Linux & GPRS

Ing. Mauro Barattin

28/10/2006

Indice

1 Introduzione 1
2 Strumenti utilizzati 1
3 Le tecnologie GPRS/EDGE/UMTS 2
4 Il Bluetooth 2
5 Il progetto Bluez 3
6 Collegamento al cellulare 4
7 Configurazione del Bluetooth 5
8 Testare la connessione 8
9 Connessione con script per pppd 8
10 Connessione con Gnome-PPP 11
11 Applicazioni Desktop per Bluetooth 13
12 Spedire SMS da PC 14

1 Introduzione

Le tariffe flat offerte dai gestori di telefonia mobile e la necessità di navigare in Internet fuori da casa e dall'ufficio, hanno fatto esplodere il fenomeno GPRS. L'intervento vuole fornire le linee guida su come configurare con Linux una connessione ad Internet sfruttando la rete GPRS e un cellulare GSM dotato di tecnologia Bluetooth.

2 Strumenti utilizzati

Sono stati utilizzati i seguenti strumenti:

- Telefono Cellulare: Sony Ericsson T68i
- Adattatore Bluetooth: D-Link DBT-120i
- Distribuzione Linux: Ubuntu Linux 6.06 LTS (Dapper)

3 Le tecnologie GPRS/EDGE/UMTS

Il GPRS (*General Packet Radio Service*) è una delle tecnologie di telefonia mobile. Viene convenzionalmente definita di generazione 2.5, vale a dire una via di mezzo fra la seconda generazione 2G (GSM) e la terza generazione 3G (UMTS). Il massimo limite teorico per la velocità è di circa 170 kbit/s, ma un valore più realistico si attesta intorno a 30-70 kbit/s. Un'evoluzione del modo in cui il GPRS utilizza le frequenze, denominato tecnologia EDGE (*Enhanced Data rates for GSM Evolution*), consente di raggiungere velocità fra 20 e 200 kbit/s. Il sistema UMTS (*Universal Mobile Telecommunications System*), per trasmissioni di tipo dati, supporta un transfer rate massimo di 384 Kbit/s , tuttavia da misure sul campo su reti scariche si sono raggiunti circa 300 kbit/s.

La velocità di trasferimento dipende dal modello di cellulare/terminale usato (classe del cellulare), dal numero di utenti collegati alla cella fra cui è frazionata la banda che, a sua volta, è funzione della densità abitativa del luogo e della fascia oraria, della distanza fra il terminale e l'antenna più vicina...

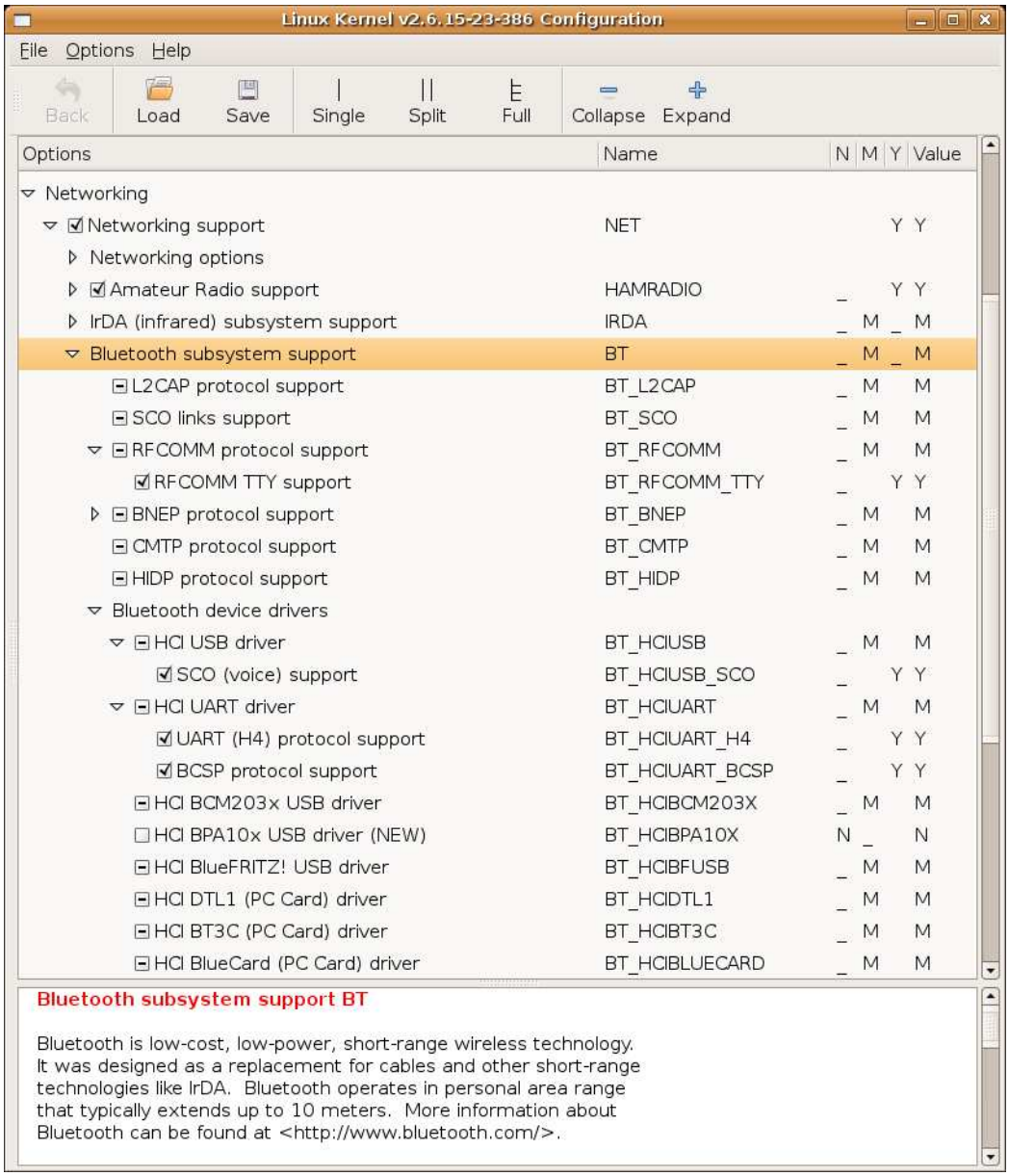
Normalmente il costo delle comunicazioni GPRS viene calcolato in base ai kilobytes ricetrasmessi, a differenza delle reti commutate dove il costo è in funzione del tempo di connessione (questo perché in quest'ultimo tipo di rete l'intera larghezza di banda disponibile è occupata anche quando nessun dato è in corso di trasferimento). A titolo di esempio, per gli operatori di telefonia italiani (Tim, Vodafone e Wind) il costo (IVA compresa) a kilobyte varia tra 0.2 e 0.6 centesimi di euro. Per chi fa largo uso del GPRS, esistono tariffe speciali e/o promozioni sia a tempo che a consumo.

4 Il Bluetooth

Lo standard Bluetooth è stato progettato con l'obiettivo primario di ottenere bassi consumi, un raggio di azione medio (da 10 a 100 metri) e un basso costo di produzione per i dispositivi compatibili. Ulteriori informazioni sul Bluetooth si possono trovare su `http://www.bluetooth.com`. Il sottosistema Bluetooth di Linux Bluetooth è composto da vari layer, tra i quali:

- Bluetooth Core (HCI device and connection manager, scheduler)
- HCI Device drivers (Interface to the hardware)
- RFCOMM Module (RFCOMM Protocol)

Per utilizzare dispositivi Bluetooth è necessario verificare se il kernel di Linux include il supporto per il Bluetooth. Nella pratica tutti i kernel forniti di default con le varie distribuzioni hanno il supporto Bluetooth. Se invece vogliamo compilare il kernel dai sorgenti, alla voce "Networking -> Networking support -> Bluetooth subsystem support" selezioniamo i seguenti moduli:



E' inoltre necessario abilitare il supporto PPP nel menù Device drivers -> Network device support.

5 Il progetto Bluez

Per utilizzare il sottosistema Bluetooth abbiamo bisogno di alcune utility in user-space (come `hciconfig` e `hcid`) che permettono alle applicazioni di comunicare con l'adattatore Bluetooth. Il progetto BlueZ (`http://www.bluez.org`) fornisce lo stack Bluetooth ufficiale di Linux. I pacchetti di cui abbiamo bisogno sono:

1. bluez-libs
2. bluez-utils
3. bluez-pin

Installiamo i pacchetti necessari (utilizzando il package manager della nostra distribuzione preferita oppure compilando i sorgenti) ed avviamo il demone `hcid` (il demone `hcid`, acronimo di *Host Controller Interface*, è necessario per tutte le operazioni relative al Bluetooth e quindi è bene lanciarlo al boot):

```
# /usr/sbin/hcid
```

6 Collegamento al cellulare

Inseriamo l'adattatore Bluetooth e verifichiamo nei messaggi di sistema se il dispositivo è stato riconosciuto da Linux:

```
root@jeeg:~ # dmesg
[4312710.179000] usb 2-2: new full speed USB device using ohci_hcd and address 5
[4312710.563000] Bluetooth: HCI USB driver ver 2.9
[4312710.565000] usbcore: registered new driver hci_usb
```

Procediamo ora alla verifica dell'operatività del nostro dispositivo:

```
root@jeeg:~ # hciconfig -a
hci0:  Type: USB
          BD Address: 00:0F:3D:48:E9:E9 ACL MTU: 192:8 SCO MTU: 64:8
          UP RUNNING PSCAN ISCAN
          RX bytes:378 acl:0 sco:0 events:17 errors:0
          TX bytes:316 acl:0 sco:0 commands:16 errors:0
          Features: 0xff 0xff 0x0f 0x00 0x00 0x00 0x00 0x00
          Packet type: DM1 DM3 DM5 DH1 DH3 DH5 HV1 HV2 HV3
          Link policy: RSWITCH HOLD SNIFF PARK
          Link mode: SLAVE ACCEPT
          Name: 'jeeg-0'
          Class: 0x3e0100
          Service Classes: Networking, Rendering, Capturing
          Device Class: Computer, Uncategorized
          HCI Ver: 1.1 (0x1) HCI Rev: 0x20d LMP Ver: 1.1 (0x1) LMP Subver: 0x20d
          Manufacturer: Cambridge Silicon Radio (10)
```

Questo output ci permette di capire che il demone `hcid` è in funzione dal momento che esiste il dispositivo `hci0` e questo appare "`UP RUNNING`", altrimenti ci sarebbe stato un "`DOWN`". Si vede anche il MAC address del dispositivo Bluetooth `00:0F:3D:48:E9:E9`. Eseguiamo ora uno scan delle periferiche Bluetooth per recuperare il MAC address del cellulare (è necessario che sul cellulare sia attivato il Bluetooth):

```
# hcitool scan
Scanning ...
            00:0A:D9:2D:7A:B8     T68i  Mauro
```

Come si vede, è stato trovato il cellulare "T68i Mauro" con MAC address `00:0A:D9:2D:7A:B8`. E' possibile "pingare" il telefono come se fosse una normale interfaccia di rete:

```
# l2ping 00:0A:D9:2D:7A:B8
Ping: 00:0A:D9:2D:7A:B8 from 00:0F:3D:48:E9:E9 (data size 44) ...
0 bytes from 00:0A:D9:2D:7A:B8 id 0 time 39.68ms
0 bytes from 00:0A:D9:2D:7A:B8 id 1 time 34.63ms
0 bytes from 00:0A:D9:2D:7A:B8 id 2 time 33.74ms
3 sent, 3 received, 0% loss
```

7 Configurazione del Bluetooth

Per configurare il Bluetooth è necessario modificare i files presenti nella directory `/etc/bluetooth/`:

```
# ls -l /etc/bluetooth/
totale 16
-rw-r--r-- 1 root root 1340 2006-03-20 06:53 hcid.conf
-rw-r--r-- 1 root root 5 2005-03-03 22:07 pin
-rw-r--r-- 1 root root 292 2006-04-01 18:21 rfcomm.conf
```

Il file `hcid.conf` contiene la configurazione del demone `hci`. Tipicamente non è necessario modificare questo file in quanto la configurazione di default è perfettamente funzionante. Al suo interno si può notare la gestione del PIN dell'interfacciamento tra i due dispositivi:

```
[...]
# Security Manager mode
# none - Security manager disabled
# auto - Use local PIN for incoming connections
# user - Always ask user for a PIN
#
security auto;
[...]
# PIN helper
pin_helper /usr/bin/pinwrapper;
[...]
```

Il passo successivo è quello di configurare il numero PIN che servirà per l'allineamento di questo dispositivo con gli altri. Il PIN si trova all'interno dell'omonimo file:

```
# cat /etc/bluetooth/pin
1234
```

Il valore "`1234`" è il valore di default. Il numero inserito deve essere lo stesso nei dispositivi che si vuole connettere alla rete Bluetooth, in modo da garantire il corretto allineamento. Deve ovviamente essere tenuto segreto, altrimenti sarà possibile, per chiunque ne sia a conoscenza, stabilire

una connessione al nostro PC. Nel momento in cui si lancia il comando che instaura la comunicazione tra PC e cellulare, da entrambi i lati viene richiesto di inserite il suddetto codice. Qualora la verifica si avvenuta con successo, sul telefono viene mostrata la richiesta del proprio sistema di sfruttare il modem del cellulare stesso per collegarsi ad Internet. Se si accetta la richiesta, si avvia il processo di comunicazione con l'ISP.

Il protocollo RFCOMM (*Radio Frequency Communication*) fornisce il meccanismo di trasporto "connection oriented" per i dispositivi Bluetooth. Il file `rfcomm.conf` contiene la configurazione della connessione seriale verso il cellulare:

```
# cat /etc/bluetooth/rfcomm.conf
#
# RFCOMM configuration file.
#
rfcomm0 {
bind yes;
# Bluetooth address of the device
device 00:0A:D9:2D:7A:B8;
# RFCOMM channel for the connection
channel 1;
# Description of the connection
comment "My T68i";
}
```

Il device da utilizzare in questo caso è `/dev/rfcomm0`. Nel campo "device" va specificato il MAC address del cellulare, mentre nel campo "comment" possiamo inserire una breve descrizione del dispositivo. Il parametro più ostico da individuare è "channel" in quanto alcuni cellulari possono utilizzare un canale RFCOMM diverso da 1 per la connessione Dial-Up. Per conoscere il canale del cellulare riservato al DUN (*Dial Up Networking*) possiamo utilizzare il tool `sdptool` (*Service Discovery Protocol tool*) in questo modo:

```
# sdptool search DUN 00:0A:D9:2D:7A:B8
Inquiring ...
Searching for DUN on 00:0A:D9:2D:7A:B8 ...
Service Name: Dial-up Networking
Service RecHandle: 0x10000
Service Class ID List:
"Dialup Networking" (0x1103)
"Generic Networking" (0x1201)
Protocol Descriptor List:
"L2CAP" (0x0100)
"RFCOMM" (0x0003)
Channel: 1
Profile Descriptor List:
"Dialup Networking" (0x1103)
Version: 0x0100
```

Si può vedere che il canale del telefono cellulare dedicato al Dial Up Networking è il canale 1. A questo punto è necessario creare il collegamento (bind) tra il dispositivo RFCOMM e il dispositivo Bluetooth remoto (il primo comando rilascia eventuali altri canali precedentemente utilizzati):

```
# rfcomm release all (opzionale)
# rfcomm bind 0 00:0A:D9:2D:7A:B8 1
```

Il primo parametro dopo il comando `bind` è il nodo TTY del dispositivo RFCOMM che verrà usato (di solito è 0). Il secondo parametro indica l'indirizzo MAC del dispositivo remoto. Il terzo invece è opzionale e specifica il canale da utilizzare. E' necessario che il nostro apparecchio rimanga sempre in ascolto delle connessioni in entrata, altrimenti non sarà possibile la connessione. Questo comando non instaura una connessione con il dispositivo remoto, ma crea solo il canale di comunicazione. La connessione verrà stabilita solo quando un'applicazione tenterà di collegarsi al dispositivo RFCOMM. Per verificare che la creazione del dispositivo RFCOMM sia avvenuta correttamente, possiamo usare il seguente comando, verificando che il canale sia "clean".

```
# rfcomm show
rfcomm0: 00:0A:D9:2D:7A:B8 channel 1 clean
```

Possiamo connetterci al telefono cellulare con il comando:

```
# rfcomm release all (opzionale)
# rfcomm connect 0 00:0A:D9:2D:7A:B8 1
```

Appena lanciato il precedente comando sul display del nostro cellulare apparirà il messaggio di avvertimento se vogliamo aggiungere alla lista dei dispositivi il nostro PC. Accettiamo ed inseriamo lo stesso PIN contenuto nel file `/etc/bluetooth/pin`. L'avvio di `rfcomm` crea automaticamente il device `/dev/rfcomm0` ma, se questo non avviene, possiamo creare manualmente il device con il comando:

```
# mknod /dev/rfcomm0 c 216 0
```

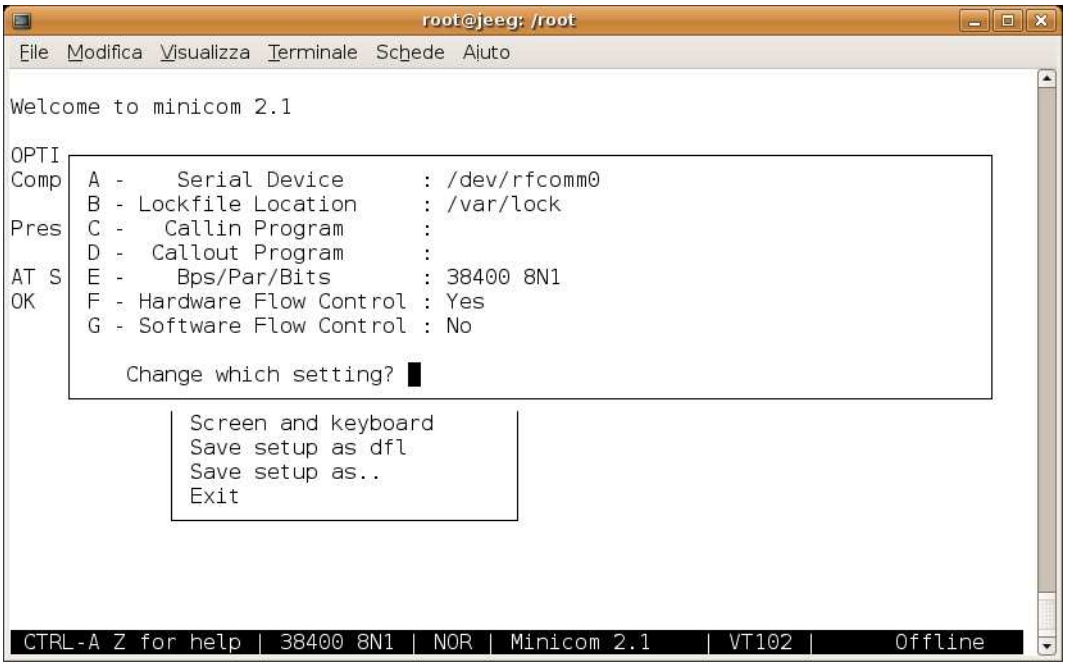
Nella pratica non è necessario utlizzare l'utility `rfcomm` per configurare il sottosistema Bluetooth in quanto tutte le moderne distribuzione possiedono degli script in grado di eseguire automaticamente tutte queste operazioni. Ad esempio su Ubuntu Dapper la configurazione del dispositivi RFCOMM viene fatta semplicemente invocando il comando:

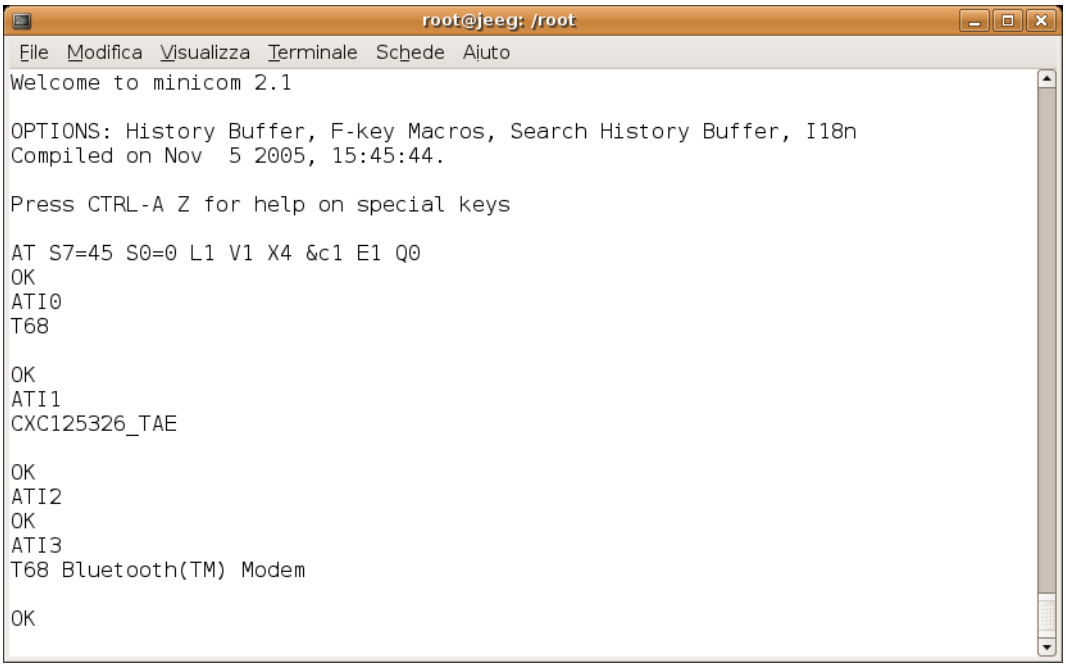
```
# /etc/init.d/bluez-utils start
  * Starting Bluetooth services...
  hcid sdpd                                              [ ok ]
```

Questo script viene automaticamente eseguito al boot.

8 Testare la connessione

Per testare il corretto funzionamento del device RFCOMM appena creato, possiamo ricorrere al software `minicom` , selezionando come porta del modem il device `/dev/rfcomm0`. Come per i comuni modem, possiamo inviare i comandi AT per conoscerne le informazioni.





9 Connessione con script per pppd

Il passo successivo riguarda la configurazione del dispositivo di connessione `ppp` (*Point To Point Protocol*). I files di configurazione sono collocati in `/etc/ppp/` e gli script di avvio della

applicazione in `/etc/chatscripts/`. Quello che si può fare è crearne uno (nel nostro caso lo script si chiama gprs) e collocarlo nella apposita directory `/etc/ppp/peers/`.
Vediamo il contenuto del file `/etc/ppp/peers/gprs`:

```
# Most GPRS phones don't reply to LCP echo's
lcp-echo-failure 0
lcp-echo-interval 0

# Keep pppd attached to the terminal
# Comment this to get daemon mode pppd
nodetach

# The chat script (be sure to edit that file, too!)
connect "/usr/sbin/chat -v -f /etc/chatscripts/vodafone_gprs_connect"
disconnect "/usr/sbin/chat -v -f /etc/chatscripts/vodafone_gprs_disconnect"

# Serial Device to which the GPRS phone is connected
/dev/rfcomm0

# Serial port line speed
115200

# The phone is not required to authenticate
noauth

# If you want to use the GPRS link as your gateway
defaultroute

# pppd must not propose any IP address to the peer
noipdefault
ipcp-accept-local
ipcp-accept-remote

# Keep modem up even if connection fails
persist

# Hardware flow control
crtscts

# Ask  the  peer  for up to 2 DNS server addresses
usepeerdns

# No ppp compression
novj
nobsdcomp
novjccomp
```

```
nopcomp
noaccomp

# For sanity, keep a lock on the serial line
lock

# Debug info from pppd
debug

# Show password in debug messages
show-password
```

Vediamo il contenuto del file `/etc/chatscripts/vodafone_gprs_connect`:

```
ABORT 'BUSY'
ABORT 'VOICE'
ABORT 'DELAYED'
ABORT 'NO CARRIER'
ABORT 'NO DIALTONE'
ABORT 'NO DIAL TONE'
ABORT 'NO ANSWER'

# Reset the line
'' ATZ

# Connection to the Vodafone Italia network
'' AT+CGDCONT=1,"IP","web.omnitel.it",0,0

# Dial the number.
'' ATDT*99***1#

# The modem is waiting for the following answer
CONNECT ''
```

Con altri operatori è necessario sostituire "web.omnitel.it" con:

Operatore	APN	Autenticazione
TIM	uni.tim.it	Si
Vodafone	web.omnitel.it	No
Wind	internet.wind	No

Se l'operatore richiede l'autenticazione sarà necessario editare il file `/etc/ppp/pap-secrets` per inserire una riga con il seguente formato:

```
"USERNAME" * "PASSWORD"
```

Editiamo ora il file `/etc/resolv.conf` (oppure il file `/etc/ppp/resolv/gprs`) ed aggiungiamo gli indirizzi DNS del nostro provider. A questo punto per collegarci ad Internet è sufficiente lanciare il comando:

```
# pon gprs
```

A questo punto la connessione è stabilita ed eseguendo il comando:

```
# ifconfig ppp0
ppp0      Link encap:Point-to-Point Protocol
          inet addr:83.225.68.39  P-t-P:83.225.68.38  Mask:255.255.255.255
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
          RX packets:4 errors:0 dropped:0 overruns:0 frame:0
          TX packets:5 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:3
          RX bytes:64 (64.0 b)  TX bytes:97 (97.0 b)
```

si potrà vedere il proprio indirizzo Internet nella rete del proprio provider. L'interfaccia in esame sarà con ogni probabilità la ppp0 (o in generale una pppx). Nei log di sistema è possibile vedere le varie fasi della connessione.

```
root@jeeg:~ # tail /var/log/messages
[...] pppd[14347]: pppd 2.4.4b1 started by mauro, uid 1000
[...] pppd[14347]: Using interface ppp0
[...] pppd[14347]: Connect: ppp0 <--> /dev/rfcomm0
[...] pppd[14347]: PAP authentication succeeded
[...] kernel: [4311555.166000] PPP BSD Compression module registered
[...] kernel: [4311555.203000] PPP Deflate Compression module registered
[...] pppd[14347]: local IP address 83.225.68.39
[...] pppd[14347]: remote IP address 83.225.68.38
[...] pppd[14347]: primary DNS address 83.224.66.134
[...] pppd[14347]: secondary DNS address 83.224.65.134
```

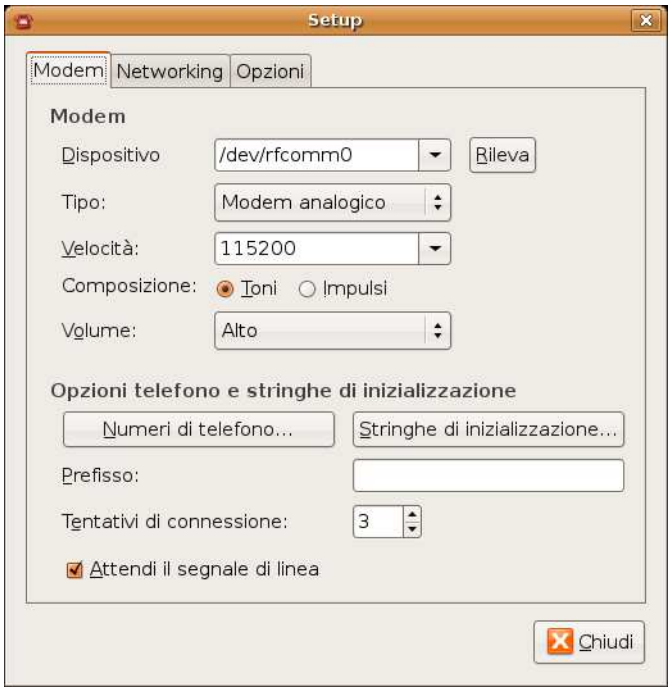
Per scollegarci utilizziamo:

```
# poff gprs
```

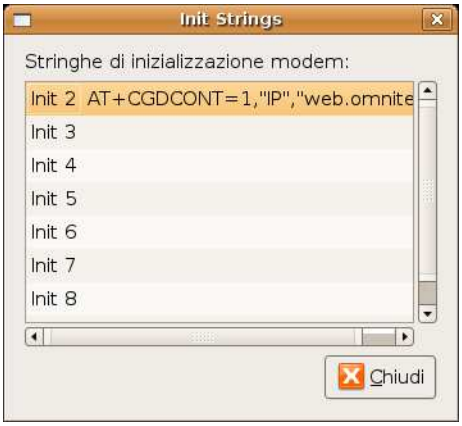
10 Connessione con Gnome-PPP

Esiste un modo molto più semplice e rapido per effettuare la connessione via Bluetooth. Ciò di cui abbiamo bisogno è `wvdial` e la sua interfaccia grafica Gnome-PPP. Qualora questi software non fossero già installati, si trovano entrambi facilmente nei repository delle principali distribuzioni. Una volta configurato correttamente il Bluetooth, apriamo Gnome-PPP e inseriamo username, password e numero di telefono. Nel caso di Vodafone, username e password sono casuali.

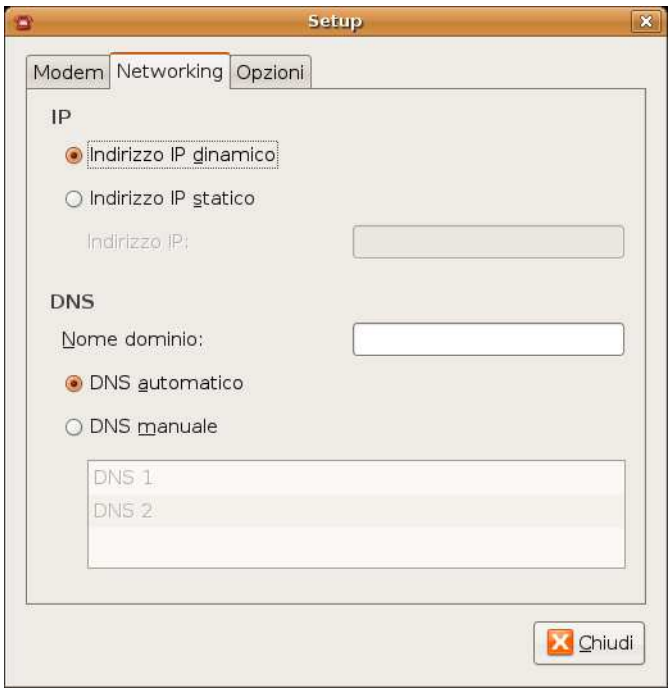
Nel menù "Configura" specifichiamo di utilizzare il device `/dev/rfcomm0`:



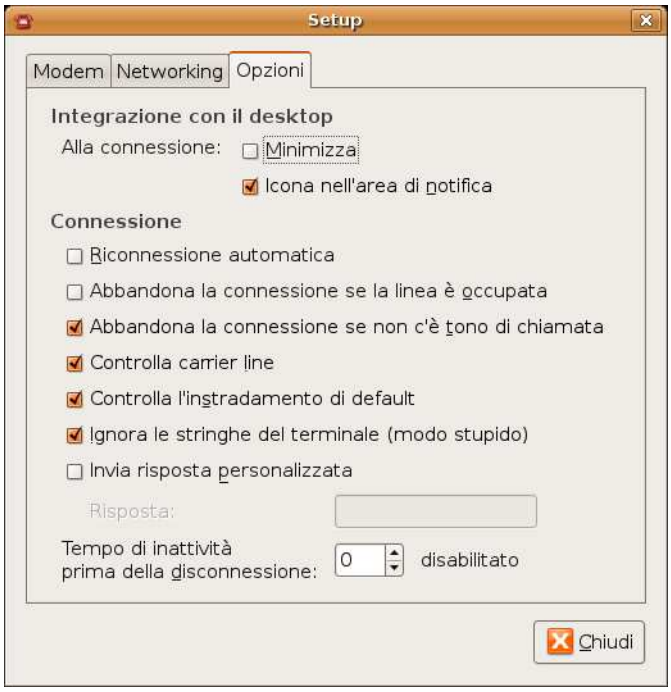
Nel menù "Stringhe di Inizializzazione" compiliamo la riga "Init 2" specificando la stringa di connessione al GPRS del vostro gestore. Nel caso di Vodafone è `AT+CGDCONT=1,"IP","web.omnitel.it"`



Nel tab "Networking" specifichiamo che l'indirizzo IP e i DNS dovranno essere configurati in modo automatico:



Nel tab “Opzioni” utilizziamo questa configurazione:



11 Applicazioni Desktop per Bluetooth

In ambiente Gnome, molto probabilmente la scelta migliore è gnome-bluetooth. Fornisce le funzioni base maggiormente utilizzate:

- gnome-bluetooth-manager: Per gestire i dispositivi remoti Bluetooth.
- gnome-obex-send: Per mandare file verso altri dispositivi.

- gnome-obex-server: Per ricevere file.

12 Spedire SMS da PC

L'applicativo `gnome-phone-manager` è un'elegante applicazione per mandare e ricevere messaggi da e verso il un telefono GSM, usando solamente il PC. Non è più necessario utilizzare il cellulare per leggere o scrivere messaggi, visto che è possibile fare tutto tramite questo programma. Selezionando l'opzione nel menù delle preferenze, è possibile essere automaticamente avvertiti con un messaggio sullo schermo quando arrivano nuovi messaggi.

In ambiente KDE la scelta migliore è probabilmente KMobileTools.

Riferimenti bibliografici

[1] Wikipedia (http://it.wikipedia.org)

[2] The Official Blueetooth Membership Site (https://www.bluetooth.org)

[3] Official Linux Bluetooth Protocol Stack (http://www

[4] Mikko Rapeli - Of Linux, GPRS Phones, Serial Cable, Irda, Bluetooth and USB.

[5] Francesco Strappini - Linux, Bluetooth e GPRS - Guida Pratica.

[6] AleXit - Connessione GPRS/EDGE/UMTS su Ubuntu con Nokia 6630 via Bluetooth e USB.

[7] Stefano Melchior - PPP, GPRS e UMTS: Internet anche in vacanza con GNU/Linux.

[8] Salvo Cuccurullo - My Linux Bluetooth Experience.