Linux Home Networking I
Inside The Home

CHAPTER 1

Introduction to Networking 1

Chapter 1

Introduction To Networking

= =

In This Chapter

= =

This chapter briefly explains some basic networking concepts for the home user.

What Is TCP/IP?

TCP/IP is a universal standard suite of protocols used to provide connectivity between networked devices. It is part of the larger OSI model upon which most data communications is based.

One component of TCP/IP is the Internet Protocol (IP) which is responsible for ensuring that data is transferred between two addresses without being corrupted.

For manageability, the data is usually split into multiple pieces or “packets” each with its own error detection bytes in the control section or “header” of the packet. The remote computer then receives the packets and reassembles the data and checks for errors. It then passes the data to the program that expects to receive it.

How does the computer know what program needs the data? Each IP packet also contains a piece of information in its header called the “type” field. This informs the computer receiving the data about the type of transportation mechanism being used.

The two most popular transportation mechanisms used on the Internet are Transmission Control Protocol (TCP) and User Datagram Protocol (UDP).

What is TCP?

TCP opens up a connection between client and server programs running on separate computers so that multiple and/or sporadic streams of data can be sent over an indefinite period of time. TCP keeps track of the packets sent by giving each one a sequence number with the remote server sending back “acknowledgement” packets confirming correct delivery. Programs that use TCP therefore have a means of detecting connection failures and requesting the retransmission of missing packets. TCP is a good example of a “connection oriented” protocol.

What is UDP?

UDP is a connectionless protocol. Data is sent on a “best effort” basis with the machine that sends the data having no means of verifying whether the data was correctly received by the remote machine. UDP is usually used for applications in which the data sent is not mission critical. It is also used when data needs to be broadcast to all available servers on a locally attached network where the creation of dozens of TCP connections for a short burst of data is considered resource hungry.

What are TCP / UDP Ports?

So the data portion of the IP packet contains a TCP or UDP segment sandwiched inside. Only the TCP segment header contains sequence information, but both the UDP and the TCP segment headers track the “port” being used. The source/destination port and the source/destination IP addresses of the client & server computers are then combined to uniquely identify each data flow.

Certain programs are assigned specific ports that are internationally recognized. For example, port 80, is reserved for HTTP web traffic and port 25 is reserved for SMTP email. Ports below 1024 are reserved for privileged system functions, those above 1024 are generally reserved for non system third party applications.

Usually when a connection is made from a client computer requesting data to the server machine that contains the data:

- the client selects a random unused "source" port greater than 1024 and queries the server on the "destination" port specific to the application. If it is an HTTP request, the client will use a source port of say, 1095 and query the server on port 80 (HTTP)
- The server recognizes the port 80 request as an HTTP request and passes on the data to be handled by the web server software. When the web server software replies to the client, it tells the TCP application to respond back to port 1095 of the client using a source port of port 80.

The client keeps track of all its requests to the server's IP address and will recognize that the reply on port 1095 isn't a request initiation for "Nicelink" (See the Bibliography for a link to a TCP/IP port listing), but a response to the initial port 80 HTTP query.

What is a TTL?

Each IP packet has a Time to Live (TTL) section that keeps track of the number of network devices the packet has passed through to reach its destination. The server sending the packet sets the TTL value and each network device that the packet passes through then reduces this value by “1”. If the TTL value reaches “0”, then the network device will discard the packet.

This mechanism helps to ensure that bad routing on the Internet won’t cause packets to aimlessly loop around the network. TTLs help to reduce the clogging of data circuits with unnecessary traffic.

What is ICMP?

There is another commonly used protocol called the Internet Control Message Protocol (ICMP). It is not strictly a TCP/IP protocol, but TCP/IP based applications use it frequently.

ICMP provides a suite of error, control, and informational messages for use by the operating system. For example, IP packets will occasionally arrive at a server with corrupted data due to any number of reasons including; a bad connection; electrical interference or even misconfiguration. The server will usually detect this by examining the packet and correlating the contents to what it finds in the IP header’s error control section. It will then issue an ICMP reject message to the original sending machine that the data should be resent as the original transmission was corrupted.

ICMP also includes echo and echo reply messages used by the Linux “**ping**” command to confirm network connectivity. More information on ICMP messages can be found in both the Appendix and the chapter on network troubleshooting.

What Do IP Addresses Look Like?

- All devices connected to the Internet have an Internet Protocol (IP) address. Just like a telephone number, it helps to uniquely identify a user of the system.
- IP addresses are in reality a string of binary digits or "bits". Each bit is either a 1 or a 0. IP addresses have 32 bits in total.

- For ease of use, IP addresses are written in what is called a "dotted decimal" format, four numbers with dots in between. None of the numbers between the dots may be greater than 255. An example of an IP address would be 97.65.25.12.
- The numbers between the dots are frequently referred to as "octets"
- Some groups of IP addresses are reserved for use only in private networks and are not routed over the Internet. These are:

Private IP Addresses

```
    10.0.0.0 - 10.255.255.255
  172.16.0.0 - 172.31.255.255
192.168.0.0 - 192.168.255.255
```

- Home networking equipment / devices usually are configured in the factory with an IP address in the range 192.168.1.1 to 192.168.1.255.
- You can check the Linux networking topics page on how to configure the IP address of your Linux box.

What Is Localhost?

Whether or not your computer has a network interface card it will have a “built in” IP address with which network aware applications can communicate with one another. This IP address is defined as 127.0.0.1 and is frequently referred to as “localhost”

What Is A Subnet Mask?

- Subnet masks are used to tell which part of the IP address represents:
  - The network on which the computer is connected (Network portion)
  - The computer's unique identifier on that network (Host portion)
- A simple analogy would be a phone number, such as (808) 225-2468. The (808) represents the area code, the 225-2468 represents the telephone within that area code.
- Subnet masks allow you to specify how long you want the area code to be (network portion) at the expense of the number of telephones in that are in the area code (Host portion)
- Most home networks use a subnet mask of 255.255.255.0. Each "255" means this octet is for the area code (network portion). So if your server has an IP address of 192.168.1.25 and a subnet mask of 255.255.255.0, then the network portion would be 192.168.1 and the server or host would be device #25 on that network.
- In this example, host #0 (192.168.1.0) is reserved to represent the network itself, and host #255 (192.168.1.255) is reserved for broadcast traffic intended to reach all hosts on the network at the same time. You can then use IP addresses from #1 to #254 on your "private" network.
- If you purchased a DSL service from your Internet service provider (ISP) that gives you fixed IP addresses, then they will most likely provide you with a subnet mask of 255.255.255.248 that defines 8 IP addresses. For example if the ISP provides you with a

"public" network address of 97.158.253.24, a subnet mask of 255.255.255.248 and a gateway of 97.158.253.25, then your IP addresses will be:

- 97.158.253.24 – Network base address
- 97.158.253.25 - Gateway
- 97.158.253.26 - Available
- 97.158.253.27 - Available
- 97.158.253.28 - Available
- 97.158.253.29 - Available
- 97.158.253.30 - Available
- 97.158.253.31 - Broadcast

How Many Addresses Do I Get With My Mask?

The method described in this section only works for subnet masks that start with “255.255.255” which should be sufficient for your home network.

- There are only 7 possible values for the last octet of a subnet mask. These are 0, 192, 128, 224, 240, 248, 252
- You can calculate the number of IP addresses for each of the above values by subtracting the value from 256
- So for example, if you have a subnet mask of 255.255.255.192 then you have 64 IP addresses in your subnet (256 - 192)

What’s The Range Of Addresses On My Network?

If someone gives you an IP address of 97.158.253.28 and a subnet mask of 255.255.255.248, how do you determine the network address and the broadcast address, in other words the boundaries of my network? Here are the steps:

Manual Calculation

- Subtract the last octet of the subnet mask from 256 to give the number of IP addresses in the subnet. (256 - 248) = 8
- Divide the last octet of the IP address by the result of step 1, don't bother with the remainder (28/8 = 3). This will give you the theoretical number of subnets of the same size that are below this IP address.
- Multiply this result by the result of step 1 to get the network address (8 x 3 = 24). Think of it as "This is the third subnet with 8 addresses in it". The Network address is therefore 97.158.253.24

- The broadcast address is the result of step 3 plus the result of step 1 minus 1. (24 + 8 - 1 = 31). Think of it as "The broadcast address is always the network address plus the number of IP addresses in the subnet minus 1". The broadcast address is 97.158.253.31

Let's do this for 192.168.3.56 with a mask of 255.255.255.224

1. 256 - 224 = 32
2. 56 / 32 = 1
3. 32 x 1 = 32. Therefore the network base address is 192.168.3.32
4. 32 + 32 - 1 = 63. Therefore the broadcast address is 192.168.3.63

Let's do this for 10.0.0.75 with a mask of 255.255.255.240

1. 256 - 240 = 16
2. 75 / 16 = 4
3. 16 x 4 = 64. Therefore the network base address is 10.0.0.64
4. 64 + 16 -1 = 79. Therefore the broadcast address is 10.0.0.79

Note: As a rule of thumb, the last octet of your network base address must be divisible by the "256 minus the last octet of your subnet mask" and leave no remainder. If you are sub-netting a large chunk of IP addresses it's always a good idea to lay it out on a spreadsheet to make sure there are no overlapping subnets. Once again, this calculation exercise only works with subnet masks that start with "255.255.255".

Calculation Using A Script

There is a BASH script in the Appendix which will do this for you. Here is a sample of how to use it, just provide the IP address followed by the subnet mask as arguments. It will accept subnet masks in dotted decimal format or "/value" format

```
[root@bigboy tmp]# ./subnet-calc.sh 216.151.193.92 /28

IP Address           : 216.151.193.92
Network Base Address : 216.151.193.80
Broadcast Address    : 216.151.193.95

Subnet Mask          : 255.255.255.240
Subnet Size          : 16 IP Addresses

[root@bigboy tmp]#
```

What Is Duplex?

- Duplex refers to the ability of a device to transmit and receive data at the same time.
- Full duplex uses separate pairs of wires for transmitting and receiving data so that incoming data flows don't interfere with outgoing data flows.

- Half duplex uses the same pairs of wires for transmitting and receiving data. Devices that want to transmit information have to wait their turn until the "coast is clear" at which point they send the data. Error detection and retransmission mechanisms ensure that data reaches the destination correctly even if it were originally garbled by multiple devices starting to transmit at the same time.
- Data transfer speeds will be low and error levels will be high if you have a device at one end of a cable set to full duplex, and another device at the other end of the cable set to half duplex.
- Most modern network cards can auto-negotiate duplex with the device on the other end of the wire. It is for this reason that duplex settings aren't usually a problem for Linux servers.

What Is A Hub?

- A hub is a device into which you can connect all devices on a home network so that they can talk together. Hubs physically cross-connect all their ports with one another which causes all traffic sent from a server to the hub to be blurted out to all other servers connected to that hub whether they are the intended recipient or not.
- Hubs have none or very little electronics inside and therefore do not regulate traffic. It is possible for multiple servers to speak at once with all of them receiving garbled messages. When this happens the servers try again, after a random time interval, until the message gets through correctly.
- It is for these reasons that devices that plug into hubs should be set to half duplex.

What Is A Switch?

- A switch is also a device into which you can connect all devices on a home network so that they can talk together. Unlike a hub, traffic sent from Server A to Server B will only be received by Server B. The only exception is broadcast traffic which is blurted out to all the servers simultaneously.
- Switches regulate traffic, thereby eliminating the possibility of message garbling. Switches therefore provide more efficient traffic flow.
- Devices that plug into switches should be set to full duplex to take full advantage of the dedicated bandwidth coming from each switch port.

What Is A LAN?

- A Local Area Network (LAN) is a grouping of ports on a hub, switch or tied to a wireless access point (WAP) that can only communicate with each other.
- It is possible to have LANs that span multiple switches. Simple home switches can be connected in a chain formation to create a LAN with more ports. This is often called "daisy chaining".
- Pure switches provide no access control between servers connected to the same LAN. This is why network administrators group trusted servers having similar roles on the same LAN. They will also ensure that they don't mix servers on different IP networks on the same LAN segment. A good rule of thumb is to have only one network per LAN.

- Communication to devices on another LAN requires a router directly connected to both LANs. The router is also capable of filtering traffic passing between the two LANs therefore providing additional security.
- Larger, more expensive switches can be configured to assign only certain ports to pre-specified Virtual LANs or (VLANs) chosen by the network administrator. In this case, the switch houses ports on multiple LANs. A router is still needs to be connected to each VLAN for inter-network communication.

What Is A Router?

- As stated before, switches and hubs usually only have servers connected to them that have been configured as being part of the same network.
- Routers will connect into multiple switches to allow these networks to communicate with one another.
- Routers can also be configured to deny communication between specific servers on different networks. They can also filter traffic based on the TCP port section of each packet. For example, it is possible to deny communication between two servers on different networks that intend to communicate on TCP port 80, and allow all other traffic between them. Routers therefore direct and regulate traffic between separate networks, much like a traffic policeman.
- If you intend to route between networks, then for each network, you must reserve an IP address for a router and make sure that the router is directly connected to the LAN associated with that network.
- In home networks, routers most frequently provide connectivity to the Internet using network address translation or NAT.

What Is A Gateway?

- Another name for a router.

What Is A Route?

- In the broader networking sense, a route refers to the path data takes to traverse from its source to its destination. Each router along the way may also be referred to as a hop.
- Usually when we speak about a route on a Linux box, we are referring to the IP address of the first hop needed to reach the desired destination network. It is assumed that this first hop will know how to automatically relay the packet.
- Routers are designed to exchange routing information dynamically, and can therefore intelligently redirect traffic to bypass failed network links. Home Linux boxes frequently don't run a dynamic routing protocol and therefore rely on "static" routes issued by the system administrator at the command line or in configuration files to determine the next hop to all desired networks.
- The Linux network topics page shows how to add static routes to your Linux box and also how you can convert it into a simple router.

What Is A Default Gateway?

- A default gateway is really a gateway of last resort. Say for example:
  - You have two routers R1 and R2
  - R1 is connected to both your SOHO home network (192.168.1.0) and the internet
  - R2 is connected to both your SOHO home network (192.168.1.0) and your credit card transaction payment the network (10.46.123.0) which is also connected to other corporate networks with addresses starting with 10.X.X.X
- You could put a route on your SOHO servers that states:
  - Go to network 10.0.0.0 255.0.0.0 via router R2
  - Go to everything else via router R1. R1 therefore would be considered your default gateway

- For most home networks, your default gateway would be the router / firewall connected to the Internet.
- You can check the Linux networking topics page on how to configure the default gateway on your Linux box.

What Is A NIC?

Your network interface card is frequently called a NIC. Currently, the most common types of NIC used in the home are Ethernet and wireless Ethernet cards.

What Does The “Link” Light On My NIC Indicate?

The link light signifies that the NIC card has successfully detected a device on the other end of the cable. This would indicate that you are using the correct type of cable and that the duplex has been negotiated correctly between the devices at both ends.

What Is A MAC Address?

The media access control address (MAC) can be equated to the serial number of the NIC. Every IP packet is sent out of your NIC wrapped inside an Ethernet frame which uses MAC addresses to direct traffic on your locally attached network.

MAC addresses therefore only have significance on the locally attached network. As the packet hops across the Internet, its source/destination IP address stays the same, but the MAC addresses are reassigned by each router on the way using a process called ARP.

What Is ARP?

The Address Resolution Protocol (ARP) is used to map MAC addresses to network IP addresses. When a server needs to communicate with another server it does the following steps:

- The server first checks its routing table to see which router provides the next hop to the destination network.
- If there is a valid router, let's say with an IP address of 192.168.1.1, the server checks it's ARP table to see whether it has the MAC address of the router's NIC. You could very loosely view this as the server trying to find the Ethernet serial number of the next hop router on the local network, thereby ensuring that the packet is sent to the correct device.
- If there is an ARP entry, the server sends the IP packet to its NIC and tells the NIC to encapsulate the packet in a frame destined for the MAC address of the router.
- If there is no ARP entry, the server will issue an ARP request asking that router 192.168.1.1 respond with its MAC address so that the delivery can be made. Once a reply is received, the packet is sent and the ARP table is subsequently updated with the new MAC address.
- As each router in the path receives the packet, it will pluck the IP packet out of the Ethernet frame, leaving the MAC information behind. It will then inspect the destination IP address in the packet and use its routing table to determine the IP address of the next router on the path to this destination.
- The router will then use the ARP-ing process to get the MAC address of this next hop router. It will then re-encapsulate the packet in an Ethernet frame with the new MAC address and will then send the frame to the next hop router. This relaying process continues until the packet reaches the target computer.
- If the target server is on the same network as the source server, a similar process occurs. The ARP table is queried. If no entry is available, an ARP request is made asking the target server for its MAC address. Once a reply is received, the packet is sent and the ARP table is subsequently updated with the new MAC address.
- The server will not send the data to its intended destination unless it has an entry in its ARP table for the next hop. If it doesn't, the application needing to communicate will issue a timeout or "time exceeded" error.
- As can be expected, the ARP table only contains the MAC addresses of devices on the locally connected network. ARP entries are not permanent and will be erased after a fixed period of time depending on the operating system used.
- The Linux network topics page shows how to see your ARP table and the MAC addresses of your server's NICs.

What Is A DTE?

DTE stands for Digital Terminal Equipment, a terminology originally intended for computer terminals located at remote offices or departments that were directly connected modems. The terminals would have no computing power and only functioned as a screen / keyboard combination for data processing.

Nowadays most PCs have their COM and Ethernet ports configured as if they were going to be connected to a modem or other type of purely networking oriented equipment.

What Is A DCE?

DCE is the acronym for Data Circuit-Terminating Equipment. Modems and other purely networking oriented equipment.

What Is A Straight Through / Crossover Cable?

When a DCE is connected to a DTE, you will need a “straight-through” type cable. DCEs connected to DCEs or DTEs connected to DTEs will always require “crossover” cables. These are the terminologies generally used with Ethernet cables.

The terminologies can be different for cables used to connect serial ports together. When connecting a PC’s COM port (DTE) to a modem (DCE) the “straight-through” cable is frequently called a “modem” cable. When connecting two PCs (DTE) together via their COM ports the “crossover” cable is often referred to as a “null modem” cable.

Unfortunately, some manufacturers configure the Ethernet ports of their networking equipment to be either of the DTE or the DCE type, so confusion can arise when selecting a cable. If you fail to get a “link” light when connecting your Ethernet devices together, try using the other type of cable.

A “straight-through” Ethernet cable is easy to identify. Hold the connectors side by side, pointing in the same direction with the clips facing away from you. The color of the wire in position #1 on connector #1 should be the same as that of position #1 on connector #2. The same would go for positions #2 to #8, ie. the same color for corresponding wires on each end. A cross over cable would have them mixed up.

Here is a good rule of thumb: PC to PC = crossover cable; PC to switch = straight through cable

What Is A Firewall?

Firewalls can be viewed as routers with more enhanced abilities to restrict traffic, not just by port and IP address like routers. Specifically, firewalls can detect malicious attempts to subvert the TCP/IP protocol. A short list of capabilities includes:

- Throttling traffic to a server when too many unfulfilled connections are made to it
- Restricting traffic being sent to obviously bogus IP addresses
- Providing network address translation or NAT

What Is NAT?

Your router / firewall will frequently be configured to give the impression to other devices on the Internet that the servers on your Home network have a valid “public” IP address, and not a “private” IP address. This is called network address translation (NAT) and is often also called IP masquerading in the Linux world. There are many good reasons for this, the two most commonly stated are:

- No one on the Internet knows your true IP address. NAT protects your home PCs by assigning them IP addresses from “private” IP address space that cannot be routed over the

internet. This prevents hackers from directly attacking your home systems as packets sent to the “private” IP will never pass over the Internet.

- Hundreds of PCs and servers behind a NAT device can masquerade as a single "public" IP address. This greatly increases the number of devices that can access the Internet without running out of "public" IP addresses.

You can configure NAT to be “one to one” in which you assign multiple IP addresses to the outside “public” interface of your firewall and pair each of these addresses to a corresponding server on the inside network. You can also use “many to one” NAT in which the firewall maps a single IP address to multiple servers on the network.

As a general rule, you won’t be able to access the public NAT IP addresses from servers on your home network. Basic NAT testing will require you to ask a friend to try to connect to your home network from the Internet.

Examples of NAT may be found in the IP masquerade section of the Linux iptables firewall chapter and also in the Cisco PIX firewall chapter.

What Is Port Forwarding With NAT?

In our simple home network, all servers accessing the Internet will appear to have the single “public” IP address of the router / firewall because of “many to one” NAT. As the router / firewall is located at the “border crossing” to the Internet it can easily keep track of all the various outbound connections to the Internet by monitoring:

- The IP addresses and TCP ports used by each home based server and mapping it to
- The TCP ports and IP addresses of the Internet servers with which they want to communicate.

This arrangement works well with a single NAT IP trying to initiate connections to many Internet addresses. The reverse isn’t true. Connections initiated from the Internet to the “public” IP address of the router / firewall face a problem. As there normally has been no prior connection association between the Internet server and any protected server on the home network, the router / firewall has no way of telling which of the many home PCs behind it should receive the relayed data.

Port forwarding is a method of counteracting this. For example, you can configure your router / firewall to forward TCP port 80 (Web/HTTP) traffic destined to the outside NAT IP to be automatically relayed to a specific server on the inside home network

As you may have guessed, port forwarding is one of the most common methods used to host websites at home with DHCP DSL.

What Is DHCP?

According to www.dhcp.org, "The Dynamic Host Configuration Protocol (DHCP) is an Internet protocol for automating the configuration of computers that use TCP/IP. DHCP can be used to automatically assign IP addresses, (and) to deliver TCP/IP stack configuration parameters such as the subnet mask and default router”.

The assignment usually occurs when the DHCP configured machine boots up, or regains connectivity to the network. The DHCP client sends out a query requesting a response from a DHCP server on the locally attached network. The DHCP server then replies to the client PC with its assigned IP address, subnet mask, DNS server and default gateway information.

The assignment of the IP address usually expires after a predetermined period of time, at which point the DHCP client and server renegotiate a new IP address from the server's predefined pool of addresses. Configuring firewall rules to accommodate access form machines who receive their IP addresses via DHCP is therefore more difficult as the remote IP address will vary from time to time. You'll probably have to allow access for the entire remote DHCP subnet.

Most home router / firewalls are configured in the factory to be DHCP servers for your home network. You can also make your Linux box into a DHCP server, once it has a fixed IP address.

The most commonly used form of DSL will also assign the outside interface of your router / firewall with a single DHCP provided IP address.

You can check the chapter on Linux networking topics page on how to configure your Linux box to get its IP address via DHCP. You can also check the chapter on Configuring a DHCP Server, to make your Linux box provide the DHCP addresses for the other machines on your network.

What Is DNS?

The domain name system (DNS) is a worldwide server network used to help translate easy to remember domain names like www.linuxhomenetworking.com into an IP address that can be used behind the scenes by your computer. Here step by step description of what happens with a DNS lookup.

- Most home computers will get the IP address of their DNS server via DHCP from their router / firewall.
- Home router / firewall providing DHCP services often provides its own IP address as the DNS name server address for home computers.
- The router / firewall will then redirect the DNS queries from your computer to the DNS name server of your Internet service provider (ISP).
- Your ISP's DNS server will then probably redirect your query to one of the 13 "root" name servers.
- The root server will then redirect your query to one of the Internet's ".com" DNS name servers which will then redirect the query to the "linuxhomenetworking.com" name server.
- The "linuxhomenetworking.com" name server will then respond with the IP address for www.linuxhomenetworking.com

As you can imagine, this process can cause a noticeable delay when you are browsing the web. Each server in the chain will store the most frequent DNS name to IP address lookups in a memory cache which helps to speed up the response. You can make your Linux box into a caching DNS server for your home network too.

How Can I Check The IP Address For A Domain?

If you have the domain, then you can use either the **nslookup** command or **host** command to get the associated IP address. **nslookup** will be removed from future releases of Linux, but can still be used with Windows .

```
[root@bigboy tmp]# nslookup www.linuxhomenetworking.com
Note:  nslookup is deprecated and may be removed from future releases.
```

```
Consider using the `dig' or `host' programs instead.  Run nslookup
with
the `-sil[ent]' option to prevent this message from appearing.
Server:         127.0.0.1
Address:        127.0.0.1#53

Name:   www.linuxhomenetworking.com
Address: 216.151.193.92

[root@bigboy tmp]# host www.linuxhomenetworking.com
www.linuxhomenetworking.com has address 216.151.193.92
[root@bigboy tmp]#
```

You can also use the **nslookup** and host commands to get the reverse information.

```
[root@bigboy tmp]# nslookup  216.151.193.92
Note:  nslookup is deprecated and may be removed from future releases.
Consider using the `dig' or `host' programs instead.  Run nslookup
with
the `-sil[ent]' option to prevent this message from appearing.
Server:         127.0.0.1
Address:        127.0.0.1#53

Non-authoritative answer:
92.193.151.216.in-addr.arpa     name = extra193-92.elan.net.

Authoritative answers can be found from:
193.151.216.in-addr.arpa        nameserver = dns1.elan.net.
193.151.216.in-addr.arpa        nameserver = dns2.elan.net.
dns1.elan.net   internet address = 216.151.192.1

[root@bigboy tmp]#
[root@bigboy tmp]# host 216.151.193.92
92.193.151.216.in-addr.arpa domain name pointer extra193-92.elan.net.
[root@bigboy tmp]#
```

How Do I Get My Own DNS Domain Name?

- There are many companies that provide DNS name registration. RegisterFree is the one I use. You can use them to determine whether the name you want is available and you can purchase the domain you want using a credit card with your web browser.
- The registration process will prompt you for your two primary DNS servers. This helps DNS root servers know exactly where to get the information for the IP address for your new website.
- If you don't have the names and/or IP addresses for you primary name servers, don't worry you can do this later, after you follow the steps below.

Static or Dynamic DNS?

- If you didn't specifically reserve static IP addresses from your ISP, then your router is probably getting its "public" Internet IP address via DHCP from your ISP. In this case you'll want to use a dynamic DNS service.
- If you bought static IP addresses, then static DNS is the way to go.

What is FTP?

This is one of the most popular applications used to copy files between computers via a network connection. There are a number of commercially available GUI based clients you can load on your PC to do this, such as WSFTP and CuteFTP. You also FTP from the command line as shown in the RPM chapter.

From the remote user's perspective, there are two types of FTP.

Regular FTP

- This is used primarily to allow specific users to download files to their systems.
- The remote FTP server will prompt you for a username, at which point the user will be the username you normally use to log into the FTP server. The password will be your regular password for your user account.

Anonymous FTP

- This is used primarily to allow any remote user to download files to their systems.
- The remote FTP server will prompt you for a username, at which point the user will type "anonymous". The password is usually your valid email address.

From the systems administrator's perspective, there are another two categories. These are active and passive FTP which is covered in more detail in the FTP Chapter.

It is good to remember that FTP isn't very secure as usernames, passwords and data are sent across the network unencrypted. More secure forms such as SFTP (Secure FTP) and SCP (Secure Copy) are available as a part of the Secure Shell package that is normally installed by default on RedHat.

Where is Linux Help?

Linux help files are accessed using the “**man**” or manual pages. From the command line you issue the **man** command followed by the Linux command or file you wish to get information about. If you want to get information on the **ssh** command, then you’d use the command “**man ssh**”. If you want to search all the man pages for a keyword, then use the man command with the **–k** switch.

Here are some examples:

Finding General Information On A Command

Here we get information on the **ssh** command:

```
[root@bigboy tmp]# man ssh
SSH(1)                    BSD General Commands
Manual                   SSH(1)

NAME
     ssh - OpenSSH SSH client (remote login program)

SYNOPSIS
     ssh [-l login_name] hostname | user@hostname [command]

     ssh [-afgknqstvxACNPTX1246] [-b bind_address] [-c cipher_spec]
         [-e escape_char] [-i identity_file] [-l login_name] [-m
mac_spec]
         [-o option] [-p port] [-F configfile] [-L
port:host:hostport] [-R
         port:host:hostport] [-D port] hostname | user@hostname
[command]

DESCRIPTION
     ssh (SSH client) is a program for logging into a remote
machine
     and for executing commands on a remote machine.  It is
intended to
     replace rlogin and rsh, and provide secure encrypted …
…
…
…
[root@bigboy tmp]#
```

Search For All Instances Of A Word

Here we discover that the search string **ssh** can be found in the TCL man pages and also in a variety of ssh related pages including ssh, ssh-add and ssh-agent. Using this information you can use the **man** command, without the **–k,** to narrow your help search.

```
[root@bigboy tmp]# man -k ssh
Tcl_DecrRefCount [Tcl_IsShared] (3)  - manipulate Tcl objects
Tcl_DuplicateObj [Tcl_IsShared] (3)  - manipulate Tcl objects
Tcl_IncrRefCount [Tcl_IsShared] (3)  - manipulate Tcl objects
Tcl_InvalidateStringRep [Tcl_IsShared] (3)  - manipulate Tcl
objects
Tcl_IsShared          (3)  - manipulate Tcl objects
Tcl_IsShared [Object] (3)  - manipulate Tcl objects
Tcl_IsShared [Tcl_DecrRefCount] (3)  - manipulate Tcl objects
Tcl_IsShared [Tcl_DuplicateObj] (3)  - manipulate Tcl objects
Tcl_IsShared [Tcl_IncrRefCount] (3)  - manipulate Tcl objects
Tcl_IsShared [Tcl_InvalidateStringRep] (3)  - manipulate Tcl
objects
```

```
Tcl_IsShared [Tcl_NewObj] (3)  - manipulate Tcl objects
Tcl_NewObj [Tcl_IsShared] (3)  - manipulate Tcl objects
ssh                   (1)  - OpenSSH SSH client (remote login
program)
ssh [slogin]          (1)  - OpenSSH SSH client (remote login
program)
ssh-add               (1)  - adds RSA or DSA identities to the
authentication agent
ssh-agent             (1)  - authentication agent
ssh-keygen            (1)  - authentication key generation,
management and conversion
ssh-keyscan           (1)  - gather ssh public keys
ssh-keysign           (8)  - ssh helper program for hostbased
authentication
ssh_config            (5)  - OpenSSH SSH client configuration files
sshd                  (8)  - OpenSSH SSH daemon
sshd_config           (5)  - OpenSSH SSH daemon configuration file
[root@bigboy tmp]#
```

Chapter 2

Linux Networking

===

In This Chapter

===

This chapter covers how to configure your Linux box's networking features.

How To Configure Your NIC's IP Address

Determining Your IP Address

Most modern PCs come with an ethernet port. When Linux is installed, this device is called "**eth0**". You can determine the IP address of this device with the "**ifconfig**" command.

```
[root@bigboy tmp]# ifconfig -a

eth0 Link encap:Ethernet HWaddr 00:08:C7:10:74:A8
BROADCAST MULTICAST MTU:1500 Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:100
RX bytes:0 (0.0 b) TX bytes:0 (0.0 b)
Interrupt:11 Base address:0x1820

lo Link encap:Local Loopback
inet addr:127.0.0.1 Mask:255.0.0.0
UP LOOPBACK RUNNING MTU:16436 Metric:1
RX packets:787 errors:0 dropped:0 overruns:0 frame:0
TX packets:787 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:0
RX bytes:82644 (80.7 Kb) TX bytes:82644 (80.7 Kb)
```

```
wlan0 Link encap:Ethernet HWaddr 00:06:25:09:6A:B5
inet addr:192.168.1.100 Bcast:192.168.1.255 Mask:255.255.255.0
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:47379 errors:0 dropped:0 overruns:0 frame:0
TX packets:107900 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:100
RX bytes:4676853 (4.4 Mb) TX bytes:43209032 (41.2 Mb)
Interrupt:11 Memory:c887a000-c887b000

wlan0:0 Link encap:Ethernet HWaddr 00:06:25:09:6A:B5
inet addr:192.168.1.99 Bcast:192.168.1.255 Mask:255.255.255.0
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
Interrupt:11 Memory:c887a000-c887b000

[root@bigboy tmp]#
```

In this example, **eth0** has no IP address as this box is using wireless interface **wlan0** as it's main NIC. Interface **wlan0** has an IP address of 192.168.1.100 and a subnet mask of 255.255.255.0

You can see that this command gives good information on the interrupts used by each card. This can also be found in less detail in the file **/proc/interrupts**

Changing Your IP Address

If you wanted, you could give this **eth0** interface an IP address using the **ifconfig** command.

```
[root@bigboy tmp]# ifconfig eth0 10.0.0.1 netmask 255.255.255.0 up
```

The "up" at the end of the command activates the interface. To make this permanent each time you boot up you'll have to add this command in your **/etc/rc.d/rc.local** file.

Linux also makes life a little easier with interface configuration files located in the **/etc/sysconfig/network-scripts** directory. Interface **eth0** has a file called **ifcfg-eth0**, **eth1** uses **ifcfg-eth1** ... etc. You can place your IP address information in these files which are then used to auto-configure your NICs when Linux boots. Here are two samples for interface **eth0**, one assumes the interface has a fixed IP address, the other assumes it requires an IP address assignment using DHCP.

network-scripts File Formats

Fixed IP Address

```
[root@bigboy tmp]#  cd /etc/sysconfig/network-scripts
[root@bigboy network-scripts]# more ifcfg-eth0

DEVICE=eth0
BROADCAST=192.168.1.255
IPADDR=192.168.1.100
NETMASK=255.255.255.0
NETWORK=192.168.1.0
ONBOOT=no

[root@bigboy network-scripts]#
```

Getting the IP Address using DHCP

```
[root@bigboy tmp]#  cd /etc/sysconfig/network-scripts
[root@bigboy network-scripts]# more ifcfg-eth0

DEVICE=eth0
BOOTPROTO=dhcp
ONBOOT=yes

[root@bigboy network-scripts]#
```

As you can see **eth0** will be activated on booting as the parameter **ONBOOT** has the value "yes" and not "no". You can read more about netmasks and DHCP on the introduction to networking chapter.

Once you change the values in the configuration files for the NIC you'll have to deactivate and activate it for the modifications to take effect. The **ifdown** and **ifup** commands can be used to do this.

```
[root@bigboy network-scripts]# ifdown eth0
[root@bigboy network-scripts]# ifup eth0
```

Multiple IP Addresses On A Single NIC

In the previous "determining your IP address" section you may have noticed that there were two wireless interfaces. One's named **wlan0** and the other **wlan0:0**. Interface **wlan0:0** is actually a "child" of interface **wlan0**, a virtual sub-interface also known as an "IP alias". IP aliasing is one of the most common ways of creating multiple IP addresses associated with

a single NIC. Aliases have the name format "*parent-interface-name:X*", where "X" is the sub-interface number of your choice.

The process for creating an IP alias is very similar to the steps outlined for the real interface in the previous "changing your IP address" section.

- First ensure the "parent" real interface exists
- Verify that no other IP aliases with the same name exists with the name you plan to use. In this we want to create interface **wlan0:0**
- Create the virtual interface with the **ifconfig** command

```
[root@bigboy tmp]# ifconfig wlan0:0 192.168.1.99 \
                        netmask 255.255.255.0 up
```

- You then have the choice of creating a **/etc/sysconfig/network-scripts/ifcfg-wlan0:0** file or adding the **ifconfig** command used above to your **/etc/rc.d/rc.local** file to ensure the IP address is assigned properly when you reboot.

IP Address Assignment For A Direct DSL Connection

If you are using a DSL connection with fixed or "static" IP addresses, then the configuration steps are the same as those outlined above. You plug your ethernet interface into the DSL modem, configure it with the IP address, subnet mask, broadcast address and gateway information provided by your ISP and you should have connectivity once you restart your interface. Remember that you may also need to configure your DNS server correctly.

If you are using a DSL connection with a DHCP or "dynamic" IP address assignment, then the process is different. Your ISP will provide you with a PPPoE "username" and "password" which will allow your computer to login transparently to the Internet each time it boots up. By default, as of version 8.0, RedHat Linux installs the **rp-pppoe** RPM software package required to support this.

Downloading and installing RPMs isn't hard. If you need a refresher, the chapter on RPMs covers how to do this in detail. The latest version of the RPM for RedHat 8.0 is **rp-pppoe-3.4-7.i386.rpm .** Install the package using the following command:

```
[root@bigboy tmp]# rpm -Uvh rp-pppoe-3.4-7.i386.rpm
Preparing...     ########################################### [100%]
1:rp-pppoe       ########################################### [100%]
[root@bigboy tmp]#
```

You'll then need to go through a number of steps to complete the connection. The PPPOE configuration will create a software based virtual interface named **ppp0** that will use the physical Internet interface **eth0** for connectivity. Here's what you need to do:

- Make a backup copy of your **ifcfg-eth0** file.

```
[root@bigboy tmp]#
[root@bigboy tmp]# cd /etc/sysconfig/network-scripts/
[root@bigboy network-scripts]# ls ifcfg-eth0
ifcfg-eth0
[root@bigboy network-scripts]# cp ifcfg-eth0 DISABLED.ifcfg-
eth0
```

- Edit your **ifcfg-eth0** file to have no IP information and also to be deactivated on boot time.

```
DEVICE=eth0
ONBOOT=no
```

- Shutdown your **eth0** interface.

```
[root@bigboy network-scripts]# ifdown eth0
[root@bigboy network-scripts]#
```

- Run the **adsl-setup** configuration script

```
[root@bigboy network-scripts]# adsl-setup
```

- It will prompt you for your ISP username, the interface to be used (eth0) and whether you want to the connection to stay up indefinitely. We'll use defaults wherever possible.

```
Welcome to the ADSL client setup.  First, I will run some checks on
your system to make sure the PPPoE client is installed properly...

LOGIN NAME

Enter your Login Name (default root): bigboy-login@isp

INTERFACE

Enter the Ethernet interface connected to the ADSL modem
For Solaris, this is likely to be something like /dev/hme0.
For Linux, it will be ethX, where 'X' is a number.
(default eth0):

Do you want the link to come up on demand, or stay up continuously?
If you want it to come up on demand, enter the idle time in seconds
after which the link should be dropped.  If you want the link to
stay up permanently, enter 'no' (two letters, lower-case.)
NOTE: Demand-activated links do not interact well with dynamic IP
addresses.  You may have some problems with demand-activated links.
Enter the demand value (default no):
```

- It will then prompt you for your DNS server information. This step will edit your **/etc/resolv.conf** file. If you're running BIND on your server in a caching DNS mode then you may want to leave this option blank. If you want your ISP to automatically provide the IP address of its DNS server then enter the word "server".

```
DNS

Please enter the IP address of your ISP's primary DNS server.
If your ISP claims that 'the server will provide dynamic DNS
addresses', enter 'server' (all lower-case) here.
If you just press enter, I will assume you know what you are
doing and not modify your DNS setup.
Enter the DNS information here:
```

- The script will then prompt you for your ISP password

```
PASSWORD

Please enter your Password:
Please re-enter your Password:
```

- Then it will ask whether you want regular users (not superuser "**root**") to be able to activate/deactivate the new **ppp0** interface

```
USERCTRL

Please enter 'yes' (two letters, lower-case.) if you want to
allow
normal user to start or stop DSL connection (default yes):
```

- The **rp-pppoe** package has two sample **ipchains** firewall scripts located in the **/etc/ppp** directory named **firewall-standalone** and **firewall-masq**. They are very basic and don't cover rules to make your Linux box a web server, DNS server nor mail server. I'd recommend selecting "none" and using a variant of the basic script samples in the firewall chapter, or the more comprehensive one found in the Appendix.

```
FIREWALLING

Please choose the firewall rules to use.  Note that these rules are
very basic.  You are strongly encouraged to use a more sophisticated
firewall setup; however, these will provide basic security.  If you
are running any servers on your machine, you must choose 'NONE' and
set up firewalling yourself.  Otherwise, the firewall rules will deny
access to all standard servers like Web, e-mail, ftp, etc.  If you
are using SSH, the rules will block outgoing SSH connections which
allocate a privileged source port.

The firewall choices are:
0 - NONE: This script will not set any firewall rules.  You are
responsible
          for ensuring the security of your machine.  You are STRONGLY
          recommended to use some kind of firewall rules.
1 - STANDALONE: Appropriate for a basic stand-alone web-surfing
workstation
```

```
2 - MASQUERADE: Appropriate for a machine acting as an Internet
gateway
                for a LAN
Choose a type of firewall (0-2): 0
```

- You'll then be asked whether you want the connection to be activated upon booting. Most people would say "yes".

```
Start this connection at boot time

Do you want to start this connection at boot time?
Please enter no or yes (default no):yes
```

- Just before exiting, you'll get a summary of the parameters you entered and the relevant configuration files will be updated to reflect your choices when you accept them.

```
** Summary of what you entered **

Ethernet Interface: eth0
User name:          bigboy-login@isp
Activate-on-demand: No
DNS:                Do not adjust
Firewalling:        NONE
User Control:       yes
Accept these settings and adjust configuration files (y/n)? y

Adjusting /etc/sysconfig/network-scripts/ifcfg-ppp0
Adjusting /etc/ppp/chap-secrets and /etc/ppp/pap-secrets
  (But first backing it up to /etc/ppp/chap-secrets.bak)
  (But first backing it up to /etc/ppp/pap-secrets.bak)
```

- At the very end it will tell you the commands to use to activate /deactivate your new **ppp0** interface and to get a status of the interface's condition.

```
Congratulations, it should be all set up!

Type '/sbin/ifup ppp0' to bring up your xDSL link and '/sbin/ifdown
ppp0'to bring it down.
Type '/sbin/adsl-status /etc/sysconfig/network-scripts/ifcfg-ppp0'
to see the link status.
```

The above example recommends using the **adsl-status** command with the name of the PPPoE interface configuration file. This command defaults to show information for interface **ppp0** and therefore listing the **ifcfg-ppp0** filename won't be necessary in most home enviornments.

Some Important Files Created By adsl-setup

- The **adsl-setup** script creates three files that will be of interest to you. The first is the **ifcfg-ppp0** file with interface's link layer connection parameters

```
[root@bigboy network-scripts]# more ifcfg-ppp0
USERCTL=yes
```

```
BOOTPROTO=dialup
NAME=DSLppp0
DEVICE=ppp0
TYPE=xDSL
ONBOOT=yes
PIDFILE=/var/run/pppoe-adsl.pid
FIREWALL=NONE
PING=.
PPPOE_TIMEOUT=20
LCP_FAILURE=3
LCP_INTERVAL=80
CLAMPMSS=1412
CONNECT_POLL=6
CONNECT_TIMEOUT=60
DEFROUTE=yes
SYNCHRONOUS=no
ETH=eth0
PROVIDER=DSLppp0
USER= bigboy-login@isp
PEERDNS=no
[root@bigboy network-scripts]#
```

- The others are the duplicate **/etc/ppp/pap-secrets** and **/etc/ppp/chap-secrets** files with the username and password needed to login to your ISP.

```
[root@bigboy network-scripts]# more /etc/ppp/pap-secrets
# Secrets for authentication using PAP
# client        server  secret                  IP addresses
"bigboy-login@isp" *       "password"
[root@bigboy network-scripts]#
```

Simple Troubleshooting

- You can run the **adsl-status** command to determine the condition of your connection. In this case the package has been installed but the interface hasn't been activated.

```
[root@bigboy tmp]# adsl-status
Note: You have enabled demand-connection; adsl-status may be
inaccurate.
adsl-status: Link is attached to ppp0, but ppp0 is down
[root@bigboy tmp]#
```

- After activation, the interface appears to work correctly.

```
[root@bigboy tmp]# ifup ppp0
[root@bigboy tmp]#  adsl-status
adsl-status: Link is up and running on interface ppp0
ppp0: flags=8051<UP,POINTOPOINT,RUNNING,MULTICAST> mtu 1462
inet
…
…
…
[root@bigboy tmp]#
```

- For further troubleshooting information you can visit the website of **rp-ppoe** at Roaring Penguin (www.roaringpenguin.com). There are some good tips there on how to avoid problems with VPN clients.

How To Change Your Default Gateway

This can be done with a simple command. This example uses a newly installed wireless interface called **wlan0**, most PCs would be using the standard ethernet interface **eth0**.

```
[root@bigboy tmp]# route add default gw 192.168.1.1 wlan0
```

In this case, make sure that the router / firewall with IP address 192.168.1.1 is connected to the same network as interface **wlan0** !

Once done, you'll need to update your **/etc/sysconfig/network** file to reflect the change. This file is used to configure your default gateway each time Linux boots.

```
NETWORKING=yes
HOSTNAME=bigboy
GATEWAY=192.168.1.1
```

Some people don't bother with this step and just place the "route add" command in the file **/etc/rc.d/rc.local**

How Configure Two Gateways

Some networks may have multiple router / firewalls providing connectivity. Here's a typical scenario:

- You have one router providing access to the Internet which you'd like to have as your default gateway (See the default gateway example above)
- You also have another router providing access to your corporate network using addresses in the range 10.0.0.0 to 10.255.255.255. Let's assume that this router has an IP address of 192.168.1.254

The Linux box used in this example uses interface **wlan0** for its Internet connectivity. You may be most likely using interface **eth0**, please adjust your steps accordingly.

Add the new route as follows:

```
route add -net 10.0.0.0 netmask 255.0.0.0 gw 192.168.1.254 wlan0
```

The file **etc/sysconfig/static-routes** will also have to updated so that the route is reinstated when you reboot. Here is a sample.

```
wlan0 net 10.0.0.0 netmask 255.0.0.0 gw 192.168.1.254
```

Some people don't bother with this step and just place the "**route add**" command in the file **/etc/rc.d/rc.local**. A more complicated **/etc/sysconfig/static-routes** file is located in a following section.

How To Delete A Route

Here's how to delete the routes added in the previous section.

```
route del -net 10.0.0.0 netmask 255.0.0.0 gw 192.168.1.254 wlan0
```

The file **etc/sysconfig/static-routes** will also have to be updated so that when you reboot the server will not reinsert the route. Delete the line that reads:

```
wlan0 net 10.0.0.0 netmask 255.0.0.0 gw 192.168.1.254
```

How To View Your Current Routing Table

The **netstat -nr** command will provide the contents of the touting table. Networks with a gateway of 0.0.0.0 are usually directly connected to the interface. As no gateway is needed to reach your own directly connected interface then an address of 0.0.0.0 seems appropriate.

- In this example there are two gateways, the default and one to 255.255.255.255 which is usually added on DHCP servers. Server bigboy is a DHCP server in this case.

```
[root@bigboy tmp]# netstat -nr
Kernel IP routing table
Destination     Gateway     Genmask         Flags MSS Window irtt
Iface
255.255.255.255 0.0.0.0     255.255.255.255
UH    40  0       0    wlan0
192.168.1.0     0.0.0.0     255.255.255.0   U     40  0       0
wlan0
127.0.0.0       0.0.0.0     255.0.0.0       U     40  0       0
lo
0.0.0.0         192.168.1.1
0.0.0.0         UG    40  0       0    wlan0
[root@bigboy tmp]#
```

- In this example, there are multiple gateways handling traffic destined for different networks on different interfaces.

```
[root@bigboy tmp]# netstat -nr
Kernel IP routing table
Destination   Gateway       Genmask         Flags MSS Window irtt
Iface
172.16.68.64  172.16.69.193 255.255.255.224
UG    40  0      0    eth1
172.16.11.96  172.16.69.193 255.255.255.224
UG    40  0      0    eth1
172.16.68.32  172.16.69.193 255.255.255.224
UG    40  0      0    eth1
172.16.67.0   172.16.67.135 255.255.255.224
UG    40  0      0    eth0
172.16.69.192 0.0.0.0       255.255.255.192
U     40  0      0    eth1
172.16.67.128 0.0.0.0       255.255.255.128
U     40  0      0    eth0
172.160.0     172.16.67.135
255.255.0.0     UG    40  0      0    eth0
172.16.0.0    172.16.67.131
255.240.0.0     UG    40  0      0    eth0
127.0.0.0     0.0.0.0       255.0.0.0       U     40  0      0
lo
0.0.0.0       172.16.69.193
0.0.0.0         UG    40  0      0    eth1
[root@bigboy tmp]#
```

- Here is what the static routes file looks like for this multi-homed (Multiple NICs) server

```
[root@bigboy tmp]# more /etc/sysconfig/static-routes
eth0 net 172.16.0.0   netmask 255.240.0.0     gw 172.16.67.131
eth0 net 172.160.0    netmask 255.255.0.0     gw 172.16.67.135
eth0 net 172.16.67.0  netmask 255.255.255.224 gw 172.16.67.135
eth1 net 172.16.68.64 netmask 255.255.255.224 gw 172.16.69.193
eth1 net 172.16.68.32 netmask 255.255.255.224 gw 172.16.69.193
eth1 net 172.16.11.96 netmask 255.255.255.224 gw 172.16.69.193
[root@bigboy tmp]#
```

How To Change The Duplex Setting Of Your NIC

There is no better Linux investment than the purchase of a fully Linux compatible NIC card. Most Linux vendors will have a list of compatible hardware on their websites, read this carefully before you start hooking up you machine to the network. If you can't find any of the desired models in your local computer store, then a model in the same family or series should be sufficient. Most cards will work, but only the fully compatible ones will provide you with error free, consistent throughput.

My experience has been that Ethernet NICs built into motherboards (onboard NICs) frequently don't negotiate port speed and duplex correctly. An onboard NIC may be adequate for a home system, but you should invest in a compatible card when using Linux in a SOHO environment.

You can manage the duplex and speed settings of your NIC with the **mii-tool** command. It is best to use this command with compatible hardware.

In the example below, we can see the output of the command verbose "–v" mode. In this case, negotiation was OK, with the NIC selecting 100Mbps, full duplex mode (FD).

```
[root@bigboy tmp]# mii-tool -v
eth1: negotiated 100baseTx-FD, link ok
  product info: vendor 00:10:18, model 33 rev 2
  basic mode:   autonegotiation enabled
  basic status: autonegotiation complete, link ok
  capabilities: 100baseTx-FD 100baseTx-HD 10baseT-FD 10baseT-HD
  advertising:  100baseTx-FD 100baseTx-HD 10baseT-FD 10baseT-HD
  link partner: 100baseTx-FD 100baseTx-HD 10baseT-FD 10baseT-HD
flow-control
[root@bigboy tmp]#
```

You can set your NIC to force itself to a particular speed and duplex by using the "-F" switch with any of the following options: 100baseTx-FD, 100baseTx-HD, 10baseT-FD, or 10baseT-HD. Remember that you could lose all network connectivity to your server if you force your NIC to a particular speed/duplex that doesn't match that of your switch.

```
[root@bigboy tmp]# mii-tool –F 100baseTx-FD eth0
```

I have seen where NICs appear to work with failed negotiation, but this is usually accompanied by many "collision" type errors being seen on the NIC when using the **ifconfig –a** command and only marginal performance. The causes for this could include an incompatible NIC, incorrect settings on your switch port or a bad cable.

How To Convert Your Linux Server Into A Router

Configuring IP Forwarding

For your Linux server to become a router, you have to enable packet forwarding. In simple terms packet forwarding lets packets flow through the Linux box from one network to another.

The configuration parameter to activate this is found in the file **/etc/sysctl.conf**. Remove the "#" from the line related to packet forwarding.

Before

```
# Disables packet forwarding
#net.ipv4.ip_forward=1
```

After

```
# Enables packet forwarding
net.ipv4.ip_forward=1
```

This will only enable it when you reboot at which time Linux will create a file in one of the subdirectories of the special RAM memory based **/proc** filesystem. To activate the feature immediately you have to create a single lined text file called **/proc/sys/net/ipv4/ip_forward** and it only contain the value "1". Here is how it's done:

```
[root@bigboy tmp] echo 1 > /proc/sys/net/ipv4/ip_forward
```

Configuring Proxy ARP

Note: If you are planning on running iptables on your Linux router, this step is unnecessary as the firewall software does this automatically.

The next step needed will be activating proxy ARP. All computers that need to communicate with a computer on another network send out an ARP request to get the Ethernet MAC address (separate from the IP address) of the most desirable router in their routing table. The router will reply with its MAC address which the server will use when forwarding the packet to the router. Proxy ARP has to be enabled for the Linux box to answer ARP requests from servers on the network. Proxy ARP activation needs to be done for each Ethernet interface on your Linux box.

Once again, the **/proc** filesystem comes into play. Proxy ARP is handled by files in the **/proc/sys/net/ipv4/conf/** directory. This directory then has sub-directories corresponding to each functioning NIC card on your server. (ie. The "**ifconfig –a**" command shows them as being "UP"). Each subdirectory then has a file called **proxy_arp**. If the value within this file is "0", then proxy ARP on the interface is disabled, if the value is "1" then it is enabled.

You can use the "echo" insert the correct values into each file. The example below activates proxy ARP for interfaces eth0 and wlan0.

```
[root@bigboy tmp] echo 1 > /proc/sys/net/ipv4/conf/eth1/proxy_arp
[root@bigboy tmp] echo 1 > /proc/sys/net/ipv4/conf/wlan0/proxy_arp
```

The following command will enable it for all interfaces.

```
[root@bigboy tmp] echo 1 > /proc/sys/net/ipv4/conf/all/proxy_arp
```

(You can determine your network interface names with the ifconfig -a command)

There is no purpose built configuration file to force Linux to do proxy ARP on booting. The best way to do this is put the commands above in your /**etc/rc.d/rc.local** file

```
echo 1 > /proc/sys/net/ipv4/conf/eth1/proxy_arp
echo 1 > /proc/sys/net/ipv4/conf/wlan0/proxy_arp
```

Remember to configure a default route on your Linux box to point to your Internet gateway. You may also want to convert your new Linux router into a firewall to protect your home network. The Netfilter - iptables pages show how to do this.

Configuring Your /etc/hosts File

The **/etc/hosts** lists the name and IP address of local hosts. Your server will typically check this file before referencing DNS, if the name is found with a corresponding IP address then DNS won't be queried. Unfortunately, if the IP address for that host changes, you'll have to update file. For ease of management, it is best to limit entries in this file to just the loopback interface, and also the local host's name and use the centralized DNS server handle the rest.

- The **/etc/hosts** file has the following format:

```
ip-address fully-qualified-domain-name alias1 alias2 alias3 etc
```

- The very first line should **always** look like this with "localhost" being the only alias:

```
127.0.0.1        localhost.localdomain  localhost
```

- If you have a NIC card in the server, then you have to add another entry in this file.
    - First determine what your true hostname is:

```
[root@bigboy mail]# hostname
bigboy
[root@bigboy mail]#
```

    - Add the corresponding entry in the **/etc/hosts** file for the NIC's IP address

Your NIC's /etc/hosts File Format

Your machine's name is NOT listed with a DNS server	Your machine's name is listed with a DNS server
IP-address hostname.localdomain hostname	IP-address hostname.my-site.com hostname

Here are some examples:

- Host bigboy with an IP address of 192.168.1.100 isn't part of any DNS domain

```
192.168.1.100    bigboy.localdomain    bigboy
```

- Host bigboy with an IP address of 192.168.1.100 is the mail and web server for domain my-site.com with corresponding entries in the DNS zone file for my-site.com

```
192.168.1.100    bigboy.my-site.com    bigboy    mail    www
```

Note: Only have one line per IP address in this file. If you server has multiple names, then just put the two or three aliases that you feel are most important.

Chapter 3

Simple Network Troubleshooting

= =

In This Chapter

= =

You will eventually find yourself trying to fix a network related problem. Here are some troubleshooting tips to help you discover what the problem could be.

How To See MAC Addresses

There are times when you lose connectivity with another server that is **directly** connected to your local network. Taking a look at the ARP table of the server from which you are troubleshooting will help determine whether or not the remote server's NIC is responding to any type of traffic from your Linux box. Lack of communication at this level may mean:

- Either server may be disconnected from the network
- There may be bad network cabling
- A NIC may be disabled or the remote server may be shut down

Here is a description of the commands you may use to determine ARP values

- The "**ifconfig -a**" command will show you both the NIC's MAC address and the associated IP addresses of the server which you are currently logged in to.

```
[root@bigboy tmp]# ifconfig -a

wlan0 Link encap:Ethernet HWaddr 00:06:25:09:6A:B5
inet addr:192.168.1.100 Bcast:192.168.1.255 Mask:255.255.255.0
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:47379 errors:0 dropped:0 overruns:0 frame:0
TX packets:107900 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:100
RX bytes:4676853 (4.4 Mb) TX bytes:43209032 (41.2 Mb)
Interrupt:11 Memory:c887a000-c887b000

wlan0:0 Link encap:Ethernet HWaddr 00:06:25:09:6A:B5
inet addr:192.168.1.99 Bcast:192.168.1.255 Mask:255.255.255.0
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
Interrupt:11 Memory:c887a000-c887b000
[root@bigboy tmp]#
```

Here you can see the wlan0 interface has two IP addresses 192.168.1.100 and 192.168.1.99 tied to the NIC hardware MAC address of 00:06:25:09:6A:B5

- The "**arp -a**" command will show you the MAC addresses in your server's ARP table. Here we see we have some form of connectivity with the router at address 192.168.1.1

```
[root@bigboy root]# arp -a
bigboypix (192.168.1.1) at 00:09:E8:9C:FD:AB [ether] on wlan0
? (192.168.1.101) at 00:06:25:09:6A:D7 [ether] on wlan0
[root@bigboy root]#
```

- You should also check the ARP table of the remote server to see whether it is populated with acceptable values too.

How To Use "Ping" To Test Network Connectivity

Whether or not your troublesome server is connected to your local network it is always a good practice to force a response from it.

One of the most common methods used to test connectivity across multiple networks is the "**ping**" command. Ping sends ICMP “echo” type packets that request a corresponding ICMP “echo-reply” response from the device at the target address. As most servers will respond to a ping query it becomes a very handy tool. A lack of response could be due to:

- A server with that IP address doesn't exist
- The server has been configured not to respond to pings
- A firewall or router along the network path is blocking ICMP traffic
- You have incorrect routing. Check routes on the local, remote servers and all routers in between. A classic symptom of bad routes on a server is the ability to only ping servers on your local network and nowhere else.

There are a variety of ICMP response codes which can help in further troubleshooting. See the appendix for a full listing of them.

The Linux ping command will send continuous pings, once a second, until stopped with a <Ctrl-C>. Here is an example of a successful ping to the server bigboy at 192.168.1.100

```
[root@smallfry tmp]# ping 192.168.1.101
PING 192.168.1.101 (192.168.1.101) from 192.168.1.100 : 56(84) bytes
of data.
64 bytes from 192.168.1.101: icmp_seq=1 ttl=128 time=3.95 ms
64 bytes from 192.168.1.101: icmp_seq=2 ttl=128 time=7.07 ms
64 bytes from 192.168.1.101: icmp_seq=3 ttl=128 time=4.46 ms
64 bytes from 192.168.1.101: icmp_seq=4 ttl=128 time=4.31 ms

--- 192.168.1.101 ping statistics ---
4 packets transmitted, 4 received, 0% loss, time 3026ms
rtt min/avg/max/mdev = 3.950/4.948/7.072/1.242 ms
[root@smallfry tmp]#
```

You may get a "Destination Host Unreachable" message. There message is caused by your router or server knowing that the target IP address is part of a valid network, but is getting no response from the target server. There are a number of reasons for this:

If you are trying to ping a host on a directly connected network:

- The server may be down, or disconnected for the network.
- Your NIC may not have the correct duplex settings, you may verify this with the **mii-tool** command.
- You may have the incorrect type of cable connecting your Linux box to the network. There are two basic types, “straight-through” and “cross-over”.
- In the case of a wireless network, your SSID or encryption keys may be incorrect

If you are trying to ping a host on remote network:

- The network device doesn’t have a route in its routing table to the destination network and sends an ICMP reply type 3 which triggers the message. The resulting message may be “**Destination Host Unreachable**” or ”**Destination Network Unreachable**”.

```
[root@smallfry tmp]# ping 192.168.1.105
PING 192.168.1.105 (192.168.1.105) from 192.168.1.100 : 56(84) bytes
of data.
From 192.168.1.100 icmp_seq=1 Destination Host Unreachable
From 192.168.1.100 icmp_seq=2 Destination Host Unreachable
From 192.168.1.100 icmp_seq=3 Destination Host Unreachable
From 192.168.1.100 icmp_seq=4 Destination Host Unreachable
From 192.168.1.100 icmp_seq=5 Destination Host Unreachable
From 192.168.1.100 icmp_seq=6 Destination Host Unreachable
--- 192.168.1.105 ping statistics ---
8 packets transmitted, 0 received, +6 errors, 100% loss, time 7021ms,
pipe 3
[root@smallfry tmp]#
```

Using "traceroute" To Test Connectivity

Another tool for network troubleshooting is the traceroute command. It gives a listing of all the router hops between your server and the target server. This helps you verify that routing over the networks in between is correct.

Traceroute works by sending a UDP packet destined to the target with a TTL of "0". The first router on the route recognizes that the TTL has already been exceeded and discards or “drops” the packet, but also sends an ICMP "time exceeded" message back to the source. The traceroute program records the IP address of the router that sent the message and knows that that is the first hop on the path to the final destination. The traceroute program tries again, with a TTL of "1". The first hop, sees nothing wrong with the packet, decrements the TTL to 0 as expected, and forwards the packet to the second hop on the path. Router 2, sees the TTL of "0", drops the packet and replies with an ICMP time exceeded message. Traceroute now knows the IP address of the second router. This continues around and around until the final destination is reached.

Here is a sample output for a query to 144.232.20.158:

```
[root@bigboy root]# traceroute 144.232.20.158
traceroute to 144.232.20.158 (144.232.20.158), 30 hops max, 38 byte
packets
1 adsl-67-120-221-110.dsl.sntc01.pacbell.net (67.120.221.110) 14.304
ms 14.019 ms 16.120 ms
2 dist3-vlan50.sntc01.pbi.net (63.203.35.67) 12.971 ms 14.000 ms
14.627 ms
3 bb1-g1-0.sntc01.pbi.net (63.203.35.17) 15.521 ms 12.860 ms 13.179
ms
4 bb2-p11-0.snfc21.pbi.net (64.161.124.246) 13.991 ms 15.842 ms
15.728 ms
5 bb1-p14-0.snfc21.pbi.net (64.161.124.53) 16.133 ms 15.510 ms 15.909
ms
6 sl-gw11-sj-3-0.sprintlink.net (144.228.44.49) 16.510 ms 17.469 ms
18.116 ms
7 sl-bb25-sj-6-1.sprintlink.net (144.232.3.133) 16.212 ms 14.274 ms
15.926 ms
8 * * *
9 * *
[root@bigboy root]#
```

If there is no response within a 5 second timeout interval a "*" is printed for that probe. Possible causes of this and other traceroute status messages are listed below:

Possible Traceroute Messages

Traceroute Symbol	Description
* * *	Time exceeded. Could be caused by: • A router on the path not sending back the ICMP "time exceeded" messages • A router or firewall in the path blocking the ICMP "time exceeded" messages • The target IP address not responding
!H, !N, or !P	Host, network or protocol unreachable
!X or !A	Communication administratively prohibited. A router Access Control List (ACL) or firewall is in the way
!S	Source route failed. Source routing attempts to force traceroute to use a certain path. Failure may be due to a router security setting

Some devices will prevent traceroute packets directed at their interfaces, but will allow ICMP packets. Using traceroute with a "**-I**" flag forces traceroute to use ICMP packets that may go through. In this case the "* * *", status messages disappear.

```
[root@bigboy root]# traceroute -I 144.232.20.158
traceroute to 144.232.20.158 (144.232.20.158), 30 hops max, 38 byte
packets
1 adsl-67-120-221-110.dsl.sntc01.pacbell.net (67.120.221.110) 14.408
ms 14.064 ms 13.111 ms
2 dist3-vlan50.sntc01.pbi.net (63.203.35.67) 13.018 ms 12.887 ms
13.146 ms
3 bb1-g1-0.sntc01.pbi.net (63.203.35.17) 12.854 ms 13.035 ms 13.745
ms
4 bb2-p11-0.snfc21.pbi.net (64.161.124.246) 16.260 ms 15.618 ms
15.663 ms
5 bb1-p14-0.snfc21.pbi.net (64.161.124.53) 15.897 ms 15.785 ms 17.164
ms
6 sl-gw11-sj-3-0.sprintlink.net (144.228.44.49) 14.443 ms 16.279 ms
15.189 ms
7 sl-bb25-sj-6-1.sprintlink.net (144.232.3.133) 16.185 ms 15.857 ms
15.423 ms
8 sl-bb23-ana-6-0.sprintlink.net (144.232.20.158) 27.482 ms 26.306 ms
26.487 ms
[root@bigboy root]#
```

Always Get A Bidirectional Traceroute

It is always best to get traceroutes from the source IP to the target IP and also from the target IP to the source IP. This is because the packet's return path from the target is sometimes not the same as the path taken to get there. A high traceroute time equates to the round trip time for both the initial traceroute query to each “hop” and the response of each “hop”.

Here is an example of one such case, using disguised IP addresses and provider names. There was once a routing issue between telecommunications carriers FastNet and SlowNet. When a user at IP address 40.16.106.32 did a traceroute to 64.25.175.200, a problem seemed to appear at the 10th. hop with OtherNet. However, when a user at 64.25.175.200 did a traceroute to 40.16.106.32, latency showed up at hop 7 with the return path being very different.

In this case, the real traffic congestion was occurring where FastNet handed traffic off to SlowNet in the second trace. The latency appeared to be caused at hop 10 on the first trace not because that hop was slow, but because that was the first hop at which the return packet traveled back to the source via the congested route. Remember, traceroute gives the packet round trip time.

Trace route to 40.16.106.32 from 64.25.175.200

```
1  0    ms 0    ms 0    [64.25.175.200]
2  0    ms 0    ms 0    [64.25.175.253]
3  0    ms 0    ms 0    border-from-40-tesser.boulder.co.coop.net
[207.174.144.169]
4  0    ms 0    ms 0    [64.25.128.126]
5  0    ms 0    ms 0    p3-0.dnvtco1-cr3.othernet.net [4.25.26.53]
6  0    ms 0    ms 0    p2-1.dnvtco1-br1.othernet.net [4.24.11.25]
7  0    ms 0    ms 0    p15-0.dnvtco1-br2.othernet.net [4.24.11.38]
8  30   ms 30   ms 30   p15-0.snjpca1-br2.othernet.net [4.0.6.225]
9  30   ms 30   ms 30   p1-0.snjpca1-cr4.othernet.net [4.24.9.150]
10 1252 ms 1212 ms 1202 h0.webhostinc2.othernet.net [4.24.236.38]
11 1252 ms 1212 ms 1192 [40.16.96.11]
12 1262 ms 1212 ms 1192 [40.16.96.162]
13 1102 ms 1091 ms 1092 [40.16.106.32]
```

Trace route to 64.25.175.200 from 40.16.106.32

```
1  1    ms 1    ms 1    ms [40.16.106.3]
2  1    ms 1    ms 1    ms [40.16.96.161]
3  2    ms 1    ms 1    ms [40.16.96.2]
4  1    ms 1    ms 1    ms [40.16.96.65]
5  2    ms 2    ms 1    ms border8.p4-2.webh02-1.sfj.fastnet.net
[216.52.19.77]
6  2    ms 1    ms 1    ms core1.ge0-1-net2.sfj.fastnet.net
[216.52.0.65]
7  993  ms 961  ms 999  ms sjo-edge-03.inet.slownet.net
[208.46.223.33]
8  1009 ms 1008 ms 971  ms sjo-core-01.inet.slownet.net
[205.171.22.29]
9  985  ms 947  ms 983  ms svl-core-03.inet.slownet.net
[205.171.5.97]
```

```
10 1028 ms 1010 ms 953  ms [205.171.205.30]
11 989  ms 988  ms 985  ms p4-3.paix-bi1.othernet.net [4.2.49.13]
12 1002 ms 1001 ms 973  ms p6-0.snjpca1-br1.othernet.net
[4.24.7.61]
13 1031 ms 989  ms 978  ms p9-0.snjpca1-br2.othernet.net
[4.24.9.130]
14 1031 ms 1017 ms 1017 ms p3-0.dnvtco1-br2.othernet.net
[4.0.6.226]
15 1027 ms 1025 ms 1023 ms p15-0.dnvtco1-br1.othernet.net
[4.24.11.37]
16 1045 ms 1037 ms 1050 ms p1-0.dnvtco1-cr3.othernet.net
[4.24.11.26]
17 1030 ms 1020 ms 1045 ms p0-0.cointcorp.othernet.net [4.25.26.54]
18 1038 ms 1031 ms 1045 ms gw234.boulder.co.coop.net [64.25.128.99]
19 1050 ms 1094 ms 1034 ms [64.25.175.253]
20 1050 ms 1094 ms 1034 ms [64.25.175.200]
```

Ping & Traceroute Troubleshooting Example

In this example, a ping to 186.9.17.153 gave a “TTL timeout” message. Ping TTLs will usually only timeout if there is a routing loop in which the packet bounces between two routers on the way to the target. Each “bounce” causes the TTL to decrease by a count of one until the TTL reaches zero at which point you get the timeout.

The routing loop was confirmed by the traceroute in which the packet was proven to be bouncing between routers at 186.40.64.94 and 186.40.64.93.

```
G:\>ping 186.9.17.153

Pinging 186.9.17.153 with 32 bytes of data:

Reply from 186.40.64.94: TTL expired in transit.
Reply from 186.40.64.94: TTL expired in transit.
Reply from 186.40.64.94: TTL expired in transit.
Reply from 186.40.64.94: TTL expired in transit.

Ping statistics for 186.9.17.153:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum =  0ms, Average =  0ms


G:\>tracert 186.9.17.153

Tracing route to lostserver.confusion.net [186.9.17.153]
over a maximum of 30 hops:

  1   <10 ms   <10 ms   <10 ms  186.217.33.1
  2    60 ms    70 ms    60 ms  rtr-2.confusion.net [186.40.64.94]
  3    70 ms    71 ms    70 ms  rtr-1.confusion.net [186.40.64.93]
  4    60 ms    70 ms    60 ms  rtr-2.confusion.net [186.40.64.94]
  5    70 ms    70 ms    70 ms  rtr-1.confusion.net [186.40.64.93]
```

```
  6    60 ms    70 ms    61 ms  rtr-2.confusion.net [186.40.64.94]
  7    70 ms    70 ms    70 ms  rtr-1.confusion.net [186.40.64.93]
  8    60 ms    70 ms    60 ms  rtr-2.confusion.net [186.40.64.94]
  9    70 ms    70 ms    70 ms  rtr-1.confusion.net [186.40.64.93]
...
...
...
Trace complete.
```

This problem was solved by resetting the routing process on both routers. The problem was initially triggered by an unstable network link that caused frequent routing recalculations. The constant activity eventually corrupted the routing tables of one of the routers.

Possible Reasons For Failed Traceroutes

Traceroutes can fail to reach their intended destination for a number of reasons, these include:

- Traceroute packets are being blocked or rejected by a router in the path. The router immediately after the last visible one is usually the culprit. It's usually good to check the routing table and/or other status of this next hop device.
- The target server doesn't exist on the network. It could be disconnected, or turned off. (!H or !N messages may be produced.)
- The network on which you expect the target host to reside doesn't exist in the routing table of one of the routers in the path (!H or !N messages may be produced.)
- You may have a typographical error in the IP address of the target server
- You may have a routing loop in which packets bounce between two routers and never get to the intended destination.
- The packets don't have a proper return path to your server. The last visible hop being the last hop in which the packets return correctly. The router immediately after the last visible one is the one at which the routing changes. It's usually good to:
  - log on to the last visible router.
  - Look at the routing table to determine what the next hop is to your intended traceroute target.
  - Log on to this next hop router.
  - Do a traceroute from this router to your intended target server.
  - **If this works**: Routing to the target server is OK. Do a traceroute back to your source server. The traceroute will probably fail at the bad router on the return path.
  - **If it doesn't work**: Test the routing table and/or other status of all the hops between it and your intended target.

Note: If there is nothing blocking your traceroute traffic, then the last visible router of an incomplete trace is either the last good router on the path, or the last router that has a valid return path to the server issuing the traceroute.

Viewing Packet Flow With TCPdump

Tcpdump is one of the most popular packages for viewing the flow of packets through your Linux box's NIC card. It is installed by default on RedHat linux and has very simple syntax, especially if you are doing simpler types of troubleshooting.

One of the most common uses of tcpdump is to determine whether you are getting basic two way communication. Lack of communication could be due to:

- Bad routing
- Faulty cables, interfaces of devices in the packet flow
- The server not listening on the port because the software isn't installed or started

Analyzing tcpdump in much greater detail is beyond the scope of this section.

Like most Linux commands, tcpdump uses command line switches to modify the output. Some of the more useful command line switches would include:

Possible TCPdump Messages

tcpdump command switch	Description
-c	Stop after viewing *count* packets.
-i	Listen on *interface*. If this is not specified, then tcpdump will use the lowest numbered interface that is UP
-t	Don't print a timestamp at the beginning of each line

You can also add expressions after all the command line switches. These act as filters to limit the volume of data presented on the screen. You can also use keywords such as "and" or "or" between expressions to further fine tune your selection criteria. Some useful expressions include:

Useful TCPdump Expressions

tcpdump command expression	Description
host *host-address*	View packets from the IP address *host-address*
icmp	View icmp packets
tcp port port-number	View TCP packets with packets with either a source or destination TCP port of port-number
udp port port-number	View UDP packets with either a source or destination UDP port of port-number

Example: tcpdump used to view ICMP "ping" packets going through interface wlan0

```
[root@bigboy tmp]# tcpdump -i wlan0 icmp

tcpdump: listening on wlan0
21:48:58.927091 smallfry > bigboy.my-site.com: icmp: echo request
(DF)
21:48:58.927510 bigboy.my-site.com > smallfry: icmp: echo reply
21:48:58.928257 smallfry > bigboy.my-site.com: icmp: echo request
(DF)
21:48:58.928365 bigboy.my-site.com > smallfry: icmp: echo reply
21:48:58.943926 smallfry > bigboy.my-site.com: icmp: echo request
(DF)
21:48:58.944034 bigboy.my-site.com > smallfry: icmp: echo reply
21:48:58.962244 bigboy.my-site.com > smallfry: icmp: echo reply
21:48:58.963966 bigboy.my-site.com > smallfry: icmp: echo reply
21:48:58.968556 bigboy.my-site.com > smallfry: icmp: echo reply

9 packets received by filter
0 packets dropped by kernel
[root@bigboy tmp]#
```

Explanation

- The first column of data is a packet time stamp.
- The second column of data shows the packet source then destination IP address or server name of the packet

- The third column shows the packet type
- Two way communication is occurring as each echo gets an echo reply

Example: tcpdump used to view packets on interface wlan0 to/from host 192.168.1.102 on TCP port 22 with no timestamps in the output

```
[root@bigboy root]# tcpdump -i wlan0 -t host 192.168.1.102 and tcp port
22

tcpdump: listening on wlan0
smallfry.32938 > bigboy.my-site.com.ssh: S 2013297020:2013297020(0)
win 5840 <mss 1460,sackOK,timestamp 75227931 0,nop,wscale 0> (DF)
[tos 0x10]
bigboy.my-site.com.ssh > smallfry.32938: R 0:0(0) ack 2013297021
win 0 (DF) [tos 0x10]
smallfry.32938 > bigboy.my-site.com.ssh: S 2013297020:2013297020(0)
win 5840 <mss 1460,sackOK,timestamp 75227931 0,nop,wscale 0> (DF)
[tos 0x10]
bigboy.my-site.com.ssh > smallfry.32938: R 0:0(0) ack 1 win 0 (DF)
[tos 0x10]
smallfry.32938 > bigboy.my-site.com.ssh: S 2013297020:2013297020(0)
win 5840 <mss 1460,sackOK,timestamp 75227931 0,nop,wscale 0> (DF)
[tos 0x10]
bigboy.my-site.com.ssh > smallfry.32938: R 0:0(0) ack 1 win 0 (DF)
[tos 0x10]
bigboy.my-site.com.ssh > smallfry.32938: R 0:0(0) ack 1 win 0 (DF)
[tos 0x10]

bigboy.my-site.com.ssh > smallfry.32938: R 0:0(0) ack 1 win 0 (DF)
[tos 0x10]
bigboy.my-site.com.ssh > smallfry.32938: R 0:0(0) ack 1 win 0 (DF)
[tos 0x10]

9 packets received by filter
0 packets dropped by kernel
[root@bigboy root]#
```

Explanation

- The first column of data shows the packet source then destination IP address or server name of the packet
- The second column shows the TCP flags within the packet
- The client named "bigboy" is using port 32938 to communicate with the server named "smallfry" on the TCP SSH port 22.
- Two way communication is occurring

Using Telnet

An easy way for you to tell if a remote server is listening on a specific TCP port is to use the telnet command. By default, telnet will try to connect on TCP port 23, but you can specify other TCP ports by typing them in after the target IP address. In this case we're testing to see if the remote server is listening on port 22 (SSH)

```
[root@bigboy tmp]# telnet 192.168.1.102 22
Trying 192.168.1.102...
Connected to 192.168.1.102.
Escape character is '^]'.
SSH-1.99-OpenSSH_3.4p1
^]
telnet> quit
Connection closed.
[root@ bigboy tmp]#
```

To break out of the connection you have to use <ctrl>], not the usual <crtl>C. You will geta "connection refused" message if the remote server isn't listening on the specified target port as shown in the example below.

```
[root@bigboy tmp]# telnet 192.168.1.102 22
Trying 192.168.1.100...
telnet: connect to address 192.168.1.100: Connection refused
[root@ bigboy tmp]#
```

I have used this technique often to troubleshoot remote webservers that aren't serving web pages. If it is listening on port 80, and no pages are being served, then the problem is usually due to a bad web application, not the web server software itself. If it isn't listening to port 80 at all, then it could be an Apache / IIS problem, or the server could be down, or there could be a network related problem that prevents connectivity from being established.

Using NMAP

NMAP is program that you can use to determine all the TCP ports on which a remote server is listening. It isn't usually an important tool in the home environment but in a corporate environment it can be helpful in getting a better idea of the function of undocumented servers that have been found by your boss. You can also use it to run a vulnerability scan against your site as NMAP is one of the tools used by malicious surfers.

Here is a sample output:

- First we see all the available options by just typing the "nmap" command

```
[root@bigboy tmp]# nmap
Nmap V. 3.00 Usage: nmap [Scan Type(s)] [Options] <host or net list>
Some Common Scan Types ('*' options require root privileges)
* -sS TCP SYN stealth port scan (default if privileged (root))
```

```
  -sT TCP connect() port scan (default for unprivileged users)
* -sU UDP port scan
  -sP ping scan (Find any reachable machines)
* -sF,-sX,-sN Stealth FIN, Xmas, or Null scan (experts only)
  -sR/-I RPC/Identd scan (use with other scan types)
Some Common Options (none are required, most can be combined):
* -O Use TCP/IP fingerprinting to guess remote operating system
  -p <range> ports to scan.  Example range: '1-1024,1080,6666,31337'
  -F Only scans ports listed in nmap-services
  -v Verbose. Its use is recommended.  Use twice for greater effect.
  -P0 Don't ping hosts (needed to scan www.microsoft.com and others)
* -Ddecoy_host1,decoy2[,...] Hide scan using many decoys
  -T <Paranoid|Sneaky|Polite|Normal|Aggressive|Insane> General timing
policy
  -n/-R Never do DNS resolution/Always resolve [default: sometimes
resolve]
  -oN/-oX/-oG <logfile> Output normal/XML/grepable scan logs to <logfile>
  -iL <inputfile> Get targets from file; Use '-' for stdin
* -S <your_IP>/-e <devicename> Specify source address or network
interface
  --interactive Go into interactive mode (then press h for help)
Example: nmap -v -sS -O www.my.com 192.168.0.0/16 '192.88-90.*.*'
SEE THE MAN PAGE FOR MANY MORE OPTIONS, DESCRIPTIONS, AND EXAMPLES
[root@bigboy tmp]#
```

- Then we scan by trying to make regular connections (**-sT**) in the extremely fast "insane" mode **(–T 5**) from ports 1 to 5000.

```
[root@bigboy tmp]# nmap -sT -T 5 -p 1-5000 192.168.1.153

Starting nmap V. 3.00 ( www.insecure.org/nmap/ )
Interesting ports on whoknows.my-site-int.com (192.168.1.153):
(The 4981 ports scanned but not shown below are in state: closed)
Port       State       Service
21/tcp     open        ftp
25/tcp     open        smtp
135/tcp    open        loc-srv
139/tcp    open        netbios-ssn
199/tcp    open        smux
465/tcp    open        smtps
507/tcp    open        crs
2103/tcp   open        unknown
2105/tcp   open        eklogin
2107/tcp   open        unknown
2301/tcp   open        compaqdiag
3300/tcp   open        unknown
Nmap run completed -- 1 IP address (1 host up) scanned in 8
seconds
[root@bigboy tmp]#
```

Who Has Used My System?

It is always important to know who has logged into your Linux box. This isn't just to help track the activities of malicious users, but mostly to figure out who made the mistake that crashed the system or blew up Apache with a typographical error in the httpd.conf file.

The "Last" Command

The most common command to do this is "last" which will list the last users who logged into the system. Here are some examples:

```
[root@rahtid log]# last -100
root     pts/0        reggae.simiya.co Thu Jun 19 09:26   still logged
in
root     pts/0        reggae.simiya.co Wed Jun 18 01:07 - 09:26
(1+08:18)
reboot   system boot  2.4.18-14        Wed Jun 18
01:07         (1+08:21)
root     pts/0        reggae.simiya.co Tue Jun 17 21:57 -
down   (03:07)
root     pts/0        reggae.simiya.co Mon Jun 16 07:24 -
00:35  (17:10)
root     pts/0        reggae.simiya.co Sun Jun 15 16:29 -
17:26  (00:56)
wtmp begins Sun Jun 15 16:29:18 2003
[root@rahtid log]#
```

In the example above someone from reggare.simiya.com logged into bigboy as user root. I generally prefer not to give out the "root" password and let all the systems administrators log in with their own individual logins. They can then get "root" privileges by using sudo. This makes it easier to track down individuals versus groups of users.

The "who" Command

This command is used to see who is currently logged in. Here we see a user logged as root from server reggae.simiya.com .

```
[root@rahtid log]# who
root     pts/0        Jun 19 09:26 (reggae.simiya.com)
[root@rahtid log]#
```

Chapter 4

Troubleshooting With Syslog

= =

In This Chapter

= =

This chapter will show you how to troubleshoot problems not only on your Linux box but also on remote networking devices using syslog.

Syslog

About syslog

Syslog is a utility for tracking and logging all manner of system messages from the merely informational to the extremely critical. Each system message sent to the syslog server has two descriptive labels associated with it that makes the message easier to handle.

- The first describes the function (facility) of the application that generated it. For example, applications such as mail and cron generate messages with easily identifiable facilities named "mail" and "cron".
- The second describes the degree of severity of the message. There are eight in all and they are listed below:

Syslog Facilities

Severity Level	Keyword	Description
0	**emergencies**	System unusable
1	**alerts**	Immediate action required
2	**critical**	Critical condition
3	**errors**	Error conditions
4	**warnings**	Warning conditions
5	**notifications**	Normal but significant conditions
6	**informational**	Informational messages
7	**debugging**	Debugging messages

The files to which syslog will write each type of message received is set in the **/etc/syslog.conf** configuration file. This file consists of two columns, the first lists the facilities and severities of messages to expect and the second lists the files to which they should be logged. By default, RedHat's **/etc/syslog.conf** file is configured to put most of the messages the file **/var/log/messages**. Here is a sample:

```
.info;mail.none;authpriv.none;cron.none                /var/log/message
s
```

In this case, all messages of severity "info" and above are logged, but none from the mail, cron or authentication facilities/subsystems. You can make this logging even more sensitive by replacing the line above with one that captures all messages from debug severity and above in the **/var/log/messages** file. This may be more suitable for troubleshooting.

```
*.debug                                                /var/log/messages
```

Certain applications will additionally log to their own application specific log files and directories independent of the syslog.conf file. Here are some common examples:

Files:

```
/var/log/maillog              : Mail
/var/log/httpd/access_log     : Apache web server page access logs
```

Directories:

```
/var/log
/var/log/samba                  : Samba messages
/var/log/mrtg                   : MRTG messages
/var/log/httpd                  : Apache webserver messages
```

NOTE: The **/etc/syslog.conf** file is very sensitive to spaces. Only use tabs on lines that don't start with the "#" comment character. Spaces in the file will cause unpredictable results.

Activating Changes To The syslog Configuration File

Changes to **/etc/syslog.conf** will not take effect until you restart syslog. Issue this command to do so:

```
[root@bigboy tmp]# /etc/init.d/syslog restart
```

How To View New Log Entries As They Happen

If you want to get new log entries to scroll on the screen as they occur, then you can use this command:

```
[root@bigboy tmp]# tail -f /var/log/messages
```

Similar commands can be applied to all log files. This is probably one of the best troubleshooting tools available in Linux. Another good command to use apart from "tail" is "**grep**". Grep will help you search for all occurrences of a string in a log file, you can pipe it through the "**more**" command so that you only get one screen at a time. Here is an example.

```
[root@bigboy tmp]# grep string /var/log/messages | more
```

You can also just use the plain old "**more**" command to see one screen at a time of the entire log file without filtering with "**grep**". Here is an example:

```
[root@bigboy tmp]# more /var/log/messages
```

Logging Syslog Messages To A Remote Linux Server

Configuring the Linux Syslog Server

By default syslog doesn't expect to receive messages from remote clients. Here's how to configure your Linux server to start listening for these messages.

As we saw previously, syslog checks its **/etc/syslog.conf** file to determine the expected names and locations of the log files it should create. It also checks the file **/etc/sysconfig/syslog** to determine the various modes in which it should operate. Syslog will not listen for remote messages unless the SYSLOGD_OPTIONS variable in this file has a "–r" included in it as shown below.

```
# Options to syslogd
# -m 0 disables 'MARK' messages.
# -r enables logging from remote machines
# -x disables DNS lookups on messages recieved with -r
# See syslogd(8) for more details
SYSLOGD_OPTIONS="-m 0 -r"
# Options to klogd
# -2 prints all kernel oops messages twice; once for klogd to
decode, and
#    once for processing with 'ksymoops'
# -x disables all klogd processing of oops messages entirely
# See klogd(8) for more details
KLOGD_OPTIONS="-2"
```

You will have to restart syslog on the server for the changes to take effect. The server will now start to listen on UDP port 514 which you can verify using either one of the following **netstat** command variations.

```
[root@bigboy tmp]# netstat -a | grep syslog
udp        0      0
*:syslog                *:*
[root@bigboy tmp]# netstat -an | grep 514
udp        0      0
0.0.0.0:514             0.0.0.0:*
[root@bigboy tmp]#
```

Configuring the Linux Client

The syslog server is now expecting to receive syslog messages. You now have to configure your remote Linux client to send messages to it. This is done by editing the **/etc/hosts** file on the Linux client named **smallfry**. Here are the steps:

- Determine the IP address and fully qualified hostname of your remote logging host
- Add an entry in the **/etc/hosts** file in the format

```
IP-address    fully-qualified-domain-
name    hostname    "loghost"
```

Example:

```
192.168.1.100   bigboy.my-site.com    bigboy     loghost
```

Now your **/etc/hosts** file has a nickname of “loghost” for server "**bigboy**"

- The next thing you need to do is edit your **/etc/syslog.conf** file to make the syslog messages get sent to your new “loghost” nickname.

```
*.debug                                                 @loghost
*.debug                                                 /var/log/messag
es
```

In this example we have all debug messages and higher being logged to both server **bigboy** (“loghost”) and the local file **/var/log/messages**. Remember to restart syslog to get the remote logging started.

You can now test to make sure that the syslog server is receiving the messages with a simple test such as restarting the **lpd** printer daemon and making sure the remote server sees the messages.

Linux Client

```
[root@smallfry tmp]# /etc/init.d/lpd restart
Stopping lpd: [  OK  ]
Starting lpd: [  OK  ]
[root@smallfry tmp]#
```

Linux Server

```
[root@bigboy tmp]# tail /var/log/messages
...
...
Apr 11 22:09:35 smallfry lpd: lpd shutdown succeeded
Apr 11 22:09:39 smallfry lpd: lpd startup succeeded
...
...
[root@bigboy tmp]#
```

Syslog Configuration and Cisco Network Devices

Syslog reserves facilities "local0" through "local7" for log messages received from remote servers and network devices. Routers, switches, firewalls and load balancers each logging with a different facility can each have their own log files for easy troubleshooting. The Cisco Home Networking companion guide has examples of how to configure syslog to do this with Cisco devices using separate log files for the routers, switches, PIX firewalls, CSS arrowpoints and LocalDirectors.

Syslog and Firewalls

Syslog listens by default on UDP port 514. If you are logging to a remote syslog server via a firewall, you'll have to allow traffic on this port to pass through the security device. Syslog messages usually have both their source and destination UDP ports being 514.

Logrotate

Logrotate is a Linux utility that renames and reuses system error log files on a periodic basis so that they don't occupy excessive disk space.

The /etc/logrotate.conf File

This is logrotate's general configuration file in which you can specify the frequency with which the files are reused.

- You can specify either "weekly" or "daily" rotation parameter. In the case below the weekly option is "commented out" with a "#", allowing for daily updates.
- The "rotate" parameter specifies the number of copies of log files logrotate will maintain. In the case below the 4 copy option is "commented out" with a "#", while allowing 7 copies.
- The "create" parameter creates a new log file after each rotation

Therefore our sample configuration file will create daily archives of **ALL** the logfiles and store them for seven days. The files will have the following names with "logfile" being current active version:

```
logfile
logfile.0
logfile.1
logfile.2
logfile.3
logfile.4
logfile.5
logfile.6
```

Sample contents of /etc/logrotate.conf

```
# rotate log files weekly
#weekly
# rotate log files daily
daily

# keep 4 weeks worth of backlogs
#rotate 4

# keep 7 days worth of backlogs
rotate 7

# create new (empty) log files after rotating old ones
create
```

The /etc/logrotate.d Directory

Most Linux applications that use syslog will put an additional configuration file in this directory to specify the names of the log files to be rotated. It is a good practice to verify that all new applications that you want to use the syslog log have configuration files in this directory. Here are some sample files which define the specific files to be rotated for each application.

The /etc/logrotate.d/syslog File (For General System Logging)

```
/var/log/messages /var/log/secure /var/log/maillog
/var/log/spooler /var/log/boot.log /var/log/cron {
    sharedscripts
    postrotate
    /bin/kill -HUP `cat /var/run/syslogd.pid 2> /dev/null` 2>
/dev/null || true
    endscript
}
```

The /etc/logrotate.d/apache File (For Apache)

```
/var/log/httpd/access_log /var/log/httpd/agent_log
/var/log/httpd/error_log /var/log/httpd/referer_log {
    missingok
    sharedscripts
    postrotate
    /bin/kill -HUP `cat /var/run/httpd.pid 2>/dev/null` 2>
/dev/null || true
    endscript
}
```

The /etc/logrotate.d/samba File (for SAMBA)

```
/var/log/samba/*.log {
    notifempty
    missingok
    sharedscripts
    copytruncate
    postrotate
    /bin/kill -HUP `cat /var/lock/samba/*.pid 2> /dev/null` 2>
/dev/null || true
    endscript
}
```

Activating logrotate

The above logrotate settings will not take effect until you issue the following command to do so:

```
[root@bigboy tmp]# logrotate –f
```

If you want logrotate to reload only a specifc configuration file, and not all of them, then issue the logrotate command with just that filename as the argument like this:

```
[root@bigboy tmp]# logrotate -f /etc/logrotate.d/syslog
```

Chapter 5

Installing RPM Software

==

In This Chapter

==

A lot of Linux system software is available using RPM packages for default Linux installs, and source RPMs for non standard installations. As the procedure for installing source RPMs involves compiling source code, they more easily installed across a wide variety of Linux flavors, thereby making life easier for the software developer who wrote the package.

Where To Get Commonly Used RPMs

Here are three commonly used sources for RPMs:

RPMs On Your Installation CDs

This is usually easier than having to download files from a remote website. See the section about using Automount to easily access your CDROM drive to obtain RPM files.

RPMs Downloaded From Redhat

Advanced searches for all versions of RedHat can be done using this web link:

http://www.redhat.com/apps/download/advanced_search.html

RedHat also has a highly used FTP site, ftp.redhat.com, start your search in the **/pub/redhat/linux/** directory and move down the directory tree. If you're new to FTP, don't worry, FTP downloading it'll be explained later.

RPMs Downloaded From Speakeasy

RedHat only has their approved software on their site. A good general purpose source is RPMfind. Always remember to select the RPM that matches your version of Linux

http://speakeasy.rpmfind.net/

How to Easily Access CD RPMs With Automount

Using the Linux installation CDs is usually easier, though you run the risk of some of the packages being obsolete due to newer releases on the RedHat website.

It is usually simplest to configure your system to Automount your CDROM. This makes the files on it immediately accessible whenever you access it without having to use the "mount" command. This will make your Linux system act more like Windows.

- **Autofs** is the package that supports Automount is installed by default with newer versions of RedHat Linux. You can check this using the following commands.

```
[root@bigboy tmp]# rpm -qa | grep autofs
autofs-3.1.7-33
[root@bigboy tmp]#
```

- You can then ensure that it runs when the system boots using the **chkconfig** command.

```
[root@bigboy tmp]# chkconfig --level 345 autofs on
[root@bigboy tmp]#
```

- There are two automount configuration files in **/etc**, one called **auto.master** and the other called **auto.misc**. My **auto.master** looks like this:

```
/misc       /etc/auto.misc     --timeout 60
```

The default version of this file normally has this line commented out so you'll have to remove the "#" at the beginning of the line for the configuration to take effect when **autofs** is restarted. The first entry is not the mount point. It's where the set of **autofs** mount points will be. The second entry is a reference to the default map file **/etc/auto.misc** and the third

option says that the mounted filesystems will automatically unmount themselves 60 seconds after use.

- Edit your **auto.misc** file to include the CDROM. It should have an entry like this.

```
cdrom -fstype=iso9660,ro,nosuid,nodev :/dev/cdrom
```

You'll find other entries such as "floppy" and "zip" commented out with a "#". If you need them, just delete the "#". The first column (the "key") is the mount point under directory **/misc**, so in this case you'll be doing auto mounting when you access **/misc/cdrom.**

Note: some versions of linux may refer to the CDROM drive in **auto.misc** as just **cd** and not **cdrom** as in the example above. In this case, you may have to find yourself referring to your automounting CDROM as **/misc/cd** instead.

- Restart autofs.

```
[root@bigboy tmp]# /etc/init.d/autofs restart
Stopping automount:[ OK ]
Starting automount:[ OK ]
[root@bigboy tmp]#
```

- Insert the CDROM and take a look at the contents. Nothing will appear in **/misc** at first as the automounting isn't triggered until you actually access the CDROM drive.

```
[root@rahtid root]# ll /misc
total 0
[root@rahtid root]#
```

We access the drive, and everything appears.

```
[root@bigboy tmp]# ll /misc/cdrom
total 113
dr-xr-xr-x    2 root     root         2048 Sep 15  1999
BonusChapter
dr-xr-xr-x    8 root     root         2048 Sep 15  1999 stage
dr-xr-xr-x    3 root     root         2048 Sep 15  1999
Translations
[root@bigboy tmp]#
```

- When finished, eject the CDROM

```
[root@bigboy tmp]# eject cdrom
```

Downloading RPMS To Your Linux Box

For casual searching and installing, I recommend using the http links above. If you are doing industrial strength stuff, then use a real FTP client such as (WSFTP or CuteFTP for GUI) or the command line.

Getting RPMs Using Web Based FTP

Let's say you are running RedHat 8.0 and need to download an RPM for the DHCP server.

RedHat

- Use your web browser to go to the RedHat link above
- Type in dhcp in the search box
- Click the search button
- Scroll down for the RPM you need for the DHCP server
- Click on the appropriate "download" link
- Click on the FTP link
- Save the file to Linux box's hard drive

Speakeasy

- Go to the Speakeasy link
- Type in dhcp in the search box
- Click the search button
- Scroll down for the RPM that matches your version of RedHat
- The right hand column has the links with the actual names of the rpm files
- Click the link
- Save the file to Linux box's hard drive

It is best to download RPMs to a directory named "RPM", so you can find them later.

Getting RPMs Using Command Line Anonymous FTP

The Web based method above transparently uses anonymous File Transfer Protocol (FTP). Anonymous FTP allows you to log in and download files from a FTP server using the username "anonymous" and a password that matches your email address. This way anyone can access the data.

- Let's try to FTP the SSH package from ftp.redhat.com

```
[root@bigboy tmp]# ftp ftp.redhat.com
Trying 66.77.185.38...
Connected to ftp.redhat.com (66.77.185.38).
220 Red Hat FTP server ready. All transfers are logged.
Name (ftp.redhat.com:root): anonymous
331 Please specify the password.
Password:
230 Login successful. Have fun.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
227 Entering Passive Mode (66,77,185,38,50,122)
150 Here comes the directory listing.
drwxr-xr-x 5 0 0 4096 Jun 09 04:20 pub
226 Directory send OK.
ftp>
```

- Let's see the available help commands

```
ftp> help
Commands may be abbreviated. Commands are:
```

!	debug	mdir	sendport	site
$	dir	mget	put	size
account	disconnect	mkdir	pwd	status
append	exit	mls	quit	struct
ascii	form	mode	quote	system
bell	get	modtime	recv	sunique
binary	glob	mput	reget	tenex
bye	hash	newer	rstatus	tick
case	help	nmap	rhelp	trace
cd	idle	nlist	rename	type
cdup	image	ntrans	reset	user
chmod	lcd	open	restart	umask
close	ls	prompt	rmdir	verbose
cr	macdef	passive	runique	?
delete	Mdelete	proxy	send	

```
ftp>
```

- The commands you'll most likely use are:

FTP Commands

Command	Description
binary	Copy files in binary mode
cd	Change directory on the FTP server
dir	List the names of the files in the current remote directory
exit	Bye bye
get	Get a file from the FTP server
lcd	Change the directory on the local machine
ls	Same as dir
mget	Same as get, but you can use wildcards like "*"
mput	Same as put, but you can use wildcards like "*"
passive	Make the file transfer passive mode
put	Put a file from the local machine onto the FTP server
pwd	Give the directory name on the local machine

- By using the search feature on the website ahead of time, I know that the RedHat 8.0RPMs are located in the **pub/redhat/linux/8.0/en/os/i386/RedHat/RPMS** directory.

```
ftp> cd pub/redhat/linux/8.0/en/os/i386/RedHat/RPMS
250 Directory successfully changed.
ftp> ls open*
227 Entering Passive Mode (66,77,185,38,45,180)
150 Here comes the directory listing.
-rw-r--r-- 1 0 0   11191  Sep 03 21:32 open-1.4-16.i386.rpm
-rw-r--r-- 1 0 0 2006950  Sep 03 21:32 openh323-1.9.3-4.i386.rpm
-rw-r--r-- 1 0 0  256971  Sep 03 21:32 openh323-devel-1.9.3-
4.i386.rpm
...
...
-rw-r--r-- 1 0 0  217326  Sep 03 21:33 openssh-3.4p1-2.i386.rpm
```

```
...
...
226 Directory send OK.
ftp>
```

- Get the file we need and place it in the local directory **/usr/rpm**. Also print "#" hash signs on the screen during the download.

```
ftp> hash
Hash mark printing on (1024 bytes/hash mark).
ftp> lcd /usr/rpm
Local directory now /usr/rpm
ftp>
ftp> get openssh-3.4p1-2.i386.rpm
local: openssh-3.4p1-2.i386.rpm remote: openssh-3.4p1-2.i386.rpm
227 Entering Passive Mode (66,77,185,38,57,102)
150 Opening BINARY mode data connection for openssh-3.4p1-
2.i386.rpm (217326 bytes).
######################################################################
###
######################################################################
###
######################################################################
##
226 File send OK.
217326 bytes received in 87.7 secs (2.4 Kbytes/sec)
ftp>
```

- Bye bye

```
ftp> exit
221 Goodbye.
[root@bigboy tmp]#
```

Getting RPMs Using WGET

The "**wget**" command can be used to quickly download files when you already know the URL at which the RPM is located. This is especially convenient when using a Microsoft Windows or X-Windows machine to log into a remote Linux server via SSH or telnet. You can browse the RedHat or Speakeasy sites for the RPM you need, right click on the desired link and select "copy shortcut" (Windows) or "Copy Link Location" (Linux). Once you have done this, you can then select your SSH / Telnet window and type in the command "**wget URL**". Here is an example downloading an SSH update from RedHat.

```
[root@bigboy tmp]# wget updates.redhat.com/9/en/os/i386/openssh-3.5p1-
11.i386.rpm
--18:54:20--  http://updates.redhat.com/9/en/os/i386/openssh-3.5p1-
11.i386.rpm
```

```
          => `openssh-3.5p1-11.i386.rpm'
Resolving updates.redhat.com... done.
Connecting to updates.redhat.com[66.187.224.52]:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 181,895 [application/x-rpm]

100%[==============================>] 181,895       98.36K/s    ETA 00:00

18:54:22 (98.36 KB/s) - `openssh-3.5p1-11.i386.rpm' saved [181895/181895]
[root@bigboy tmp]
```

How To Install The RPMs

Using Downloaded Files

- Download the source RPMs which usually have a file extension ending with (.rpm) into a temporary directory such as **/tmp**
- As user root, issue the following command:

```
[root@bigboy tmp]# rpm -Uvh filename.rpm
```

Using CDROMs

- Insert the CDROM and check the files in **/misc/cdrom/RedHat/RPMS**

```
[root@bigboy tmp]# cd /misc/cdrom/RedHat/RPMS
[root@bigboy RPMS]# ls filename*
filename.rpm
[root@bigboy RPMS]# rpm -Uvh filename.rpm
```

- When finished, eject the CDROM

```
[root@bigboy RPMS]# cd /tmp
[root@bigboy tmp]# eject cdrom
[root@bigboy tmp]#
```

How to Install Source RPMs

Sometimes the packages you want to install need to be compiled in order to match your kernel version. This requires you to use source RPM files.

- Download the source RPMs or locate them on your CD collection. They usually have a file extension ending with (.src.rpm)
- Run the following commands as root:

Newer Linux Versions

Compiling and installing source RPMs with newer RedHat Linux versions can be done simply with the **rpmbuild** command

```
[root@bigboy tmp]# rpmbuild --rebuild filename.src.rpm
```

- Here is an example in which we install the tacacs plus package.

```
[root@bigboy rpm]# rpmbuild --rebuild tac_plus-4.0.3-2.src.rpm
Installing tac_plus-4.0.3-2.src.rpm
Executing(%prep): /bin/sh -e /var/tmp/rpm-tmp.61594
+ umask 022
+ cd /usr/src/redhat/BUILD
+ cd /usr/src/redhat/BUILD
+ rm -rf tac_plus-4.0.3
+ /usr/bin/gzip -dc /usr/src/redhat/SOURCES/tac_plus-4.0.3.tgz
+ tar -xvvf -
drwxr-xr-x nsen/25           0 1999-08-04 00:33:15 tac_plus-
4.0.3/
-rw-r----- root/root      9029 1999-04-02 22:03:45 tac_plus-
4.0.3/CHANGES
...
...
...
...
...
...
Checking for unpackaged file(s): /usr/lib/rpm/check-files
/var/tmp/tacacsd
Wrote: /usr/src/redhat/RPMS/i386/tac_plus-4.0.3-2.i386.rpm
Executing(%clean): /bin/sh -e /var/tmp/rpm-tmp.60207
+ umask 022
+ cd /usr/src/redhat/BUILD
+ cd tac_plus-4.0.3
+ rm -rf /var/tmp/tacacsd
+ exit 0
Executing(--clean): /bin/sh -e /var/tmp/rpm-tmp.60207
+ umask 022
+ cd /usr/src/redhat/BUILD
+ rm -rf tac_plus-4.0.3
+ exit 0
[root@bigboy rpm]#
```

- The compiled RPM file can now be found in one of the architecture subdirectories under **/usr/src/redhat/RPMS** directory. For example, if you compiled an i386 architecture version of the RPM it will placed in the i386 subdirectory.
- You will then have to install the compiled RPMs found in their respective subdirectories as you normally would.

Older Linux Versions

The process is more complicated with older RedHat Linux versions as can be seen below.

- The source files are first exported into the directory **/usr/src/redhat/SPECS** with the **rpm** command.
- You then have to run the rpm command again to compile the source files into a regular RPM file which will be placed in either the **/usr/src/packages/RPMS/i386/** or the **/usr/src/redhat/RPMS/i386/** directories.
- You then have to install the new RPM file from this directory.

```
[root@bigboy tmp]# rpm -Uvh filename.src.rpm
[root@bigboy SPECS]# cd /usr/src/redhat/SPECS
[root@bigboy SPECS]# rpm –ba filename
[root@bigboy tmp]# cd /usr/src/redhat/RPM/i386
[root@bigboy i386]# rpm -Uvh filename.rpm
```

How To List Installed RPMs

- The **rpm –qa** command will list all the packages installed on your system

```
[root@bigboy tmp]# rpm –qa
perl-Storable-1.0.14-15
smpeg-gtv-0.4.4-9
e2fsprogs-1.27-9
libstdc++-3.2-7
audiofile-0.2.3-3
…
…
…
[root@bigboy tmp]#
```

- You can also pipe the output of this command through the **grep** command if you are interested in only a specific package. In this example we are looking for all packages containing the string “ssh” in the name, regardless of case (“-i” meaning ignore case)

```
[root@bigboy tmp]# rpm -qa | grep -i ssh
openssh-server-3.4p1-2
openssh-clients-3.4p1-2
openssh-askpass-gnome-3.4p1-2
openssh-3.4p1-2
openssh-askpass-3.4p1-2
[root@bigboy tmp]#
```

How To List All The Files Inside An RPM

Listing Files For Already Installed RPMs

- You can use the "**-ql**" qualifier to list all the files associated with an installed RPM. In this example we test to make sure that the NTP package is installed using the"**-qa**" qualifier, then we use the "**-ql**" qualifier to get the file listing.

```
[root@bigboy tmp]# rpm -qa ntp
ntp-4.1.2-0.rc1.2
cd /tmp
[root@bigboy tmp]# rpm -ql ntp
/etc/ntp
/etc/ntp.conf
/etc/ntp/drift
/etc/ntp/keys
…
…
…
/usr/share/doc/ntp-4.1.2/rdebug.htm
/usr/share/doc/ntp-4.1.2/refclock.htm
/usr/share/doc/ntp-4.1.2/release.htm
/usr/share/doc/ntp-4.1.2/tickadj.htm
[root@bigboy tmp]#
```

Listing Files In RPM Files

- You can use the "**-qpl**" qualifier to list all the files in a RPM file.

```
[root@bigboy updates]# rpm -qpl dhcp-3.0pl1-23.i386.rpm
/etc/rc.d/init.d/dhcpd
/etc/rc.d/init.d/dhcrelay
/etc/sysconfig/dhcpd
/etc/sysconfig/dhcrelay
…
…
…
/usr/share/man/man8/dhcrelay.8.gz
/var/lib/dhcp
/var/lib/dhcp/dhcpd.leases
[root@bigboy updates]#
```

How Uninstall RPMs

- The **rpm –e** command will erase an installed package. The package name given must match that listed in the **rpm –qa** command as the version of the package is important.

```
[root@bigboy tmp]# rpm –e package-name
```

Which RPMs Will Start Up At Boot Time?

The best way to view and configure this is by using the **chkconfig** command. A more detailed explanation will be provided in the chapter on the Linux boot process.

RedHat Up2date

RedHat has a program called **up2date** which will update your Linux installation with the latest revisions of the RPMs from the RedHat website via a HTTPS/SSL connection running in the background. Here's what to do:

- After installing the operating system issue the **up2date** command. It will prompt you to change the initial settings. Just quit by typing "**q**" and up2date will give you the command to run to get the encryption keys from RedHat.

```
[root@bigboy tmp]# up2date
0.  debug               No

1.  isatty              Yes

2.  depslist            []
…
…
…

Enter number of item to edit <return to exit, q to quit without
saving>:
Your GPG keyring does not contain the Red Hat, Inc. public key.
Without it, you will be unable to verify that packages Update
Agent downloads are securely signed by Red Hat.

Your Update Agent options specify that you want to use GPG.

To install the key, run the following as root:

    rpm --import /usr/share/rhn/RPM-GPG-KEY

[root@bigboy tmp]#
```

- Issue the **rpm** command to get the keys

```
[root@bigboy tmp]# rpm --import /usr/share/rhn/RPM-GPG-KEY
[root@bigboy tmp]#
```

- Issue the **up2date** command again and it will prompt you through a number of registration screens which will ask for information such as:

- The login name & password of your choice
- Your, name, address and email address
- A profile name for your server

- It will then present you with a list of all the packages installed on your server and ask you whether you want to register this software information with RedHat
- The **up2date** updater will then register your system and exit back to the command prompt.
- Now you have to actually update the software using **up2date**. This is done with the **up2date –u** command. This is what it looks like:

```
[root@bigboy tmp]# up2date -u
Fetching package list for channel: redhat-linux-i386-8.0...
########################################
Fetching Obsoletes list for channel: redhat-linux-i386-8.0...
########################################
Fetching rpm headers...
########################################
Testing package set / solving RPM inter-dependencies...
########################################
cups-libs-1.1.17-0.2.i386.r ##########################
Done.
…
…
…
Preparing           ##########################################
[100%]

Installing...
   1:cups-libs      ##########################################
[100%]
   2:cvs            ##########################################
[100%]
   3:cyrus-sasl     ##########################################
[100%]
   …
   …
   …

The following Packages were marked to be skipped by your
configuration:

Name                            Version        Rel  Reason
-----------------------------------------------------------------
kernel                          2.4.18         24.8.0Pkg
name/pattern

[root@bigboy tmp]#
```

Some Necessary Facts About up2date

- You can update your contact information afterwards using the RedHat Network link http://www.redhat.com/network
- RedHat will regularly send you emails with the packages you need to update. You can selectively update the package mentioned in each email using the command:

  ```
  [root@bigboy tmp]# up2date package-name
  ```

- Only one profile per login name is free. All additional profiles under the login name have an annual fee.
- **up2date** uses HTTPS/SSL to do its updating. If you have a firewall protecting your system, you will need TCP port 443 access to the internet
- Updating packages could cause programs written by you to stop functioning especially if they rely on the older version's features or syntax.
- Some RPMs won't install unless other RPMs have been installed previously. **up2date** automatically figures out these package inter-dependencies and will install all the required foundation packages as well.
- Sometimes you may get the following error when attempting to do up2date:

  ```
  There was an SSL error: [('SSL routines',
  'SSL3_GET_SERVER_CERTIFICATE', 'certificate verify failed')]
  A common cause of this error is the system time being incorrect.
  Verify that the time on this system is correct.
  ```

  A more probable cause is that the SSL keys used by your version of up2date have expired. You can get the most recent copy from the RedHat Network. If the rpm command complains about a Gnome dependency when doing the update you may have to use the "-i –force" options and then update the Gnome version of up2date immediately afterwards.

- You can write a small script to periodically update your system. The "**–u**" switch will update all packages and the **"-p**" will register any additional packages you have installed without using **up2date**. Here is a sample script that you can run weekly using **cron**

  ```
  #!/bin/sh
  #
  # Updates system every week
  #
  up2date -p
  up2date -u
  ```

Chapter 6

The Linux Boot Process

==

In This Chapter

==

The way Linux boots up is very important information to know. You can alter it to change the type of login screen you get and also which programs get started.

The RedHat Boot Sequence

When RedHat boots, the boot process will run a number of scripts located in subdirectories under directory **/etc/rc.d**. The boot process first runs the scripts found in **/etc/rc.d/rc1.d** which provides only the most basic functionality and the ability to only handle a single user. This stage is known as "single user mode". After completing this first phase, the boot process will run scripts in only one of the other directories depending on the startup mode (aka. run level). These are listed below.

Mode/Run Level	Directory	Run Level Description
0	/etc/rc.d/rc0.d	Halt
1	/etc/rc.d/rc1.d	Single-user mode
2	/etc/rc.d/rc2.d	Not used (user-definable)
3	/etc/rc.d/rc3.d	Full multi-user mode (No GUI interface)
4	/etc/rc.d/rc4.d	Not used (user-definable)
5	/etc/rc.d/rc5.d	Full multi-user mode (With GUI interface)
6	/etc/rc.d/rc6.d	Reboot

Determining The Default Boot runlevel

The default boot runlevel is set in the file **/etc/inittab** with the "initdefault" variable. When set it to “3”, the system boots up with the text interface on the VGA console; when set to “5”, you get the GUI. Here is a sample snippet of the file: (Delete the initdefault line you don't need)

```
# Default runlevel. The runlevels used by RHS are:
# 0 - halt (Do NOT set initdefault to this)
# 1 - Single user mode
# 2 - Multiuser, without NFS (The same as 3, if you do not have
networking)
# 3 - Full multiuser mode
# 4 - unused
# 5 - X11
# 6 - reboot (Do NOT set initdefault to this)
#
id:3:initdefault:                          # Console Text Mode
id:5:initdefault:                          # Console GUI Mode
```

- Most home users boot up with a Windows like GUI (Run Level 5)
- Most techies will tend to boot up with a plain text based command line type interface (Run level 3)
- Changing "**initdefault**" from 3 to 5 or vice-versa will only have an effect upon your next reboot. See the section below on how to get a GUI login all the time until the next reboot.

Get A GUI Console

You have two main options if your system comes up in a text terminal mode on the VGA console and you want to get the GUI:

- ***Manual Method:*** You can start the X terminal GUI application each time you need it by running the “**startx**” command at the VGA console. Remember that when you log out you will get the regular text based console again.

```
[root@bigboy tmp]# startx
```

- ***Automatic Method:*** You can have Linux automatically start the X terminal GUI console for every login attempt until your next reboot by using the **init** command. You will need to edit your “**initdefault**” variable in your **/etc/inittab** file as mentioned in the preceding section to keep this functionality even after you reboot.

```
[root@bigboy tmp]# init 5
```

Get A Basic Text Terminal Without Exiting The GUI

Using A GUI Terminal Window

You can open a GUI based window with a command prompt inside by doing the following:

- Click on the “Red Hat” Start button in the bottom left hand corner of the screen.
- Click on Systems Tools, then Terminal

Using Virtual Terminals

Linux actually has seven virtual console sessions running on the VGA console.

- Sessions one through six are text sessions. If the GUI is running, it will run under session number seven.
- You can step through each text session by using the <CTL> <ALT> <F1> through <F6> key sequence. You'll get a new login prompt for each attempt.
- You can get the GUI login with the sequence <CTL> <ALT> <F7>, only in run level 5, or if the GUI is running after launching "**startx**"

System Shutdown And Rebooting

The "**init**" command will allow you to change the current runlevel.

Halt / Shutdown The System

```
[root@bigboy tmp]# init 0
```

Reboot The System

```
[root@bigboy tmp]# init 6
```

How To Set Which Programs Run At Each runlevel

Most RedHat packages place a startup script in the directory **/etc/init.d** and place symbolic links (pointers) to this script in the appropriate **/etc/rc.d/rc.X** directory. The typical home/SOHO user doesn't have to be a scripting / symbolic linking guru to make sure everything works right because RedHat comes with a nifty utility called "**chkconfig**" to do it for you.

- Use this command to get a full listing of packages listed in **/etc/init.d** and the runlevels at which they will be "on" or "off"

```
[root@bigboy tmp]# chkconfig --list
keytable 0:off 1:on  2:on  3:on 4:on  5:on 6:off
atd      0:off 1:off 2:off 3:on 4:on  5:on 6:off
syslog   0:off 1:off 2:on  3:on 4:on  5:on 6:off
gpm      0:off 1:off 2:on  3:on 4:on  5:on 6:off
kudzu    0:off 1:off 2:off 3:on 4:on  5:on 6:off
wlan     0:off 1:off 2:on  3:on 4:on  5:on 6:off
sendmail 0:off 1:off 2:off 3:on 4:off 5:on 6:off
netfs    0:off 1:off 2:off 3:on 4:on  5:on 6:off
network  0:off 1:off 2:on  3:on 4:on  5:on 6:off
random   0:off 1:off 2:on  3:on 4:on  5:on 6:off
...
...
```

Chkconfig Examples

You can use **chkconfig** to change runlevels for particular packages. Here we see Sendmail will start with a regular startup at runlevel 3 or 5. Let's change it so that Sendmail doesn't startup at boot.

Use Chkconfig To Get A Listing Of Sendmail's Current Startup Options

```
[root@bigboy tmp]# chkconfig --list | grep mail
sendmail 0:off 1:off 2:off 3:on 4:off 5:on 6:off
[root@bigboy tmp]#
```

Switch Off Sendmail Starting Up In Levels 3 and 5

```
[root@bigboy tmp]# chkconfig --level 35 sendmail off
[root@bigboy tmp]#
```

Doublecheck That Sendmail Will Not Startup

```
[root@bigboy tmp]# chkconfig --list | grep mail
sendmail 0:off 1:off 2:off 3:off 4:off 5:off 6:off
[root@bigboy tmp]#
```

Turn it back on again

```
[root@bigboy tmp]# chkconfig --level 35 mail on
[root@bigboy tmp]# chkconfig --list | grep mail
sendmail 0:off 1:off 2:off 3:on 4:off 5:on 6:off
[root@bigboy tmp]#
```

Final Tips On chkconfig

- In most cases you'll want to modify runlevels 3 and 5 simultaneously AND with the same values.
- Don't add/remove anything to other runlevels unless you absolutely know what you are doing. Don't experiment.
- Chkconfig doesn't start the programs in the **/etc/init.d** directory, it just configures them to be started or ignored when the system boots up. The commands for starting and stopping the programs covered in this book are covered in each respective chapter.

Chapter 7

Configuring The DHCP Server

= =

In This Chapter

= =

Normally if you have a cable modem or DSL you get your home PC's IP address dynamically assigned from your service provider. If you install a home cable/DSL router between your modem and home network your PC will most likely get its IP address at boot time from the home router instead. You can choose to disable the DHCP server feature on your home router and set up a Linux box as the DHCP server.

This chapter only covers the configuration of a DHCP server that provides IP addresses. The configuration of a Linux DHCP client that gets its IP address from a DHCP server is covered in the Linux Networking Topics chapter.

Download and Install The DHCP Package

Most RedHat Linux software products are available in the RPM format. Downloading and installing RPMs isn't hard. If you need a refresher, the RPM chapter covers how to do this in detail.

- For example, the RedHat 9.0 RPM as of this writing was:

```
dhcp-3.0pl1-23.i386.rpm
```

- Install the package using the following command:

```
[root@bigboy tmp]# rpm -Uvh dhcp-3.0pl1-20.i386.rpm
```

The /etc/dhcp.conf File

When DHCP starts it reads the **file /etc/dhcp.conf**. It uses the commands here to configure your network. Normally you can find a sample copy of dhcpd.conf in the following directory which you can always use as a guide.

```
/usr/share/doc/dhcp-<version-number>/dhcpd.conf.sample
```

Here is a quick explanation of the **dhcp.conf** file: Most importantly, there ***must*** be a "subnet" section for each interface on your Linux box.

```
ddns-update-style interim # Redhat Version 8.0+

subnet 192.168.1.0 netmask 255.255.255.0 {

   # The range of IP addresses the server
   # will issue to DHCP enabled PC clients
   # booting up on the network

   range 192.168.1.201 192.168.1.220;

   # Set the amount of time in seconds that
   # a client may keep the IP address

   default-lease-time 86400;
   max-lease-time 86400;

   # Set the default gateway to be used by
   # the PC clients

   option routers 192.168.1.1;

   # Don't forward DHCP requests from this
   # NIC interface to any other NIC
   # interfaces

   option ip-forwarding off;

   # Set the broadcast address and subnet mask
   # to be used by the DHCP clients

   option broadcast-address 192.168.1.255;
   option subnet-mask 255.255.255.0;

   # Set the DNS server to be used by the
   # DHCP clients
```

```
    option domain-name-servers 192.168.1.100;

    # Set the NTP server to be used by the
    # DHCP clients

    option nntp-server 192.168.1.100;

    # If you specify a WINS server for your Windows clients,
    # you need to include the following option in the dhcpd.conf file:

    option netbios-name-servers 192.168.1.100;

}
#
# List an unused interface here
#
subnet 192.168.2.0 netmask 255.255.255.0 {
}

# You can also assign specific IP addresses based on the clients'
# ethernet MAC address as follows (Host's name is "smallfry":

host smallfry {
    hardware ethernet 08:00:2b:4c:59:23;
    fixed-address 192.168.1.222;
}
```

There many more options statements you can use to configure DHCP. These include telling the DHCP clients where to go for services such as finger and IRC. Check the dhcp-options man page after you do your install. The command to do this follows:

```
[root@bigboy tmp]# man dhcp-options
```

Upgrading Your DHCP Server

Always refer to this sample file after doing an upgrade as new required commands may have been added. For example, in Redhat Version 8.0 (dhcpd version 3.0b2pl11) you will need to add the line at the very top of the config file or else you will get errors:

```
ddns-update-style interim
```

How to get DHCP started

- Before you start the DHCP server for the first time, it will fail unless there is an existing dhcpd.leases file. Use the command "**touch /var/lib/dhcp/dhcpd.leases**" to create the file if it does not exist.

```
[root@bigboy tmp]# touch /var/lib/dhcp/dhcpd.leases
```

- Use the chkconfig command to get DHCP configured to start at boot:

```
[root@bigboy tmp]# chkconfig --level 35 dhcpd on
```

- Use the **/etc/init.d/dhcpd** script to start/stop/restart DHCP after booting

```
[root@bigboy tmp]# /etc/init.d/dhcpd start
[root@bigboy tmp]# /etc/init.d/dhcpd stop
[root@bigboy tmp]# /etc/init.d/dhcpd restart
```

- Remember to restart the DHCP process every time you make a change to the conf file for the changes to take effect on the running process. You also can test whether the DHCP process is running with the following command, you should get a response of plain old process ID numbers:

```
[root@bigboy tmp]# pgrep dhcpd
```

- Finally, always remember to set your PC to get its IP address via DHCP.

Modify Your Routes for DHCP on Linux Server

When a DHCP configured PC boots, it will request its IP address from the DHCP server. It does this by sending a standardized DHCP broadcast request packet to the DHCP server with a source IP address of 255.255.255.255.

You will have to add a route for this address on your Linux DHCP server so that it knows the interface on which to send the reply, if not, it sends it to the default gateway. (In both examples below, we're assuming that DHCP requests will be coming in on interface eth0).

Note: More information on adding Linux routes and routing may be found in the Linux Networking chapter.

Note: You can't run your DHCP sever on multiple interfaces as you can only have one route to network 255.255.255.255. If you try to do it, you'll discover that DHCP serving working on only one interface.

Temporary solution

- Add the route to 255.255.255.255 from the command line.

```
[root@bigboy tmp]# route add -host 255.255.255.255 dev eth0
```

- If the message **255.255.255.255: Unknown host** appears then try adding the following entry to your /etc/hosts file:

```
255.255.255.255 dhcp
```

Then, try:

```
route add -host dhcp dev eth0
```

Permanent Solution

- Edit **/etc/sysconfig/static-routes**, which is read on booting, and add the following line:

```
eth0 host 255.255.255.255
```

- If this doesn't work properly try adding the following entry to your **/etc/hosts** file:

  ```
  255.255.255.255 dhcp
  ```

  Then, try:

  ```
  eth0 host dhcp
  ```

Configuring Linux clients to use DHCP

Remember to have your Linux based clients configured to have DHCP obtained IP addresses

Error Found When Upgrading From Redhat 7.3 To 8.0

This dhcpd startup error is caused by not having the following line at the very top of your /etc/dhcpd.conf file:

```
ddns-update-style interim
```

Sample error:

```
Starting dhcpd: Internet Software Consortium DHCP Server V3.0pl1
Copyright 1995-2001 Internet Software Consortium.
All rights reserved.
For info, please visit http://www.isc.org/products/DHCP

** You must add a ddns-update-style statement to /etc/dhcpd.conf.
To get the same behaviour as in 3.0b2pl11 and previous
versions, add a line that says "ddns-update-style ad-hoc;"
Please read the dhcpd.conf manual page for more information. **

If you did not get this software from ftp.isc.org, please
get the latest from ftp.isc.org and install that before
requesting help.

If you did get this software from ftp.isc.org and have not
yet read the README, please read it before requesting help.
If you intend to request help from the dhcp-server@isc.org
mailing list, please read the section on the README about
submitting bug reports and requests for help.

Please do not under any circumstances send requests for
help directly to the authors of this software - please
send them to the appropriate mailing list as described in
the README file.

exiting.
[FAILED]
```

Chapter 8

Adding Linux Users

= =

In This Chapter

= =

One of the most important activities in administering a Linux box is the addition of users. I have included some simple examples to provide a foundation for future chapters. A more detailed description of the process is beyond the focus of this book. You may use the command "**man useradd**" to get the help pages on adding users with the **useradd** command or the "**man usermod**" to become more familiar with modifying users with the **usermod** command.

Who Is The Super User?

The super user with unrestricted access to all system resources and files is the user named "**root**". You will need to log in as user **root** to add new users to your Linux box

How To Add Users

Adding users takes some planning, read through the steps below before starting:

- Arrange your list of users into groups by function. In this example there are three groups "parents", "children" and "soho".

```
Parents     Children     Soho

Paul        Alice        Accounts
Jane        Derek        Sales
```

- Add the Linux groups to your server:

```
[root@bigboy tmp]# groupadd parents
[root@bigboy tmp]# groupadd children
[root@bigboy tmp]# groupadd soho
```

- Add the Linux users, assign them to their respective groups

```
[root@bigboy tmp]# useradd -g parents  paul
[root@bigboy tmp]# useradd -g parents  jane
[root@bigboy tmp]# useradd -g children derek
[root@bigboy tmp]# useradd -g children alice
[root@bigboy tmp]# useradd -g soho     accounts
[root@bigboy tmp]# useradd -g soho     sales
```

If you don't specify the group with the "-g", RedHat Linux will create a group with the same name as the user you just created. When each new user first logs in, they will be prompted for their new permanent password.

- Each user's personal directory will be placed in the /**home** directory. The directory name will be the same as their user name.

```
[root@bigboy tmp]# ll /home
drwxr-xr-x    2 root     root        12288 Jul 24 20:04 lost+found
drwx------    2 accounts soho         1024 Jul 24 20:33 accounts
drwx------    2 alice    children     1024 Jul 24 20:33 alice
drwx------    2 derek    children     1024 Jul 24 20:33 derek
drwx------    2 jane     parents      1024 Jul 24 20:33 jane
drwx------    2 paul     parents      1024 Jul 24 20:33 paul
drwx------    2 sales    soho         1024 Jul 24 20:33 sales
[root@bigboy tmp]# ll /home
```

How To Change Passwords

You'll need to create passwords for each account. This is done with the "**passwd**" command. You will be prompted once for your old password and twice for the new one.

- User "**root**" changing the password for user "**paul**"

```
[root@bigboy root]# passwd paul
Changing password for user paul.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
[root@bigboy root]#
```

- Users may wish to change their passwords at a future date. Here is how unprivileged user "**paul**" would change his own password.

```
[paul@bigboy paul]$ passwd
```

```
Changing password for paul
Old password: your current password
Enter the new password (minimum of 5, maximum of 8 characters)
Please use a combination of upper and lower case letters and
numbers.
New password: your new password
Re-enter new password: your new password
Πασσωορδ χηανγεδ.
[παυλ≅βιγβοψ παυλ]∃
```

How to Delete User

- The **userdel** command is used. The "-r" flag removes all the contents of the user's home directory

```
[root@bigboy tmp]# userdel -r paul
```

How To Tell The Groups To Which A User Belongs

- Use the "groups" command with the username as the argument

```
[root@bigboy root]# groups paul
paul : parents
[root@bigboy root]#
```

Chapter 9

Windows, Linux And Samba

= =

In This Chapter

= =

Samba is a suite of utilities that allows your Linux box to share files and other resources such as printers with Windows boxes. This chapter describes how you can make your Linux box into a Windows Primary Domain Controller (PDC) or a server for a Windows Workgroup. Either configuration will allow everyone at home to have:

- their own logins on all the home windows boxes while having their files on the Linux box appear to be located on a new Windows drive
- shared access to printers on the Linux box
- shared files accessible only to members of their Linux user group.

What's the difference between a PDC and Windows Workgroup member? A detailed description is beyond the scope of this chapter, but this simple explanation should be enough.

- A PDC stores the login information in a central database on its hard drive. This allows each user to have a universal username and password when logging in from all PCs on the network.
- In a Windows Workgroup, each PC stores the usernames and passwords locally so that they are unique for each PC.

This chapter will only cover the much more popular PDC methodology used at home. By default, Samba mimics a Windows PDC in almost every way needed for simple file sharing. Linux functionality doesn't disappear when you do this. Samba Domains and Linux share the same usernames so you can log into the Samba based Windows domain using your and immediately

gain access to files in your Linux user's home directory. For added security you can make your Samba and Linux passwords different.

When Samba starts up it reads the configuration file **/etc/samba/smb.conf** to determine its various modes of operation. You can create your own **smb.conf** using a text editor or using the easier web based SWAT utility. Keep in mind that you will lose all your comments inserted in **/etc/samba/smb.conf** with a text editor if you subsequently use SWAT to edit it. Explanations of how to use both SWAT and a text editor to configure Samba are given in this chapter.

Download and Install Packages

Samba is comprised of a suite of RPMs that come on the RedHat CDs. As of this writing, the latest version of the Samba suite for RedHat 9.0 was version 2.2.7a-8. Install all the packages in this order:

```
[root@bigboy tmp]# rpm -Uvh samba-2.2.7a-8.9.0.i386.rpm
[root@bigboy tmp]# rpm -Uvh samba-common-2.2.7a-8.9.0.i386.rpm
[root@bigboy tmp]# rpm -Uvh samba-client-2.2.7a-8.9.0.i386.rpm
[root@bigboy tmp]# rpm -Uvh samba-swat-2.2.7a-8.9.0.i386.rpm
```

Downloading and installing RPMs isn't hard. If you need a refresher, the RPM chapter covers how to do this in detail.

How To Get SAMBA Started

- You can configure Samba to start at boot time using the chkconfig command:

```
[root@bigboy tmp]# chkconfig --level 35 smb on
```

- You can start/stop/restart Samba after boot time using the smb initialization script as in the examples below:

```
[root@bigboy tmp]# /etc/init.d/smb start
[root@bigboy tmp]# /etc/init.d/smb stop
[root@bigboy tmp]# /etc/init.d/smb restart
```

- Remember to restart the smb process every time you make a change to the conf file for the changes to take effect on the running process.
- You can test whether the smb process is running with the following command, you should get a response of plain old process ID numbers:

```
[root@bigboy tmp]# pgrep smb
```

Configuring SWAT

SWAT is a very intuitive web based Samba configuration tool that allows you to configure Samba without all the memorization of the keywords needed for text based configuration. Unfortunately it doesn't encrypt your login password. This may be a security concern in a corporate environment. Because of this, you may want to create a Samba administrator user that has no root privileges whatsoever or do your configuration using the configuration files.

The enabling/disabling, starting and stopping of SWAT is controlled by xinetd via a configuration file named **/etc/xinetd.d/swat**. Here is a sample.

```
service swat
{
   port            = 901
   socket_type     = stream
   protocol        = tcp
   wait            = no
   user            = root
   server          = /usr/sbin/swat
   log_on_failure  += USERID
   disable         = no
   only_from       = localhost
}
```

The formatting of the file is fairly easy to understand, especially as there are only two entries of interest.

- The disable parameter must be set to "no" to accept connections.
- By default, you can only log into SWAT from the VGA console as user "root". The URL must point to your localhost IP address on port 901 (http://127.0.0.0:901) as defined by the **only_from** and **port** parameters.

You can make SWAT accessible from other servers by adding IP address entries separated to the **only_from** parameter. Here's an example of an entry to allow connections only from 192.168.1.3 and localhost:

```
only_from = localhost, 192.168.1.3
```

Therefore in this case you can also configure Samba on your Linux server "Bigboy" IP with address 192.168.1.100 from PC 192.168.1.3 using the URL http://192.168.1.100:901

Samba and PC Firewall Software

Firewall software installed on Windows PCs may cause Samba to not function. Here are some ways to solve the problem with two popular packages.

Zone Alarm

The default installation of Zone Alarm assumes that your PC is directly connected to the Internet. This means that the software will deny all inbound connections that attempt to connect with your PC. The NetBIOS traffic that Samba uses to communicate with the PCs on the network will therefore be considered as hostile traffic. The easiest way around this is to configure Zone Alarm to consider your home network as a trusted network too.

This can be done by clicking on the **firewall** tab and editing the settings for your home network that will most likely have a 192.168.x.x/255.255.255.0 type entry. Make this network a trusted network, instead of an Internet network and ZoneAlarm should cease to interfere with Samba.

The Windows XP Built In Firewall

You may also need to disable the firewall feature of Windows XP by doing the following:

- Bring up Control Panel
- Go through the Network and Internet Connections and then Network Connections menus.
- Right click your on your LAN connection icon and select **Properties**
- Click on the **Advanced** tab.
- Uncheck the **Internet Connection Firewall** box and it will be turned on.

Once you get SAMBA to work, you may want to experiment with the firewall software settings to optimize your security with the need to maintain a valid relationship with the SAMBA server

How To Create A Samba PDC Administrator User

To do both SWAT and user administration with Samba you'll need to create an Administrator account on the Samba PDC Linux box. Here is how it's done:

Create The Administrator's User Group and Directories

- First create a Linux group for administrators:

```
[root@bigboy tmp]# /usr/sbin/groupadd sysadmin
```

- Then create a Linux directory to house all the administrator directories:

```
[root@bigboy tmp]# mkdir /home/sysadmin
[root@bigboy tmp]# chgrp sysadmin /home/sysadmin
[root@bigboy tmp]# chmod 0770 /home/sysadmin
```

Create The Administrator User Under Linux

- For each administrator user, create a Linux user with the **adduser** command:

```
[root@bigboy tmp]# /usr/sbin/adduser  -d \
       /home/sysadmin/administrator \
       -g sysadmin -m -k /etc/skel.smb  -n administrator
```

- As this user may not need a real Linux login, we won't assign a real Linux password. This provides an added level of security, especially if you're using SWAT. The table below explains what each of the adduser command switches used.

Adduser's Command Switches

useradd command switch	Description
`-g group`	Sets the group to which the user should be added. In this case the administrator user has been made a member of the sysadmin group.
`-m`	Forces linux to create the directory specified with the -d switch
-d dir_path	Home directory for the new user
-k */etc/skel.smb*	Tells adduser to copy the contents of the directory /etc/skel.smb to the users new home directory. Usually default login scripts.
-n	Tells RedHat NOT to create a default group with the same name as the user. (Redundant in this case)

Create An Administrator Domain Password

The Linux Administrator now needs a Samba password to log into the Windows domain. This is done with the **smbpasswd** command.

```
[root@bigboy tmp]# /usr/bin/smbpasswd -a administrator password
```

The -a adds the user administrator to the **/etc/smbpasswd** file. Use a generic password then have users change it immediately from their workstation the usual way.

Note: If you want user "administrator" to be able to log into the Linux box as a regular user via Telnet or SSH, then you'll have to use the Linux **passwd** command to give this user a Linux (not a Samba domain) password. Samba domain logins use the **smbpasswd** password.

Make The Administrator One Of The Samba Admin Users

Edit **/etc/samba/smb.conf** file and add the sysadmin group to the list of Samba system administrator users.

```
[global]
admin users = @sysadmin
```

This can also be set via SWAT in the expanded “global settings” section. You’ll need to restart Samba for this to take effect.

How to Configure a Samba PDC

Create A Samba PDC

By far the easiest way to configure a Samba PDC is by using SWAT. Log into SWAT and click on the “globals” section and make sure the key highlighted parameters below are set correctly.

```
# Samba config file created using SWAT
# from 192.168.1.101 (192.168.1.101)
# Date: 2002/11/10 19:54:45

# Global parameters
[global]

##
## The name I want to give my DOMAIN
##
     workgroup = HOMENET
```

```
##
## The name of my Linux Box
##
     server string = BigBoy

##
## Make sure passwords are always encrypted
##
     encrypt passwords = Yes

##
## Samba will log errors via syslog to
## directory /var/log/samba
##
     log file = /var/log/samba/%m.log
     max log size = 0

##
## The command Samba uses to add new printers
## directory /var/log/samba
##
     addprinter command = /usr/bin/addprinter

##
## Only domain users in the sysadmin group can act as
## universal administrators. Ie administer the PDC
## and do activities on the local PC like
## software installation. You can also have local
## administrator accounts on PCs to do software
## installation, but they wont be PDC administrators
## to do things like adding new PCs to the domain
##
     domain admin group = @sysadmin

##
## The script logon.cmd in the Linux user's home directory
## will be run whenever the user logs in. (Usually blank)
##
     logon script = logon.cmd

##
## My user's Linux login directory will always appear
## to be Windows drive H:
##
     logon drive = h:

##
## Activate domain type logins and make
## Linux box be the master Windows server
## in all domain type functions
##
     domain logons = Yes
     os level = 99
     preferred master = True
     domain master = True
     local master = True
```

```
        dns proxy = No

##
## Only users in the sysadmin group can use SWAT
## to modify Samba and
##
admin users = @sysadmin
        printer admin = @sysadmin
        printing = lprng
```

Next you will have to use SWAT to click on the "**shares**" button. Here you will use the drop down menu to edit the **netlogon** and **profiles** shares

```
[netlogon]
##
## Store all Samba PDC overhead data in the directory
## /home/netlogon (or whatever you desire)
##
path = /home/netlogon
        write list = administrator
        guest ok = Yes

[profiles]
##
## Store user profiles in this directory
##
        path = /home/ntprofile
        read only = No
        create mask = 0600
        directory mask = 0700
        guest ok = Yes
        browseable = No
```

Click on the "**Status"** button at the top of the screen and restart Samba to make your settings take effect.

Create Your PC Machine Trusts

PDCs will only accept user logins from trusted PCs that have been placed in its PC client database. Samba can create these "Machine Trusts" in two ways, either manually or automatically.

Manual Creation Of Machine Trust Accounts (NT Only)

When manually creating a machine trust account you need to manually create the corresponding Unix account in **/etc/passwd** and **/etc/smbpasswd** files. Pay careful attention to the **"$"** at the end and replace **machine_name** with the name of the Windows client machine.

```
[root@bigboy tmp]# /usr/sbin/useradd -g 100 -d /dev/null -c
"machine nickname" -s /bin/false machine_name$
```

```
[root@bigboy tmp]# passwd -l machine_name$
[root@bigboy tmp]# smbpasswd -a -m machine_name
```

This is the only way to configure machine trusts using Windows NT.

Dynamic Creation Of Machine Trust Accounts

The second (and recommended) way of creating machine trust accounts is simply to allow the Samba server to create them as needed when the client joins the domain. This can be done by editing **/etc/samba/smb.conf** to automatically add the required users. This method is also referred to as making a machine account "on the fly".

```
[global]
# <...remainder of parameters...>
add user script = /usr/sbin/useradd -d /dev/null -g 100 -s
/bin/false -M %u
```

This is probably easier to do if you use SWAT in the “Global” menu

Make Your PC Clients Aware Of Your Samba PDC

When you log into the domain, the PDC will send your new client PC a list of universal “look and feel” related features that may have been previously set by the Administrator. These include things like suppressing the splash screen, defining your wall paper and setting the way dates are formatted. This information is stored in the netlogon directory. How to configure the netlogon directory is beyond the scope of this chapter, but suffice it to say that the directory must be created on the Linux box for Samba to operate correctly.

```
[root@bigboy tmp]# mkdir /home/netlogon
[root@bigboy tmp]# chmod 0755 /home/netlogon
```

You’ll then have to log into each PC client and do the following steps depending on their operating system.

Windows 95/98/ME

Windows 9x machines do not implement full domain membership and therefore don't require machine trust accounts

- Navigate to the Network section of the Control Panel (Start ->Settings->Control Panel->Network)
- Select the Configuration tab
- Highlight "Client for Microsoft Networks"
- Click the Properties button.
- Check "Log onto Windows NT Domain", and enter the domain name.
- Click all the OK buttons and reboot!

Windows NT

Create a manual SAMBA machine trust account as explained above, then go through the following steps:

- Navigate to the Network section of the Control Panel (Start ->Settings->Control Panel->Network)
- Select the “**identification**” tab
- Click the "**change**" button
- Enter the domain name and computer name, **do not** check the box "Create a Computer Account in the Domain." In this case, the existing machine trust account is used to join the machine to the domain.
- Click "**OK**"
- You should then get a confirmation that you’ve been added with a "Welcome to <DOMAIN>" message.
- Reboot.
- Log in using any account in **the /etc/smbpasswd** file with your domain as the domain name.

Windows 2000

Create a dynamic SAMBA machine trust account as explained above, then go through the following steps:

- Press Windows-key Break-key simultaneously to get the System Properties dialogue box
- Click on the '**Network Indentification**' tab on the top
- Click the "**Properties**" button
- Click on the “**Member of Domain**” button
- Also enter your domain name and computer name and then click "OK"
- You will now be prompted for a user account and password with rights to join a machine to the domain. This should be your Samba administrator. In our case that would be user "administrator" with the corresponding smbpasswd password. (See this note before proceeding.)
- You should then get a confirmation that you’ve been added with a "Welcome to <DOMAIN>" message.
- Reboot.
- Log in using any account in the **/etc/smbpasswd** file with your domain as the domain name.

Windows XP

Create a dynamic SAMBA machine trust account as explained above, then go through the following steps:

- Press Windows-key Break-key simultaneously to get the System Properties dialogue box
- Click on the '**Computer Name**' tab on the top
- Click on the '**Change**' button
- Click on the "**Member of Domain**" button
- Also enter your domain name and computer name and then click "OK"
- You will now be prompted for a user account and password with rights to join a machine to the domain. This should be your Samba administrator. In our case that would be user "administrator" with the corresponding smbpasswd password.
- Reboot.
- Log in using any account in the **/etc/smbpasswd** file with your domain as the domain name.

How To Add Users To Your Samba Domain

Add The Users In Linux

First go through the process of adding users in Linux just like you would normally do. Passwords won't be necessary unless you want the users to log in to the Samba server via Telnet or SSH.

Map The Linux Users To An smbpassword

Next you need to create Samba domain login passwords for all users

```
[root@bigboy tmp]# /usr/bin/smbpasswd -a username password
```

The -a adds the user to **the /etc/smbpasswd** file. Use a generic password then have users change it immediately from their workstation the usual way.

Map A Drive Share

By default, Samba will automatically give each user logged into the domain an H: drive that really maps to the **/home/username** directory on the Linux box.

Mapping Using "My Computer"

If the auto-mapping doesn't work then do the following:

- Let the user log into the domain
- Right click on the "My Computer" icon on the desktop
- Click on "Map Network Drive"
- Select a drive letter
- Browse to the HOMENET domain, then the Samba server, then the user's home directory.
- Click on the check box "Reconnect at Logon", to make the change permanent

Mapping From The Command Line

If you find the "My Computer" method too time consuming for dozens of users or if the PC doesn't have the feature available, then you can do the following and possibly make it into a script.

- Create a master logon batch file for administrator users. We will add the contents of this file to all administrator's logon scripts.

```
[root@bigboy tmp]# vi /home/netlogon/administrator.bat
```

- Add the following contents to mount the user's share as drive P: (for 'private').

```
REM administrator
net use P: \\bigboy\administrator
```

- Make the file world readable:

```
[root@bigboy tmp]# chmod +r
/home/samba/netlogon/administrator.bat
```

- Linux and Windows format text files slightly differently. As the file resides on a Linux box, but will be interpreted bay a Windows machine, you'll have to convert the file to the Windows format. This requires the '**todos**' program to be installed. (You can get this package from http://speakeasy.rpmfind.net)

```
[root@bigboy tmp]# todos
/home/samba/netlogon/administrator.bat
```

- The next page will show you how to add regular users to your new SAMBA domain

Domain Groups And Samba

Samba supports domain groups which will allow users who are members of the group to be able to have Administrator rights on each PC in the domain. This will allow them to do such things as add software and configure network settings. In Windows, Domain Groups also have the ability to join machines to the domain, but this is not currently supported in Samba.

The domain admin group parameter specifies users who will have domain admin rights. The argument is a space-separated list of user names or group names (group names must have an @ sign prefixed). For example:

```
domain admin group = <USER1> <USER2> @<GROUP>
```

How To Delete Users From Your Samba Domain

Delete The Users In Linux

First go through the process of deleting users in Linux just like you would normally do. Here we are deleting the user zmeekins and all zmeekin's files from the Linux server:

```
[root@bigboy tmp]# userdel -r zmeekins
```

Delete The Users Using smbpasswd

Next, use the smbpasswd command with the "-x" switch

```
[root@bigboy tmp]# smbpasswd -x zmeekins
Deleted user zmeekins.
[root@bigboy root]#
```

Chapter 10

Sharing Resources With Samba

==

In This Chapter

==

Now that you have Samba up and running, you may want to allow users to share resources such as floppy drives, directories and printers via the Samba server. Here's how to do it all.

Adding A Printer To A Samba PDC

Sharing printers amongst all your PCs is one of the advantages of creating a home network. Here's how to connect your printer directly to your PDC and make it available to all your windows workstations. This makes your Samba PDC a print server too!

The method explained here requires the Windows printer driver to be loaded on every client machine. This may be OK for a small home network but may be impractical for a huge corporate network.

Adding The Printer To Linux

By far the easiest way to add a printer in Linux is to use the GUI from the VGA console. The way to get a GUI on the console is outlined in the runlevels chapter. We'll assume the printer is locally attached to the parallel port. Here are the menus to use:

- Click on the RedHat icon in the bottom left hand corner of the screen
- Click on System Settings
- Click on Printing
- Click on New
- You'll now get the "Add a New Print Queue" menu
  - Click "forward"
- You'll get the "Set the Print Queue name and Type" menu
  - Give the printer an easy to remember name. (My printer is an Epson Stylus C60, I called the printer queue EpsonC60)

- Click the local printer button
- Click "forward"

- You'll get the "Configure a Local Printer" menu
  - Select /dev/lp0 as I assume the printer is on the parallel port (not USB)
- You'll get the "Select Print Driver" menu
  - Scroll to the printer
  - Double click on the name
  - Select the driver
  - Click "forward"
- You'll get the "Finish and Create the New Print Queue" menu
  - Click finished
- Click "Apply"
  - Do a test print to make sure all is OK

Make Samba Aware Of The Printer

The easiest way to do this is using the Samba SWAT web interface. Once you are in SWAT:

- Select the "Printers" button
- Find your printer in the drag down menu
- If the printer name has a [*] beside it, then it has been auto configured by Samba, but may not be visible on your network because Samba hasn't been restarted since creating the printer. If this is the case, restart Samba and go to the next section.
- If this isn't the case, edit/create the printer
- Click on the "**Commit Changes**" button to create an updated **/etc/samba/smb.conf** file.
- Click on the "**Status**" tab at the top of the screen and restart **smbd** and **nmbd** to restart Samba.

Configure The Printer Driver On The Workstations

- Download the Windows printer driver from the manufacturer and install it.
- Go to the Add printer menu
  - Click the "Next" button
- Select the "Network Printer" button to get the "Local or Network Printer" menu
  - Click the "Next" button
- You should be on the "Locate Your Printer" menu
  - Don't enter a name, Click next so you can browse for your printer

- You should be on the "Browse for Printer" menu
  - Double Click on the name of your Linux Samba Box
  - You should see the new printer
  - Click on the printer name
  - Click the "Next" button
- You may get a message stating "The server on which the printer resides does not have the correct printer driver installed. If you want to install the driver on your local computer, click OK". Fortunately, you pre installed the driver
  - Click the "OK" button
- The "Add Printer Wizard" will appear
  - Select the manufacturer of your printer
  - Select the printer model
  - Click the "OK" button
- The "Add Printer Wizard" will prompt you whether you want to use this new printer as the default printer.
  - Select "Yes" or "No" depending on your preference
  - Click the "Next" button
- The "Completing the Add Printer Wizard" menu will appear
  - Click the "Finish" button
- The new printer should now show up on the Windows Printers menu in "Control Panel"
- Send a test print.

Creating Group Shares in SAMBA

On occasion, subgroups of a family need a share that is fully accessible by all members of the group. For example, parents working in a home office environment may need a place where they can share, distribute or collaboratively work on documents. Here's how it's done.

Create The Directory And User Group

- Create a new Linux group parents:

```
[root@bigboy tmp]# /usr/sbin/groupadd parents
```

- Create a new directory for the group's files. If one user is designated as the leader, you might want to change the chown statement to make them owner

```
[root@bigboy tmp]# mkdir /home/parent-files
[root@bigboy tmp]# chown parents /home/parent-files
[root@bigboy tmp]# chmod 0770 /home/parent-files
```

- Next we add the group members to the new group. For instance, let's add user "father" to the group.

```
[root@bigboy tmp]# /usr/sbin/usermod -G parents father
```

Configure The Share In SWAT

- Finally, create the share in Samba using SWAT.
- Click on the **shares** button then enter the name of the share you want to create, let's say "**only-parents**".
- Click on the "Create Share" button. Make sure the path maps to "**/home/parent-files**" and make the valid users be **@parents**, where parents is the name of the Linux user group.
- Click on the "**Commit Changes**" button to create a new **/etc/samba/smb.conf** file.
- Click on the "**Status**" tab at the top of the screen and restart **smbd** and **nmbd** to restart Samba.
- Your **/etc/samba/smb.conf** file should have an entry like this at the end:

```
# Parents Shared Area
[only-parents]
        path = /home/parent-files
        valid users = @parents
```

Map The Directory Using "My Computer"

- Let the user log into the domain from a remote PC
- Right click on the "My Computer" icon on the desktop
- Click on "Map Network Drive"
- Select a drive letter
- Browse to the HOMENET domain, then the Samba server, then the share named **only-parents**
- Click on the check box "Reconnect at Logon", to make the change permanent

Windows Drive Sharing With Your SAMBA Server

You can also access a CD, DVD, ZIP, floppy or hard drive installed on a Windows Client from the Samba server. In this section we'll attempt to share a ZIP drive.

Windows Setup

The Windows client box should first be setup as a member of a Samba domain or workgroup. The next step is to make the ZIP drive shared.

Windows 98/ME

- Double click 'My Computer'
- Right click on the ZIP drive and choose 'Sharing'
- Set the Share Name as 'zip' with the appropriate access control
- Restart windows

Windows 2000

- Double click 'My Computer'
- Right click on the ZIP drive and choose 'Sharing'
- Set the Share Name as 'zip' and the appropriate access control
- Logout and login again as normal using your current login

Windows XP

- Double click '**My Computer**'
- Right click on the ZIP drive and choose '**Sharing and Properties**'
- Set the Share Name as 'zip' and the appropriate access control
- Logout and login again as normal using your current login

Test Your Windows Client Configuration

Use the **smbclient** command to test your share. You should substitute "WinClient" with the name of your widows client PC and "username" with a valid workgroup/domain username that normally has access to the Windows client. You should get output like this when using the username's corresponding password:

```
[root@bigboy tmp]# smbclient -L WinClient -U username
added interface ip=192.168.1.100 bcast=192.168.1.255
nmask=255.255.255.0
added interface ip=127.0.0.1 bcast=127.255.255.255 nmask=255.0.0.0
Got a positive name query response from 192.168.1.253
( 192.168.1.253 )
Password:
Domain=[HOMENET] OS=[Windows 5.1] Server=[Windows 2000 LAN Manager]

Sharename Type Comment
--------- ---- -------
IPC$ IPC Remote IPC
D$ Disk Default share
print$ Disk Printer Drivers
SharedDocs Disk
zip Disk
Printer2 Printer Acrobat PDFWriter
ADMIN$ Disk Remote Admin
C$ Disk Default share
```

```
Server Comment
--------- -------

Workgroup Master
--------- -------
```

Note: You could have got the same result using the following command, though it is less secure:

```
[root@bigboy tmp]# smbclient -L WinClient -U username%password
```

Create A ZIP Drive Mount Point On Your Samba Server

You'll need to create the mount point on the Linux server in order to mount and access the ZIP floppy. Here's how to do it.

Prompted For Password Method

```
[root@bigboy tmp]# mkdir /mnt/zip
[root@bigboy tmp]# mount -t smbfs -o username=username
//winclient/zip /mnt/zip
```

Not Prompted For Password Method

```
[root@bigboy tmp]# mkdir /mnt/zip
[root@bigboy tmp]# mount -t smbfs -o
username=username,password=password //winclient/zip /mnt/zip
```

Using The smbmount Command Method

Some versions of Linux support the **smbmount** command to mount the remote drive. Incompatible versions will give errors like this:

```
[root@bigboy tmp]# smbmount //winclient/zip /mnt/zip  -o
username=username
Password:
27875: session setup failed: ERRDOS - ERRnoaccess (Access
denied.)
SMB connection failed
```

Chapter 11

Linux Wireless Networking

= =

In This Chapter

= =

This chapter will show you how to configure wireless NIC cards on your Linux box. As of version 8.0, RedHat did not ship with the Linux Wireless driver set (Linux-WLAN) installed. You have to download and install them after installing Linux. For this reason it may be good to keep your regular Ethernet NIC installed in the machine to provide connectivity to the web until you get the wireless NIC to work, or you could ask someone to burn a CD with the files needed for installation.

If you are not running RedHat Linux, or have upgraded your RedHat distribution's kernel then you can refer to the sections on how to install WLAN from universally usable tar files.

Wireless Linux Compatible NICs

Not all wireless NIC cards work with Linux-WLAN. For this reason it is best to check the Linux-WLAN group's website at www.linux-wlan.org for the latest hardware compatibility list.

The Linksys WMP11 NIC and Linux

You have to be especially carefull with Linksys WMP series of wireless PCI cards. The older version of the card that uses the Intersil chipset works with Linux, but the newer version 2.7 card using a Broadcom chipset will not. Even so, the original WMP won't work without upgrading the firmware.

Pre Version 2.7 WMP 11 Card

This card uses the Linux-WLAN compatible Intersil chipset and doesn't have any version number stamped on it. You'll have to download and install the latest firmware for a card from the Linksys website, then install the card in a windows box and upgrade the

firmware. If you don't, your Linux box may not detect your NIC card at all and you will get kernel error messages like this one in /var/log/messages after you finish installing the software.

```
Aug 25 21:07:06 hostname kernel: p80211knetdev_hard_start_xmit:
Tx attempt prior to association, frame dropped.
```

Be careful as this message can also be due to you using an SSID in your configuration files that doesn't match the SSID of your WAP / wireless router.

The WMP 11 Version 2.7 Card

In September 2002, Linksys launched a Version 2.7 (or v2.7) model of the WMP11 card using a Broadcom chipset. The linux-wlan.org site's hardware compatibility page now lists the WMP v2.7 as being an incompatible device. You can determine whether you have this model by looking for the "V2.7" which is very clearly stamped on the front side of these cards.

Installing the WMP11 v2.7 with the linux-WLAN tarball will give the following error in the log file **/var/log/messages**

```
00:0c.0 Network controller: BROADCOM Corporation: Unknown device
4301 (rev01)
Subsystem: Unknown device 1737:4301
Flags: bus master, fast devsel, latency 64, IRQ 5
Memory at f4000000 (32-bit, non-prefetchable) [size=3D8K]
Capabilities: [40] Power Management version 2
```

Installing the WMP11 v2.7 with the linux-WLAN tarball using RPMs will give the following error message on the screen:

```
Dec 1 01:28:14 bigboy insmod: /lib/modules/2.4.18-
14/net/prism2_pci.o: init_module: No such device
Dec 1 01:28:14 bigboy insmod: Hint: insmod errors can be caused
by incorrect module parameters, including invalid IO or IRQ
parameters. You may find more information in syslog or the
output from dmesg
Dec 1 01:28:14 bigboy insmod: /lib/modules/2.4.18-
14/net/prism2_pci.o: insmod wlan0 failed
```

Linux-WLAN Preparation

All devices on a wireless network must use the same "Network Identifier" or SSID in order to communicate with each other. The default SSID for Linux-WLAN is "linux-wlan", the default SSID for your windows NIC cards may be different. It's a good idea to decide on a common SSID and stick with it.

Once configured, Linux-WLAN doesn't identify the wireless NIC as an Ethernet "eth" device, but as a "wlan" device. This is good to know in order to avoid confusion when troubleshooting.

Always be prepared to check your syslog **/var/log/messages** file for errors if things don't work. It is a good source of information. The syslog chapter will also show you how to set up syslog error logging to be more sensitive to error types.

You may get "device unknown" or "no such device" errors related to the wlan device in the **/var/log/messages** file if you use older unpatched versions of the Linux-WLAN software. Always use the most recent versions to reduce the installation mental stress.

PCMCIA Type Card Specific Information

Before installing the linux-wlan software for PCMCIA type cards such as the (Linksys WPC11) you will need to install the RedHat Linux "pcmcia-cs" RPM package. This step isn't necessary for true PCI cards such as the Linksys WMP11. According to the linux-wlan documentation, this will have to be done from a source RPM. The latest version as of this writing was:

```
kernel-pcmcia-cs-3.1.31-9.i386.rpm
```

Downloading and installing RPMs isn't hard. If you need a refresher, the RPM chapter covers how to do this in detail.

Installing The Linux-WLAN Drivers

Linux-WLAN Installation - Using RPMs

- Download the latest version of linux-wlan RPM. RPM versions of the driver files can be found at http://prism2.unixguru.raleigh.nc.us. Remember to download the files for the correct kernel type, OS version and kernel version. Downloading and installing RPMs isn't hard. If you need a refresher, the RPM chapter covers how to do this in detail.

Determining The Kernel Type

Use the "uname -p" command. The Bigboy server discussed in the Topology chapter is running a i686 version of Linux. The Linux version may not match the CPU you have installed, always use the uname version.

```
[root@bigboy tmp]# uname -p
i686
[root@bigboy tmp]#
```

Determining The OS Version

One of the easiest ways is to view the **/etc/ redhat-release** file. In this case server Bigboy is running RedHat version 9.0. You can also look at the **/etc/issue** file for non RedHat versions of Linux.

```
[root@bigboy tmp]# more /etc/redhat-release
Red Hat Linux release 9 (Shrike)
[root@bigboy tmp]#
```

Determining The Kernel Version

You can use the "uname -r" command to do this. In this case, Bigboy is running version 2.4.20-8

```
[root@bigboy tmp]# uname -r
2.4.20-8
[root@bigboy tmp]#
```

Installing the RPMs

If you upgrade the version of your Linux, you'll have to do these steps all over again. The combined Linux / Linux-WLAN upgrade will also create new versions of your **/etc/sysconfig/network-scripts/ifcfg-wlan0**, **/etc/wlan/wlan.conf** and **/etc/pcmcia/wlan-ng.opts** files which you may have to restore from the automatically saved versions.

- Once you have all this information, you'll need to download and install the base, module and interface packages. Here are examples for an i686 installation using a PCI card on Redhat 9.0 running kernel version 2.4.20-8. The packages to be installed are:

  kernel-wlan-ng-0.2.0-7.i686.rpm
  kernel-wlan-ng-modules-rh9.8-0.2.0-7.i686.rpm
  kernel-wlan-ng-pci-0.2.0-7.i686.rpm

- Here is the installation procedure, the sequence of installation is important. DOublecheck your preparation steps and the RPM versions if the very last line of the installation gives a result code that is not "success".

```
[root@bigboy tmp]# rpm -Uvh kernel-wlan-ng-0.2.0-7.i686.rpm
Preparing...                ########################################
[100%]
   1:kernel-wlan-ng         ########################################
[100%]
[root@bigboy tmp]# rpm -Uvh  kernel-wlan-ng-modules-rh9.8-0.2.0-
7.i686.rpm
Preparing...                ########################################
[100%]
   1:kernel-wlan-ng-modules-########################################
[100%]
[root@bigboy tmp]#

[root@bigboy tmp]# rpm -Uvh kernel-wlan-ng-pci-0.2.0-7.i686.rpm
Preparing...                ########################################
[100%]
   1:kernel-wlan-ng-pci     ########################################
[100%]
Adding prism2_pci alias to /etc/modules.conf file...
***NOTE***  YOU MUST CHANGE THIS IF YOU HAVE A PLX CARD!!!
The default wlan0 network configuration is DHCP.  Adjust accordingly.

ACHTUNG!  ATTENTION!  WARNING!
   YOU MUST configure /etc/wlan/wlan.conf to define your SSID!
   YOU ALSO must configure /etc/wlan/wlancfg-SSID to match WAP
settings!
       (---> replace SSID in filename with the value of your SSID)

If you get an error after this point, there is either a problem with
```

```
your drivers or you don't have the hardware installed!  If the
former,
get help!

Starting WLAN Devices:message=dot11req_mibset
  mibattribute=dot11PrivacyInvoked=false
  resultcode=success
message=dot11req_mibset
  mibattribute=dot11ExcludeUnencrypted=false
  resultcode=success
[root@bigboy tmp]#
```

- Under version 8.o of RedHat I have seen the **kernel-wlan-ng-pcmcia rpm** installation give errors stating that the kernel-pcmcia-cs rpm hadn't been previously installed even when it had been. Installing the rpm with --force and --nodeps switches does the trick by forcing the installation while not checking for dependencies. Always remember that under normal circumstances this wouldn't be a good idea, error messages are there for a reason.

```
[root@bigboy tmp]# rpm -Uvh kernel-wlan-ng-pcmcia-0.1.15-
6.i686.rpm
error: Failed dependencies:
      kernel-pcmcia-cs is needed by kernel-wlan-ng-pcmcia-
0.1.15-6
[root@bigboy tmp]# rpm -Uvh --force --nodeps kernel-wlan-ng-
pcmcia-0.1.15-6.i686.rpm
Preparing...                ##################################
########## [100%]
   1:kernel-wlan-ng-
pcmcia  ########################################### [100%]

Adding prism2_cs alias to /etc/modules.conf file...
Shutting down PCMCIA services: cardmgr modules.
Starting PCMCIA services: modules cardmgr.
The default wlan0 network configuration is DHCP.  Adjust
accordingly.

ACHTUNG!  ATTENTION!  WARNING!
     YOU MUST configure /etc/pcmcia/wlan-ng.opts to match WAP
settings!!!

[root@bigboy tmp]#
```

Linux-WLAN Installation – Using TAR files

If you are running a non standard version of your RedHat kernel or using a version of Linux that is incompatible with RPMs, then you'll need to use the TAR file installation method. If you are running standard RedHat Linux use the RPMs unless you have excess patience. Remember that if you upgrade your Linux version or kernel, you'll probably have to do these steps all over again

Install the Kernel Source Files

Installing Linux-WLAN using TAR files involves compiling the software to make it match the particular flavor of the Linux kernel you are running. It is therefore important to install your kernel sources files. For RedHat version 7.3 it was version 2.4.18-3.

```
[root@bigboy tmp]# rpm -Uvh kernel-source-2.4.18-3.i386.rpm
```

Download And Install The Linux-WLAN TAR File

- Download the latest version of Linux-WLAN from www.linux-wlan.org. in this example we'll be using the file **linux-wlan-ng-0.1.14-pre1.tar.gz**.

- Unzip and install the Linux-WLAN files

```
[root@bigboy tmp]# gunzip linux-wlan-ng-0.1.14-pre1.tar.gz
[root@bigboy tmp]# tar -xvf linux-wlan-ng-0.1.14-pre1.tar
[root@bigboy tmp]# cd  linux-wlan-ng-0.1.14
[root@bigboy linux-wlan-ng-0.1.14]# make clean
[root@bigboy linux-wlan-ng-0.1.14]# make config
```

==

Running "make config" command will prompt you for information:

- ❖ (PCI cards only) Say 'y' to pci and 'n' to pcmcia, plx, and usb driver questions
- ❖ (PCMCIA cards only) Say 'y' to pcmcia and 'n' to pci, plx, and usb driver questions
- ❖ When you are prompted for the "Module install directory" enter /lib/modules/"linux-kernel-version", where "linux-kernel-version" is the version of the kernel. Get a directory listing of /lib/modules/ beforehand to make sure you are providing the correct kernel directory that both matches your kernel version and that also actually has files in it. Select only a single module directory as using more than one can lead to future "make" problems.
- ❖ Use other defaults

==

- Continue with the installation

```
[root@bigboy linux-wlan-ng-0.1.14]# make all
[root@bigboy linux-wlan-ng-0.1.14]# make install
```

Configure The New wlan0 Interface Driver (PCI Cards)

- Edit /etc/modules.conf and insert the following line to load the driver on booting:

```
alias wlan0 prism2_pci
```

- Create a startup driver configuration file called wmp11 (or whatever you NIC card is named) in the /etc/init.d directory

```
[root@bigboy tmp]# vi /etc/init.d/wmp11
```

- Add the following 4 lines to the file.

```
#!/bin/bash
modprobe prism2_pci
wlanctl-ng wlan0 lnxreq_ifstate ifstate=enable
wlanctl-ng wlan0 lnxreq_autojoin ssid=linux_wlan
authtype=opensystem
exit 0
```

- Remember to modify the SSID in the above commands to match that of your WAP. You can also test these commands from the command line to see if they work. The response should be:

```
message=lnxreq_autojoin
ssid=linksys
authtype=opensystem
resultcode=success
```

If you get a resultcode=error or something else, then start over making sure you are using the latest versions of the Linux-WLAN software. At this time, the "Link" LED on your NIC card will come on solid, as it has established a link with the WAP11 access point.

- Make the file executable so that it will be able to run on the next system reboot.

```
[root@bigboy tmp]# chmod 755 /etc/init.d/wmp11
```

- The next step is to create a link to this file in the startup directories. When booting, the system needs to load the drivers for the interface before it will activate the interface. Some web sites recommend putting the driver loading commands in **/etc/rc.d/rc.local**, but this makes the driver load at the end of the booting process and the wlan0 interface will be inactive till then.

  This may not be a problem for many installations, but applications such as Samba, DHCP server, DNS (named) and SSH, when configured to specifically run on the IP address of your interface, may fail to start if the interface is down.

  If your applications are set to promiscuous listening, which is the default setting for the applications above, it may not matter and you could put these commands in your /etc/rc.d/rc.local file instead and save yourself a lot of grief.

  If you don't want to use the /etc/rc.d/rc.local file then you need to ensure that you run your custom driver script before the Linux "network" script starts up the wlan0 interface device you will create later. In RedHat the default network startup script link in /etc/rc3.d and /etc/rc5.d is named "S10network". You will need to create a symbolic

link called "S09wmp11" to make /etc/init.d/wmp11 be run before "S10network" during the boot process.

```
[root@bigboy tmp]# cd /etc/rc3.d
[root@bigboy tmp]# ls *network*    (Verify the "network" filename)
[root@bigboy tmp]# ln -s ../init.d/wmp11 S09wmp11
[root@bigboy tmp]# cd /etc/rc5.d
[root@bigboy tmp]# ln -s ../init.d/wmp11 S09wmp11
```

Configure The New wlan0 Interface Driver (PCMCIA Cards)

Open and edit the configuration options file, **/etc/pcmcia/wlan-ng.opts**. Locate the lines containing "ssid=linux_wlan" and set the SSID to whatever value you've decided to use on your wireless LAN.

NOTE: Never alias for the PCMCIA cards in **/etc/modules.conf**, as it is not neccessary, and also it will cause the system to try to bring up wlan0 before the PCMCIA services, which won't work.

Post Installation Steps

Configure The New wlan0 Interface

- Edit /etc/sysconfig/network-scripts/ifcfg-wlan0 to include the following lines:

```
DHCP Version            Fixed IP Version
============            ================

DEVICE=wlan0            DEVICE=wlan0
USERCTL=yes             BROADCAST=192.168.1.255
ONBOOT=yes              IPADDR=192.168.1.100
BOOTPROTO=dhcp          NETMASK=255.255.255.0
                        NETWORK=192.168.1.0
                        ONBOOT=yes
```

In the fixed IP version you will also need to:

- Substitute your selected IP, netmask, network, broadcast address with those above.
- Make sure you have correct gateway statement in your **/etc/sysconfig/network** file. eg. GATEWAY=192.168.1.1

Disable Your Existing Ethernet NIC

- You may want to disable your existing eth0 Ethernet interface after installing the drivers. Edit **/etc/sysconfig/network-scripts/ifcfg-eth0** file to have an "**ONBOOT=no"** entry. This will disable the interface on reboot or when **/etc/init.d/network** is restarted.

Select the Wireless mode and SSID

Using RedHat 8.0 WLAN RPMs

- Edit your **/etc/wlan.conf** file (PCI type NIC) or your **/etc/pcmcia/wlan-ng.opts** file (PCMCIA type NICs) configuration file. Locate the lines containing "ssid=linux_wlan" and set the SSID to whatever value you've decided to use on your wireless LAN.

  Also modify the IS_ADHOC option to make your NIC either support "adhoc" mode for peer to peer networks or "infrastructure" mode if you are using a WAP.

  Here is a sample snippet.

```
#=======SELECT STATION MODE==================
IS_ADHOC=n              # y|n, y - adhoc, n - infrastructure

#=======INFRASTRUCTURE STATION START==================
# SSID is all we have for now
AuthType="opensystem" # opensystem | sharedkey (requires WEP)
# Use DesiredSSID="" to associate with any AP in range
DesiredSSID="linksys"
```

Using RedHat 9.0 WLAN RPMs

- All the configuration files are located in the **/etc/wlan** directory.
- The RedHat 9.0 package allows your server to be connected to up to 3 wireless LANs. You specify the SSIDs (LAN IDs) for each wireless LAN in the **/etc/wlan/wlan.conf** file. In the example below, we make the wlan0 interface join the "linksys" WLAN. We are also making the WLAN driver scan all wireless channels for SSIDs.

```
#
# Specify all the wlan interfaces on the server
#
WLAN_DEVICES="wlan0"
#
# Specify whether the server should scan the network channels
# for valid SSIDs
#
WLAN_SCAN=y
#
# Specify expected SSIDs and the wlan0 interface to which it
should
# be tied
#
SSID_wlan0="linksys"
ENABLE_wlan0=y
```

- Each WLAN specified in the **/etc/wlan/wlan.conf** file has it's own configuration file. Copy the **/etc/wlan/wlancfg-DEFAULT** file to a file named **/etc/wlan/wlancfg-SSID** (replace SSID with the actual SSID for your WAP). In the example below, we're configuring for the "linksys" SSID.

```
[root@bigboy wlan]# cp wlancfg-DEFAULT wlancfg-linksys
```

Simulate a Reboot

- Start the wlan process and test for errors in the file **/var/log/messages**. All the resultcodes in the status messages should be “success”
- When using RPMs, you may receive the following error which the WLAN RPM website claims is “harmless”.

```
Error for wireless request "Set Encode" (8B2A) :
    SET failed on device wlan0 ; Function not implemented.
Error for wireless request "Set ESSID" (8B1A) :
    SET failed on device wlan0 ; Function not implemented.
```

PCI Cards – Installed Using RPMs

```
[root@bigboy tmp]# /etc/init.d/wlan restart
[root@bigboy tmp]# ifup wlan0
```

PCI Cards – Installed Using TAR Files

```
[root@bigboy tmp]# /etc/init.d/wmp11
[root@bigboy tmp]# /etc/init.d/network restart
```

PCMCIA Cards

```
[root@bigboy tmp]# /etc/rc.d/init.d/pcmcia restart
[root@bigboy tmp]# /etc/init.d/network restart
```

Now check to see if IP address of the wlan interface is OK

```
[root@bigboy tmp]# ifconfig -a
[root@bigboy tmp]# ping <gateway-address>
```

Check For Interrupt Conflicts

Before installing the software you should ensure that the wireless NIC card doesn’t have an interrupt that clashes with another device in your computer. Insert the card in an empty slot in your Linux box and reboot. Inspect your **/var/log/messages** file again:

```
[root@bigboy tmp]# tail -300 /var/log/messages
```

Look carefully for any signs that the card is interfering with existing card IRQs. If there is a conflict there will usually be a warning, or "IRQ also used by..." message. If that is the case, move the card to a different slot, or otherwise eliminate the conflict by disabling the conflicting device if you don’t really need it.

After you’ve installed the software, you can also inspect your /proc/interrupts file for multiple devices having the same interrupt.

```
[root@bigboy tmp]# more /proc/interrupts
11:      4639     XT-PIC     wlan0, eth0      (bad)

[root@bigboy tmp]# more /proc/interrupts
11:      4639     XT-PIC     wlan0            (good)
```

Interrupt conflicts are usually more problematic with old style PC-AT buses, newer PCI based systems generally handle conflicts better. The above (bad) **/proc/interrupts** example came from a functioning PCI based Linux box, the reason why it worked was that though the interrupt was the same, the base memory address which was used by Linux to communicate with the cards were different. You can check both the interrupts and base memory of your NIC cards ***after*** doing the software installation by using the "ifconfig -a" command:

```
[root@bigboy tmp]# ifconfig -a
eth0 Link encap:Ethernet HWaddr 00:08:C7:10:74:A8
BROADCAST MULTICAST MTU:1500 Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:100
RX bytes:0 (0.0 b) TX bytes:0 (0.0 b)
Interrupt:11 Base address:0x1820

lo Link encap:Local Loopback
inet addr:127.0.0.1 Mask:255.0.0.0
UP LOOPBACK RUNNING MTU:16436 Metric:1
RX packets:88418 errors:0 dropped:0 overruns:0 frame:0
TX packets:88418 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:0
RX bytes:7678679 (7.3 Mb) TX bytes:7678679 (7.3 Mb)

wlan0 Link encap:Ethernet HWaddr 00:06:25:09:6A:B5
inet addr:192.168.1.100 Bcast:192.168.1.255 Mask:255.255.255.0
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:215233 errors:0 dropped:0 overruns:0 frame:0
TX packets:447594 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:100
RX bytes:39394014 (37.5 Mb) TX bytes:126738425 (120.8 Mb)
Interrupt:11 Memory:c887a000-c887b000
[root@bigboy tmp]#
```

Linux-WLAN Encryption For Security

One of the flaws of wireless networking is that all the wireless clients can detect the presence of all available network SSIDs and have the option of joining any of them. With encryption, the client must have a membership encryption password which can also be represented as a series of Wireless Encryption Protocol (WEP) keys. The **wlan.conf** file (RedHat 8.0 RPMs) or **wlan-SSID** file (RedHat 9.0 RPMs) or the **/etc/pcmcia/wlan-ng.opts** file (PCMCIA type NICs) file is also used to activate this feature.

Note: I must strongly recommend that you first set up your network ***without*** encryption. Only migrate to an encrypted design after you are satisfied that the unencrypted design works satisfactorily.

To invoke encryption, you have to set the "dot11PrivacyInvoked" parameter to "true" and state which of the keys will be used as the default starting key via the "dot11WEPDefaultKeyID" parameter. You then have the option of either providing a key generating string (simple password) or all four of the keys. In the example below, "ketchup" is the password used to automatically generate the keys.

```
#=======WEP==========================================
# [Dis/En]able WEP. Settings only matter if PrivacyInvoked is true
lnxreq_hostWEPEncrypt=false # true|false
lnxreq_hostWEPDecrypt=false # true|false
dot11PrivacyInvoked=true
dot11WEPDefaultKeyID=1
dot11ExcludeUnencrypted=true # true|false, in AP this means WEP
# is required for all STAs

# If PRIV_GENSTR is not empty, use PRIV_GENTSTR to generate
# keys (just a convenience)
PRIV_GENERATOR=/sbin/nwepgen # nwepgen, Neesus compatible
PRIV_KEY128=false # keylength to generate
PRIV_GENSTR="ketchup"

# or set them explicitly. Set genstr or keys, not both.
dot11WEPDefaultKey0= # format: xx:xx:xx:xx:xx or
dot11WEPDefaultKey1= # xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx
dot11WEPDefaultKey2= # e.g. 01:20:03:40:05 or
dot11WEPDefaultKey3= # 01:02:03:04:05:06:07:08:09:0a:0b:0c:0d
```

Not all devices on your network will use the same algorithm method to generate the encryption keys. You may find the same generator string will not create the same keys, and intra-network communication will be impossible. If this is the case, you can use the **/sbin/nwepgen** program to generate the keys once you provide your easy to remember key generator string. Once you have the four sets of keys, you'll have to add them individually and in sequence to the **wlan.conf**, **wlan-SSID** or **/etc/pcmcia/wlan-ng.opts** file and set the PRIV_GENSTR parameter to "". Here is how you can use nwepgen to create the keys with a generator string of "ketchup".

```
[root@bigboy tmp]# /sbin/nwepgen ketchup
64:c1:a1:cc:db
2b:32:ed:37:16
b6:cc:9e:1b:37
d7:0e:51:3f:03
[root@bigboy tmp]#
```

In this case your **wlan.conf** or **wlan-SSID** file would look like this:

```
PRIV_GENSTR=""

# or set them explicitly. Set genstr or keys, not both.
dot11WEPDefaultKey0= 64:c1:a1:cc:db
dot11WEPDefaultKey1= 2b:32:ed:37:16
dot11WEPDefaultKey2= b6:cc:9e:1b:37
dot11WEPDefaultKey3= d7:0e:51:3f:03
```

Remember that all devices on your network will need to have the same keys and default key for this to work. This includes all wireless NICs and WAPs

De-activating Encryption

In some cases, NIC cards without full Linux-WLAN compatibility will freeze up after a number of hours of working with encryption. The steps to reverse encryption are:

- Set the configuration file parameter "**dot11PrivacyInvoked**" to "**false**"
- Stop Linux-WLAN and disable the wireless wlan0 interface

```
[root@bigboy tmp]# /etc/init.d/wlan stop
Shutting Down WLAN Devices:message=lnxreq_ifstate
  ifstate=disable
  resultcode=success
[root@bigboy tmp]# ifdown wlan0
```

- Even though you have done these two steps, the driver is still loaded in memory, though not active. Your next steps will be to list all the active drivers in memory with the **lsmod** command, and remove the Linux-WLAN related entries using **rmmod**

```
[root@bigboy tmp]# lsmod
Module                  Size  Used by    Not tainted
…
…
prism2_pci             66672   1  (autoclean)
p80211                 20328   1  [prism2_pci]
…
…
[root@bigboy tmp]# rmmod prism2_pci
[root@bigboy tmp]# rmmod p80211
```

- Restart Linux-WLAN and reactivate the wlan0 interface and you should be functional again.

```
[root@bigboy tmp]# /etc/init.d/wlan start
Starting WLAN Devices:message=lnxreq_hostwep
  resultcode=no_value
  decrypt=false
  encrypt=false
[root@bigboy tmp]# ifup wlan0
```

- If you fail to reload the driver modules you'll get errors like these below in your **/var/log/messages** file.

```
Jan 2 18:11:12 bigboy kernel: prism2sta_ifstate:
hfa384x_drvr_start() failed,result=-110
Jan 2 18:11:18 bigboy kernel: hfa384x_docmd_wait: hfa384x_cmd
timeout(1), reg=0x8021.
Jan 2 18:11:18 bigboy kernel: hfa384x_drvr_start: Initialize
command failed.
Jan 2 18:11:18 bigboy kernel: hfa384x_drvr_start: Failed,
result=-110
```

Troubleshooting Your Wireless LAN

Always check the **/var/log/messages** file for possible errors arising from the software installation. The chapter on logging covers how to do this in more detail.

p80211 Kernel errors in **/var/log/messages** usually point to an incorrectly configured SSID

```
Nov 13 22:24:54 bigboy kernel: p80211knetdev_hard_start_xmit: Tx
attempt prior to association, frame dropped.
```

If there are no errors in **/var/log/messages** and you can't ping your gateways or obtain an IP address, then check your **/etc/sysconfig/network-scripts/ifcfg-wlan0** file for a correct IP configuration and your routing table to make sure your routes are OK. You can also check to see if your Linux box is out or range of the WAP.

Chapter 12

Using Sudo

===

In This Chapter

===

You can give selected users temporary "root" privileges using the "sudo" command, here's how.

What is sudo?

- Sudo is a command that allows users defined in the **/etc/sudoers** configuration file to have temporary **root** access to run certain privileged commands.
- The command you want to run must first begin with the word "**sudo**" followed by the regular command syntax.
- When running the command you will be prompted for your regular password before it is executed.You may run other privileged commands using **sudo** within a five minute period without being re-prompted for a password
- All commands run as **sudo** are logged in the log file **/var/log/messages**

Download and Install The sudo Package

Fortunately the package is installed by default by RedHat

The visudo Command

- "**visudo**" is the command used to edit the **/etc/sudoers** configuration file. It is not recommended that you use any other editor to modify your **sudo** parameters. "**visudo**" uses the same commands as the "**vi**" text editor.
- "**visudo**" is best run as user "root"

```
[root@aqua tmp]# visudo
```

The /etc/sudoers File

General Guidelines

- The **/etc/sudoers** file has the general format:

  usernames/group target-servername = command

- Groups are the same as user groups and are differentiated from regular users by a % at the beginning
- The "#" at the beginning of a line signifies a comment line
- You can have multiple usernames per line separated by commas
- Multiple commands can be separated by commas too. Spaces are considered part of the command.
- The keyword "ALL" can mean all usernames, groups, commands and servers.
- If you run out of space on a line, you can end it with a "\" and continue on the next line.
- The NOPASSWD keyword provides access without you being prompted for your password

Simple sudoers Examples

- Users "paul" and "mary" have full access to all privileged commands

```
paul, mary  ALL=(ALL) ALL
```

- Users with a groupid of "operator" has full access to all commands and won't be prompted for a password when doing so.

```
%operator  ALL=(ALL) NOPASSWD: ALL
```

How To Use sudo

- In this example, user "paul" attempts to view the contents of the **/etc/sudoers** file

```
[paul@bigboy paul]$ more /etc/sudoers
/etc/sudoers: Permission denied
[paul@bigboy paul]$
```

- Paul tries again using **sudo** and his regular user password and is successful

```
[paul@bigboy paul]$ sudo more /etc/sudoers
Password:
...
...
...
[paul@bigboy paul]$
```

Using syslog To Track All sudo Commands

All **sudo** commands are logged in the log file **/var/log/messages**. Here is sample output from the above example.

```
[root@bigboy tmp]# grep sudo /var/log/messages
Nov 18 22:50:30 bigboy sudo(pam_unix)[26812]: authentication failure;
logname=paul uid=0 euid=0 tty=pts/0 ruser= rhost= user=paul
Nov 18 22:51:25 bigboy sudo: paul : TTY=pts/0 ; PWD=/etc ; USER=root ;
COMMAND=/bin/more sudoers
[root@bigboy tmp]#
```

Appendix 1

Miscellaneous Topics

= =

In This Chapter

= =

We briefly discuss some miscellaneous topics in this chapter that are beyond the scope of this book with the intention that you will be stimulated to consider utilizing some of the technologies discussed to improve your website and / or your SOHO network.

VPN Terminologies

A Virtual Private Network (VPN) provides security for transmission of sensitive information over unprotected networks such as the Internet. VPN relationships are established between trusted sites on the Internet making the public network appear to be virtually the same as a private network to the VPN members. What follows is an introduction to VPN terminology

Authentication

The process of ensuring that the VPN data received is both unchanged and from the expected source.

Encryption

The process of encoding VPN data to protect it from unauthorized viewing except by the intended recipient who has the decoder key.

IPSec

The name given to a number of data communications protocols designed to authenticate and encrypt VPN data to protect it from unauthorized viewing or modification as it is transmitted across a network.

Authentication Header (AH)

One of two IPSec security protocols. Provides authentication and anti-replay services, without encryption. It does this by adding its own security header to the original IP packet.

Encapsulating Security Protocol (ESP)

The other IPSec security protocol. Provides authentication, encryption, and anti-replay services. It does this by encrypting the data within the packet and then adding its own security header to the original IP packet. As ESP headers don't authenticate the outer IP header like AH headers, AH and an ESP are often used in combination with each other. This is called Transport Adjacency.

Transport mode VPNs

The original source and destination address of the data being sent over the VPN is unchanged. Here are some examples of what transport mode VPN IP packets will look like. (For more information on the IP protocol, please refer to the OSI model page)

Transport mode AH packet format

Original IP Header	**Inserted AH Header**	Original TCP Header	DATA

Transport mode AH / ESP packet format

Original IP Header	**Inserted AH Header**	**Inserted ESP Header**	Original TCP Header	DATA

Tunnel mode VPNs

The original source and destination address of the data being sent over the VPN is changed by encapsulating the original IP packet within another IP packet. The original packet is frequently encrypted, header and all in an effort to provide an additional layer of security by not revealing the true identities of the servers communicating with each other.

Here are some examples of what tunnel mode VPN IP packets will look like. (For more information on the IP protocol, please refer to the OSI model page)

Tunnel mode AH packet format

New IP Header	Inserted AH Header	Original IP Header	Original TCP Header	DATA

Tunnel mode AH / ESP packet format

New IP Header	Inserted AH Header	Inserted ESP Header	Original IP Header	Original TCP Header	DATA

Authentication methods

IPSec data integrity is usually provided by one of two Hashed Message Authentication Code (HMAC) methods:

- Message Digest 5 (MD5)
- Secure Hash Algorithm (SHA-1).

Encryption methods

IPSec usually uses one of two methods to encrypt data:

- The Data Encryption Standard (DES) using a 56-bit encryption key
- Triple DES using a 168-bit encryption key.

Internet Key Exchange (IKE)

IKE provides authentication of the IPSec peers, negotiates IPSec security associations, and establishes IPSec keys.

IKE authentication methods

There are two main methods of establishing a trusted relationship between two devices that want to create a VPN between themselves:

Public key cryptography using RSA encryption

RSA overview

Each VPN device has its own public and private keys. Anything encrypted with one of the keys can only be decrypted with the other. This allows you to create a signature when the message is encrypted with a sender's private key. The receiver verifies the signature by decrypting the message with the sender's public key.

As the message could be decrypted using the sender's public key means that the holder of the private key created the message. A successful exchange requires the receiver to have a copy of the sender's public key and knowing with a high degree of certainty that it really does belong to the sender, and not to someone pretending to be the sender. This is done using Certification Authorities.

CA overview

A digital certificate contains information that identifies a user or device, such as a name, serial number, company, department, or IP address. It will also contain a copy of the entity's public key.

Certificates are managed and issued by Certification authorities (CAs). A CA can either be a trusted public third party, such as VeriSign, or a "in-house" private server that you establish within your organization.

Prior to installing a certificate based VPN, each VPN device must be pre-configured with the certificate generated for them by the CA. The VPN devices will also be pre-configured with the CA's certificate.

During the key exchange, the VPN peers authenticate by sending each other the certificate issued to them by the CA, but encrypted using their private key.

Each peer then uses the pre-installed CA certificate they have to authenticate with the CA and securely receive the other peer's certificate from the CA using public key cryptography.

Each peer then extracts the public key from the certificate they receive from the CA and then uses it to decrypt the certificate they just received from the other peer.

Once the the certificates received from the CA and the other peer match, authentication is complete.

Shared keys

The devices at each end of the VPN use a shared key or password. The disadvantage is that each pair of VPN connections need set of keys, making it difficult for large scale implementations. Unlike the RSA method, there is no CA to provide an impartial audit trail of VPN connection initiations.

IKE's role in creating Security Associations

Once authentication is complete, IKE is then used by the VPN peers to negotiate the security associations (SAs) to be used at each end point. SAs are comprised of two factors:

Transforms

Describes how the data will be transformed by the VPN to provide the desired security. This includes:

- Packet encryption methods
- Packet authentication methods
- Transport versus tunnel mode
- AH and or ESP usage
- SA lifetime before it is renegotiated. (SAs are permanent for when manually established)

Shared keys

The actual keyword used by the encryption and authentication to protect the data.

IKE and ISAKMP

IKE uses special ISAKMP IP packets using "protocol 50" to establish an Security Association.

VPN Security And Firewalls

- All security devices in the path of a VPN connection will have to allow "protocol 50" between the two VPN devices to ensure that IKE works properly.
- The VPN uses a separate channel through which the encrypted data passes. This uses UDP packets using port 500. Unusually, the source and destination port is 500. This must also be allowed to pass through unimpeded.
- In permanent VPNs, you may have to open up these ports and protocols to the CA as well.

VPN User Authentication Methods For Temporary Connections

The above sections have been slanted towards a permanent connection between purpose built VPN devices. Frequently the device at the other end of the connection is a PC. Here are some authentication methods used in such

Types Of Dial Up VPN Authentication

Method	Description
IKE-XAUTH secured RADIUS	Usernames and passwords entered into the VPN remote login software are relayed by the VPN device at the remote end to a trusted RADIUS server. If the username / password combination is valid for remote login then the RADIUS server will authorize the VPN device to continue with the IKE interchange.
ACE/SecurID	Software uses a username, password in conjunction with a digital key FOB whose authentication serial number changes every few minutes for a login to occur. In order to login, the user not only has to enter the username & password, but also the PIN tied to the FOB plus the FOB's dynamic serial number which is synchronized with the authentication server at the other end of the VPN.
Windows Domain	Remote home user authentication relies on the same username / password combination of the Windows Domain Controller that the user would normally use to login when they are at work.
Local user database	Valid usernames and passwords are configured into the VPN device at the other end of the VPN

Running Linux Without A Monitor

You can reduce the cost of ownership of your Linux system by not using a VGA monitor. This creates what is also known as a “headless” system. Operating costs may not be important at home, but will be in a corporate environment with large numbers of Linux servers racked in data centers. In such cases, access to the Linux box can be more cheaply provided via the COM port.

I’ve included this section as I have occasionally hosted the website www.linuxhomenetworking.com at friends’ homes and felt badly about borrowing their monitors. Having access via the COM ports has also helped me in both the home and business situations. The most common occurrence is when the system is hung, locking out network access, and I need to get to it by using:

- A notebook PC with a console cable connected to the COM port.
- A modem connected to the COM port
- Telnet to login to a terminal server that has one of its ports connected to the Linux box’s COM port

Preparing To Go “Headless”

- One of the advantages of this method is that you don't need a keyboard either. Unfortunately your BIOS may halt the system during the Power On Self Test (POST) if it doesn't detect a keyboard. Make sure you disable this feature in the BIOS setup of your PC before proceeding. This feature can usually be found on the very first screen under the “Halt On” option.
- You will also need to make sure that you have activated your COM ports in your BIOS settings.
- For non-modem connectivity (PC to PC) connect a **NULL** modem cable to the COM port you want to test, connect the other end to the client PC running "Hyperterm" or whatever terminal emulation software you are using. One popular Linux equivalent to Hyperterm is “minicom”. A brief configuration guide for minicom follows the section below.
- If you're using a modem for connectivity, then you'll need a **FULL** modem cable and testing will have to be done using a dial up connection.

Configuration Steps

In RedHat Linux, the COM1 and COM2 ports are controlled by a program called "agetty", but "agetty" usually isn't activated when you boot up unless its configuration file **/etc/inittab** is modified. In other versions of Linux, "agetty" may be called just plain "getty". Here is a table that lists the physical ports to their equivalent Linux device names.

Port	Linux "agetty" Device Name
COM1	ttyS0
COM2	ttyS1

The following lines added to **/etc/inittab** will configure your COM ports for terminal access:

```
# Run COM1 and COM2 gettys in standard runlevels
S0:235:respawn:/sbin/agetty -L 19200 ttyS0 vt102
S1:235:respawn:/sbin/agetty -L 19200 ttyS1 vt102
```

In summary, these lines mean:

- At boot time, when the system enters runlevels 2, 3 or 5, "agetty" must attach itself to devices ttyS0 and ttyS1 and emulate a VT102 terminal running at 19200 baud.
- The "-L" means ignore modem control signals, this option should be omitted if you are connecting the port to a modem.
- The respawn means that agetty will restart automatically if, for whatever reason, it dies.

The next step is to restart the "init" process to re-read **/etc/inittab**

```
[root@bigboy tmp]# init q
```

Now you need to configure the terminal client such “as Hyperterm” to match the speed settings in **/etc/inittab**. Connect the console / modem cable between the client and your Linux box. Hit "enter" a couple times, and celebrate when you see something like this:

```
Red Hat Linux release 8.0 (Psyche)
Kernel 2.4.18-14 on an i586

bigboy login:
```

Note: By default, user "root" will not be able to log in from a terminal. To do this you'll have to edit the **/etc/securetty** file which contains the device names of tty lines on which root is allowed to login. Just add ttyS0 and ttyS1 to the list if you need this access.

Make Your Linux Box Emulate A VT100 Dumb Terminal

Dumb terminals can be loosely defined as devices that allow you to log in to your system via the COM port. You can make your Linux box emulate a dumb terminal quite easily. There are a number of reasons to do this:

- You run Linux on a notebook and you need to use it to access a hung “headless” Linux server via the COM port
- You need to gain access to a modem connected to the COM port. (This section will focus on the notebook scenario, not the use of using Linux to dial a modem.)

Configuration Steps

The most commonly used Linux terminal emulation program is minicom. It is simple to use mainly because it uses a text based GUI. Here are the steps you’ll need to go through to get it working.

- You will first need to go through all the relevant steps listed in the “Preparing to go Headless” section of this chapter to ensure you have the right type of cable and correct BIOS settings.
- Minicom will clash with your **agetty** configuration explained in the previous section. In other words a “headless” system cannot be used to access another “headless” system using the “headless” COM port. You therefore have to disable the agetty configuration for the port on which you wish to run minicom.

  In the case below we disable agetty on COM1 by commenting out the ttyS0 agetty statements in the **/etc/inittab** file.

COM1 will therefore be used for outbound minicom connections to other systems. Other systems using minicom can use COM2 to access this system.

We then need to restart the init process to reload the new **/etc/inittab** settings.

Edit /etc/inittab

```
# Run COM1 and COM2 gettys in standard runlevels
#S0:235:respawn:/sbin/agetty -L 19200 ttyS0 vt102
S1:235:respawn:/sbin/agetty -L 19200 ttyS1 vt102
```

Restart init

```
[root@bigboy tmp]# init q
```

- Run minicom in setup mode using the **minicom –s** command

```
[root@bigboy tmp]# minicom –s
```

- You will get the setup menu

```
------[configuration]-------
| Filenames and paths       |
| File transfer protocols   |
| Serial port setup         |
| Modem and dialing         |
| Screen and keyboard       |
| Save setup as dfl         |
| Save setup as..           |
| Exit                      |
| Exit from Minicom         |
---------------------------
```

- Select the serial port setup menu item. Make the speed match that of the remote "headless" system and make sure the correct serial COM device is chosen. Device **/dev/ttyS0** is COM1 and **/dev/ttyS1** is COM2. Also make sure that flow control is off.

```
-----------------------------------------
| A -    Serial Device      : /dev/ttyS0  |
| B - Lockfile Location     : /var/lock   |
| C -   Callin Program      :             |
| D -  Callout Program      :             |
| E -    Bps/Par/Bits       : 19200 8N1   |
| F - Hardware Flow Control : No          |
| G - Software Flow Control : No          |
|                                         |
|    Change which setting?                |
-----------------------------------------
```

- Select the "Modem and dialing" option and make sure the "Init string" and "Reset string" settings are blank.
- Select the "Save setup as dfl" to make this your saved default setting and then "Exit from Minicom"

- Make sure the other system is correctly configured for headless operation. Connect the cables between the systems
- Re-enter minicom, this time without the "**-s**"

```
[root@bigboy tmp]# minicom
```

- Hit enter and you should get a login prompt

```
Welcome to minicom 2.00.0

OPTIONS: History Buffer, F-key Macros, Search History Buffer,
I18n
Compiled on Jun 23 2002, 16:41:20.

Press CTRL-A Z for help on special keys

bigboy login:
```

- To exit minicom you type CTRL-A, then Z, then X
- Non "root" users will get a "permission denied" message if they use minicom as the COM ports are not normally accessible to regular users. The way to get around this is for user "root" to either give everyone read/write access using the **chmod** command below, or add selected trusted users to your sudo configuration.

  Remember that minicom will reset the privileges to the COM port each time you change the configuration with "**minicom –s**" so you may find yourself having to run **chmod** from time to time.

```
[root@bigboy tmp]# chmod o+rw /dev/ttyS0
```

Disk Partitioning Explained

Here's some interesting information on how Linux handles hard drives and their partitions.

What Is A Partition?

A partition is a means of dividing your hard disk into multiple sections, each of which is treated as a separate disk by your operating system. This allows you to be able to boot different operating systems from the same disk, for example, Linux and Windows. Partitions cannot be moved or resized without destroying the data on them.

What Is A Filesystem?

Filesystems can be considered as being the directory structures on a disk partition that contains all the files. Most Windows users would be familiar with the analogous terms

"folders" and "sub-folders", whereas Linux users would be familiar with the terms "directories" and "sub-directories".

How Linux Links Filesystems And Partitions

In Windows, a disk with two partitions would most likely find itself with a "C:" drive and a "D:" drive each with a separate set of folders.

In Linux, everything appears to be a single set of "folders" or directories, the partitions hide underneath unseen to the regular user. In other words, Linux partitions handle all files in the subdirectory of your choice. The choice of which subdirectories belong in which partition is made when you partition the hard drive.

If you create a file system called "/var" and another called "/var/log". The directory allocation will be:

Partition	Directory Allocation
/var/log	All files in directory /var/log and all the subdirectories underneath /var/log
/var	All files in /var except the contents of directory /var/log and all the subdirectories underneath /var/log

What Partitions Are Mandatory?

The mandatory partitions are:

"/", Also Known As "root"

The root filesystem ("/") contains the files necessary for the system to boot up in single user mode with the bare minimum of functionality. Most Linux systems then will then become multi-user systems by changing their runlevel and executing the associated startup scripts including those that will mount the remaining file systems.

In most cases, a corrupted root filesystem will make your system unbootable from the hard drive. Usually recovery requires reformatting the root file system and doing a Linux reinstall. If all your files are located in a root filesystem that becomes corrupted, there is a high possibility that you will lose all your data. It is for this reason that you should consider placing similar files in dedicated partitions.

/boot

The /boot partition contains the Linux kernel which is the "master control program", not only for controlling the boot process, but also the normal functioning of Linux.

Redhat Linux creates this partition automatically. This reduces the need to reformat the "root" partition if it becomes corrupted.

swap

Used as a location to place data temporarily if RAM memory becomes full. RedHat automatically creates this partition and usually makes it about twice the amount of system RAM.

Recommended Sizes For Disk Partitions

Here are some allocation suggestions that may be useful. This is a generous guide, you may look at the bottom of the table to see the settings I used for the 4GB hard disk I used to first run www.linuxhomenetworking.com

The RedHat default partitioning scheme used during the Linux installation should be sufficient for most home / SOHO systems, especially if you have more than 4GB.

Some Recommended Partition Sizes

Partition	Size Workstation / Home Web Server (MB)	Size Mail Server (MB)	Purpose of the Partition
Swap	2 X Physical Memory	2 X Physical Memory	Used as a location to place data temporarily if RAM memory becomes full
/ Usually called "root"	100	100	Base system program files Configuration files Libraries Device drivers
/boot	Automatically determined by RedHat 50 (approx)	Automatically determined by RedHat 50 (approx)	Assigned automatically by RedHat. Contains the system kernel files.
/usr	2.0+ GB	1.0+ GB	Third party software

Partition	Size Workstation / Home Web Server (MB)	Size Mail Server (MB)	Purpose of the Partition
/var	750+	500+	Man pages (Help files)
/var/log	500+	1.5+ GB	Error log files
/var/spool	750+	500	Print queues (Needed only if you plan to do printing)
/var/spool/mail	N/A	4.0+ GB	Mail queues
/tmp	250+	250+	Temporary files
/home	Variable	Variable	Multiply the expected number of users by the amount of disk space you want to assign per user. Remember MP3s fill disks quickly.

How Much Space Do I Have On My Partitions?

You can use the "df" command. These are the settings I used for the hard disk I used to first run this site.

```
[root@bigboy root]# df -k
Filesystem           1K-blocks      Used Available Use% Mounted on
/dev/hda3               505636     90002    389529  19% /
/dev/hda1                46636      9164     35064  21% /boot
/dev/hda5               505605     87915    391586  19% /home
/dev/hda7               830104     17632    770304   3% /tmp
/dev/hda2              4633108   1797504   2600252  41% /usr
/dev/hda6               256667    169577     73838  70% /var
[root@bigboy root]#
```

What Can I Do When I Run Out Of Disk Space?

- If it is in the /home partition, you can ask users to delete unnecessary files such as downloaded software and MP3s

- If it is in the /var partition, then you can consider deleting some of your log files in /var/log. See the logging page for a description of the log files and how you can use the "logrotate" command to help reduce their size.
- You may want to also consider backing up files and then deleting them.

The OSI Networking Model

The Open System Interconnection (OSI) protocol suite acts as a framework for designing network based applications. It consists of layers of sub-applications, each building on the lower ones to provide a complete connectivity solution. Each layer generally has "hooks" into the layer immediately above and below it so that the data can flow smoothly through the sub-applications designed to handle each layer. Detailed descriptions of TCP, UDP, IP and ARP can be found in the Introduction to Networking chapter.

The Seven OSI Layers

Layer	Name	Description	Application
7	Application	• The user interface to the application	Telnet FTP Sendmail
6	Presentation	• Converts data from one presentation format to another. For example, email text entered into Outlook express being converted into SMTP mail formatted data.	
5	Session	• Manages continuing requests and responses between the applications at both ends over the various established connections.	
4	Transport	• Manages the establishment and tearing down of a connection. Ensures that unacknowledged data is retransmitted. Correctly re-sequences data packets that arrive in the wrong order.	TCP UDP
3	Network	• Handles the routing of data between links that are not physically connected together. • Used as a means of not tying the address of the server to its MAC address. Your network address can stay the same if the NIC is replaced. • Also allows you to select your own numbering scheme for the global network independent of the MAC address which in rare cases could also be duplicated. • Also provides the option of having multiple addresses of the same networking protocol being assigned to the same MAC address.	IP ARP
2	Link	• Error control and timing of bits speeding down the wire between two directly connected devices. In the home environment, data is frequently sent using the MAC addresses of the NIC cards of the communicating devices.	Ethernet ARP

Layer	Name	Description	Application
1	Physical	• Defines the electrical and physical characteristics of the network cabling and interfacing hardware	Ethernet

TCP/IP Packet Format

The TCP/IP packet contains an IP header followed by a TCP or UDP header followed by the TCP/UDP data.

IP Header	TCP/UDP Header	DATA

Contents Of The IP Header

Field	Description
IP Version	The version of IP being used. Version 4, is the current version used by most devices on the Internet. Version 6, is a newer format which allows for a much more vast range of addresses.
IHL	Internet Header Length. Total length of the IP header
Total Length	Total length of the IP packet
DF Bit	Indicator to tell whether the data in the packet may be fragmented into smaller packets due to limitations of the communications line
MF Bit	Indicator to tell whether this data in this packet is the last one of a stream of fragments.
Fragment Offset	If this packet is part of a fragmented datagram, then this specifies where in the complete datagram the data in this packet should be inserted.
TTL	Time to live. This value is decremented by each router through which the packet has passes. When the value reaches zero, the packet is discarded by the router. The server sending the data usually sets the TTL to a value high enough to reach it's destination without being discarded. The TTL decrement feature is used by routers as an additional precaution to prevent the packet from mistakenly being routed around the Internet in an infinite loop due to a routing error.
Protocol	Defines the type of protocol header to expect at the end of this header. For example, 6 = TCP. 17 = UDP

Field	Description
Header checksum	Used to ensure that the header contents are error free.
Source Address	Indicates the IP address of the server sending the data
Destination Address	Indicates the IP address of the server intended to receive the data

Contents Of The TCP Header

Field	Description
Source and Destination Port	Identifies points at which upper-layer source and destination processes receive TCP services.
Sequence Number	Usually specifies the number assigned to the first byte of data in the current message. In the connection-establishment phase, this field also can be used to identify an initial sequence number to be used in an upcoming transmission.
Acknowledgment Number	Contains the sequence number of the next byte of data the sender of the packet expects to receive.
Data Offset	Length of the TCP header.
Flags	Carries a variety of control information, including the SYN and ACK bits used for connection establishment, and the FIN bit used for connection termination.
Window	Specifies the size of the sender's receive window (that is, the buffer space available for incoming data)
Checksum	Used to ensure that the header contents are error free.
Data	Contains upper-layer information

Contents Of The UDP Header

Field	Description
Source and Destination Port	Identifies points at which upper-layer source and destination processes receive TCP services.
Length	Specifies the length of the UDP header and data
Checksum	Used to ensure that the header contents are error free.

Appendix 2

Bibliography

===

In This Chapter

===

An informal listing of some of the sites I visited to create this manual.

Wireless Linux

A good reference page (Has GUI screens for PCMCIA type cards)

- http://www.saragossa.net/LinuxG3/ls-wlan.shtml

The LDP

- http://linuxdoc.org/HOWTO/Wireless-HOWTO

Wireless LAN Resources for Linux

- http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/

The drivers are here...

- http://www.linux-wlan.org/

RPM versions of the drivers can be found here

- http://prism2.unixguru.raleigh.nc.us

Netfilter - iptables Configuration

Netfilter iptables HOWTO tutorial site

- http://www.netfilter.org/documentation/tutorials/blueflux/iptables-tutorial.html

General Home Networking Resource Pages

Linux networking HOWTO

- http://www.tldp.org/HOWTO/Net-HOWTO/
- http://www.tldp.org/HOWTO/Networking-Overview-HOWTO.html
- Linux Networking-HOWTO

ICMP codes listing

- http://www.nwconnection.com/oct.99/icmp09/code.html

TCP and UDP port assignments

- http://www.iana.org/assignments/port-numbers

SSH Servers and SSH Clients

PuTTY homepage (Free)

- http://www.chiark.greenend.org.uk/~sgtatham/putty/

SecureCRT homepage (Purchase Required)

- http://www.vandyke.com/

The Windows SCP client called WinSCP

- http://winscp.vse.cz/eng/

FTP Server and FTP Clients

A good site on how to install one of the many Windows GUI FTP clients

- Zen and the Art of FTP: An FTP Tutorial

Perl Script to do Automatic FTP File Copies to Update Your Website

- http://www.unixreview.com/documents/s=2426/uni1018361705067/0204e.htm

DHCP Server

RedHat's guide to configuring DHCP

- http://www.redhat.com/docs/manuals/linux/RHL-8.0-Manual/custom-guide/s1-dhcp-configuring-server.html

Linux Magazine's detailed guide

- http://www.linux-mag.com/2000-04/networknirvana_05.html

Apache Web Server Software

Apache's documents on setting up virtual hosts

- http://httpd.apache.org/docs/vhosts/examples.html

Apache week's description of how to do virtual hosting

- http://www.apacheweek.com/features/vhost

Sendmail Mail Configuration

Good sendmail quick setup guide

- http://www.neilgunton.com/network_howto/part3/

Good page on how to stop SPAM with sendmail

- http://www.brettglass.com/spam/paper.html

Some more was found here

- http://www.freebsdsystems.com/handbook/sendmail.html

Masquerading explanations

- http://www.sendmail.org/m4/features.html

Explanation on the difference between email headers and envelopes

- http://www.primemail.com/permanentEmailAddress.hml

RedHat's recommendations for configuring sendmail

- http://www.redhat.com/docs/manuals/linux/RHL-8.0-Manual/ref-guide/s1-email-sendmail.html

Dynamic DNS - Hosting Your Website at Home

What is Dynamic DNS (DDNS)?

- http://www.technopagan.org/dynamic/

MiniDNS.net - DDNS Provider

- http://www.minidns.net

dynDNS.org - DDNS Provider

- http://www.dyndns.org

Ez-ipupdate DDNS Update Script

- http://gusnet.cx/proj/ez-ipupdate/

DDclient DDNS Update Script

- See the README for more information. Mirrored here.

Static DNS

What is DNS?

- http://www.technopagan.org/dynamic/

All you ever need to know about setting up DNS for a home / SOHO environment

- http://www.tldp.org/HOWTO/DNS-HOWTO.html

NTP Server

NTP Configurations

- http://www.eecis.udel.edu/~mills/ntp/servers.html

POP Mail Server

From the creators of the default Redhat POP server

- http://www.washington.edu/imap/

POP server compilation and configuration details

- http://www.openna.com/community/articles/security/v1.3-xml/imapop.html

Samba - Linux as a Windows File Server

The Samba Project

- http://www.samba.org

The Unoffical Samba HOWTO, by David Lechnyr

- http://hr.uoregon.edu/davidrl/samba.html

Simple Windows Workgroup Networking Tutorial

- http://www.azstarnet.com/~jemorrow/samba/implementation.html

SAMBA organization's description of creating a PDC

- http://us6.samba.org/samba/ftp/docs/htmldocs/Samba-PDC-HOWTO.html

Mandrake's resource page on Samba PDCs

- http://www.mandrakeuser.org/docs/connect/csamba6.html

Sharing a Windows hard drive from a Linux box (Windows steps to do this)

- http://playstation2-linux.com/download/cfyc/HOWTO_setup_samba.html

Sharing a Windows hard drive from a Linux box (Linux steps to do this)

- http://plug.org.in/pipermail/plug-mail/2002-June/004274.html

General Linux Resource Pages

Text Terminal HOWTO

- http://www.linuxpowered.com/archive/howto/Text-Terminal-HOWTO.html#toc14

Complete Linux Networking HOWTO

http://www.linux.uni-bayreuth.de/howtos/html/NET-3-HOWTO.html

Automount HOWTO with FAQs

- http://home.rosko.net/rosko/howto/en/mini/Automount.html

A very comprehensive guide to sudo

- http://unix.about.com/library/weekly/aa102500c.htm

Disk Partitioning

- http://www.linuxsa.org.au/tips/disk-partitioning.html

Network Monitoring

The MRTG homepage

- http://www.mrtg.org

My Other Sites

Simiya - Caribbean Art and Photos

- Simiya