Linux Networking

Iowa State University
Information Technology Services

Linux Networking

- How Networking Works
- Configuring Networking in Linux
- Using system-config-network
- Network debugging
- Wireless networking
- IPv6

Iowa State University Information Technology Services
Last update 9/20/05 by jbalvanz

UNIX and the Internet grew up together. In this class, we'll look at UNIX networking as practiced in Linux, beginning with the basics of how networking works. We'll show you two ways to configure networking, and how to debug network problems. Finally, we'll look briefly at wireless network setup and the future of the Internet, Ipv6.

Networking

- Networking uses the TCP/IP protocol by default, but Linux can use other protocols to interact with other operating systems:

  MS Networking (samba)
  Novell Netware (ipx)
  AppleShare (netatalk)

There are a number of different protocols that can be used to communicate on a network, but UNIX (and therefore Linux) has historically preferred the TCP/IP protocol set. Linux is ecumenical, though; it can also communicate with Microsoft Windows, Novell Netware and Mac OS through their respective native protocols. This class, however, will concentrate on TCP/IP and leave the others for another time.

Networking References

- Much more info on TCP/IP available at:

  Linux Network Administrator's Guide by Dawson, Bautts and Purdy (O'Reilly)

  `http://www.redhat.com/docs/manuals/enterprise/RHEL-4-Manual/sysadmin-guide/`

  `http://www.redhat.com/docs/manuals/enterprise/RHEL-4-Manual/ref-guide/`

  `http://www.freeprogrammingresources.com/tcp.html`

There are a number of good references to Linux networking. The best I've seen is the *Linux Network Administrator's Guide*, available from O'Reilly. For Red Hat Linux, the first two URLs are the manuals pertinent to networking. The third URL is a good compendium of online tutorials on TCP/IP.

Network Interfaces

- All interfaces treated as devices, found under /dev/*devname*
- lo -- loopback device
  Used for testing and accessing servers running on the local machine.
  Named “localhost” with IP 127.0.0.1

Like most things in UNIX, network interfaces are treated as files under the /dev directory. Unfortunately, you can't just `cat` network devices and expect predictable results; there are special commands for working with them.

The first network device we'll discuss is lo, the loopback device. It is used for testing purposes and for accessing servers running on the local machine. That's handy when you want to use software that works with a Web or database server, but you don't want it accessible by everyone on the Internet.

Ethernet/wireless interfaces

- eth0, eth1, ...
  Ethernet and wireless cards

  Most common connections on campus

  Drivers are usually loaded as a module during startup (using insmod) but some older drivers may have to be compiled into the Linux kernel to work correctly. Usually detected at installation unless you add them later.

Ethernet and wireless cards are the most common network devices nowadays, and Linux treats them identically, giving them the names `/dev/eth0`, `/dev/eth1`, etc. I'll probably lump them together under “Ethernet interfaces” or “network cards”. The software for Ethernet interfaces is usually loaded into the kernel during startup with the `insmod` command. Some older network cards may not work (or give reasonable performance) unless their modules are compiled into the kernel.

The first Ethernet card will usually be detected when you install Linux, but others will have to be added by hand. If the Red Hat installer doesn't find it and you're not sure what it is, try booting up a Knoppix disk. The Knoppix hardware detection routines are really good.

PPP interfaces

- ppp0, ppp1, ...

  Point-to-point protocol, usually telephone modem

  Appears when you invoke the PPP client daemon, usually by running something like wvdial or kdial, to connect to an Internet service provider

Point to Point Protocol is used for Internet connections over serial interfaces, usually modems. As a general rule, making a PPP connection involves using an application like wvdial or kdial, and the /dev/pppN device becomes active when the connection is actually made. (It's also possible to run a PPP daemon on a Linux machine to provide your own dialup access, but that's beyond the scope of this class.

Other interfaces

- tk0, tk1... Token ring interfaces
- sl0, sl1... SLIP (serial, usually modem, outdated and replaced by PPP)
- plip0... Parallel-Line Interface Protocol (parallel port, like LapLink)
- cipebc0... Crypto IP Encapsulation (IP tunnel)
- isdn0... ISDN modems
- ax0, ax1... AX.25 (for amateur radio buffs)

There are lots of other interfaces, many of which are hardly ever used unless you're trying to make a connection to an old machine that doesn't have Ethernet available (or you need to transfer the Ethernet driver to in the first place). In that case, a LapLink parallel cable and a plip server (easy enough to set up) make for a relatively simple connection between two machines.

Naming of Parts

- **NIC**: Network Interface Card, the hardware that connects the machine to the network (might be built in to motherboard, might be a PCI or ISA or PCCard addon)
- **MAC Address:** hardware address of the NIC, assigned by the card manufacturer when the card is made
  Ex: 00:0a:95:a7:62:d8
- **IP Address:** 32-bit address relating machine to its "physical" location on the network (also called IP number)

I include this slide just so you know:

- The terms I'm using;
- That a MAC address has nothing to do with a Macintosh (although all modern Macs do have MAC addresses);
- I'm going to be discussing version 4 of the IP protocols, so IP addresses are 32 bits long. I'll touch on IPv6, which uses 128 bit addresses, at the end of the course.

IP Packet

- Contains the data to be transferred plus MAC addresses and IP addresses for both source and destination

Header info	IP Addresses	Actual data to be transferred

  Header info includes packet identifier, packet type, protocol to be used, time to live and checksum

- This string of bytes is "yelled" on to the subnet, and the router forwards it where it's supposed to go.

In the days of 10Base2 (coax) Ethernets and bridging hubs, a subnet of computers was sort of like people in a crowded room. When a machine wanted to send information somewhere, it was transmitted to the subnet and all the machines on the subnet “heard” the packet go by. Nowadays, using switches instead of hubs, the traffic from each machine goes to the switch and the switch forwards packets only to the machines they're intended for. It's more secure because packet sniffing becomes impossible (unless you can compromise the switch somehow) and it makes your computer run better because it doesn't have to decide what to do with all those packets that weren't intended for it.

IP Numbers

- Each interface must have a unique 32-bit IP number (at least, on its subnet)
- Usually written as four 8-bit numbers 129.186.142.36
- First three numbers determine the subnet
- Each subnet has a router that transfers packets to and from the subnet

Originally, each machine on the Internet had to have a unique IP number. If you're on a private network (about which more later) with a router doing Network Address Translation (NAT) it only has to be unique to the subnet served by the router (but the router has to have a unique IP address). It's up to the router to make sure the traffic (which will come to it with its IP address on it) finds its way to the appropriate machine.

Bear in mind that if a machine has more than one network interface (like an Ethernet card and a wireless card) it may have more than one MAC address and more than one IP address.

Where does the IP Number Come From?

- localhost is always 127.0.0.1
- PPP connections are assigned an IP when they connect
- Ethernet, wireless and similar connections get an IP number in one of two ways:
  - Static IP number
  - Dynamic addressing (DHCP or BOOTP)

So, how do you get an IP number? Ultimately, IP address ranges are assigned to organizations by INTERNIC (send mail to Hostmaster@INTERNIC.NET); from there, the organization's network adminstrator assigns IP addresses either manually or automatically (or both) through a variety of means. At ISU, that's Rod Eldridge, and IP addresses are assigned either through NetReg (the Network Registration service at http://netreg.ait.iastate.edu) or through ASW (Authenticated Services on the Web, http://asw.iastate.edu).

Static IP Number

- Assigned by the network administrator
- At ISU, primarily used for machines that must be at predictable locations (servers)
- At ISU, log in to http://asw.iastate.edu and go to Request for Services > Domain Name Service > IP Request to request a static IP number
- Also used on private networks (isolated behind a router with no DHCP, or in a location with no Internet connection)

Static IP numbers are just that; a machine gets assigned a number and it keeps that number “forever”. Originally all IP numbers were static; nowadays static IP numbers are used only for servers that must be at a predictable IP address (like DNS servers or WINS servers). But even Web and ftp servers can use dynamic addresses, so it's not as necessary anymore.

You might also use static IP addresses on small, isolated private networks for which it's too much trouble to set up a DNS server.

Dynamic IP Numbers

- A machine connecting to the net requests an address and a DHCP server gives it one
  - Most of campus buildings (including residence)
  - Most DSL routers and cable modems
  - Many Linux-based router projects
  - Microsoft's Internet Connection Sharing (Windows 98+)

Because assigning IP addresses to thousands of machines that may well (in the case of laptops) change subnets on a regular basis is a lot of work, techniques have been developed to assign machines IP numbers automatically as they connect to the network. This is called “Dynamic Host Configuration Protocol” or DHCP. If I leave the subnet and come back tomorrow, I may be given a different IP address. As long as the domain name servers keep track of where my machine is, and everyone refers to my machine by its name instead of its IP address, it doesn't matter if the IP number changes.

DHCP servers are built into most DSL routers, cable modems and wireless access points. They're also incorporated into Windows and Mac software for Internet connection sharing. It makes setting up a home network much simpler.

Private Networks

- Private network IP addresses fall in the following ranges:

  10.0.0.0 through 10.255.255.255
  172.16.0.0 through 172.31.0.0
  192.168.0.0 through 192.168.255.0

  As long as your network is isolated or connected to the Internet by a router, you can use IP numbers in this range. Assign one to each machine and keep track of what you use. Just don't use them on campus!

Private networks used to be limited to isolated nets that weren't connected to the Internet. Now they're used extensively with Network Address Translation. A router (often built in to a DSL or cable modem) provides DHCP services to the machines on the private network. When traffic comes in to the router, it figures out which of the machines on the subnet the traffic is for and routes it accordingly. This has two advantages:

- I don't have to request IP numbers for all the machines on my subnet, but only for the router;
- It's harder for the machines on my subnet to be attacked from the Internet. Since their addresses aren't valid Internet addresses, the machines can't be “seen” from the outside.

You can use a machine with a private address on a normal Internet subnet, but normally routers are set to block traffic from private addresses, so you won't get outside the subnet. At ISU, we also use private networks for two special purposes:

- A machine connected via a virtual private network (VPN) is given a 10.15 address. The internal ISU routers will pass traffic for these addresses, so the machine can access ISU servers, but the border routers will not allow them to pass outside into the Internet.
- A machine that has not been registered with NetReg is given a 10.11 address. These machines can only get to selected servers on the campus network, including a Web proxy server that will connect them only to places where antivirus and Windows updates may be obtained.

Configuring Networking

- Basic networking (first Ethernet card) is configured during installation
- Networking changes can be done in two ways:
  - Edit configuration files, restart interface with ifdown *iface* and ifup *iface* (or "ifconfig *iface* down" and "ifconfig *iface* up")
  - Use the Network Administration Tool
    system-config-network

When you install most distributions of Linux, the installer detects the first Ethernet card and asks you for configuration information. If you have more than one Ethernet/wireless card, you'll have to add and configure the subsequent cards manually.

In Red Hat Linux you can configure networking either by editing the configuration files manually, or with the Network Administration Tool, system-config-network. Be warned, however, that using system-config-network will overwrite the configuration files you've edited by hand, so if you have anything special you've done (for whatever reason) it will be lost.

Network Configuration Files

- **/etc/sysconfig/network**
  Basic information about the computer; whether or not networking is on and the hostname
- **/etc/sysconfig/network-scripts/ifcfg-*iface***
  Configuration information for interface *iface*, plus ifup and ifdown scripts
- **/etc/hosts**
  IP numbers for "special" machines
- **/etc/resolv.conf**
  Info on domain name servers and search domains

Networking in Linux is governed by three special files and the scripts in a directory. These files can be modified in Red Hat Linux using system-config-network or with a text editor (but not, as we have seen, both).

/etc/sysconfig/network

- Sets whether or not networking is on, the hostname and (optionally) the gateway device and IP.

```
NETWORKING=yes
HOSTNAME=hostname.subdomain.iastate.edu
GATEWAYDEV=eth0
GATEWAY=129.186.144.254
```

- When using DHCP, the DHCP client will create this file for you.
- The hostname should also be in /etc/hostname for compatibility.

The file `/etc/sysconfig/network` determines whether or not networking is on and what device acts as the network gateway. (It's possible to have multiple gateways, but this is beyond the scope of this class.) In this example, eth0 is the network connection. Somewhere on the subnet the gateway device is connected to is a gateway router, which must be specified here. At ISU the router's IP address usually has the final number 254, but this is a convention rather than a requirement.

If you're using DHCP most of the information in this file comes from the DHCP server, and so you don't have to create or modify this file at all.

/etc/sysconfig/network-scripts

- Configurations for the different network interfaces are located in /etc/sysconfig/network-scripts/ifcfg-*iface*
- The scripts ifup and ifdown can be used to start and stop interfaces:

```
ifup interface
ifdown interface
```

- Changes to configurations can be done manually or with the Network Configuration tool (system-config-network) --- we'll show you both before we're done.

The manual procedure for modifying network settings is as follows:

1. Make changes to the appropriate `/etc/sysconfig/network-scripts/ifcfg-iface` file;
2. Type `/sbin/ifdown iface` to shut down the interface.
3. Type `/sbin/ifup iface` to restart the interface; the new settings will be read from the configuration file at startup.

All these actions have to be done as root unless the interface configuration file contains the line `USERCTL=yes`. Making that change is probably a bad idea...

---

ifcfg-eth0, static IP on isolated network

```
DEVICE=eth0
# static IP, do not use a boot protocol
BOOTPROTO=none
# activate interface at startup
ONBOOT=yes
NETWORK=192.168.1.0
NETMASK=255.255.255.0
IPADDR=192.168.1.27
# do not allow users to enable and disable
USERCTL=no
```

The first line of this file, DEVICE=eth0, tells which device the file governs. Whether or not the machine asks for an IP number at startup is determined by the BOOTPROTO variable; here's it's “none”, which means “don't ask for an address, I already have one”. “ONBOOT=yes” means that the interface should be activated during startup.

If you're using a static IP number, you'll have to include the network, the netmask and the IP address in the ifcfg-eth0 file. Notice that this machine is using a private network address.

Lines beginning with a pound sign (#) are comments; you can put anything you want there.

---

ifcfg-eth0, dynamic IP with DHCP

```
DEVICE=eth0
# use DHCP for configuration information
BOOTPROTO=dhcp
# activate on startup
ONBOOT=yes
```

Here's an example using DHCP, possibly the simplest approach you can take. The device is eth0, and you use DHCP at startup time when the interface is activated.

[Exercises:

1. Become root and look at the file /etc/sysconfig/network-scripts/ifcfg-eth0.
2. Bring down the eth0 interface with `/sbin/ifdown eth0`
   Open Mozilla and try accessing any Web page.
3. Restart eth0 with `/sbin/ifup eth0`. Try opening a Web page again.]

PPP connections

- For the most part you don't need to modify ifcfg-ppp*n*; if you use wvdial, Kppp or a similar tool to make your connections, it will manage that file for you.
- You may need to modify ifcfg-ppp*n* and/or dialing scripts manually if your PPP service has strange requirements (ISU's doesn't) or you have a cranky modem

Looking at the PPP configuration files is mostly academic. Whatever tool you use to dial in to your Internet service provider (wvdial, Kppp, etc.) will modify that file for you with a neater interface. Still, you may have to make changes to the scripts if you have a weird modem or your ISP has weird requirements (ISU doesn't).

Typical ifcfg-ppp0 (page 1 of 2)

```
DEVICE=ppp0
NAME=test
# Name in WVDIAL's configuration list
WVDIALSECT=test
# Modem device and serial port speed
MODEMPORT=/dev/modem
LINESPEED=115200
# name used for PAP authentication at dialup
PAPNAME=jbalvanz
# User can activate and deactivate PPP
USERCTL=true
# Do not activate on startup
ONBOOT=no
```

This particular ifcfg-ppp0 was created by wvdial. Notice that it buries its own configuration variables (like WVDIALSECT) into this file. The device used to connect in MODEMPORT is the symbolic device /dev/modem. You will need to either change this in wvdial or use the command

```
ln -s /dev/ttyS0 /dev/modem
```

to associate the /dev/modem device with the right physical device.

---

ifcfg-ppp0 (p. 2 of 2)

```
# do not force reconnect if connection drops
PERSIST=no
# use this interface as the default route
DEFROUTE=yes
# modify /etc/resolv.conf with host's DNS info
PEERDNS=yes
# do not automatically open PPP on demand
DEMAND=no
# hang up after 10 minutes inactivity
IDLETIMEOUT=600
```

Domain Names

- Used so you can remember easy names (like www.iastate.edu) instead of 129.186.1.122
- When it doesn't know, your machine asks the domain name server (DNS) what IP number corresponds to the name it has.
- Controlled by two files in Red Hat Linux:
  `/etc/hosts`
  `/etc/resolv.conf`

In a simpler world everything could be done by IP address, but humans don't remember crazy strings of numbers very well. So, we created domain names. Your machine may have a table of domain names, called `/etc/hosts`, that it can use as a local address book to find an IP number's given domain name. If it isn't there, it will query a domain name server to look up the IP number. It finds the IP numbers for the domain name servers in `/etc/resolv.conf`.

/etc/hosts

- Normally used only on isolated networks without domain name servers, or for those machines you have to be able to connect to even if the DNS isn't available. On a typical isolated network:

```
127.0.0.1      localhost.localdomain localhost
192.168.0.1    pavillion
192.168.0.2    jeffs486
192.168.0.10   duron    fileserver
```

- Second names are called aliases

In the beginning everyone on the Internet copied the hosts file, which contained the IP numbers and names of all the machines on the Internet, to their local hosts file. That's not practical any more, but hosts still has a couple of uses. If you have a small, isolated network you can just put the names in the hosts file and not bother with DNS. And if there are machines that you really need to be able to connect to even when DNS isn't available, put those entries in the hosts file. Your machine will always look there first for an IP number, so it can know those addresses even if it can't connect to a domain name server. (Just be sure that you keep it up to date; if the IP number of the machine changes, you won't be able to reach it until you change or remove the hosts file entry for it.)

/etc/resolv.conf

- Tells Linux what machines to ask for DNS info if the name given isn't in /etc/hosts
- You may not have to make this if using DHCP or PPP; those clients can create "/etc/resolv.conf" on connection
- Typical /etc/resolv.conf for Iowa State's network:

```
search ait.iastate.edu
nameserver 129.186.142.200
nameserver 129.186.140.200
nameserver 129.186.1.200
```

You'll have to create `/etc/resolv.conf` if you're using a static IP number. If you're using DHCP or PPP, the client will receive this information from the server and create the file on its own.

The "search" line lets you abbreviate domain names. If you enter "jeff3" as a domain name, and if the DNS can't find that name, your machine will add the search string to the name and try again.

You can have more than one nameserver specified (at ISU, we have three), just add more nameserver lines to the file.

Networking Commands

- Most of network configuration can be done with a small number of text-mode commands:

```
hostname
ifconfig
route
```

- From a GUI, you can use the Network Administration Tool (system-network-config)

There are only a few commands required to make changes to network configuration in a text console. The hostname, ifconfig and route commands allow you to change the network interfaces on the fly, without restarting the machine and in many cases, without restarting the interface. You can also use system-network-config to do this, of course, but that requires an X connection. You can make changes through a serial terminal with text mode.

But if you're connected to the machine through an ssh session, be careful not to bring down the interface you're connecting to the machine through; you'll lose your session. (In other words, put the commands for bringing the interface up and down in a script and run it in the background. You'll fall off, but the interface will come right back and you can log in again.)

---

hostname

- Sets the hostname in /etc/sysconfig/network and /etc/hostname

```
hostname machinename
```

- This is normally done during startup by the script /etc/rc.d/sysinit; you shouldn't need to do it manually.
- Use the shorthand name (emperor) instead of the fully-justified domain name (emperor.ait.iastate.edu)

The hostname command just changes the machine name in the two files mentioned in the slide. The `/etc/hostname` file is there for compatibility with older UNIX software; most new stuff knows to look in `/etc/sysconfig/network`.

---

ifconfig

- Used to get statistics and set configuration info about network devices
- Common to distributions other than Red Hat (is used in Debian, for instance) so will probably be available even if you're not on your standard machines.
- To examine the settings and statistics for an interface, type

```
ifconfig iface
```

The `ifconfig` command can be used to make changes to network configuration without changing the configuration files. This command is common to most Linux distributions, so it will work even if you don't have system-config-network (or X) on the machine.

“ifconfig eth0” output

```
eth0      Link encap:Ethernet  HWaddr 00:0B:DB:67:18:CA
          inet addr:129.186.139.204  Bcast:129.186.139.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:982598 errors:0 dropped:0 overruns:0 frame:0
          TX packets:114372 errors:0 dropped:0 overruns:0 carrier:0
          collisions:9214 txqueuelen:100
          RX bytes:238620678 (227.5 Mb)  TX bytes:45184277 (43.0 Mb)
          Interrupt:9 Base address:0xdcc0 Memory:ff6e0000-ff700000
```

- “Hwaddr” is the MAC address NetReg needs
- “inet addr” is the IP address of the machine
- Note RX and TX (receive and transmit) statistics (useful in diagnosing interface problems)
- Info about ethernet card hardware appears in last line

[Exercise:

Type

```
ifconfig eth0
```

What is your machine's IP address? MAC address? Do you receive more packets, or send more? Are you seeing packet collisions? How many? (Collisions are signs of congestion on the network.)]

Activating and deactivating interfaces

- To activate an interface, type

```
ifconfig iface up
```

- To deactivate an interface, type

```
ifconfig iface down
```

This is another way to reset the network interface for those machines that don't have the ifup and ifdown scripts in /sbin.

[Exercise:

1. Start a web browser, if you haven't already.
2. Type
   `ifconfig eth0 down`
3. Verify that the web browser doesn't work.
4. Type
   `ifconfig eth0 up`
5. Verify that the browser works again.]

Setting configuration with ifconfig

- Type

```
/sbin/ifconfig iface address netmask nn.nn.nn.nn
```

Example:

```
/sbin/ifconfig eth0 129.186.139.205 netmask 255.255.255.0
```

- Setting an address triggers an automatic “up”; to change IP, bring interface down, then set address
- Other options can be used to set Ethernet card settings, IP tunneling, point-to-point connections, etc.

Changing the IP number of a machine is pretty arcane, but a server adminstrator might use it to switch machines providing a service:

1. Bring down the ethernet interface(s) on both servers;
2. Use `ifconfig` to set the new machine's IP address(es) to the one(s) formerly on the old server.

Now the new server looks just like the old one, except (hopefully) faster.

route

- Which interface and gateway does a packet use to get to a particular host? The answer is in the routing tables.
- route edits the routing tables, determining which interface packets use to get to which network host.
- route is not always in the path; if not, try /sbin/route

Usually the only routing you need on a typical Linux machine is the default gateway set way back there in `/etc/sysconfig/network`. But if you have two or more Ethernet interfaces, we may want to send some traffic through one interface and other traffic through another. That's where the `route` command comes in; it lets us specify which interface the machine will use to reach a particular subnet.

Seeing the routing table

- Type
  `/sbin/route`

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
129.186.139.0   *               255.255.255.0   U     0      0        0 eth0
127.0.0.0       *               255.0.0.0       U     0      0        0 lo
default         router-129-186- 0.0.0.0         UG    0      0        0 eth0
```

- Items with a gateway of "*" are on the same subnet as this machine; they don't need a gateway to reach them.
- The default router must be on the local subnet, or on a subnet with an explicitly defined route.
- Note that the gateway is described by domain name, not IP number. Use `/sbin/route -n` to get IP numbers.

Each of the lines in the route table describes what gateway is used to reach a particular network. In this case, the machine is on the subnet 129.186.139, so it doesn't need a gateway to get there. The subnet 127.0.0 is the local domain, so it doesn't need a gateway there either. Everything else goes to default, which is the router out of that subnet. Notice that the router's name is too long to fit in the field. The `/sbin/route -n` command will just display the IP number of the router; if you have to figure out what the name is, use the `nslookup` or `host` command.

[Exercise: use the `/sbin/route` command to view the routing table on your machine.]

Adding routes and gateways

`route add -net 129.186.141.0 netmask 255.255.255.0 dev eth1`

-- sets route to the network 129.186.141.* via device eth1

`route add default gw 129.186.141.254`

-- sets default gateway (for all traffic not otherwise routed) to 129.186.141.254

- If you need a router and have an old 486 machine lying about, see http://www.freesco.org

The `route add` command is used to specify which interface is used to reach a subnet. In this example, traffic for the 129.186.141 subnet goes through the eth1 device. We might use this where a lab is on an isolated subnet, or just for improved network performance.

The `gw` option indicates that the following address is the gateway for the previous network (in this case, `default` means that this is the address for all traffic).

When you have specified routes to different networks, they apply only to this machine; it does not automatically route traffic from other machines through this machine. If you wanted to do that, you need to run the `routed` service. Since the cost of routers has plummeted, it's not as important as it used to be.

The Network Administration Tool

- aka `system-config-network`
- A GUI tool for doing network configuration
- Really just edits the appropriate files and runs scripts to start and restart interfaces, but some people find it useful because everything is in one big place.
- Click Start > System Settings > Network, or type

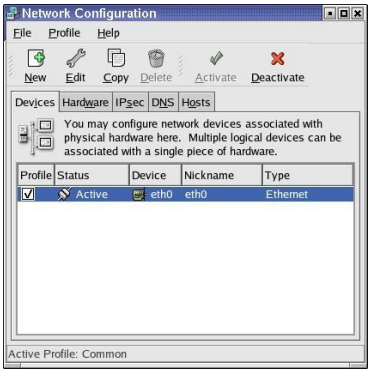
```
system-config-network &
```

Okay, you don't need the GUI tool to configure the network, and if the machine isn't running X you can't use it. But many people like system-config-network because it brings all these crazy commands together into one place and lets you click buttons to use them.

[Exercise: start system-config-network by either of the techniques at left.]

Configuring Devices

- From the Device tab you can activate or deactivate a network interface with the buttons at lower right
- To edit configuration for an interface, highlight and click Edit; to add an interface, click New



To configure a network interface, click the Device tab. Highlight the device you want to work with and:

- Click Deactivate to bring the interface down;
- Click Activate to bring it back up;
- Click Edit to modify the interface's properties;
- Click New to add a new interface. (You must already have the card added under “Hardware” to add an interface.)

[Exercise: highlight the eth0 interface and click Edit to examine the settings.]

Configuring an Interface with SCN

- To set a static IP address, turn on “Statically set IP addresses” and enter values for address, subnet mask and default gateway address

You can set a static IP address in the General tab of the interface properties. Click the radio button and enter values for IP address, netmask and default gateway address.

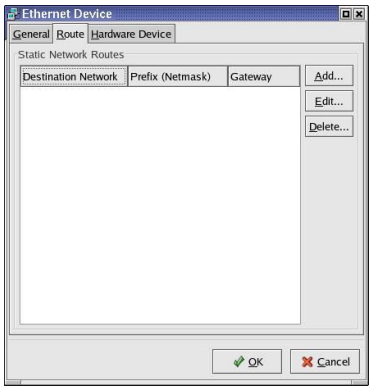
At ISU, the netmask is almost always 255.255.255.0, and the gateway is the first three parts of your IP number followed by 254. (For example, if your IP number is 129.186.140.23, your gateway will be 129.186.140.254.)

[Exercise:

1. Find out the IP number that DHCP has assigned your computer with the ifconfig command.
2. Set your IP number statically to the number you were assigned by DHCP.
3. Set it back to DHCP.]

Modifying routing with SCN

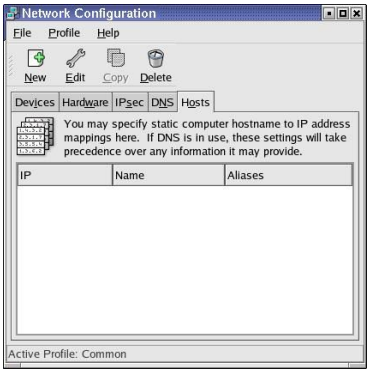
- To add a route, click "Add" and enter the network, netmask and gateway addresses (as in the route command)



If you need to add a static route, the Route tab will let you do the same thing as the route command. Click Add and enter the subnet, the netmask and the IP number of the gateway.

Managing /etc/hosts with SCN

- The Hosts tab is an interface to /etc/hosts. Click "New" to add a host, and enter IP number, name and aliases.

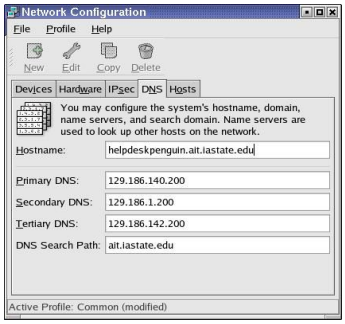


[Exercise: one use of /etc/hosts is to block access to hosts that you don't want people to contact. For example, suppose you didn't want people to get to www.cnn.com. You could do this:

1. Click New to add a host.
2. Enter the following:
   Address: 129.186.1.121
   Hostname: www.cnn.com
3. Go to the File menu and choose Save.
4. Now start Mozilla and try opening www.cnn.com. Notice that you wind up at www.iastate.edu instead!
5. Go back to the Network Administration tool and delete the static route to www.cnn.com. Try contacting CNN again (you may have to close Mozilla and open it again).]

Changing DNS with SCN

- Set machine hostname with the Hostname field.
- Add up to three domain name servers in the Primary DNS, Secondary DNS and Tertiary DNS fields.
- To specify the search path (s), enter a Search Domain and click "Add".



[Exercise:

1. Click on DNS.
2. Make up a hostname using your NetID followed by "aitlabs.iastate.edu".
3. Add the three domain name servers shown in the slide.
4. Specify the search domain as "aitlabs.iastate.edu".
5. Go to the File menu and click "Save".]

Debugging Tools

- ping – test connection to a machine
- host – get DNS information about a computer
- traceroute – follow the path of routers to a machine
- netstat – see what ports are open and what is connected to them

When you're first setting up networking on a machine – or figuring out what's wrong with networking later – there are a number of commands that are useful for testing network connections. Most of them don't require that you log in as root.

Ping

- Tests the connection to a machine

```
ping hostname
```

```
PING jeffnt.ait.iastate.edu (129.186.144.80) from 129.186.139.205 : 56(84) bytes of data.
64 bytes from jeffnt.ait.iastate.edu (129.186.144.80): icmp_seq=1 ttl=127 time=0.647 ms
64 bytes from jeffnt.ait.iastate.edu (129.186.144.80): icmp_seq=2 ttl=127 time=0.640 ms
64 bytes from jeffnt.ait.iastate.edu (129.186.144.80): icmp_seq=3 ttl=127 time=0.550 ms
64 bytes from jeffnt.ait.iastate.edu (129.186.144.80): icmp_seq=4 ttl=127 time=0.594 ms
64 bytes from jeffnt.ait.iastate.edu (129.186.144.80): icmp_seq=5 ttl=127 time=0.626 ms
64 bytes from jeffnt.ait.iastate.edu (129.186.144.80): icmp_seq=6 ttl=127 time=0.685 ms

--- jeffnt.ait.iastate.edu ping statistics ---
6 packets transmitted, 6 received, 0% loss, time 5042ms
rtt min/avg/max/mdev = 0.550/0.623/0.685/0.051 ms
```

- Press <Ctrl/C> to stop pinging (it's not polite)

If you can't reach a Web or other server, the first test is to determine if the machine is still there. We do that with ping. The ping command sends a “please answer” packet to the machine, which according to the protocol the machine must answer with a response packet. The ping command reports how long it took the machine to respond to each packet. When you have several responses, press {Ctrl/C} and ping will stop pinging and summarize its results.

It's not polite to ping a machine too many times; if enough computers were doing it it would be a distributed denial of service attack.

[Exercise: ping emperor.ait.iastate.edu, and let it run for five to seven pings. What is the average response time? Does it seem to change as more people ping it?]

---

host

- `host` returns the DNS information about a domain name or an IP number.

```
vincent% host jeffnt.ait.iastate.edu

jeffnt.ait.iastate.edu has address 129.186.144.80

vincent% host 129.186.144.80

80.144.186.129.in-addr.arpa domain name pointer jeffnt.ait.iastate.edu.
```

- `host -a` returns information about the DNS servers returning the information as well.

The host command can be used to look up a machine's IP number from its domain name or vice versa.

[Exercise: use `host` to determine the IP number of www.iastate.edu. Notice that more than one number is returned. What does this mean?]

---

traceroute

- traceroute traces the path from your machine to a remote host.

  `/usr/sbin/traceroute hostname`

```
#/usr/sbin/traceroute www.uiowa.edu
traceroute: Warning: www.uiowa.edu has multiple addresses; using 128.255.56.81
traceroute to www.uiowa.edu (128.255.56.81), 30 hops max, 38 byte packets
 1  b11sr1-10-145.tele.iastate.edu (10.10.145.251)  0.467 ms  0.395 ms  0.829 ms
 2  b31gb1-lan254-128.tele.iastate.edu (129.186.254.131)  0.441 ms  0.554 ms  0.464 ms
 3  b31br2-437.gw.iastate.edu (192.245.179.154)  0.938 ms  0.737 ms  0.714 ms
 4  rtr-border-lc.uiowa.edu (198.49.182.17)  11.432 ms  11.547 ms  11.831 ms
 5  rtr-core-lc.uiowa.edu (128.255.2.130)  11.686 ms  12.065 ms  11.197 ms
 6  lime.weeg.uiowa.edu (128.255.56.81)  12.059 ms  12.668 ms  11.079 ms
```

- Each line is a "hop" or router. The three times on each line are times to return from that machine in milliseconds. An asterisk will appear if it is unable to connect before timeout.

The `traceroute` command can be used to figure out why you can't reach a machine, or why the connection is slow. It shows every router the connection must pass through and how long it took to get a response from it. If you start getting asterisks, you know where the problem is. (TIP: you can identify who a domain belongs to with the `whois domain` command, as in `whois iastate.edu`.)

[Exercise: do a traceroute to a host of your choice – like your favorite Website.]

Netstat

- Used to determine network connections by and to your machine.

  `netstat`

```
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0    256 mommy.ait.iastate.e:ssh jlbg5.ait.iastate:49264 ESTABLISHED
tcp        0      0 mommy.ait.iastate:37248 du139-205.aitlabs.i:ssh TIME_WAIT
Active UNIX domain sockets (w/o servers)
Proto RefCnt Flags       Type       State         I-Node Path
unix  7      [ ]         DGRAM                    964    /dev/log
unix  3      [ ]         STREAM     CONNECTED     5213642
unix  3      [ ]         STREAM     CONNECTED     5213641
unix  2      [ ]         DGRAM                    1727409
unix  2      [ ]         DGRAM                    1244
unix  2      [ ]         DGRAM                    1198
unix  2      [ ]         DGRAM                    1097
unix  2      [ ]         DGRAM                    979
```

To determine what machines your computer is connected to (or vice versa) enter the `netstat` command. You'll probably want to pipe it to more (as in `netstat | more`) so you can see what is connected.

[Exercise:

1. From a shell, use `ssh -X sas.iastate.edu` to make an ssh connection.
2. From another shell, use `netstat | more`. Notice the connection to sas.iastate.edu.]

Wireless (802.11?) Networking

- Linux does support wireless networking, BUT...
  - Not all wireless cards have Linux drivers
  - There are ways around this (on i386 machines) but only sometimes
- To avoid problems in configuration, check that the card you're looking at is supported before you buy
- To use PC Cards you must also have the PCMCIA support loaded (it's a service)

The trick to remember is this: Windows XP supports new hardware better than Linux, Linux supports older hardware better than Windows XP. Many hardware vendors don't have Linux drivers and keep their hardware secret so their proprietary Windows drivers are all there is. So, before you buy a wireless card, make sure that support is available for it.

Adding a Wireless Card

- Choose System Settings -> Network
- Click Devices, then New.
- Choose Wireless connection and click Forward.
- Choose “Other wireless card”, click Forward.
- Choose your adapter from the list, click Forward three times, then click Apply.

You can use `insmod` and `ipconfig` to install and configure drivers for wireless cards, but if you're using Red Hat it's much simpler to use the Network Administration tool if the adapter matches one in the list.

You can add adapters that aren't installed, but SCN will complain when it tries to contact the card for the first time.

Working with Wireless

- Once installed, a wireless card looks like a normal Ethernet card.
- IP address, DNS, etc. is set with `ipconfig`
- Channels, etc. set with `iwconfig`

```
iwconfig ethN options
```

See the iwconfig man page for options. Or you can do it all with redhat-config-network instead.

Once you get it installed a wireless card looks pretty much like an Ethernet card and even has an eth*N* device name. There is an extra command called `iwconfig` that selects card frequency, SSID, etc. for the wireless card.

Wireless with SCN

- An extra tab appears when editing a wireless device
- For an encrypted network, enter name under SSID and key below

When you edit a wireless device in SCN, there's an extra tab in the Device Configuration window that sets the wireless parameters. By default a wireless card will connect automatically to whatever network it can find, but you can force it to connect to a specified network here. If that network is encrypted, enter the key below. (Prefix the key with 0x if you have the key in hexadecimal.)

More wireless help

- The current Wireless HOWTO is located at

  http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/
- The pertinent section of the Red Hat Enterprise manual is at

  http://www.redhat.com/docs/manuals/enterprise/RHEL-4-Manual/sysadmin-guide/s1-network-config-wireless.html

We haven't covered using wireless in any detail here. You are strongly recommended to go read Jean Tourrihes' Wireless HOWTO and/or the pertinent section of the Red Hat manual (assuming you're using Red Hat) for more information.

Ndiswrapper

- Support for NICs (particularly wireless) that don't have Linux drivers available
- Provides a Windows API-compatible layer letting the Windows NDIS driver work with Linux
- You'll need to compile from source and install manually
- http://ndiswrapper.sourceforge.net

The manufactures of some NICs don't believe in Linux, hence have not taken the time to create Linux drivers for their cards. (That's one reason you want to be careful when you buy a laptop you're planning to run Linux on.) The ndiswrapper project has created a driver that allows Linux to talk to *some* Windows NDIS drivers, letting you use them with Linux anyway. For most distributions you'll have to compile ndiswrapper from source and install it manually. (Debian does include a package for it, so you can just install it with apt-get.)

---

rfswitch

- Many laptops allow you to switch the wireless adapter on and off to save power
- Unfortunately, some laptops use a software switch – which doesn't work with Linux!
- rfswitch gives you control of the radio in *some* laptops
- http://rfswitch.sourceforge.net

(Most Dell laptops use a hardware switch, so don't need this utility.)

Wireless cards require a lot of power (they're radio transceivers, after all) and can seriously reduce your battery life if you're not using networking, so some laptops allow you to turn them on and off. Unfortunately, many laptops use a software switch which, guess what, only works with Windows. The rfswitch utility replaces that software for Linux.

IPv6

- The current version of IP (IPv4) has a problem; it only allows about 4 billion addresses (32 bits)
- IPv6 uses a 128 bit address, allowing 340282366920938463463374607431768211456 (three hundred forty undecillion) addresses
- There are also many other improvements in security and efficiency (IPsec is required, NATs are unnecessary...)

The big problem with IP is that there soon won't be enough IP numbers. With only four billion possible IP numbers (and a lot of those wasted on subnets that don't have 255 machines in them) we'll run out long before everyone on earth has their own Internet connection. IPv6 uses a larger address field (128 bits) thus giving the possbilitity of a unique IP address for every grain of sand on the planet.

Linux Support for IPv6

- Supported (badly) in the 2.4 kernel, better in 2.6; check for "file" `/proc/net/if_inet6`
- Support can be added with `insmod ivp6`
- Some utilities support it, others not
- Howto and status at http://www.tldp.org/HOWTO/Linux+IPv6-HOWTO/

The Linux 2.4 kernel did have support for IPv6, but it works much better in Linux 2.6. Red Hat Enterprise Linux V. 3 does have support but doesn't load it by default. You can load it with `insmod ipv6`, but you can't unload the module; you have to restart to remove it. Not all of the utilities believe in a 128-bit address space, anyway. Check the HOWTO listed below for more details.
