0>  NotReachableReason  Voice  active number: ++34637670xxxxxx  subaddr:
 time: -1
<FORW3.1>  NotReachableReason  Data  active number:   subaddr:   time:
1073785150
<FORW3.2>  NotReachableReason  Fax  active number:   subaddr:   time: 3
```

Esto son los desvios.

No sé exactamente muy bién, pero el MobileBusyReason es cuando está ocupado, el NoReplyReason cuando no se contesta y el NotReachableReason cuando está apagado Lo hace para voz, fax y datos.

El number es el número donde se desvía, en este caso el contestador. (las xxx las he puesto por paranoia ;)).

El time es el tiempo que tarda en ponerse. No sé si por segundos, décimas, o por número de tonos, la verdad. Lógicamente me da que -1 es porque no espera nada.

```
<BATT0>  0 ME is powered by the battery
<BATT1>  50
```

Forma de suministro de energia y estado de la batería. Se lee perfectamente, alimentado por la bateria y al 50%.
Otro caso podria ser:

```
<BATT0>  1 ME has a battery connected, but is not powered by it
<BATT1>  70
```

Aquí el teléfono está enchufado a la luz.

```
<BITERR0>  99
```

En caso de agún tipo de error se muestra aquí. Sería un error de 0 a 7 y en caso de 99 es que no es detectable o no hay.

```
<SCA0>
```

Algo del centro de mensajes. Siempre lo he visto vacio.

```
<CSET0>  'GSM' 'HEX' 'IRA' 'PCCP437' 'PCDN' '8859-1
<CSET1>  'GSM'
```

Juegos de caracteres disponibles y cual es el usado.

```
<SIG0>  31
```

Señal de recepción:

0

-113 dBm or less

1

-111 dBm

2...30

-109... -53 dBm (in steps of 2 dBm)

31

-51 dBm or greater

99

not known or not detectable

NOTA: Por cierto, el parametro ALL del gsmctl con el 7110 no me ha funcionado. A la hora de optener los desvíos da un fallo por lo que se corta el ALL (no se si es cosa de este 7110 o general).
Lo que SI se puede hacer es ir llamando parámetro por parámetro: BATT, OP, SIG etc etc...

6.4 Conexión a Internet

6.4.1 Nokia 6150

El Nokia 6150 tiene la particularidad de no tener un módem AT que nos solventase la faena. El módem es mas bien parecido a lo que conocemos como Winmodem. Por llamarlo de alguna forma.

Por otro lado, el stack IRDA no cumple al 100% con las especificaciones. Por lo tanto todo tipo de comunicación via 6150 a traves de IR tiene la particularidad de no entenderse con los estándar.

Todo esto nos sirve de algo?. Pues con el 6150 que he probado yo, absolutamente para nada.

No he conseguido hacer ir ninguno de los programas con el 6150. En el tráfico observamos algo como:

```
12:47:28.642811 xid:cmd ffffffff < fb0c0000 S=6 s=* Nokia 6100 hint=8101 [ PnP
Telephony ] (28)
ff3f0100000cfbffffffff01ff008101004e6f6b6961203631303000 . ? . . . . . . . . . .
. . . . . N o k i a   6 1 0 0 .
12:47:28.702404 snrm:cmd ca=fe pf=1 2b6dcfa5 < fb0c0000 new-ca=46 (32)
ff9300000cfba5cf6d2b4601013e8201078301018401018501fc8601010801ff . . . . . . . .
m + F . . > . . . . . . . . . . . . . . . . . .
```

<

El anuncio de IrCOMM o IrOBEX no aparece. Los tios de gnokii dicen que si lo han hecho ir desde SIR. Pos fale, yo llevo probando unos tres dias y no hay manera santa de que vaya.

Parece ser que a lo mejor funciona con un firm superior al que tenía.

NOTA: el 6150 de las pruebas usaba v5.22

Eso si, no puedo decir lo mismo desde el cable. Con el cable funciona todo de cojones. Por eso puse en lo de "elementos necesarios" que haría falta un cable. Pues bien, con el 6150 va todo, pero con cable.

Lógicamente, como he dicho y no me he equivocado: ¿Cómo has hecho ir el módem aunque sea desde cable si es un winmodem?. Pues efectivamente no puedes enchufarle un wvdial directamente. Los tios de gnokii han creado un demonio (gnokiid) que permite utilizar el módem con comandos AT, como si de uno 100% compatible se tratase.

Aun así el tema de gnokki es algo raro. Al parecer hay una persona que les ha enviado multitud de parches que han sido ignorados. Visto lo visto, lo que han hecho es, como siempre, sacar una nueva rama. ¡Vivan los esfuerzos absurdos!. El programa se llama 'mygnokii' basado al 100% en el original, pero con algunas cosas retocadas... no puedo hablar mucho de esto porque aín estoy de pruebas. En futuras revisiones del artículo se irá completando la información.

La versión que tengo aquí se llama *0.3.3_pre8-gold_2001_11_03*.

Se puede bajar de:

6.4.2 Nokia 7110/8210

La interconexión del portátil con un 7110 o un 8210 es muchísimo mas fácil. Una vez activado el IR en el móvil tan solo hace falta que el wvdial apunte al dispositivo para que funcione. De esta forma en el wvdial.conf ponéis:

```
   Modem = /dev/ircomm0
Baud = 9600
Phone = telefonillo
Username = nombre_De_usuario
Password = contraseña_de_la_conexion
New PPPD = yes
```

Atentos sobre todo a los Baud, he intentado hacer ir ambos móviles, tanto 7110 como 8210 (que se suponen soportan 14400), a 14400 y no ha habido manera de conectar.

Por lo demás, a partir de ese momento es como un módem normal.

NOTA: Firm del 7110 de las pruebas: v4.84.

Chapter 7

Agradecimientos

No puede faltar esta sección por nada del mundo.

Como habré dicho antes, y si no lo he dicho lo digo ahora, no tengo móvil con IR, por lo que he dependido al 100% de otra gente sin los cuales no podría haber hecho ni la mitad de historias.

El primer agradecimiento de todos es para mi padre, XDDD, sin su 8210 a ver como podría haber probado todo esto.

Importantísima la ayuda de mi muy buen amigo Pe-n0. Que puso el 7110, el 6210, el cable de datos, la paciencia y el coche, desde el que hice las pruebas del B (APENDICE B).

Luego he realizado pruebas con teléfonos como el de Txevi (en medio de las clases del CCNA), Romario (en medio de una conferencia de Panda, imaginaros que tostón de conferencia) y Charlie (cuyo móvil me sirvió para comprobar los últimos matices).

Y como no, a la gente que forma Undersec, que me ayuda a tener un respaldo y un toma y daca que tiene como producto final textos así.

Appendix A

Protocolo IrLAP (Apéndice A)

A.1 Introducción

Como he introducido varios fragmentos donde se mostraba el trafico IR, es absurdo no poner un apéndice un poco mas técnico para los interesados en analizar las diferentes comunicaciones....

Lo primero decir que las estaciones que intervienen en la comunicación se pueden poner en dos modos.

NRM - Normal Response Mode NDM - Normal Disconect Mode

Mas o menos sería en modo de conectado y en modo de desconectado.

La secuenciación de las tramas se realiza con dos registros: *NS y NR*. De esta manera es posible seguir un orden en el envío de las diferentes tramas. Estos dos registros están en el campo de control de la trama (Véase mas adelante).

La secuenciación es bastante sencilla.

NS lleva un contador de las tramas que va de 0 a 7.

NR Envía qué trama se espera recibir.

Hay secuencias DIFERENTES para las tramas de tipo *cmd* (que mas adelante explicamos para que se usan) y las de tipo *rsp* (que tambien están explicadas mas adelante).

De esta forma por ejemplo vemos que, independientemente del resto de información de la trama, este ejemplo se explica de la siguiente manera:

```
19:17:56.783218 i:cmd  > ca=1e pf=1 nr=2 ns=3 LM slsap=12 dlsap=00 CONN_CMD (6)
        1f5680120100
         . V . . . .
```

Esta es la trama 3 en modo *CMD* y espero la 2. Seguramente la 2 se perdería.

```
19:17:56.792779 i:rsp  < ca=1e pf=1 nr=4 ns=2 LM slsap=00 dlsap=12 CONN_RSP (6)
        1e9492008100
         . . . . . .
```

Esta es la trama 2 (la que queríamos) y espero la 4.

```
19:17:56.792794 i:cmd  > ca=1e pf=1 nr=3 ns=4 LM slsap=12 dlsap=00
GET_VALUE_BY_CLASS: "IrDA:IrCOMM" "IrDA:TinyTP:LsapSel" (37)
        1f780012840b497244413a4972434f4d4d13497244413a54696e7954503a4c73
         . x . . . . I r D A : I r C O M M . I r D A : T i n y T P : L s
```

Esta es la trama 4 y vuelvo a esperar la 3, pero esta vez es de tipo *RSP*.

```
19:17:56.807427 i:rsp  < ca=1e pf=1 nr=5 ns=3 LM slsap=00 dlsap=12
GET_VALUE_BY_CLASS: Success Integer: 04 (15)
        1eb612008400000100040100000004
         . . . . . . . . . . . . . . . .
```

Aquí tienes la 3 y espero la 5

```
19:17:56.807438 i:cmd  > ca=1e pf=1 nr=4 ns=5 LM slsap=12 dlsap=00 DISC (6)
        1f9a80120201
         . . . . . .
```

Esta es la 5 y espero la 4 pero *RSP*. Y así todo el rato.
Cuando llega a 7, se confirma y se empieza. Ejemplo:

```
19:17:56.834072 i:cmd  > ca=1e pf=1 nr=5 ns=7 LM slsap=10 dlsap=04 TTP
credits=0 (24)
        1fbe041000120001041004000025801101131201042001 0c
         . . . . . . . . . . . . . . % . . . . . . . .   . .
19:17:56.843203 rr:rsp < ca=1e pf=1 nr=0 (2)
        1e11
         . .
19:17:56.843211 rr:cmd > ca=1e pf=1 nr=5 (2)
        1fb1
         . .
19:17:56.855087 i:rsp  < ca=1e pf=1 nr=0 ns=5 LM slsap=04 dlsap=10 TTP credits=1
(32)
        1e1a1004011a010101100400004b0011010312013f13021113140213112101 30
         . . . . . . . . . . . . . . K . . . . . . ? . . . . . . . . . ! . 0
```

Vamos pues a adentrarnos mas en lo que es el contenido de cada trama.

La trama de IrLAP esta formada por un campo address, un campo control y un campo para información.

Los dos primeros tienen una tamaño de 8 bits mientras el ultimo debe ser multiplo de 8.

```
|------------|--------------|------------.............|

Address         Control         Informacion
8 bits          8 bits          N*8 bits
```

A.2 Campo ADDRESS

En lo que es el campo address, el bit de menor peso es el conocido como C/R. Sirve para diferenciar si la trama es un cmd (command) o una rsp (response) Dicho vulgarmente, es para saber si es una trama de petición/comando o si es una trama de contestación/respuesta precisamente a esa petición/comando.

C/R=1 cmd

C/R=0 rsp

00000000= NULL conection

11111111= Broadcast

El resto son direcciones normales

A.3 Campo CONTROL

En el campo control de 8 bits encontramos funciones muy importantes. Por ejemplo el 4º bit es el de P/F.

Bit P/F (Poll/Final): En una comunicación tenemos siempre el que pregunta y el que responde. En lenguaje mas técnico hablamos de la estación primaria (primary) y de la secundaria (secondary). Cuando el bit P/F=1 en la estación primaria entonces es POLL (P). Mientras que si P/F=1 en la secundaria entonces es FINAL (F). POLL exige a la estación secundaria una respuesta o serie de respuestas. Cuando es FINAL entonces es porque es la respuesta o fin de respuestas a una solicitud (POLL). Todo esto nos sirve para saber cuando un paquete es respuesta de otro y no hacernos la picha un lio.

Despues de detenernos en la particularidad del bit P/F, hay que decir que encontramos tres tipos de tramas según su campo de control:

U - Unnumered Format

S - Supervisor

I - Information Format

Vamos uno por uno:

A.4 Tipos de tramas

A.4.1 Tramas U (Unnumered Format)

Es usado para la conexión y desconexión de la comunicación, reportar errores y es posible transferir información en una trama de tipo 'U' cuando la localización de los datos de una secuencia no está comprobada. Vamos, que no se tiene ni puta idea ni se reconoce de qué viene o a donde va lo que estamos enviado. Estas tramas son las se utilizan cuando se reconoce por primera vez un dispositivo.

El famoso:

```
19:17:56.046160 xid:cmd 1a6a9e08 > ffffffff S=6 s=0 (14)
        ff3f01089e6a1affffffff010000
         . ? . . . j . . . . . . . .
19:17:56.136126 xid:cmd 1a6a9e08 > ffffffff S=6 s=1 (14)
        ff3f01089e6a1affffffff010100
         . ? . . . j . . . . . . . .
19:17:56.226120 xid:cmd 1a6a9e08 > ffffffff S=6 s=2 (14)
        ff3f01089e6a1affffffff010200
         . ? . . . j . . . . . . . .
19:17:56.316120 xid:cmd 1a6a9e08 > ffffffff S=6 s=3 (14)
        ff3f01089e6a1affffffff010300
         . ? . . . j . . . . . . . .
19:17:56.400635 xid:rsp 1a6a9e08 < ab580000 S=6 s=3 Nokia 7110 hint=b125 [ PnP
Modem Fax Telephony IrCOMM IrOBEX ] (27)
        febf01000058ab089e6a1a010300b125004e6f6b69612037313130
         . . . . . X . . . j . . . . . % . N o k i a   7 1 1 0
```

son tramas de tipo U. Dicho de otra manera, los XID son tramas de tipo U según su campo de control.

Si os dais cuenta, se cumplen dos cosas. La primera es que está buscando un dispositivo y la segunda es que no tenemos ni puta idea de quien recibe la solicitud.

Ya que estamos os explico los tipos de tramas U (Unnumered Format).

XID (Exchange station IDentification):

Es usado para descubrir dispositivos y para cuando hay conflictos con direcciones. Sirven tanto como solicitud como para respuesta. Lógicamente, la diferencia está en que en su campo address las de solicitud (cmd) tendrán el bit de C/R=1 y las de respuesta C/R=0. Véase el ejemplo anterior.

La estructura de la trama sería:

```
struct xid_frame {
        guint8  caddr; /* Connection address */
        guint8  control;
        guint8  ident; /* Should always be XID_FORMAT */
        guint32 saddr; /* Source device address */
        guint32 daddr; /* Destination device address */
        guint8  flags; /* Discovery flags */
        guint8  slotnr;
        guint8  version;
        guint8  discovery_info[0];
} __attribute__((packed));
```

SNRM (Set Normal Response Mode):

Este control es usado para establecer o reiniciar una conexión. Cuando estamos en medio de una conexión entre dispositivos, la estacion que lo envía (la trama SNRM) se convierte en primaria, mientras que la que lo contesta es la secundaria. Así distinguimos quién llama a quién.
El CAMPO ADDRESS de la trama del SNRM cuando se está realizando la conexión esta colocado todo a 1. Dicho de otra forma, es una petición de difisión con el C/R=1, o sea, como solicitud.
El CAMPO ADDRESS de la la trama del SNRM en una solicitud de desconexión que lleva la direccion del dispositivo a desconectar.

El CAMPO INFORMACIÓN de la trama SNRM cuando se está estableciendo la conexión lleva la dirección de origen y destino y los parámetros para la negociación. En la desconexión el CAMPO INFORMACIÓN no contiene nada.

La estructura sería:

```
struct snrm_frame {
        guint8  caddr;
        guint8  control;
        guint32 saddr;
        guint32 daddr;
        guint8  ncaddr;
        guint8  params[0];
} __attribute__((packed));
```

UA (Unnumered Acknowledgmen):

Es la respuesta a tramas SNRM y DISC. Sirven para confirmar ambas tramas.

Ejemplo:

```
19:17:56.601954 snrm:cmd ca=fe pf=1 1a6a9e08 > ab580000 new-ca=1e (33)
        ff93089e6a1a000058ab1e0102fe0182010183013f84017f8501ff8601070801
         . . . . j . . . X . . . . . . . . . . . ? . . . . . . . . . . .
19:17:56.741781 ua:rsp ca=1e pf=1 1a6a9e08 < ab580000 (31)
        1e73000058ab089e6a1a01013e8201018301028401018501fc860107080107
         . s . . X . . . j . . . > . . . . . . . . . . . . . . . . . . .
```

Estructura:

```
struct ua_frame {
        guint8 caddr;
        guint8 control;

        guint32 saddr; /* Source device address */
        guint32 daddr; /* Dest device address */
        guint8  params[0];
} __attribute__((packed));
```

TEST:

Pues eso, sirve para, sin ir mas lejos, hacer comprobaciones. Sería el supuesto PING en el stack TCP/IP. La distinción entre cmd y rsp lo de siempre, el bit C/R.

Ejemplo:

```
21:01:33.414173 test:cmd ca=0xfe pf=1 00b9fab2 > ffffffff (32)
        fff3b2fab900ffffffff0600ad95ed3bcf5106000001020304050607080 90a0b
         . . . . . . . . . . . . . . . . ; . Q . . . . . . . . . . . . . .
21:01:33.517694 test:rsp ca=0xfe pf=1 a2309075 < 00b9fab2 (32)
```

```
        fef3759030a2b2fab9000600ad95ed3bcf5106000001020304050607080 90a0b
         . . u . 0 . . . . . . . . . . . ; . Q . . . . . . . . . . . . . . .
```

Estructura:

```
struct test_frame {
        guint8 caddr;          /* Connection address */
        guint8 control;
        guint32 saddr;         /* Source device address */
        guint32 daddr;         /* Destination device address */
        guint8 info[0];        /* Information */
} __attribute__((packed));
```

DISC:

Sirve para poner fin a la comunicación. Una vez recibido la secundaria confirma con un UA y se pone en modo A.1 (NDM):

```
19:37:56.976127 disc:cmd > ca=0x1e pf=1 (2)
        1f53
         . S
19:37:57.006146 ua:rsp ca=1e pf=1 1a6a9e08 < ab580000 (10)
        1e73000058ab089e6a1a
         . s . . X . . . j .
```

La clasificación entera de datagramas de tipo U sería, espero no dejarme ninguno:

SNRM, DISC, UI, XID, TEST, RNRM, UA, FRMR, DM, RD

Para mas detalles de cada una remitiros a la especificaciones en:

A.4.2 Tramas S (Supervisor)

RR (Receive Ready):

Confirma la secuencia de tramas recibidas y dice que está listo para recibir nuevas. Puede ser enviado tanto por la estación primaria como con la secundaria. Ejemplo:

```
19:17:56.741839 rr:cmd > ca=1e pf=1 nr=0 (2)
        1f11
         . .
19:17:56.750896 rr:rsp < ca=1e pf=1 nr=0 (2)
        1e11
```

RNR (receive Not Ready):

Enviado por una estación primaria o secundaria. Indica un estado de ocupado, bien por que no quedan mas buffers libres o por lo que sea.
Se envía un RR para indicar el fin de este estado.

REJ (Reject):

Solicita la retransmisión a partir de una trama que se ha perdido o ha llegado con errores.

SREJ (Selective Reject):

Solicita únicamente una trama.

A.4.3 Tramas de Información (I)

Son las únicas tramas en las que se da el caso de que pueden secuenciarse sin necesidad de ningún tipo de interactuación. Cuando se llega a las 7 tramas secuenciales se empieza de nuevo.

El fin de la secuencia se marca con el bit de P/F=1, ya sea por parte de la primaria o la secundaria.

A.5 Campo INFORMACIÓN

Pues sencillamente es el campo que posee toda la información que se quiere tranmitir. Según el tipo de tramas, este campo trae algún dato en su interior o no lo lo trae. Su longitud no está definida pero tiene que ser un múltiplo de 8.

Appendix B

Análisis real (Apéndice B)

El Apéndice B no es mas ni menos que un ejemplo 100% práctico y analizado. Se trata de una conexión del portátil con un 7110 a Internet....con Eresmas mismo. Alla vamos....

Voy a desmenuzar una trama antes que nada.

```
19:18:01.847849 = Hora

i:rsp = Tipo de trama:Flag C/R. Si 1 cmd, si 0 rsp.

< = sentido de la comunicación

ca=1e = ?????

pf=1 = bandera poll/final

nr=1  = trama que se espera

ns=7  = número de trama

LM slsap=04 dlsap=10 TTP credits=1 = nivel superior TinyTP y SAP

(19) = tamaño del campo de información, creo.
```

NOTA: Decir que el contenido de la trama que muestra el irdadump viene en hexadecimal primero y en ascii en la linea inferior y que no muestra todo el contenido del paquete. No es un sniffer.

```
[ Empezamos con las típicas señales de difusión con paquetes U de tipo XID ]

19:17:56.046160 xid:cmd 1a6a9e08 > ffffffff S=6 s=0 (14)
        ff3f01089e6a1affffffff010000
         . ? . . . j . . . . . . . .
19:17:56.136126 xid:cmd 1a6a9e08 > ffffffff S=6 s=1 (14)
        ff3f01089e6a1affffffff010100
         . ? . . . j . . . . . . . .
19:17:56.226120 xid:cmd 1a6a9e08 > ffffffff S=6 s=2 (14)
        ff3f01089e6a1affffffff010200
```

```
         . ? . . . j . . . . . . . .
19:17:56.316120 xid:cmd 1a6a9e08 > ffffffff S=6 s=3 (14)
        ff3f01089e6a1affffffff010300
         . ? . . . j . . . . . . . .
19:17:56.400635 xid:rsp 1a6a9e08 < ab580000 S=6 s=3 Nokia 7110 hint=b125 [ PnP
Modem Fax Telephony IrCOMM IrOBEX ] (27)
        febf01000058ab089e6a1a010300b125004e6f6b69612037313130
         . . . . . X . . . j . . . . . % . N o k i a   7 1 1 0


[ Respuesta del Nokia con otro XID y el bit C/R cambiado ]


19:17:56.406121 xid:cmd 1a6a9e08 > ffffffff S=6 s=4 (14)
        ff3f01089e6a1affffffff010400
         . ? . . . j . . . . . . . .
19:17:56.496120 xid:cmd 1a6a9e08 > ffffffff S=6 s=5 (14)
        ff3f01089e6a1affffffff010500
         . ? . . . j . . . . . . . .
19:17:56.586121 xid:cmd 1a6a9e08 > ffffffff S=6 s=* annapurna hint=8404 [
Computer IrCOMM ] (26)
        ff3f01089e6a1affffffff01ff00840400616e6e617075726e61
         . ? . . . j . . . . . . . . . . a n n a p u r n a

[ Aquí es donde el PC ha visto la contestación del Nokia, antes habia soltado
dos señales de difusión mas. El PC anuncia en el campo I su nombre y el protocolo IrCOMM de
nivel superior ]


19:17:56.601954 snrm:cmd ca=fe pf=1 1a6a9e08 > ab580000 new-ca=1e (33)
        ff93089e6a1a000058ab1e0102fe0182010183013f84017f8501ff8601070801
         . . . . j . . . X . . . . . . . . . . . ? . . . . . . . . . . . .
19:17:56.741781 ua:rsp ca=1e pf=1 1a6a9e08 < ab580000 (31)
        1e73000058ab089e6a1a01013e8201018301028401018501fc860107080107
         . s . . X . . . j . . . > . . . . . . . . . . . . . . . . . . . .

[ Paquete SNRM con la respuesta UA. Ahora se reinician los registros NR Y NS para
empezar la comunicación. Hemos pasado de NDM a NRM ]



19:17:56.741839 rr:cmd > ca=1e pf=1 nr=0 (2)
        1f11
         . .
19:17:56.750896 rr:rsp < ca=1e pf=1 nr=0 (2)
        1e11
         . .

[ Veis, el nr de los dos está a 0. Estamos ya con tramas Supervisor, el PF es el
bit Poll/final. ]
```

```
19:17:56.750911 i:cmd  > ca=1e pf=1 nr=0 ns=0 LM slsap=11 dlsap=00 CONN_CMD (6)
        1f1080110100
         . . . . . .
19:17:56.760470 i:rsp  < ca=1e pf=1 nr=1 ns=0 LM slsap=00 dlsap=11 CONN_RSP (6)
        1e3091008100
         . 0 . . . .
19:17:56.760489 i:cmd  > ca=1e pf=1 nr=1 ns=1 LM slsap=11 dlsap=00
GET_VALUE_BY_CLASS: "IrDA:IrCOMM" "Parameters" (28)
        1f320011840b497244413a4972434f4d4d0a506172616d6574657273
         . 2 . . . . I r D A : I r C O M M . P a r a m e t e r s
19:17:56.775481 i:rsp  < ca=1e pf=1 nr=2 ns=1 LM slsap=00 dlsap=11
GET_VALUE_BY_CLASS: Success N/A (19)
        1e52110084000001000402000600010401010 1
         . R . . . . . . . . . . . . . . . . .
19:17:56.775496 i:cmd  > ca=1e pf=1 nr=2 ns=2 LM slsap=11 dlsap=00 DISC (6)
        1f5480110201
         . T . . . .
19:17:56.783205 rr:rsp < ca=1e pf=1 nr=3 (2)
        1e71
         . q

[ Acabamos de negociar el IrCOMM ]


19:17:56.783218 i:cmd  > ca=1e pf=1 nr=2 ns=3 LM slsap=12 dlsap=00 CONN_CMD (6)
        1f5680120100
         . V . . . .
19:17:56.792779 i:rsp  < ca=1e pf=1 nr=4 ns=2 LM slsap=00 dlsap=12 CONN_RSP (6)
        1e9492008100
         . . . . . .
19:17:56.792794 i:cmd  > ca=1e pf=1 nr=3 ns=4 LM slsap=12 dlsap=00
GET_VALUE_BY_CLASS: "IrDA:IrCOMM" "IrDA:TinyTP:LsapSel" (37)
        1f780012840b497244413a4972434f4d4d13497244413a54696e7954503a4c73
         . x . . . . I r D A : I r C O M M . I r D A : T i n y T P : L s
19:17:56.807427 i:rsp  < ca=1e pf=1 nr=5 ns=3 LM slsap=00 dlsap=12
GET_VALUE_BY_CLASS: Success Integer: 04 (15)
        1eb61200840000010004010000000 4
         . . . . . . . . . . . . . . . .
19:17:56.807438 i:cmd  > ca=1e pf=1 nr=4 ns=5 LM slsap=12 dlsap=00 DISC (6)
        1f9a80120201
         . . . . . .

19:17:56.815513 rr:rsp < ca=1e pf=1 nr=6 (2)
        1ed1
         . .

[ Hemos negociado el Tiny TP con el SAP. Fijaros que ahora los paquetes
```

```
incorporarán mas info ]


19:17:56.815522 i:cmd  > ca=1e pf=1 nr=4 ns=6 LM slsap=10 dlsap=04 CONN_CMD TTP
credits=0(7)
        1f9c841001000e
         . . . . . . .
19:17:56.825177 i:rsp  < ca=1e pf=1 nr=7 ns=4 LM slsap=04 dlsap=10 CONN_RSP TTP
credits=0(7)
        1ef89004810005
         . . . . . . .

[ Habría que mirar especificaciones del TinyTP para entender este par de
paquetes ]


19:17:56.825196 rr:cmd > ca=1e pf=1 nr=5 (2)
        1fb1
         . .
19:17:56.834064 rr:rsp < ca=1e pf=1 nr=7 (2)
        1ef1
         . .
19:17:56.834072 i:cmd  > ca=1e pf=1 nr=5 ns=7 LM slsap=10 dlsap=04 TTP credits=0
(24)
        1fbe0410001200010410040000258011011312010420010c
         . . . . . . . . . . . . . . % . . . . . . . .   . .
19:17:56.843203 rr:rsp < ca=1e pf=1 nr=0 (2)
        1e11
         . .
19:17:56.843211 rr:cmd > ca=1e pf=1 nr=5 (2)
        1fb1
         . .
19:17:56.855087 i:rsp  < ca=1e pf=1 nr=0 ns=5 LM slsap=04 dlsap=10 TTP credits=1
(32)
        1e1a1004011a010101100400004b0011010312013f13021113140213112101 30
         . . . . . . . . . . . . . . K . . . . . . ? . . . . . . . . . ! . 0
19:17:56.855115 rr:cmd > ca=1e pf=1 nr=6 (2)
        1fd1
         . .
19:17:56.866899 i:rsp  < ca=1e pf=1 nr=0 ns=6 LM slsap=04 dlsap=10 TTP credits=0
(9)
        1e1c10040003210130
         . . . . . . ! . 0
19:17:56.866918 rr:cmd > ca=1e pf=1 nr=7 (2)
        1ff1
         . .
19:17:56.875511 rr:rsp < ca=1e pf=1 nr=0 (2)
        1e11
         . .
19:17:56.916120 rr:cmd > ca=1e pf=1 nr=7 (2)
```

```
        1ff1
         . .
19:17:56.926282 rr:rsp < ca=1e pf=1 nr=0 (2)
        1e11
         . .
19:17:57.026120 rr:cmd > ca=1e pf=1 nr=7 (2)
        1ff1
         . .
19:17:57.037053 rr:rsp < ca=1e pf=1 nr=0 (2)
        1e11
         . .
19:17:57.186119 rr:cmd > ca=1e pf=1 nr=7 (2)
        1ff1
         . .
19:17:57.193977 rr:rsp < ca=1e pf=1 nr=0 (2)
        1e11
         . .
19:17:57.386119 rr:cmd > ca=1e pf=1 nr=7 (2)
        1ff1
         . .
19:17:57.397065 rr:rsp < ca=1e pf=1 nr=0 (2)
        1e11
         . .
19:17:57.646119 rr:cmd > ca=1e pf=1 nr=7 (2)
        1ff1
         . .

[ Mala comunicación. Mil perdones, estoy en el coche haciendo la prueba]


19:18:01.686119 rr:cmd > ca=1e pf=1 nr=7 (2)
        1ff1
         . .
19:18:01.694030 rr:rsp < ca=1e pf=1 nr=0 (2)
        1e11
         . .
19:18:01.776296 i:cmd  > ca=1e pf=1 nr=7 ns=0 LM slsap=10 dlsap=04 TTP credits=2
(7)
        1ff0041002000d
         . . . . . . .
19:18:01.786340 rr:rsp < ca=1e pf=1 nr=1 (2)
        1e31
         . 1
19:18:01.786348 rr:cmd > ca=1e pf=1 nr=7 (2)
        1ff1
         . .
19:18:01.795571 rr:rsp < ca=1e pf=1 nr=1 (2)
        1e31
         . 1
19:18:01.836119 rr:cmd > ca=1e pf=1 nr=7 (2)
```

```
        1ff1
         . .
19:18:01.847849 i:rsp  < ca=1e pf=1 nr=1 ns=7 LM slsap=04 dlsap=10 TTP credits=1
(19)
        1e3e100401000d0a4f4b0d0a0d0a4f4b0d0a0d
         . > . . . . . . O K . . . . O K . . .
19:18:01.847866 rr:cmd > ca=1e pf=1 nr=0 (2)
        1f11
         . .
19:18:01.855573 rr:rsp < ca=1e pf=1 nr=1 (2)
        1e31
         . 1
19:18:01.855595 i:cmd  > ca=1e pf=1 nr=0 ns=1 LM slsap=10 dlsap=04 DISC (6)
        1f1284100201
         . . . . . .
19:18:01.864802 rr:rsp < ca=1e pf=1 nr=2 (2)
        1e51
         . Q

[ MMMMm....prometí que era real el registro, así que ahora se ve que ha fallado algo
porque decide volver a empezar ]


19:18:01.864811 i:cmd  > ca=1e pf=1 nr=0 ns=2 LM slsap=14 dlsap=00 CONN_CMD (6)
        1f1480140100
         . . . . . .
19:18:01.874376 i:rsp  < ca=1e pf=1 nr=3 ns=0 LM slsap=00 dlsap=14 CONN_RSP (6)
        1e7094008100
         . p . . . .
19:18:01.874394 i:cmd  > ca=1e pf=1 nr=1 ns=3 LM slsap=14 dlsap=00
GET_VALUE_BY_CLASS: "IrDA:IrCOMM" "Parameters" (28)
        1f360014840b497244413a4972434f4d4d0a506172616d6574657273
         . 6 . . . . I r D A : I r C O M M . P a r a m e t e r s
19:18:01.889388 i:rsp  < ca=1e pf=1 nr=4 ns=1 LM slsap=00 dlsap=14
GET_VALUE_BY_CLASS: Success N/A (19)
        1e92140084000001000402000600010401010 1
         . . . . . . . . . . . . . . . . . . .
19:18:01.889401 i:cmd  > ca=1e pf=1 nr=2 ns=4 LM slsap=14 dlsap=00 DISC (6)
        1f5880140201
         . X . . . .
19:18:01.897110 rr:rsp < ca=1e pf=1 nr=5 (2)
        1eb1
         . .
19:18:01.897121 i:cmd  > ca=1e pf=1 nr=2 ns=5 LM slsap=15 dlsap=00 CONN_CMD (6)
        1f5a80150100
         . Z . . . .
19:18:01.906683 i:rsp  < ca=1e pf=1 nr=6 ns=2 LM slsap=00 dlsap=15 CONN_RSP (6)
        1ed495008100
         . . . . . .
19:18:01.906694 i:cmd  > ca=1e pf=1 nr=3 ns=6 LM slsap=15 dlsap=00
```

```
GET_VALUE_BY_CLASS: "IrDA:IrCOMM" "IrDA:TinyTP:LsapSel" (37)
        1f7c0015840b497244413a4972434f4d4d13497244413a54696e7954503a4c73
         . | . . . . I r D A : I r C O M M . I r D A : T i n y T P : L s
19:18:01.921331 i:rsp  < ca=1e pf=1 nr=7 ns=3 LM slsap=00 dlsap=15
GET_VALUE_BY_CLASS: Success Integer: 04 (15)
        1ef615008400000100040100000004
         . . . . . . . . . . . . . . . .
19:18:01.921341 i:cmd  > ca=1e pf=1 nr=4 ns=7 LM slsap=15 dlsap=00 DISC (6)
        1f9e80150201
         . . . . . .
19:18:01.929417 rr:rsp < ca=1e pf=1 nr=0 (2)
        1e11
         . .
19:18:01.929426 i:cmd  > ca=1e pf=1 nr=4 ns=0 LM slsap=13 dlsap=04 CONN_CMD TTP
credits=0(7)
        1f90841301000e
         . . . . . . .
19:18:01.939083 i:rsp  < ca=1e pf=1 nr=1 ns=4 LM slsap=04 dlsap=13 CONN_RSP TTP
credits=0(7)
        1e389304810005
         . 8 . . . . .
19:18:01.939103 rr:cmd > ca=1e pf=1 nr=5 (2)
        1fb1
         . .
19:18:01.947880 rr:rsp < ca=1e pf=1 nr=1 (2)
        1e31
         . 1
19:18:01.947888 i:cmd  > ca=1e pf=1 nr=5 ns=1 LM slsap=13 dlsap=04 TTP credits=0
(24)
        1fb204130012000104100400002580110113120104200l0c
         . . . . . . . . . . . . . % . . . . . . .   . .
19:18:01.957111 rr:rsp < ca=1e pf=1 nr=2 (2)
        1e51
         . Q
19:18:01.957119 i:cmd  > ca=1e pf=1 nr=5 ns=2 LM slsap=13 dlsap=04 TTP credits=0
(11)
        1fb4041300000d0d0d0d0d
         . . . . . . . . . . .
19:18:01.968995 i:rsp  < ca=1e pf=1 nr=3 ns=5 LM slsap=04 dlsap=13 TTP credits=1
(32)
        1e7a1304011a010101100400004b0011010312013f13021113140213112l0130
         . z . . . . . . . . . . . . K . . . . . ? . . . . . . . . ! . 0
19:18:01.969024 i:cmd  > ca=1e pf=1 nr=6 ns=3 LM slsap=13 dlsap=04 TTP credits=0
(21)
        1fd60413000f100400002580110113120104200100
         . . . . . . . . . . % . . . . . .   . .
19:18:01.980806 i:rsp  < ca=1e pf=1 nr=4 ns=6 LM slsap=04 dlsap=13 TTP credits=0
(9)
        1e9c13040003210130
         . . . . . . ! . 0
```

```
19:18:01.980821 rr:cmd > ca=1e pf=1 nr=7 (2)
        1ff1
         . .

[ Parece que ahora va bien. Las banderas de slsap y dlsap están por encima del
IrLAP. O son del Tiny TP a nivel de transporte o pertenecen al SAP también en
ese nivel. Vamos a observar estas tramas que vienen ]


19:18:01.989420 rr:rsp < ca=1e pf=1 nr=4 (2)
        1e91
         . .
19:18:02.036120 rr:cmd > ca=1e pf=1 nr=7 (2)
        1ff1
         . .
19:18:02.045604 i:rsp  < ca=1e pf=1 nr=4 ns=7 LM slsap=04 dlsap=13 TTP credits=2
(11)
        1e9e130402000d0d0d0d0d
         . . . . . .
19:18:02.045616 rr:cmd > ca=1e pf=1 nr=0 (2)
        1f11
         . .
19:18:02.054036 rr:rsp < ca=1e pf=1 nr=4 (2)
        1e91
         . .
19:18:02.096119 rr:cmd > ca=1e pf=1 nr=0 (2)
        1f11
         . .
19:18:02.104804 rr:rsp < ca=1e pf=1 nr=4 (2)
        1e91
         . .
19:18:02.196119 rr:cmd > ca=1e pf=1 nr=0 (2)
        1f11
         . .
19:18:02.206346 rr:rsp < ca=1e pf=1 nr=4 (2)
        1e91
         . .
19:18:02.356119 rr:cmd > ca=1e pf=1 nr=0 (2)
        1f11
         . .
19:18:02.363269 rr:rsp < ca=1e pf=1 nr=4 (2)
        1e91
         . .
19:18:02.456140 i:cmd  > ca=1e pf=1 nr=0 ns=4 LM slsap=13 dlsap=04 TTP credits=3
(21)
        1f180413030f10040000258011011312013f20010c
         . . . . . % . . . . . ?   . .
19:18:02.469426 rr:rsp < ca=1e pf=1 nr=5 (2)
        1eb1
         . .
```

```
19:18:02.469434 rr:cmd > ca=1e pf=1 nr=0 (2)
        1f11
         . .
19:18:02.478655 rr:rsp < ca=1e pf=1 nr=5 (2)
        1eb1
         . .
19:18:02.478663 i:cmd  > ca=1e pf=1 nr=0 ns=5 LM slsap=13 dlsap=04 TTP credits=0
(11)
        1f1a041300000d0d0d0d0d
         . . . . . . . . . . .
19:18:02.487886 rr:rsp < ca=1e pf=1 nr=6 (2)
        1ed1
         . .
19:18:02.487894 rr:cmd > ca=1e pf=1 nr=0 (2)
        1f11
         . .
19:18:02.497117 rr:rsp < ca=1e pf=1 nr=6 (2)
        1ed1
         . .
19:18:02.546119 rr:cmd > ca=1e pf=1 nr=0 (2)
        1f11
         . .
19:18:02.557637 i:rsp  < ca=1e pf=1 nr=6 ns=0 LM slsap=04 dlsap=13 TTP credits=2
(11)
        1ed0130402000d0d0d0d0d
         . . . . . . . . . . .
19:18:02.557648 rr:cmd > ca=1e pf=1 nr=1 (2)
        1f31
         . 1
19:18:02.566348 rr:rsp < ca=1e pf=1 nr=6 (2)
        1ed1
         . .
19:18:02.616119 rr:cmd > ca=1e pf=1 nr=1 (2)
        1f31
         . 1
19:18:02.626349 rr:rsp < ca=1e pf=1 nr=6 (2)
        1ed1
         . .
19:18:02.656234 i:cmd  > ca=1e pf=1 nr=1 ns=6 LM slsap=13 dlsap=04 TTP credits=1
(10)
        1f3c0413010041545a0d
         . < . . . . A T Z .
19:18:02.667980 rr:rsp < ca=1e pf=1 nr=7 (2)
        1ef1
         . .
```

[<Kiroooooooopa! este ATZ me suena..... inicialización del módem]

```
19:18:02.667988 rr:cmd > ca=1e pf=1 nr=1 (2)
        1f31
         . 1
19:18:02.677736 i:rsp  < ca=1e pf=1 nr=7 ns=1 LM slsap=04 dlsap=13 TTP credits=1
(9)
        1ef213040103210112
         . . . . . ! . .
19:18:02.677750 rr:cmd > ca=1e pf=1 nr=2 (2)
        1f51
         . Q
19:18:02.686442 rr:rsp < ca=1e pf=1 nr=7 (2)
        1ef1
         . .
19:18:02.736119 rr:cmd > ca=1e pf=1 nr=2 (2)
        1f51
         . Q
19:18:02.747584 i:rsp  < ca=1e pf=1 nr=7 ns=2 LM slsap=04 dlsap=13 TTP credits=0
(16)
        1ef41304000041545a0d0d0a4f4b0d0a
         . . . . . A T Z . . . O K . .


[ El módem contesta, todo OK.]


19:18:02.747595 rr:cmd > ca=1e pf=1 nr=3 (2)
        1f71
         . q
19:18:02.755672 rr:rsp < ca=1e pf=1 nr=7 (2)
        1ef1
         . .
19:18:02.796119 rr:cmd > ca=1e pf=1 nr=3 (2)
        1f71
         . q
19:18:02.806442 rr:rsp < ca=1e pf=1 nr=7 (2)
        1ef1
         . .
19:18:02.846547 i:cmd  > ca=1e pf=1 nr=3 ns=7 LM slsap=13 dlsap=04 TTP credits=2
(21)
        1f7e0413020041544454203930383235303235300d
         . ~ . . . . A T D T   9 0 8 2 5 0 2 5 0 .

[ Marcamooooooooos. >Eresmas?....Atención al detalle de a continuación, que me
sirve para explicar algo muy bien. ]


19:18:02.857121 rr:rsp < ca=1e pf=1 nr=0 (2)
        1e11
```

```
         . .
19:18:02.857129 rr:cmd > ca=1e pf=1 nr=3 (2)
        1f71
         . q
19:18:02.866351 rr:rsp < ca=1e pf=1 nr=0 (2)
        1e11
         . .
19:18:02.916119 rr:cmd > ca=1e pf=1 nr=3 (2)
        1f71
         . q
19:18:02.928028 i:rsp  < ca=1e pf=1 nr=0 ns=3 LM slsap=04 dlsap=13 TTP credits=1
(21)
        1e161304010041544454203930383235303235300d
         . . . . . . A T D T   9 0 8 2 5 0 2 5 0 .

[Aquí es donde uno se da cuenta que el infrarrojo falla mas que una escopeta de
feria. Normal, tengo el portátil encima de mis piernas, el móvil a un lado
apollado en el posabrazos del coche y estoy en un parking XDDDD.
Analizo la secuencia, fijaros en los nr ns. Lo represento como un cuento:
el primer ATDT está en la trama 7 y espera la 3.
El otro, que creo que está sordo, nos contesta con la cero: Hola, soy 0.
No no melón, espero 3. (que sordo esta el cabrón)
Si, je, soy cero.
La puta, espero el 3!!!! (cago en la leche que soooordo)
Hostias, mandando la 3. ]

19:18:02.928042 rr:cmd > ca=1e pf=1 nr=4 (2)
        1f91
         . .
19:18:02.935584 rr:rsp < ca=1e pf=1 nr=0 (2)
        1e11
         . .
19:18:02.976119 rr:cmd > ca=1e pf=1 nr=4 (2)
        1f91
         . .
19:18:02.986362 rr:rsp < ca=1e pf=1 nr=0 (2)
        1e11
         . .

[ y mas rr:cmp, rr:rsp seguidos, es que estoy en medio del coche sorry ]



19:18:07.256119 rr:cmd > ca=1e pf=1 nr=4 (2)
        1f91
         . .
19:18:07.264147 i:rsp  < ca=1e pf=1 nr=0 ns=4 LM slsap=04 dlsap=13 TTP credits=0
(9)
        1e1813040003210132
         . . . . . . ! . 2
```

```
19:18:07.264172 rr:cmd > ca=1e pf=1 nr=5 (2)
        1fb1
         . .
19:18:07.274087 i:rsp  < ca=1e pf=1 nr=0 ns=5 LM slsap=04 dlsap=13 TTP credits=0
(17)
        1e1a130400000d0a434152524945520d0a
         . . . . . . . . C A R R I E R . .


[ Ale, ya tenemos línea ]



19:18:07.274101 rr:cmd > ca=1e pf=1 nr=6 (2)
        1fd1
         . .
19:18:07.281991 rr:rsp < ca=1e pf=1 nr=0 (2)
        1e11
         . .
19:18:07.326119 rr:cmd > ca=1e pf=1 nr=6 (2)
        1fd1
         . .
19:18:07.337374 rr:rsp < ca=1e pf=1 nr=0 (2)
        1e11
         . .
19:18:07.436119 rr:cmd > ca=1e pf=1 nr=6 (2)
        1fd1
         . .
19:18:07.443529 rr:rsp < ca=1e pf=1 nr=0 (2)
        1e11
         . .
19:18:07.586119 rr:cmd > ca=1e pf=1 nr=6 (2)
        1fd1
         . .

[ ............ y mas todavía de rr:cdm, rr:rsp ..........]



19:18:23.193761 rr:rsp < ca=1e pf=1 nr=0 (2)
        1e11
         . .
19:18:23.686132 rr:cmd > ca=1e pf=1 nr=6 (2)
        1fd1
         . .
19:18:23.697464 i:rsp  < ca=1e pf=1 nr=0 ns=6 LM slsap=04 dlsap=13 TTP credits=0
(9)
        1e1c130400032101b8
         . . . . . ! . .
19:18:23.697516 rr:cmd > ca=1e pf=1 nr=7 (2)
```

```
        1ff1
         . .
19:18:23.707839 i:rsp  < ca=1e pf=1 nr=0 ns=7 LM slsap=04 dlsap=13 TTP credits=0
(22)
        1e1e130400000d0a434f4e4e45435420393630300d0a
         . . . . . . . . C O N N E C T   9 6 0 0 . .

[ Conectamos, cómo no..... a 9600 ]


19:18:23.707890 rr:cmd > ca=1e pf=1 nr=0 (2)
        1f11
         . .
19:18:23.715302 rr:rsp < ca=1e pf=1 nr=0 (2)
        1e11
         . .
19:18:23.756149 rr:cmd > ca=1e pf=1 nr=0 (2)
        1f11
         . .
19:18:23.766081 rr:rsp < ca=1e pf=1 nr=0 (2)
        1e11
         . .
19:18:23.856130 rr:cmd > ca=1e pf=1 nr=0 (2)
        1f11
         . .
19:18:23.867616 rr:rsp < ca=1e pf=1 nr=0 (2)
        1e11
         . .
19:18:24.016147 rr:cmd > ca=1e pf=1 nr=0 (2)
        1f11
         . .

[ ......... nada interesante hasta que ..............]



19:18:26.376135 rr:cmd > ca=1e pf=1 nr=1 (2)
        1f31
         . 1
19:18:26.387650 rr:rsp < ca=1e pf=1 nr=4 (2)
        1e91
         . .
19:18:26.536129 rr:cmd > ca=1e pf=1 nr=1 (2)
        1f31
         . 1
19:18:26.551641 i:rsp  < ca=1e pf=1 nr=4 ns=1 LM slsap=04 dlsap=13 TTP credits=0
(60)
        1e92130400007eff7d23c0217d212e7d207d397d227d267d207d2a7d207d207d
         . . . . . . ~ . } # . ! } ! . }   } 9 } " } & }   } * }   }   }
```

```
[ Empieza la negociación del PPP. Joer que lento es esto. ]


19:18:26.551681 rr:cmd > ca=1e pf=1 nr=2 (2)
        1f51
         . Q
19:18:26.563121 rr:rsp < ca=1e pf=1 nr=4 (2)
        1e91
         . .
19:18:26.606136 rr:cmd > ca=1e pf=1 nr=2 (2)
        1f51
         . Q
19:18:26.613803 rr:rsp < ca=1e pf=1 nr=4 (2)
        1e91
         . .
19:18:26.706139 rr:cmd > ca=1e pf=1 nr=2 (2)
        1f51
         . Q
19:18:26.715346 rr:rsp < ca=1e pf=1 nr=4 (2)
        1e91
         . .
19:18:26.856140 rr:cmd > ca=1e pf=1 nr=2 (2)
        1f51
         . Q
19:18:26.867657 rr:rsp < ca=1e pf=1 nr=4 (2)
        1e91
         . .
19:18:26.976157 i:cmd  > ca=1e pf=1 nr=2 ns=4 LM slsap=13 dlsap=04 TTP credits=1
(24)
        1f580413011220010c10040000258011011312013f20010c
         . X . . . .   . . . . . % . . . . . ?   . .
19:18:26.987659 rr:rsp < ca=1e pf=1 nr=5 (2)
        1eb1
         . .
19:18:26.987696 rr:cmd > ca=1e pf=1 nr=2 (2)
        1f51
         . Q
19:18:26.996889 rr:rsp < ca=1e pf=1 nr=5 (2)
        1eb1
         . .
19:18:27.032719 i:cmd  > ca=1e pf=1 nr=2 ns=5 LM slsap=13 dlsap=04 TTP credits=0
(53)
        1f5a041300007eff7d23c0217d217d217d207d347d227d267d207d207d207d20
         . Z . . . . ~ . } # . ! } ! } ! }   } 4 } " } & }   }   }   }


[ Siiiigueeee negociando ]
```

```
19:18:27.047662 rr:rsp < ca=1e pf=1 nr=6 (2)
        1ed1
         . .
19:18:27.047699 rr:cmd > ca=1e pf=1 nr=2 (2)
        1f51
         . Q
19:18:27.056892 rr:rsp < ca=1e pf=1 nr=6 (2)
        1ed1
         . .
19:18:27.106148 rr:cmd > ca=1e pf=1 nr=2 (2)
        1f51
         . Q
19:18:27.116891 rr:rsp < ca=1e pf=1 nr=6 (2)
        1ed1
         . .
19:18:27.216149 rr:cmd > ca=1e pf=1 nr=2 (2)
        1f51
         . Q
19:18:27.227666 rr:rsp < ca=1e pf=1 nr=6 (2)
        1ed1
         . .
19:18:27.376140 rr:cmd > ca=1e pf=1 nr=2 (2)
        1f51
         . Q
19:18:27.384587 rr:rsp < ca=1e pf=1 nr=6 (2)
        1ed1
         . .
19:18:27.576136 rr:cmd > ca=1e pf=1 nr=2 (2)
        1f51
         . Q
19:18:27.587668 rr:rsp < ca=1e pf=1 nr=6 (2)
        1ed1
         . .
19:18:27.836144 rr:cmd > ca=1e pf=1 nr=2 (2)
        1f51
         . Q
19:18:27.852402 i:rsp  < ca=1e pf=1 nr=6 ns=2 LM slsap=04 dlsap=13 TTP credits=2
(53)
        1ed4130402007eff7d23c0217d227d217d207d347d227d267d207d207d207d20
         . . . . . ~ . } # . ! } " } ! }   } 4 } " } & }   }   }   }

[ ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ ]

19:18:27.852467 rr:cmd > ca=1e pf=1 nr=3 (2)
        1f71
         . q
```

```
19:18:27.859983 rr:rsp < ca=1e pf=1 nr=6 (2)
        1ed1
         . .
19:18:27.906142 rr:cmd > ca=1e pf=1 nr=3 (2)
        1f71
         . q
19:18:27.915364 rr:rsp < ca=1e pf=1 nr=6 (2)
        1ed1
         . .
19:18:28.006145 rr:cmd > ca=1e pf=1 nr=3 (2)
        1f71
         . q
19:18:28.016902 rr:rsp < ca=1e pf=1 nr=6 (2)
        1ed1
         . .
19:18:28.166143 rr:cmd > ca=1e pf=1 nr=3 (2)
        1f71
         . q
19:18:28.173830 rr:rsp < ca=1e pf=1 nr=6 (2)
        1ed1
         . .
19:18:28.366151 rr:cmd > ca=1e pf=1 nr=3 (2)
        1f71
         . q
19:18:28.376908 rr:rsp < ca=1e pf=1 nr=6 (2)
        1ed1
         . .
19:18:28.626143 rr:cmd > ca=1e pf=1 nr=3 (2)
        1f71
         . q
19:18:28.642251 i:rsp  < ca=1e pf=1 nr=6 ns=3 LM slsap=04 dlsap=13 TTP credits=0
(59)
        1ed6130400007eff7d23c0217d212f7d207d397d227d267d207d2a7d207d207d
         . . . . . ~ . } # . ! } ! / }   } 9 } " } & }   } * }   }   }
19:18:28.642294 rr:cmd > ca=1e pf=1 nr=4 (2)
        1f91
         . .
19:18:28.653836 rr:rsp < ca=1e pf=1 nr=6 (2)
        1ed1
         . .
19:18:28.653876 i:cmd  > ca=1e pf=1 nr=4 ns=6 LM slsap=13 dlsap=04 TTP credits=2
(59)
        1f9c041302007eff7d23c0217d222f7d207d397d227d267d207d2a7d207d207d
         . . . . . ~ . } # . ! } " / }   } 9 } " } & }   } * }   }   }
19:18:28.672387 rr:rsp < ca=1e pf=1 nr=7 (2)
        1ef1
         . .
19:18:28.672420 rr:cmd > ca=1e pf=1 nr=4 (2)
        1f91
         . .
```

```
19:18:28.681615 rr:rsp < ca=1e pf=1 nr=7 (2)
        1ef1
         . .
19:18:28.726135 rr:cmd > ca=1e pf=1 nr=4 (2)
        1f91
         . .
19:18:28.737005 rr:rsp < ca=1e pf=1 nr=7 (2)
        1ef1
         . .

[ .......... blablabla y .........]


19:18:29.480099 rr:rsp < ca=1e pf=1 nr=7 (2)
        1ef1
         . .
19:18:29.480149 i:cmd  > ca=1e pf=1 nr=5 ns=7 LM slsap=13 dlsap=04 TTP credits=1
(42)
        1fbe04130100c2230260001f1017175ff6dded3f92d9f17f50069944e3747540
         . . . . . . . # . ‘ . . . . . _ . . . ? . . . . P . . D . t u @


[ Páralo POOOOOOOOOL. Aquí, aunque parezca mentira, estoy enviado la contraseña, el
famoso tu@eresmas ]


19:18:29.493849 rr:rsp < ca=1e pf=1 nr=0 (2)
        1e11
         . .
19:18:29.493891 rr:cmd > ca=1e pf=1 nr=5 (2)
        1fb1
         . .
19:18:29.503079 rr:rsp < ca=1e pf=1 nr=0 (2)
        1e11
         . .


[ ..... ZZ ZZ ZZ zz zz zz zz ZZ ZZ ZZ .......]



19:20:38.575350 rr:rsp < ca=1e pf=1 nr=0 (2)
        1e11
         . .
19:20:38.816130 rr:cmd > ca=1e pf=1 nr=6 (2)
        1fd1
         . .
19:20:38.824588 rr:rsp < ca=1e pf=1 nr=0 (2)
        1e11
         . .
```

```
19:20:39.116129 rr:cmd > ca=1e pf=1 nr=6 (2)
        1fd1
         . .
19:20:39.130874 i:rsp  < ca=1e pf=1 nr=0 ns=6 LM slsap=04 dlsap=13 TTP credits=1
(72)
        1e1c130401007e214500003ca79240002e061a73d8ef23653e52501000508000
         . . . . . ~ ! E . . < . . @ . . . . s . . # e > R P . . P . .
19:20:39.130928 rr:cmd > ca=1e pf=1 nr=7 (2)
        1ff1
         . .
19:20:39.138443 rr:rsp < ca=1e pf=1 nr=0 (2)
        1e11
         . .
19:20:39.138475 i:cmd  > ca=1e pf=1 nr=7 ns=0 LM slsap=13 dlsap=04 TTP credits=1
(63)
        1ff0041301007e2145000034dd6940004006d2a33e525010d8ef236580000050
         . . . . . ~ ! E . . 4 . i @ . @ . . . > R P . . . # e . . . P
19:20:39.152284 rr:rsp < ca=1e pf=1 nr=1 (2)
        1e31
         . 1
19:20:39.152309 i:cmd  > ca=1e pf=1 nr=7 ns=1 LM slsap=13 dlsap=04 TTP credits=0
(130)
        1ff2041300002145000163dd6a40004006d1733e525010d8ef23658000005085
         . . . . . ! E . . c . j @ . @ . . s > R P . . . # e . . . P .
19:20:39.170743 rr:rsp < ca=1e pf=1 nr=2 (2)
        1e51
         . Q


[ Abrimos el navegador yyyyyyyyyyy..... ]


19:20:39.170772 i:cmd  > ca=1e pf=1 nr=7 ns=2 LM slsap=13 dlsap=04 TTP credits=0
(130)
        1ff4041300003436360d0a486f73743a207777772e676f6f676c652e636f6d0d
         . . . . . 4 6 6 . . H o s t :   w w w . g o o g l e . c o m .
19:20:39.189205 rr:rsp < ca=1e pf=1 nr=3 (2)
        1e71
         . q
19:20:39.189232 i:cmd  > ca=1e pf=1 nr=7 ns=3 LM slsap=13 dlsap=04 TTP credits=0
(117)
        1ff604130000742d4c616e67756167653a20656e0d0a4163636570742d456e63
         . . . . . t - L a n g u a g e :   e n . . A c c e p t - E n c
19:20:39.207918 i:rsp  < ca=1e pf=1 nr=4 ns=7 LM slsap=04 dlsap=13 TTP credits=4
(5)
        1e9e130404
         . . . . .


[ jejejejee, efectivamente buscamos algo en google.com, >por qué no? ]
```

```
19:20:39.207953 rr:cmd > ca=1e pf=1 nr=0 (2)
        1f11
         . .
19:20:39.216901 rr:rsp < ca=1e pf=1 nr=4 (2)
        1e91
         . .
19:20:39.266138 rr:cmd > ca=1e pf=1 nr=0 (2)
        1f11
         . .


[ ......un poco lento si es si....ZZZZZ ....]


19:20:40.695991 rr:cmd > ca=1e pf=1 nr=2 (2)
        1f51
         . Q
19:20:40.707686 rr:rsp < ca=1e pf=1 nr=4 (2)
        1e91
         . .
19:20:40.756128 rr:cmd > ca=1e pf=1 nr=2 (2)
        1f51
         . Q
19:20:40.779123 i:rsp  < ca=1e pf=1 nr=4 ns=2 LM slsap=04 dlsap=13 TTP credits=0
(130)
        1e94130400000d0a436f6e74656e742d547970653a20746578742f68746d6c0d
         . . . . . C o n t e n t - T y p e :   t e x t / h t m l .
19:20:40.779180 rr:cmd > ca=1e pf=1 nr=3 (2)
        1f71
         . q
19:20:40.790771 rr:rsp < ca=1e pf=1 nr=4 (2)
        1e91
         . .
19:20:40.790805 i:cmd  > ca=1e pf=1 nr=3 ns=4 LM slsap=13 dlsap=04 TTP credits=3
(63)
        1f78041303007e2145000034dd6b40004006d2a13e525010d8ef236580000050
         . x . . . . ~ ! E . . 4 . k @ . @ . . . > R P . . . # e . . . P
19:20:40.804613 rr:rsp < ca=1e pf=1 nr=5 (2)
        1eb1
         . .
19:20:40.804645 rr:cmd > ca=1e pf=1 nr=3 (2)
        1f71
         . q
19:20:40.813846 rr:rsp < ca=1e pf=1 nr=5 (2)
        1eb1
         . .
19:20:40.856141 rr:cmd > ca=1e pf=1 nr=3 (2)
```

```
        1f71
         . q
19:20:40.875959 i:rsp  < ca=1e pf=1 nr=5 ns=3 LM slsap=04 dlsap=13 TTP credits=1
(130)
        1eb6130401001996d54c0000dd213c68746d6c3e3c686561643e3c4d45544120
         . . . . . . . . . . L . . . ! < h t m l > < h e a d > < M E T A
19:20:40.876038 rr:cmd > ca=1e pf=1 nr=4 (2)
        1f91
         . .
19:20:40.887700 rr:rsp < ca=1e pf=1 nr=5 (2)
        1eb1
         . .
19:20:40.936128 rr:cmd > ca=1e pf=1 nr=4 (2)
        1f91
         . .
19:20:40.963891 i:rsp  < ca=1e pf=1 nr=5 ns=4 LM slsap=04 dlsap=13 TTP credits=0
(130)
        1eb8130400000a626f64792c74642c612c702c2e687b666f6e742d66616d696c
         . . . . . . . b o d y , t d , a , p , . h { f o n t - f a m i l
19:20:40.963937 rr:cmd > ca=1e pf=1 nr=5 (2)
        1fb1
         . .
19:20:40.975377 rr:rsp < ca=1e pf=1 nr=5 (2)
        1eb1
         . .
19:20:41.016128 rr:cmd > ca=1e pf=1 nr=5 (2)
        1fb1
         . .
19:20:41.026153 rr:rsp < ca=1e pf=1 nr=5 (2)
        1eb1
         . .
19:20:41.126137 rr:cmd > ca=1e pf=1 nr=5 (2)
        1fb1
         . .
19:20:41.148073 i:rsp  < ca=1e pf=1 nr=5 ns=5 LM slsap=04 dlsap=13 TTP credits=0
(130)
        1eba130400000a3c212d2d0a66756e6374696f6e20736628297b646f63756d65
         . . . . . . . < ! - - . f u n c t i o n   s f ( ) { d o c u m e
19:20:41.148118 rr:cmd > ca=1e pf=1 nr=6 (2)
        1fd1
         . .
19:20:41.155379 rr:rsp < ca=1e pf=1 nr=5 (2)
        1eb1
         . .
19:20:41.196128 rr:cmd > ca=1e pf=1 nr=6 (2)
        1fd1
         . .
19:20:41.217589 i:rsp  < ca=1e pf=1 nr=5 ns=6 LM slsap=04 dlsap=13 TTP credits=0
(130)
        1ebc13040000756d656e742e662e712e76616c75652e6c656e677468296c6f63
```

```
        . . . . . u m e n t . f . q . v a l u e . l e n g t h ) l o c
19:20:41.217638 rr:cmd > ca=1e pf=1 nr=7 (2)
        1ff1
         . .
19:20:41.229226 rr:rsp < ca=1e pf=1 nr=5 (2)
        1eb1
         . .

[ Todo esto nos sirve como referencia para ver la rapidez de envío/recepción de
las tramas, otra petición si eso.....]

         . .
19:21:05.901502 i:cmd  > ca=1e pf=1 nr=0 ns=5 LM slsap=13 dlsap=04 TTP credits=0
(130)
        1f1a0413000021450001b3d2f640004006d7973e525010d8ef27658001005085
         . . . . . ! E . . . . . @ . @ . . . > R P . . . ' e . . . P .
19:21:05.927221 i:rsp  < ca=1e pf=1 nr=6 ns=0 LM slsap=04 dlsap=13 TTP credits=1
(63)
        1ed0130401007e214500003461e240002e065c2bd8ef27653e52501000508001
         . . . . . ~ ! E . . 4 a . @ . . . \ + . . ' e > R P . . P . .
19:21:05.927284 i:cmd  > ca=1e pf=1 nr=1 ns=6 LM slsap=13 dlsap=04 TTP credits=0
(130)
        1f3c0413000020687474703a2f2f7777772e676f6f676c652e636f6d2f0d0a43
         . < . . . .  h t t p : / / w w w . g o o g l e . c o m / . . C
19:21:05.949613 rr:rsp < ca=1e pf=1 nr=7 (2)
        1ef1
         . .
19:21:05.949646 i:cmd  > ca=1e pf=1 nr=1 ns=7 LM slsap=13 dlsap=04 TTP credits=0
(130)
        1f3e041300006c612f352e3020285831313b20553b204c696e757820322e342e
         . > . . . . l a / 5 . 0  ( X 1 1 ;  U ;  L i n u x  2 . 4 .
19:21:05.967983 rr:rsp < ca=1e pf=1 nr=0 (2)
        1e11
         . .
19:21:05.968013 i:cmd  > ca=1e pf=1 nr=1 ns=0 LM slsap=13 dlsap=04 TTP credits=0
(73)
        1f3004130000652c636f6d70726573732c6964656e746974790d0a4b6565702d
         . 0 . . . . e , c o m p r e s s , i d e n t i t y . . K e e p -
19:21:05.982073 i:rsp  < ca=1e pf=1 nr=1 ns=1 LM slsap=04 dlsap=13 TTP credits=4
(5)
        1e32130404
         . 2 . . .
19:21:05.982090 i:cmd  > ca=1e pf=1 nr=2 ns=1 LM slsap=13 dlsap=04 TTP credits=1
(63)
        1f52041301002145000034d2f740004006d9153e525010d8ef27658001005085
         . R . . . . ! E . . 4 . . @ . @ . . . > R P . . . ' e . . . P .

[ ..... colguemos ..... ]
```

```
19:37:56.856123 rr:cmd > ca=1e pf=1 nr=7 (2)
        1ff1
         . .
19:37:56.866967 rr:rsp < ca=1e pf=1 nr=1 (2)
        1e31
         . 1
19:37:56.976127 disc:cmd > ca=0x1e pf=1 (2)
        1f53
         . S
19:37:57.006146 ua:rsp ca=1e pf=1 1a6a9e08 < ab580000 (10)
        1e73000058ab089e6a1a
         . s . . X . . . j .

24280 packets received by filter
```

Appendix C

IrOBEX (Apéndice C)

C.1 Introducción y pruebas

El protocolo IrOBEX en un principio estaba integrado en el paquete de las irda-tools. Sin embargo, se separó para formar por si solo un proyecto de desarroyo conjunto.
Lo forman dos paquetes. Uno perteneciente a las librerías y otro sería el de los binarios:

```
-rw-r--r--    1 root     root       167002 Oct  5 11:57 openobex-0.9.8.tar.gz
-rw-r--r--    1 root     root        53957 Oct  5 11:58 openobex-apps-0.9.8.tar.gz
```

El IrOBEX es una especie de protocolo de intercambio de objetos. Es el que utilizan los teléfonos para intercambiar las VCARDS (tarjetas de visita) o bien la Palm cuando transmite aplicaciones de una a otra.
Por supuesto, ni que decir que también es posible hacer funcionar dicho protocolo en Linux.
Después de la instalación de las susodichas librerías y aplicaciones, encontramos la utilidad perfecta para Palm, que es con la que he hecho las pruebas:

irobex_palm3

A modo de ejemplo, lo ejecutamos:

```
annapurna:~# irobex_palm3
Send and receive files to Palm3
Waiting for files

.....................................HEADER_LENGTH = 35611
put_done() Skipped header 05
CREATORID = 0x6c6e6368
Filename = BigClock.prc
Wrote /tmp/BigClock.prc (35611 bytes)
```

No tiene mayor misterio, es otro modo de transmisión de los ficheros. Algo mas enfocado a como si fueran objetos.

Destacar el anuncio de las tramas XID donde ahora el campo I en vez de anunciar el IrCOMM lo hace con el IrOBEX y que la comunicación esta vez es desde la Palm al portátil y no al revés, como en la mayoria de casos anteriores.

```
13:03:55.498363 xid:cmd ffffffff < cd1af64f S=6 s=5 (14)
        ff3f014ff61acdfffffffff010500
         . ? . 0 . . . . . . . . . .
13:03:55.498385 xid:rsp 2c18ee8c > cd1af64f S=6 s=5 annapurna hint=8420 [
Computer IrOBEX ] (26)
        febf018cee182c4ff61acd010500842000616e6e617075726e61
         . . . . . , 0 . . . . . . .   . a n n a p u r n a
13:03:55.591249 xid:cmd ffffffff < cd1af64f S=6 s=* quasar hint=8220 [
PDA/Palmtop IrOBEX ] (23)
        ff3f014ff61acdfffffffff01ff00822000717561736172
         . ? . 0 . . . . . . . . . .   . q u a s a r

13:03:55.640515 snrm:cmd ca=fe pf=1 2c18ee8c < cd1af64f new-ca=42 (32)
        ff934ff61acd8cee182c4201013f82010183010f840101850108601070801ff
         . . 0 . . . . . , B . . ? . . . . . . . . . . . . . . . . . .
13:03:55.640560 ua:rsp ca=42 pf=1 2c18ee8c > cd1af64f (31)
        42738cee182c4ff61acd01013e82010183013f84017f8501ff860107080107
         B s . . . , 0 . . . . . > . . . . . ? . . . . . .
```

Vemos cómo en la negociación hemos cambiado de protocolo.

```
13:03:55.716862 i:cmd  < ca=42 pf=1 nr=1 ns=1 LM slsap=01 dlsap=00
GET_VALUE_BY_CLASS: "OBEX" "IrDA:TinyTP:LsapSel" (30)
        4332000184044f42455813497244413a54696e7954503a4c73617053656c
         C 2 . . . . O B E X . I r D A : T i n y T P : L s a p S e l
13:03:55.716889 i:rsp  > ca=42 pf=1 nr=2 ns=1 LM slsap=00 dlsap=01
GET_VALUE_BY_CLASS: Success Integer: 10 (15)
        4252010084000001fe810100000010
         B R . . . . . .
```

Y así aparecen otros campos en las tramas, que el irdadump nos los muestra de la siguiente manera:

```
13:03:55.747489 i:rsp  > ca=42 pf=1 nr=4 ns=3 LM slsap=10 dlsap=03 TTP credits=1
        OBEX SUCCESS len=7 ver=1.1 flags=0 mtu=1024 (12)
        4296031001a0000711000400
         B . . . . . .
13:03:55.765767 i:cmd  < ca=42 pf=1 nr=4 ns=4 LM slsap=03 dlsap=10 TTP credits=1
        OBEX PUT final=0 len=63 Name="BigClock.prc" Lenght=35611
Description="BigClock" custom=1819173736 (68)
        439810030102003f01001d00420069006700430006c006f0063006b002e007000
         C . . . . . ? . . . . B . i . g . C . l . o . c . k . . . p .
13:03:55.765795 rr:rsp > ca=42 pf=1 nr=5 (2)
        42b1
         B .
13:03:55.772290 rr:cmd < ca=42 pf=1 nr=4 (2)
        4391
         C .
13:03:55.772301 i:rsp  > ca=42 pf=1 nr=5 ns=4 LM slsap=10 dlsap=03 TTP credits=1
OBEX CONTINUE (8)
        42b8031001900003
         B . . . . . .
```

No he probado el IrOBEX con móviles.....queda para futuras versiones....

Appendix D

Revisiones

Original: 2 Noviembre 2001

Revisión: 12 Noviembre 2001

Añadido WinCE: 1 Febrero 2002

Revisión (linuxdoc): 1 Marzo 2002

Gramática y ortografía: 1 Noviembre 2002 (macana@macana-es.con)