Network Debugging mini-HOWTO

This HOWTO is aimed at solving linux newbie problems with ethernet setup's similar to those here at Rice. I.e. if you have:

- Multiple ethernet cards.
- Dynamic IP assignment
- A modem

then this document is not for you. Also, I realize that their may be easier ways to do this (such as netcfg under Redhat), but the below method should work for all linux distros, and you can automate it later once you verify that you have good hardware and good drivers.

Getting Static Ethernet to Work

First off you need some information from your ISP. You can't guess at these values or make them up (nor can you use my example values from Rice). An IP address is a series of 4 numbers from 0 to 255 usually connected with periods, such as 128.42.5.4 or 192.168.1.1. You'll need to know:

- The make and model of your network card. You may need to know the chipset instead. This can usually be discovered by looking at the physical card in your machine and reading the manufacturers names printed on the chipset.
- The IP address of your machine. I'm gonna call this *<yaddr>*.
- The IP address of your gateway. I'm gonna call this *<gaddr>*.
- The netmask for your network or alternately, the network address for your network. I'm gonna call this *<maddr>* for the mask address or *<naddr>* for the network address. If you just have a network IP address, then you can calculate the netmask IP by looking at the last few digits of the network address:

network addr	netmask
ddd.ddd.ddd.ddd	255.255.255.248 (probably)
ddd.ddd.ddd.0	255.255.255.0
ddd.ddd.0.0	255.255.0.0
ddd.0.0.0	255.0.0.0

  where `ddd` is any number from 1 to 254.
- The IP Address of your primary DNS (you may have an IP for the secondary and tertiary DNS too, this is a good thing but not essential). I'm gonna call this *<daddr>*.

Step 1: Check if your interface is configured

The first thing to look at is to see if linux believes you have a network card set up and configured. You change network parameters with a program called `ifconfig` and if you run it with no arguments, you should get a list of network interfaces (which is a fancy name for network cards). For example, here at Rice, I get the following:

```
% ifconfig
eth0      Link encap:Ethernet  HWaddr 00:60:08:93:0E:05
          inet addr:128.42.14.176  Bcast:128.42.14.255  Mask:255.255.255.0
          EtherTalk Phase 2 addr:49600/201
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:33933225 errors:0 dropped:0 overruns:0 frame:0
          TX packets:38375051 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          EtherTalk Phase 2 addr:0/0
          UP LOOPBACK RUNNING  MTU:3924  Metric:1
          RX packets:3527806 errors:0 dropped:0 overruns:0 frame:0
          TX packets:3527806 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0
```

You should have at least two entries, one for the loopback device (lo), and one for your ethernet card (eth0). If you have them both, then you can skip the next section.

Step 2: Setting up your interface

There's two parts to this. The first is loading up the drivers to your card. The second is configuring the interface. In order to check if the drivers are loaded, run the command `dmesg | grep -3 -i eth`. You should get back some information about your card:

```
% dmesg | grep -3 -i eth
SCSI device sda: hdwr sector= 512 bytes. Sectors= 8498506 [4149 MB] [4.1 GB]
SCSI device sdb: hdwr sector= 512 bytes. Sectors= 8498506 [4149 MB] [4.1 GB]
3c59x.c:v0.99H 11/17/98 Donald Becker http://cesdis.gsfc.nasa.gov/linux/drivers/vortex.html
eth0: 3Com 3c905 Boomerang 100baseTx at 0xef00,  00:60:08:93:0e:05, IRQ 11
  8K word-wide RAM 3:5 Rx:Tx split, autoselect/MII interface.
  MII transceiver found at address 24, status 786f.
  Enabling bus-master transmits and whole-frame receives.
```

dmesg returns information that your system discovered at boot time. grep allows us to focus on the ethernet part. Here you see that linux discovered a 3Com card in my machine. If this doesn't work, it means that you don't have support for the card. You'll need to recompile your kernel with support for your card. See the kernel-HOWTO for more detailed information on this, but you basically need to run: `cd /usr/src/linux`, `make menuconfig`, `make deps`, `make clean`, `make zImage`, `cp /usr/src/linux/arch/boot/i386/zImage /boot/vmlinuz`, `lilo`, and `shutdown -r 5`. In menuconfig, you'll need to add support for your card, as well as support for TCP/IP networking. You really should read the kernel-HOWTO in order to do it right.

If dmesg returns info about your card, then you just need to configure it. You'll need to run the following command as root `ifconfig eth0 *<yaddr>* netmask *<naddr>* up` where*<naddr>* is your netmask IP address and *<yaddr>* is your IP address. For example, I use:

```
# ifconfig eth0 128.42.14.176 netmask 255.255.255.0 up
```

If you don't have the loopback interface up (as reported by ifconfig), then you should set it also at this time by running `ifconfig lo up` as root. You should now be able to ping your own machine, by running both:

```
% ping -c 1 <yaddr>
PING <yaddr> (<yaddr>) from <yaddr> : 56 data bytes
64 bytes from <yaddr>: icmp_seq=0 ttl=255 time=0.1 ms

--- <yaddr> ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 0.1/0.1/0.1 ms
```

and

```
% ping -c 1 127.0.0.1
PING 127.0.0.1 (127.0.0.1) from 127.0.0.1 : 56 data bytes
64 bytes from 127.0.0.1: icmp_seq=0 ttl=255 time=0.1 ms

--- 127.0.0.1 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 0.1/0.1/0.1 ms
```

If that doesn't work, then you can't move on until it does. Try reading the Ethernet-HOWTO, NET-3 HOWTO, and the Kernel-HOWTO for more information, or ask your local guru for help. :-(

Step 3: Check if you can ping the gateway

The gateway is the machine that routes all your packets to the outside world. If you can get ping packets (simple network traffic) to it, then you should be good to go. Most of the problems after this are relatively easy to solve. In order to test this out, run \`ping -c 3 *gateway ip*\` , with the actual IP address of the gateway substituted in for the gateway. For example, here at Rice my gateway is 128.42.14.1 and I get output that looks like:

```
% ping -c 3 128.42.14.1
PING 128.42.14.1 (128.42.14.1) from 128.42.14.176 : 56 data bytes
64 bytes from 128.42.14.1: icmp_seq=0 ttl=255 time=0.8 ms
64 bytes from 128.42.14.1: icmp_seq=1 ttl=255 time=0.6 ms
64 bytes from 128.42.14.1: icmp_seq=2 ttl=255 time=0.6 ms

--- 128.42.14.1 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 0.6/0.6/0.8 ms
```

This means that it worked. I tried to send 3 packets to the gateway and all 3 came back. If you can ping your gateway, then skip the next section.

Step 4: You can't ping the gateway.

There's several possible things that could be wrong at this point:

- Check that the cable is plugged in. This solves more networking problems than all other solutions that I've tried.
- you could have a bad cable: Check the link light (it's green) on the card or the hub (not all cards have them), it should be on. try swapping your ethernet cable with a friend, and see if that solves the problem.
- you could have the card setup incorrectly, see step 2.
- you could have a hardware conflict with your card, check the output of \`cat /proc/interrupts\` , \`cat /proc/ioports, and \`cat /proc/pci\` and see if any other device is

trying to use the same hardware resources as your card. If this is the case, then try pulling other devices out of your box and see if the card works, or talk to your local hardware guru.

If the network card is installed correctly (with both drivers and the interface set up), then you have passed beyond the capabilities of this document to help you. bummer.

Step 5: You can ping the gateway.

Woohoo! Everything is easy at this point. Next thing to try is to ping past the gateway. Your DNS server is a good place to try:

```
% ping -c 1 <daddr>
PING <daddr> (<daddr>) from <yaddr> : 56 data bytes
64 bytes from <daddr>: icmp_seq=0 ttl=253 time=1.6 ms

--- <daddr> ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 1.6/1.6/1.6 ms
```

If you can ping the DNS server then skip the next section.

Step 6: Setting up routing

The packets are going to the gateway but you can't talk to anybody else, you need to setup your computer to route the packets. Routing is just a set of rules that control how network traffic is directed. You can check these rules by running `route`. Usually, your routing table will have four entries as follows:

```
% route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use
Iface
<yaddr>         *               255.255.255.255 UH    0      0        0 eth0
127.0.0.1       *               255.255.255.255 UH    0      0        0 lo
<naddr>         *               <maddr>         U     0      0        0 eth0
default         <gaddr>         0.0.0.0         UG    0      0        0 eth0
```

These four entries correspond to the following:

- Packets bound for my machine. Accept them.
- Packets bound for my machine. Accept them.
- Packets bound for my network. Pass them out through my ethernet card.
- Packets bound for anywhere else (default). Give them to my gateway to handle.

If you are missing any of these entries, then you need to setup the missing routes with the route command. You need to run the following commands as root to setup routing correctly:

```
# route add -net <naddr> netmask <maddr> eth0
# route add default gw <gaddr> eth0
```

You should probably also setup the loopback route if it's missing but this isn't essential:

```
# route add -host 127.0.0.1 lo
```

Step 7: Name resolution

This is the part of networking that maps www.yahoo.com onto an IP address. You'll need to have 3 files: /etc/host.conf, /etc/hosts, /etc/resolv.conf. Respectively, they should contain:

/etc/host.conf:

```
order hosts,bind
multi on
```

/etc/hosts:

```
127.0.0.1       localhost loopback
<yaddr>         this.host.name
```

If you don't have a hostname, then you can leave off the second line.

/etc/resolv.conf:

```
domain yourdept.yourcompany.com
search yourdept.yourcompany.com yourcompany.com
nameserver <daddr>
```

You can add additional 'nameserver *<daddr>*' lines if you have a secondary (or tertiary) DNS. The search line is just a list of domains to be searched when you type an incomplete hostname. The domain line corresponds to the domain that you belong to.

Last step: Automation

You should have networking up and working but you don't want to run the ifconfig, and route commands every time you reboot. First off, there maybe a "better" way to do this (via linuxconf or netcfg, or whatever). Read their documentation if you want to do it that way. To automate the above process, you just need to put the correct steps for your box (ifconfig and route only) into a file, and then put it in /etc/rc.d/init.d/net and put an appropriate symlink to the file from the /etc/rc.d hierarchy.

Other problems

First off, don't mail me with your networking problem. I don't want to solve it. sorry. mail me if you have suggestion about this document. In the meantime, here's some other things to consider:

- Check that the cable is plugged in. This solves more networking problems than all other solutions that I've tried.
- Command not found. Program *xyz* that you told me to run isn't on my system. Actually, they probably are, but they might not be in your path, some common locations are:
  - /bin/ping
  - /sbin/ifconfig
  - /sbin/route

  Try the full pathname and see if that solves the problem. Else you might need to get the net-tools. See the NET-3 HOWTO.

- Other resources:
  - The Networking HOWTO. Has all the above information and more.
  - The Ethernet HOWTO. Useful for getting your card to work.
  - The Linux Kernel HOWTO. Info on recompiling your kernel.
- Useful manpages:
  - ifconfig(8)
  - route(8)
  - resolv+(4) or resolv(4) or resolv.conf(4)
  - hosts(4)