Wireless Networking in Linux

Types of Wireless Networking

There are many types of wireless networking / communication out there. In this guide were going to concentrate mostly on installing and setting up 802.11b wireless networking. The IEEE[1] 802.11 spec wasn't the first wireless networking solution but it has become one of the most dominate and commercial driven solution. A small breakdown of the other wireless networking standards is below.

Standard	***Speed***	***Distance***	***Pro***	***Con***
802.11a	54Mbps	25-75ft	High Bandwidth	Short Distance
802.11b	11Mbps	100-150ft	Lower Bandwidth	Interference
802.11g	54Mbps	100-150ft	High Bandwidth	Long Distance, Interference
HomeRF	10Mbps	100-150ft	Lower Bandwidth	Not common
Bluetooth	723Kbps	30ft	Low Power	Low Bandwidth, Interference
X.25	9600bps	10-100mi	Long Distance	Low Bandwidth
IrDA	115Kbps	15ft	Common in Laptops	Low Bandwidth, Line of sight

The 802.11 spec is the most common. It comes in three flavors so far. The 802.11a[2] is high bandwidth and communicates using the 5GHz spectrum. There isn't much interference with using this spectrum but it is limited to distance by doing so. The in-door range is usually around half of what 802.11b[3] or 802.11g[4] is. The 802.11b spec is a lower bandwidth spec than the 802.11a but can transmit up to about 150ft in-doors using the 2.4GHz spectrum. The 802.11g spec is a improvement to the 802.11b, boosting the transfer rate to around that of 802.11a. Unfortunately 2.4GHz frequency is one of the public domain frequencies that any company is allowed to use. So there are plenty of other devices that are using this frequency. Cordless phones and microwaves spring to mind. Currently there is support in Linux for 802.11a and 802.11b, but the 802.11g drivers are still in the works.

The main competitor to 802.11b when it came out was HomeRF[5]. HomeRF has all the same specs as 802.11b but the companies that were backing it really didn't do a good job of marketing it. I haven't seen any HomeRF drivers for Linux, or any hardware in any of the common computer shops.

Bluetooth[6] was made by IBM to replace the aging IrDA[7] spec for short range communication. It is relatively low bandwidth, only about 700Kbps max, and is designed for low power use. Most newer mobile devices such as PDAs, cell phones, laptops are coming with bluetooth already installed. There are drivers in Linux for both IrDA and bluetooth.

The X.25[8] amature radio spec uses multiple different VHF and UHF frequencies that are regulated. It has really low bandwidth, but what it lacks in bandwidth it makes up in distance.

Security

Real security is 802.11 is lacking in most areas. WEP (Wired Equivalence Protocol) has been cracked multiple ways and tends to only act as a deterrent. WEP cracking can take a long time, and increasing the bit level only prolongs the inevitable.

For most people turning on 128-bit WEP along with MAC Address filtering, and changing your SSID will stop most people. Most war drivers are just out either mapping networks or looking for free Internet. If the 802.11 is being used for business purpose you should consider using the above along with a vpn connection between your device and the wired portion of the network.

802.11 Drivers

The Linux Kernel has basic support for Aironet and Hermes based chipsets. The Aironet driver mainly supports the Cisco Wi-Fi card, and the Hermes supports almost any Hermes/Wavelan based chipset like the Orinoco and the Prism2. These drivers seems to work well if your not trying to do network debugging, but you still need the pcmcia-cs[9] scripts to bring the card on line and set everything up. The pcmcia-cs package support more types of cards than the kernel does and is usually a lot more up to date.

The next 802.11 driver is the host-ap[10] drivers. This driver set was originally interned to run access points for custom wireless routers but has found its way into being used on laptops because of its good support for the latest wireless extensions and for its sniffing capabilities. This driver has good support for most of the generic Prism2/2.5/3 cards. The wireless extensions allow the driver to really status information thought the /proc for real time strength and noise ratio readings.

Finally there is the linux-wlan-ng drivers[11]. This is one of the more popular 802.11 drivers as they support the most cards and are usually on the bleeding edge of all new hardware. The scripts that are provided by the package are worlds better than the pcmcia-cs wireless scripts. As they are always working on the newer hardware, the wireless extensions version are usually out of date and you can't always use the newer status monitors with this driver. This driver tends to support most cards on the market. Everything from basic Prism2 pcmcia card to newer USB Prism3 cards are usually supported.

Kernel/PCMCIA-CS Setup

For the pcmcia-cs wireless drivers or the kernel drivers using the pcmcia-cs scripts. You need to edit the /etc/pcmcia/wireless.opts file in you favorite editor.

First thing is to remove the lines in between the “START SECTION TO REMOVE” and the “END SECTION TO REMOVE”. This will activate this configuration file. By default wireless support is turned off for some reason. You also need to comment out the 'Pick up any Access Point' block of code. If you see your card in the listing you need to comment out that block also, we're going to be using the generic block at the very bottom of the file that looks like below, (NOTE: I've removed all the comments from the block for space consideration):

```
# Generic example (decribe all possible settings)
*,*,*,*)
    INFO="Fill with your own settings..."
    ESSID=""
    NWID=""
    MODE=""
    FREQ=""
    CHANNEL=""
    SENS=""
    RATE=""
    KEY=""
    RTS=""
    FRAG=""
    IWCONFIG=""
    IWSPY=""
    IWPRIV=""
    ;;
```

First we're going to setup the generic information Thats not access point dependent. Setup the following accordingly:

```
NWID=”0100”
FREQ="2.425G"
RATE="auto"
```

Next we're going to be setting up the access point dependent sections. You want to set the ESSID to that of your access point. The ESSID is the SSID (Service Set Identifier) of you access point. This is a string up to 32 characters that identifies your network. Even if you have two networks with-in range of each other on the same channel they wont be able to see each other if the SSIDs are set differently.

Most wireless access points are setup in infrastructure mode by default so set the MODE=”Managed”. In infrastructure mode all communication goes thought a access point. The access point acts as kind of a switch. Compared to ad-hoc mode, which is more of a peer-to-peer mode. In ad-hoc mode you don't need a access point and just need to connect multiple wireless devices together.

You need to now set the channel for the access point your going to be using. The channel is a 1-11 number and specifies a different sub frequency for the card to communicate on. Its a good idea to change the default channel on your access point. Channel 6 is the most common, and if you have another access point near that is using the same channel they'll cut into each others bandwidth.

Finally you can set the KEY variable if your using WEP. This value can be set using the colon delimited hex values like those that are shown in the linux-wlan-ng section below, or you can use a pure ASCII string. There are tools that come with the wireless-tools package for generating the wep keys in both formats.

As you can see this method of setting up your wireless network isn't all that flexible. In the next section we'll go over the linux-wlan-ng driver which is much easier to setup multiple access points.

Linux-wlan-ng Setup

For the linux-wlan-ng driver setup is a little more flexible. All the configuration files are in the /etc/wlan directory or in the /etc/conf.d directory for Gentoo/Linux users. Your main configuration file is the wlan.conf, this is where you setup all the SSIDs for the networks your going to be accessing. Just change the SSID_wlan0=”” value to the ssid of the network your going to be accessing, like SSID_wlan0=”linksys”.

Next you need to copy the wlancfg-DEFAULT to the wlancfg-{new ssid}, like wlancfg-linksys. Then open the new file in you favorite editor. Now comes the fun part. There are a lot of options in this file but once you understand the basics its not really that hard.

The first section in that file is the WEP section, if your using WEP (which you probably should be) you need to change the following lines to true:

```
lnxreq_hostWEPEncrypt=
lnxreq_hostWEPDecrypt=
dot11PrivacyInvoked=
```

This tells the driver that WEP is required for this network. Now is the fun part, you either need access to the WEP “key” for need to know the password used to generate the key. If you know the password used to generate the key then you can run the keygen program to generate the WEP key. The WEP key should look like:

```
xx:xx:xx:xx:xx for 64-bit WEP
xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx for a 128-bit WEP
```

Enter the WEP key after the equal sign after the equal sign on the dot11WEPDefaultKey0 line. Most people don't have to worry about entering the other default WEP keys.

The AuthType can be found on the access point setup. Usually most access points will be set to opensystem, unless you turn on WEP then it changes to sharedkey.

The last thing you need to change in this file is the CHANNEL setting to match that on your access point. And you should be already to go. If you have a pcmcia card or usb device just plug it in, else if your using a pci device you need to start the wlan service on your box. This usually means running /etc/init.d/wlan start. If everything work you should be able to ifconfig and see the wlan0 device in the listing.

Links

[1] IEEE Homepage	http://www.ieee.org
[2] 802.11a Spec	http://standards.ieee.org/getieee802/802.11.html
[3] 802.11b Spec	http://standards.ieee.org/getieee802/802.11.html
[4] 802.11g Spec	http://standards.ieee.org/getieee802/802.11.html
[5] HomeRF Resource Center	http://www.homerf.org
[6] The Official Bluetooth Website	http://www.bluetooth.com
[7] Infrared Data Association	http://www.irda.org
[8] Linux Amateur Radio AX.25 HOWTO	http://www.tldp.org/HOWTO/AX25-HOWTO/
[9] pcmcia-cs homepage	http://pcmcia-cs.sf.net
[10] Host-ap driver homepage	http://hostap.epitest.fi
[11] Linux-wlan-ng driver homepage	http://www.linux-wlan.org