Routing

The task of finding how to get a datagram to its destination is referred to as "routing". In fact, many of the details depend upon the particular implementation. However, some general things can be said.

First, it is necessary to understand the model on which IP is based. IP assumes that a system is attached to some local network. We assume that the system can send datagrams to any other system on its own network. (In the case of Ethernet, it simply finds the Ethernet address of the destination system, and puts the datagram out on the Ethernet.) The problem comes when a system is asked to send a datagram to a system on a different network. This problem is handled by gateways. A gateway is a system that connects a network with one or more other networks. Gateways are often normal computers that happen to have more than one network interface. For example, we have a Unix machine that has two different Ethernet interfaces. Thus, it is connected to networks 128.6.4 and 128.6.3. This machine can act as a gateway between those two networks. The software on that machine must be set up so that it will forward datagrams from one network to the other. That is, if a machine on network 128.6.4 sends a datagram to the gateway, and the datagram is addressed to a machine on network 128.6.3, the gateway will forward the datagram to the destination. Major communications centers often have gateways that connect a number of different networks. (In many cases, special-purpose gateway systems provide better performance or reliability than general-purpose systems acting as gateways. A number of vendors sell such systems.)

Routing in IP is based entirely upon the network number of the destination address. Each computer has a table of network numbers. For each network number, a gateway is listed. This is the gateway to be used to get to that network. Note that the gateway doesn't have to connect directly to the network. It just has to be the best place to go to get there. For example, at Rutgers, our interface to NSFnet is at the John von Neuman Supercomputer Center (JvNC). Our connection to JvNC is via a high-speed serial line connected to a gateway whose address is 128.6.3.12. Systems on net 128.6.3 will list 128.6.3.12 as the gateway for many off-campus networks. However, systems on net 128.6.4 will list 128.6.4.1 as the gateway to those same off-campus networks. 128.6.4.1 is the gateway between networks 128.6.4 and 128.6.3, so it is the first step in getting to JvNC.

When a computer wants to send a datagram, it first checks to see if the destination address is on the system's own local network. If so, the datagram can be sent directly. Otherwise, the system expects to find an entry for the network that the destination address is on. The datagram is sent to the gateway listed in that entry. This table can get quite big. For example, the Internet now includes several hundred individual networks. Thus, various strategies have been developed to reduce the size of the routing table. One strategy is to depend upon "default routes". Often, there is only one gateway out of a network. This gateway might connect a local Ethernet to a campus-wide backbone network. In that case, we don't need to have a separate entry for every network in the world. We simply define that gateway as a "default". When no specific route is found for a datagram, the datagram is sent to the default gateway. A default gateway can even be used when there are several gateways on a network. There are provisions for gateways to send a message saying "I'm not the best gateway -- use this one instead." (The message is sent via ICMP. See RFC 792.) Most network software is designed to use these messages to add entries to their routing tables. Suppose network 128.6.4 has two gateways, 128.6.4.59 and 128.6.4.1. 128.6.4.59 leads to several other internal Rutgers networks. 128.6.4.1 leads indirectly to the NSFnet. Suppose we set 128.6.4.59 as a default gateway, and have no other routing table entries. Now what happens when we need to send a datagram to MIT? MIT is network 18. Since we have no entry for network 18, the datagram will be sent to the default, 128.6.4.59. As it happens, this gateway is the wrong one. So, it will forward the datagram to 128.6.4.1. But it will also send back an error saying in effect: "to get to network 18, use 128.6.4.1". Our software will then add an entry to the routing table. Any future datagrams to MIT will then go directly to 128.6.4.1. (The error message is sent using the ICMP protocol. The message type is called "ICMP redirect.")

Most IP experts recommend that individual computers should not try to keep track of the entire network. Instead, they should start with default gateways, and let the gateways tell them the routes, as just described. However, this doesn't say how the gateways should find out about the routes. The gateways can't depend upon this strategy. They have to have fairly complete routing tables. For this, some sort of routing protocol is needed. A routing protocol is simply a technique for the gateways to find each other, and keep up to date about the best way to get to every network. RFC 1009 contains a review of gateway design and routing. However, rip.doc is probably a better introduction to the subject. It contains some tutorial material, and a detailed description of the most commonly-used routing protocol.

If you are going to physically separate your network into several networks, or are going to connect your network to a larger network such as the internet, you must be able to "route" packets from a system on your network to a system that isn't. This is accomplished via a physical device that connects the two networks (a router), and the use of a the "route" command under UNIX.

The route command:

route

the destination can be a default route. A default route is used to send all packets that your network does not know about through either a physical connection or a specified route.

The system stores the information in a routing table. You use the netstat -r command to print the routing table.

```
Routing Table:
Destination Gateway Flags Ref Use Interface
-------------------- -------------------- ----- ----- ---
--- ---------
192.168.35.0 192.168.35.13 U 2 3453 hme1
192.168.28.0 192.168.28.23 U 3 15809 hme0
224.0.0.0 192.168.28.23 U 3 0 hme0
default 192.168.35.1 UG 0 4
default 192.168.28.1 UG 0 5
127.0.0.1 127.0.0.1 UH 0 656704 lo0
```

This system is connected to two networks, but is not acting like a router. It has two default routes. The O/S will round/robin between the routes.

On many Unix systems you can create a virtual interface and place a different IP address on that interface.

Why, would you do this? For example, if multiple networks were sharing the same physical network hardware, but they were completely separate networks and there was no router between them, you can use a virtual network to connect them.

At work, my company was acquired by another company. We were relocated to a single location, but ran separate networks. Before we could merge the two networks, I wanted to put a machine on both networks, but I did not have a second ethernet card. So, I physically attached the two networks with a cross-over cable between the switches. Now, the two networks were physically one network. When I added the virtual interface, I could now see both networks from one machine. This was very helpful as we transitioned the 206.156.173.0/24 to the 192.168.20.0/24 network.

Two Networks:

Network A -- 192.168.28.0/24
Network B -- 206.156.173.0/24

My machine (solaris 8) was connected to Network A
192.168.28.76 mysun

I created a host entry for Network B
206.156.173.250 mysunb

I ran these commands:

```
# create the virtual interface
/sbin/ifconfig hme0:1 plumb
# start the interface
/sbin/ifconfig hme0:1 inet mysun-b up

# ifconfig hme0:1 plumb
# ifconfig -a
lo0: flags=1000849<up,loopback,running,multicast,ipv4> mtu 8232
index 1
inet 127.0.0.1 netmask ff000000
hme0: flags=1000843<up,broadcast,running,multicast,ipv4> mtu
1500 index 2
inet 192.168.28.76 netmask ffffff00 broadcast 192.168.28.255
ether 8:0:20:f8:e7:86
hme0:1: flags=1000842<broadcast,running,multicast,ipv4> mtu 1500
index 2
inet 0.0.0.0 netmask 0
# the interface is up, but does not have an IP address
```

```
# ifconfig hme0:1 inet mysun-b up
#: ifconfig -a
lo0: flags=1000849<up,loopback,running,multicast,ipv4> mtu 8232
index 1
inet 127.0.0.1 netmask ff000000
hme0: flags=1000843<up,broadcast,running,multicast,ipv4> mtu
1500 index 2
inet 192.168.28.76 netmask ffffff00 broadcast 192.168.28.255
ether 8:0:20:f8:e7:86
hme0:1: flags=1000843<up,broadcast,running,multicast,ipv4> mtu
1500 index 2
inet 206.156.173.178 netmask ffffff00 broadcast 206.156.173.255
```

The interface is up -- it used the default netmask and broadcast addresses.
This machine can see both networks --