Collegarsi ad una rete wireless 802.11 con Linux!

(extended edition)

Scano Alessandro
alescano@libero.it

GULCh – Gruppo Utenti Linux Cagliari

Wireless Linux

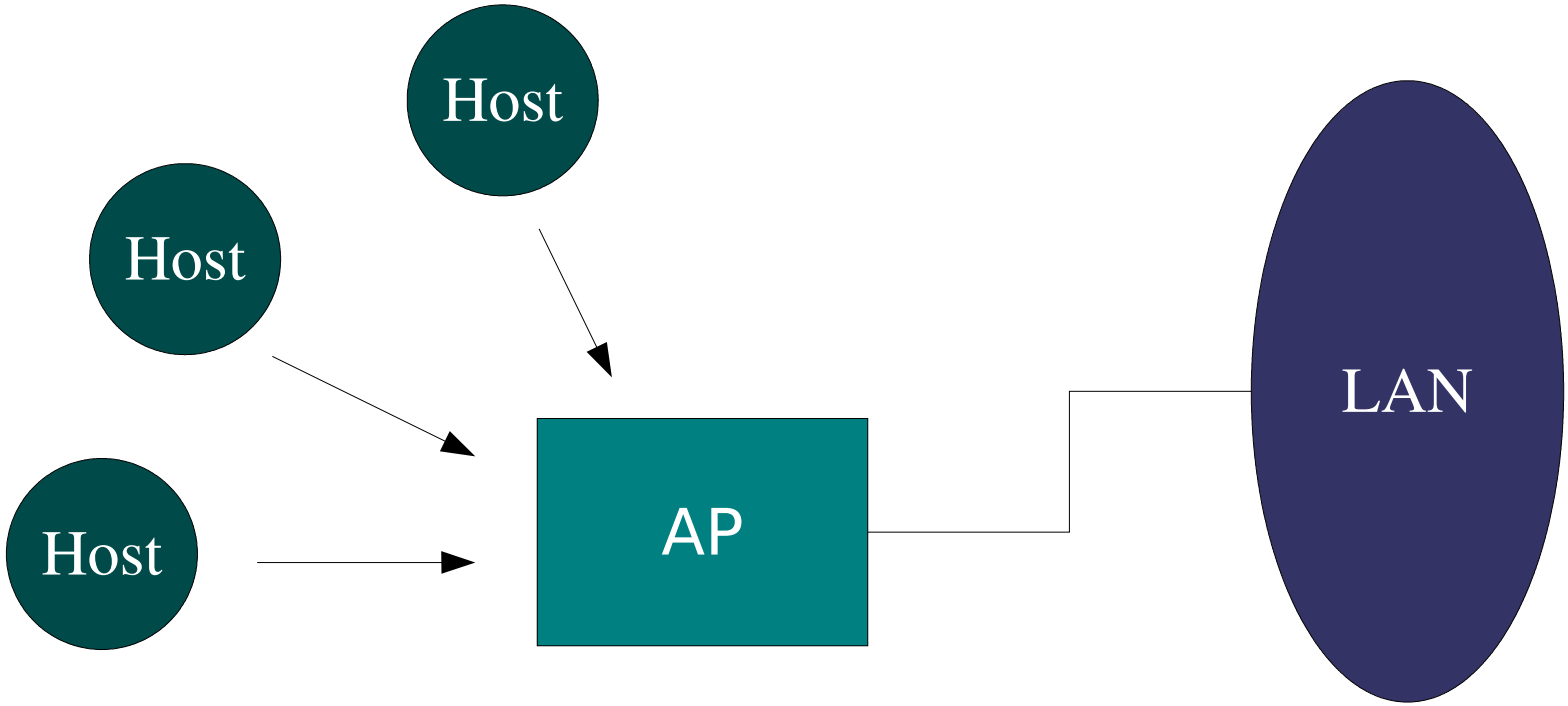
Lo standard 802.11 (Wi – Fi) viene sempre più utilizzato per la creazione di LAN radio non cablate soprattutto in piccole realtà per via della comodità di installazione ed utilizzo.

I dispositivi più venduti ultimamente sono quelli 802.11g a 54Mb, ma sono ancora molto diffusi quelli a 11Mb

Vi sono principalmente due modalità di utilizzo:

Ad-Hoc:	host -> host
Infrastructure	host -> access point (AP)

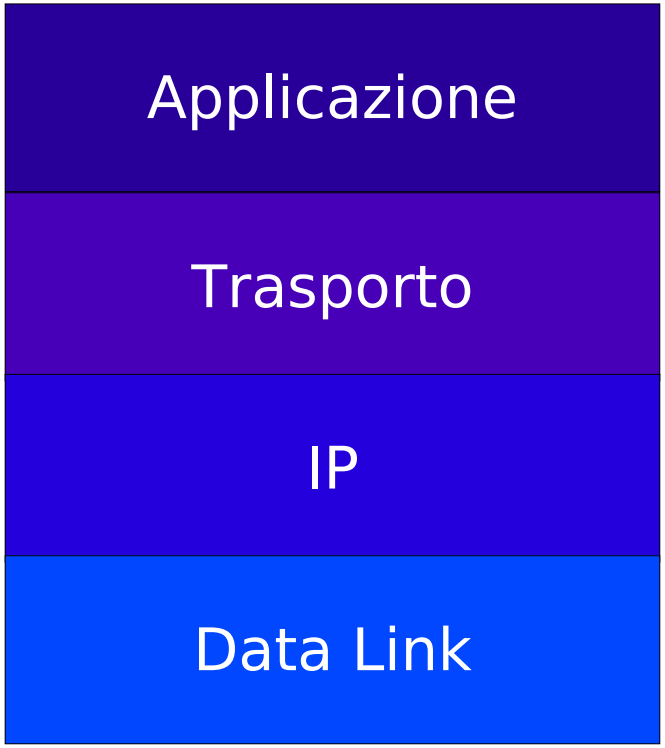
Chiamate tecnicamente:
IBSS (Indipendent Basic Service Set)
BSS (Basic Service Set)



L'access point fa da bridge (ponte) tra le reti cablate e la wi-fi collegando fisicamente gli host wi-fi tra loro e con quelli in LAN

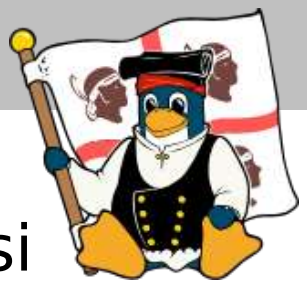
Un Bridge tecnicamente è uno switch a livello 2 dello stack ISO / OSI

Il protocollo 802.11 è collocato alla base dello stack TCP/IP



Tra 802.11b e 802.11g vi è una differenza di velocità;
tra 802.11g e 802.11i di sicurezza

802.11(b/g/j)



Prima di poter comunicare l'host wi-fi deve autenticarsi ed associarsi presso l'AP

Questo passa attraverso gli stati:

- Non autenticato, non associato
- Autenticato, non associato
- Autenticato, Associato

Tipologie di autenticazione standard:

- Open (aperta)
- Shared Key (host ed AP condividono una chiave)
- SSID based (Service Set identifier)
- MAC Address (access list)
- etc... (vendor extensions)

Wireless Linux

Il wi-fi essendo basato su onde radio è ancora più facilmente intercettabile

Il protocollo WEP (Wired Equivalent privacy) basato su chiave condivisa è notoriamente poco sicuro (http://www.nwfusion.com/details/715.html)

L' 802.11 prevede due modalità di gestione chiavi:

- Finestra di 4 chiavi (l'AP può decifrare con ognuna delle 4, ll peer può trasmettere con una qualsiasi delle 4)
- Associazione MAC -> chiave (una chiave per ogni peer)

Oltre a questo l'AP può essere configurato per non mandare in broadcast ad intervalli regolari il *beacon* management frame in modo da non rivelare il SSID in reti di tipo “closed” .. ma anche questa è ben poca cosa.

il WEP si basa su una chiave conosciuta dall'AP e dai client, che serve per criptare il contenuto dei frame 802.11 mediante l'algortimo di cifratura RC4.

Il problema di questa chiave è che 24 dei suoi bit sono derivati direttamente dall'IV (initialization vector), che sono trasmessi anche in chiaro nei frame 802.11: infatti solo i dati sono criptati, mentre header e IV rimangono in chiaro. Visto che i bit dedicati all'IV sono così pochi, con un ampio scambio di dati gli IV possibili (2^24 ~ 17 M) si esauriscono in fretta:
la sicurezza del protocollo viene meno una volta che vengono utilizzate le chiavi più di una volta.

Inoltre il WEP prevede una reinizializzazione delle chiavi ogni qualvolta avviene una collisione nelle trasmissione, quindi in una rete piuttosto trafficata è possibile che vengano riutilizzate le stesse chiavi in pochi minuti.

L'802.11i recentemente approvato ha due principali implementazioni:

WPA

Basato sul draft 3.0 della specifica 802.11i implementa un sottoinsieme della specifica per poter essere implementato con l'hardware esistente.
Al posto del WEP usa TKIP che usa ancora RC4 ma con chiave evolutiva per pacchetto.
Può usare chiave condivisa come il WEP oppure un server di autenticazione (Radius) ed EAP (Extensible Authentication Protocol). Tutti e due i meccanismi servono a generare comunque una chiave master di sessione tra l'AP ed il cosiddetto *supplicant* (il nodo client)
La fase di generazione delle chiavi di criptazione dati avviene in 4 fasi (four way handshake)

http://hostap.epitest.fi/wpa_supplicant/

WPA2

Implementazione effettiva dello standard 802.11i: rispetto al WPA già visto si ha che:

Usa AES (Advanced Encryption Standard) con chiave a 128 bit (questo lo rende incompatibile con l'hardware esistente) Il IV (initialization vector) è a 48 bits, L'integrità dell'header e dei dati è garantita dall'algoritmo CBC-MAC

La gestione delle chiavi è basata sul 'EAP (Extensible Authentication Protocol).

Wireless Linux

Linux supporta i dispositivi 802.11 con le wireless-extensions sin dalla versione 2.0.30

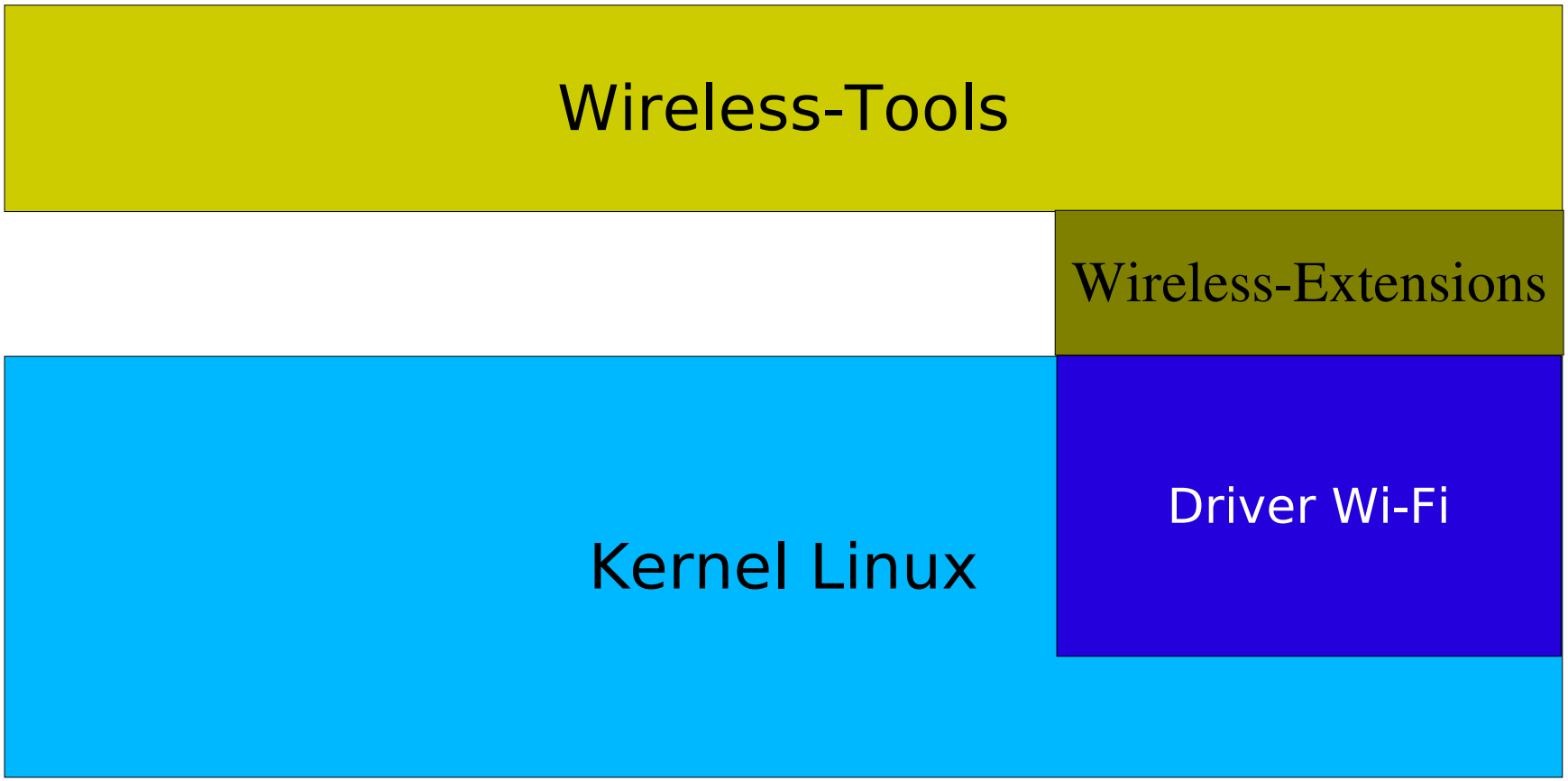
Le wireless extensions si dividono in due parti:

- Tools userspace per la configurazione dei device
- Driver kernel con supporto alle stesse

I device wi-fi che vogliono essere supportati sotto linux dai wireless-tools devono avere dei driver che supportano le wireless-extensions del kernel, che presentano cioè quell'interfaccia di configurazione

Wireless Linux

Le wireless extension rappresentano uno strato di compatibilità di configurazione tra drivers del kernel e tools user-space di configurazione:



Parametri di configurazione importanti per un dispositivo wi-fi:

ESSID: Extended service set id (è il nome della rete, serve ad associare gli host tra loro)
channel: frequenza di lavoro specificata in canali
freq: frequenza di lavoro
mode: Modalità del peer (BSS o IBSS) (Ad-Hoc, Infrastructure)
ap: MAC address dell'AP desiderato
rate: Velocità di lavoro (Bit Rate) Es. 54M
key: Chiave di criptazione utilizzata

Wireless Linux

La configurazione su linux si fa con gli wireless-tools, programmi in userspace:

iwconfig: analogo di *ifconfig*, serve a settare i parametri citati prima.
iwlist: fa uno scan per i peer o gli AP, o elenca le funzionalità del driver
iwspy: colleziona statistiche su un peer remoto della LAN
iwpriv: parametri specifici del driver
iwevent: monitor di eventi sull'interfaccia

Alcuni esempi di configurazione:

```
iwconfig eth1 essid “LinuxDay” mode Managed
```

Infrastructure, senza chiave WEP

```
iwconfig wlan0 essid TUX mode Ad-Hoc
channel 6 rate 54M
```

peer to peer, senza chiave WEP, a 54Mbit, canale 6

```
iwconfig wlan0 essid TUX mode Infrastucture
channel 6 rate 54M key s:PENGUINBOOTX24
```

peer to peer, con chiave ASCII, a 54Mbit, canale 6

```
iwlist eth1 scanning
```

scan dei peer e degli Access Point sull'interfaccia

Passi da seguire per l'installazione

I passi sono sempre gli stessi:

1. Identificare il chipset della scheda
2. Verificare il supporto su linux (ufficiale o no)
3. Configurare ed installare il driver
4. Verificarne il funzionamento

I chipset supportati ufficialmente dal kernel sono i più indicati (aironet, prism2, lucent, alcuni chipset cisco); ma esistono numerosi altri progetti opensource che supportano il wi-fi

Wireless Linux

Root Console - Konsole

Session Edit View Bookmarks Settings Help

```
Linux Kernel v2.6.6 Configuration

                    Wireless LAN (non-hamradio)
  Arrow keys navigate the menu.  <Enter> selects submenus --->.  Highlighted letters are
  hotkeys.  Pressing <Y> includes, <N> excludes, <M> modularizes features.  Press
  <Esc><Esc> to exit, <?> for Help.  Legend: [*] built-in  [ ] excluded  <M> module  < >
  module capable

          [*] Wireless LAN drivers (non-hamradio) & Wireless Extensions
          ---   Obsolete Wireless cards support (pre-802.11)
          <*>   STRIP (Metricom starmode radio IP)
          < >   Aironet Arlan 655 & IC2200 DS support
          < >   AT&T/Lucent old WaveLAN & DEC RoamAbout DS ISA support
          < >   AT&T/Lucent old WaveLAN Pcmcia wireless support
          < >   Xircom Netwave AirSurfer Pcmcia wireless support
          ---   Wireless 802.11 Frequency Hopping cards support
          < >   Aviator/Raytheon 2.4MHz wireless support
          ---   Wireless 802.11b ISA/PCI cards support
          < >   Cisco/Aironet 34X/35X/4500/4800 ISA and PCI cards
          < >   Hermes chipset 802.11b support (Orinoco/Prism2/Symbol)
          < >   Atmel at76c50x chipset  802.11b support
          ---   Wireless 802.11b Pcmcia/Cardbus cards support
          <M>   Cisco/Aironet 34X/35X/4500/4800 PCMCIA cards
          <M>   Planet WL3501 PCMCIA cards
          ---   Prism GT/Duette 802.11(a/b/g) PCI/Cardbus support
          <M>   Intersil Prism GT/Duette/Indigo PCI/Cardbus

                <Select>    < Exit >    < Help >
```

Shell | Root Console

GULCh – Gruppo Utenti Linux Cagliari

Wireless Linux

Root Console - Konsole

Session Edit View Bookmarks Settings Help

```
Linux Kernel v2.6.6 Configuration

                    Intersil Prism GT/Duette/Indigo PCI/Cardbus
CONFIG_PRISM54:

Enable PCI and Cardbus support for the following chipset based cards:

ISL3880 - Prism GT            802.11 b/g
ISL3877 - Prism Indigo        802.11 a
ISL3890 - Prism Duette        802.11 a/b/g

For a complete list of supported cards visit <http://prism54.org>.
Here is the latest confirmed list of supported cards:

3com OfficeConnect 11g Cardbus Card aka 3CRWE154G72
Allnet ALL0271 PCI Card
Compex WL54G Cardbus Card
Corega CG-WLCB54GT Cardbus Card
D-Link Air Plus Xtreme G A1 Cardbus Card aka DWL-g650
I-O Data WN-G54/CB Cardbus Card
Kobishi XG-300 aka Z-Com Cardbus Card
Netgear WG511 Cardbus Card
Ovislink WL-5400PCI PCI Card
Peabird WLG-PCI PCI Card
Sitecom WL-100i Cardbus Card
Sitecom WL-110i PCI Card
SMC2802W -    EZ Connect g 2.4GHz 54 Mbps Wireless PCI Card
SMC2835W -    EZ Connect g 2.4GHz 54 Mbps Wireless Cardbus Card
SMC2835W-V2 - EZ Connect g 2.4GHz 54 Mbps Wireless Cardbus Card
Z-Com XG-900 PCI Card
Zyxel G-100 Cardbus Card

If you enable this you will need a firmware file as well.
You will need to copy this to /usr/lib/hotplug/firmware/isl3890.
You can get this non-GPL'd firmware file from the Prism54 project page:
<http://prism54.org>
You will also need the /etc/hotplug/firmware.agent script from
                                                              ( 78%)
                              < Exit >
```

Shell | Root Console

Il driver Prism54 supporta le nuove schede a 54MB

E' uno dei più supportati e flessibili su Linux

GULCh – Gruppo Utenti Linux Cagliari

Se si è incerti consultare il sito del Vendor per capire il chipset installato nella scheda

Altrimenti ecco alcuni progetti di riferimento per i driver:

Linux w-lan-ng: http://www.linux-wlan.org/
(qua c'è anche una discreta lista di schede con relativi chipset all'indirizzo:
http://www.linux-wlan.org/docs/wlan_adapters.html.gz)

Madwifi: http://www.madwifi.net/
(sopratutto per quanto riguarda i chipset atheros)

ACX100: http://acx100.sourceforge.net/
(supporto dei chipset Texas Instruments)

I device Wi-Fi USB non sono supportati quanto gli altri sotto linux, ma la soluzione migliora costantemente:

Su:
http://atmelwlandriver.sourceforge.net

Vi è un progetto che riguarda device Wi-Fi USB basati su chipset **atmel**; le ultime versioni di questo driver consentono l'autenticazione WPA (tramite il comando “iwpriv” e supportano completamente le wireless extensions.

Se siete veramente sfortunati:

http://www.linuxant.com/driverloader/
(loader di driver wi-fi windows a pagamento)

Oppure:

http://ndiswrapper.sourceforge.net/
(loader di driver wi-fi windows open source)

Il risultato dell'installazione di un driver è sempre un modulo del kernel, ma le modalità possono variare di molto.

Una volta che le cose funzionano, si può configurare la scheda all'avvio del sistema...

Debian 3.0 e successive:

Nel file `/etc/network/interfaces` è possibile aggiungere dei parametri che riguardano le impostazioni wireless: Ex.

```
iface eth1 inet static
        wireless-essid Floppy
        wireless-mode Managed
        address 192.168.1.10
        netmask 255.255.255.0
        broadcast 192.168.1.255
        gateway 192.168.1.1
```

La forma generale è:
`wireless-[param] [param]`

Suse 8.0 e successive:

Nel file `/etc/sysconfig/network/wireless` si specificano i parametri globali wireless nella forma:

```
WIRELESS_ESSID = “[essid]”
```

Mentre negli script: `/etc/sysconfig/network/ifcfg-[device]`

è possibile specificare gli stessi parametri, che sovrascrivono quelli globali.

E' possibile configurare il tutto anche con Yast2

RedHAT 7.3 e successive:

il tool redhat-config-network semplifica di molto la configurazione di un device wireless in questa distribuzione.

La configurazione a “basso livello” risiede, similarmente alla suse negli script:

```
/etc/sysconfig/network-scripts/ifcfg-[device]
```

Con parametri del tipo:

```
MODE = <mode>
ESSID = “<essid>”
RATE =  <rate>
```

Mandrake 8.2 e successive:

La configurazione si effettua negli script:

```
/etc/sysconfig/network-scripts/ifcfg-[device]
```

Dove [device] è il device di rete wireless

Con parametri del tipo:

```
WIRELESS_MODE=<mode>
WIRELESS_ESSID=<essid>
WIRELESS_FREQ=<freq/channel>
WIRELESS_SENS=<sensitivity>
WIRELESS_RATE=<rate>
WIRELESS_ENC_KEY=<keys>
```

Front-End grafici

Come al solito, esistono anche i front-end grafici:

KDE 3.3 ha un modulo di configurazione per il control center

GNOME ha dei front-end:
http://gwifiapplet.sourceforge.net/
(include un monitor ed un tool di configurazione)

Esistono numerosi tool di monitor:
kwifimanager (per KDE)
TKWifi: http://tkwifi.sourceforge.net/ (tcl / tk)
http://wmwifi.digitalssg.net/ (windowmaker)

Configurazione KDE:

Wireless Network - Control Center

File View Settings Help

Index Search Help

- Appearance & Themes
- Desktop
- Internet & Network
  - Desktop Sharing
  - File Sharing
  - Local Network Browsing
  - Preferences
  - Proxy
  - Samba Configuration
  - Web Browser
  - Wireless Network
- KDE Components
- Peripherals
- Power Control
- Regional & Accessibility
- Security & Privacy
- Sound & Multimedia
- System Administration

Wireless Network

Config 1 | Config 2 | Config 3 | Config 4

Network name: LINUXDAY

Operational Mode: Infrastructure / Peer-to-peer

Speed Setting: Auto, 1 MBit/s, 2 MBit/s, 5.5 MBit/s, 11 MBit/s

Execute script on connect:

Use cryptography — Key to use: 0

Crypto Mode: Open mode / Restricted mode

Crypto Keys: String: 1, String: 2, String: 3, String: 4

Enable power management — Sleep duration (ms): 0 — Awaken time (ms): 0

Receive Packets: Unicast only / Multi/broadcast only / Both

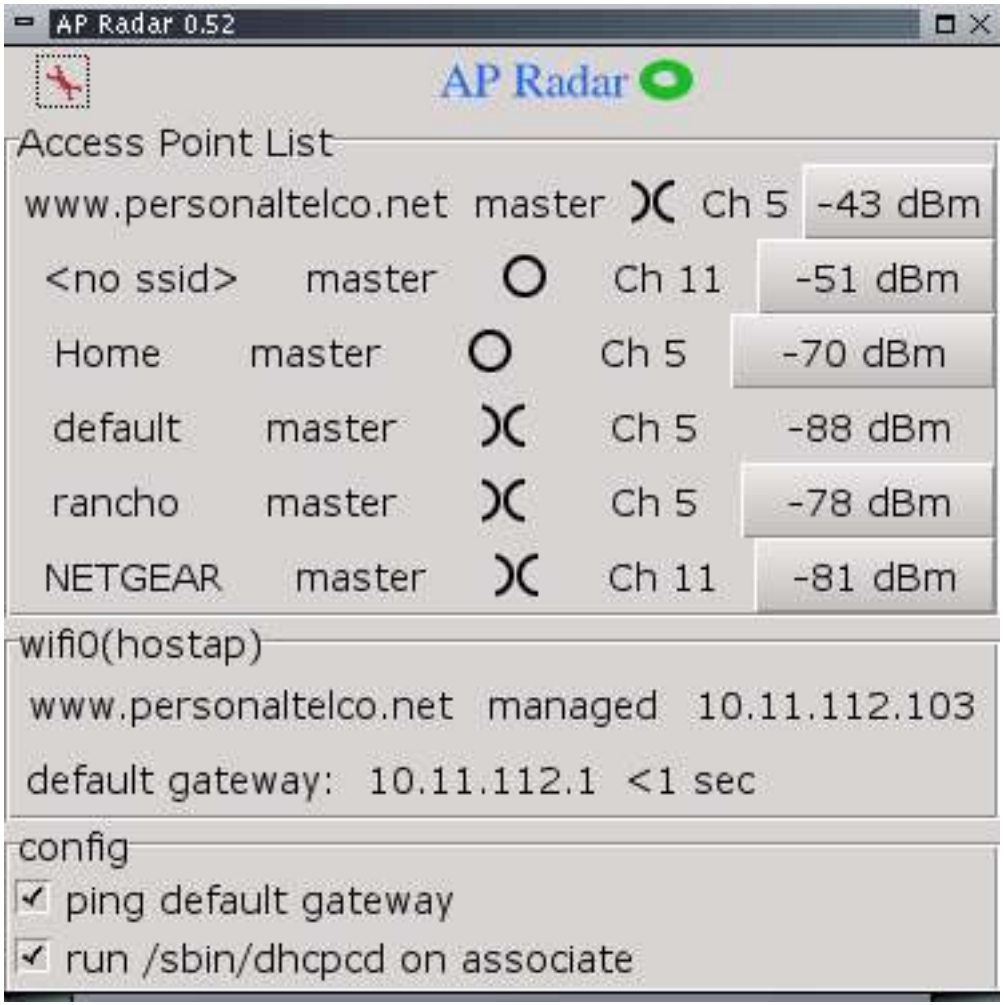
Load preset configuration on KDE startup

Configuration to load: Config 1 — Activate Configuration

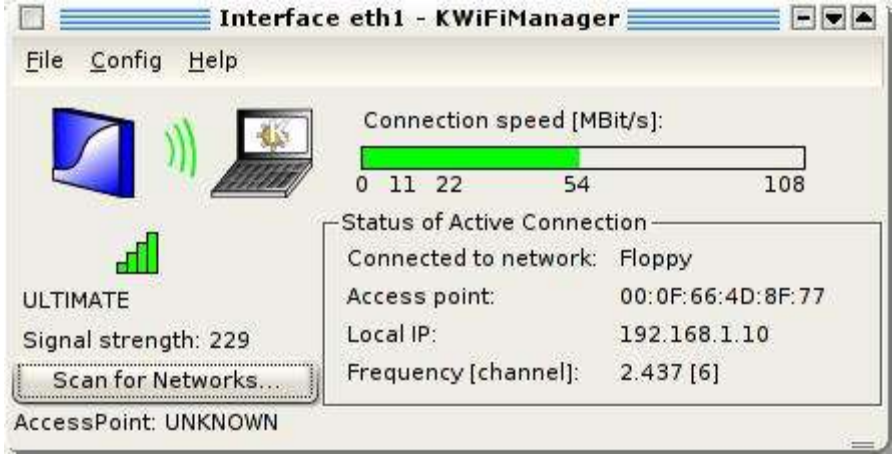
Settings apply to interface: eth1 — Autodetect Interface

Defaults — Apply — Reset

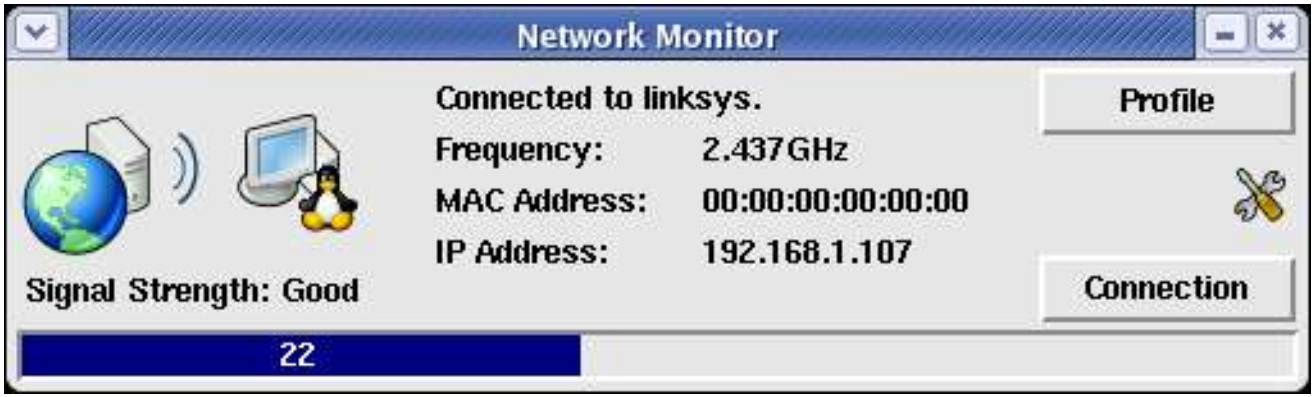
AP Radar: Configuratore scritto in GTK
(http://apradar.sourceforge.net)



Monitors:



KwifiManager



TKWifi

Linux come Access Point

Abbiamo detto prima che un access point è un nodo wi-fi che ha la possibilità di funzionare da master e che fa da bridge tra la LAN cablata ed la rete 802.11

Alcuni device 802.11 come quelli con chipset Prism2 (ampiamente supportati da Linux) possono funzionare da nodi Master

Linux con le ultime versioni del kernel può funzionare da BRIDGE (switch di Layer 2)

In sostanza, una macchina linux opportunamente configurata può fare da Access Point!

Mettere in “master” mode il device

Si risolve in:

```
iwconfig [device] mode Master [altri parametri]
```

Esempio:

```
iwconfig wlan0 mode Master essid “LINUXDAY”
```

Un nodo master può accettare connessioni da altri nodi client e gestire la loro comunicazione wireless

Configurare il bridge

1. In Device Drivers -> Networking support -> Networking options abilitare: 802.1 Ethernet Bridging e ricompilare il kernel ;)

2. scaricare ed installare i bridge-tools (http://bridge.sourceforge.net/) ; con debian: apt-get install bridge-tools

3. Tirare su il bridge con brctl
(configurazione per debian 3.0):

```
iface br0 inet dhcp
     bridge_ports eth0 wlan0
     pre-up ifconfig eth0 0.0.0.0 up
     pre-up ifconfig wlan0 0.0.0.0 up
     pre-up iwconfig wlan0 mode master
     pre-up brctl addbr br0
     pre-up brctl addif eth0
     pre-up brctl addif wlan0
     post-down ifconfig eth0 0.0.0.0 down
     post-down ifconfig wlan0 0.0.0.0 down
     post-down brctl delif br0 eth0
     post-down brctl delif br0 wlan0
     post-down brctl delbr br0
```

Configurazione alternativa

Su http://hostap.epitest.fi/ è possibile scaricare uno speciale driver per device con chipset Prism, che una volta compilato ed installato consente ad un host linux di funzionare da access point, ma suddividendo il carico tra la scheda ed il processore del pc in modo da lasciare compiti time-critical alla scheda, e facendo gestire all'host compiti quali: autenticazione, associazione e trasmissione tra clients. Questo si può fare grazie ad uno speciale stato del device chiamato HOSTAP

Abbinato ad un demone in user-space consente di implementare su linux criteri di sicurezza dell' 802.11i quali WPA

WPA e Linux

Occorre rivolgersi principalmente al progetto:
wpa_supplicant

http://hostap.epitest.fi/wpa_supplicant/

Drivers supportati:
- Host AP driver for Prism2/2.5/3 (WPA and WPA2)
- Linuxant DriverLoader (WPA, but not WPA2)
- Agere Systems Inc. Linux Driver (WPA, but not WPA2)
- madwifi (Atheros ar521x)
- ATMEL AT76C5XXx
- Linux ndiswrapper
- Broadcom wl.o driver
- Intel ipw2100
- Intel ipw2200

Wireless Linux

Per testare la sicurezza delle vostre LAN wireless su Linux bastano uno sniffer LAN ed un tool di crack delle chiavi WEP:

- Tool di sniffing
  http://airsnort.shmoo.com/

  Da utilizzarsi con una scheda in grado di operare in “Monitor” mode: si sniffano dai 5 a 10 milioni di pacchetti e si ha la chiave.

- Tool di cracking (airodump / aircrack)
  http://www.cr0.net:8040/code/network/

  Altra accoppiata vincente.

Un articolo molto interessante sull'argomento:

http://www.kuht.it/modules/sections/index.php?op=viewarticle&artid=152

Links di riferimento

- Driver HOSTAP
  http://hostap.epitest.fi/

- Esempio di AP embedded
  http://www-106.ibm.com/developerworks/library/l-wap.html?ca=dnt-429

- Linux AP - HOW-TO
  http://oob.freeshell.org/nzwireless/LWAP-HOWTO.html

- Bridge-HOWTO
  http://www.linux.org/docs/ldp/howto/BRIDGE-STP-HOWTO/

Links consigliati per il wireless

- Wireless-tools (un ottimo punto di partenza)
  http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/Tools.html
- Wireless HOW-TO (ottima lista di drivers)
  http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/Wireless.html
- Distribuzione adatta a soluzioni di AP embedded
  http://www.nycwireless.net/pebble/
- Tool di sniffing (per chi vuole sperimentare)
  http://airsnort.shmoo.com/

Fine trasmissioni