FDDI

Fiber Distributed Data Interface (FDDI)

Standard is ANSI X3T9.5 . Topology is **ring with two counter rotating rings for reliability** with no hubs. Cable type is fiber-optic. Connectors are specialized. The media access method is token passing. The maximum length is 100 kilometers. The maximum number of nodes on the network is 500. **Speed is 100 Mbps.** FDDI is normally used as a backbone to link other networks. **A typical FDDI network can include servers, concentrators, and links to other networks.**

Devices called concentrators provide functions similar to hubs. Most concentrators use dual attachment station network cards but single attachment concentrators may be used to attach more workstations to the network.

FDDI token passing allows multiple frames to circulate around the ring at the same time. Priority levels of a data frame and token can be set to allow servers to send more data frames. Time sensitive data may also be given higher priority. The second ring in a FDDI network is a method of adjusting when there are breaks in the cable. The primary ring is normally used, but if the nearest downstream neighbor stops responding the data is sent on the secondary ring in attempt to reach the computer. Therefore **a break in the cable will result in the secondary ring being used.** There are two network cards which are:

1. Dual attachment stations (DAS) used for servers and concentrators are attached to both rings.
2. Single Attachment stations (SAS) attached to one ring and used to attach workstations to concentrators.

A router or switch can link an FDDI network to a local area network (LAN). Normally FDDI is used to link LANs together since it covers long distances.

IPX/SPX

IPX/SPX is a routable protocol and can be used for small and large networks. The following protocols are part of the IPX/SPX suite:

- SAP - Service Advertising Protocol packets are used by file and print servers to periodically advertise the address of the server and the services available. It works at the application, presentation, and session levels.
- NCP - NetWare Core Protocol provides for client/server interactions such as file and print sharing. It works at the application, presentation, and session levels.
- SPX - Sequenced Packet Exchange operates at the transport layer providing connection oriented communication on top of IPX.
- IPX - Internetwork Packet Exchange supports the transport and network layers of the OSI network model. Provides for network addressing and routing. It provides fast, unreliable, communication with network nodes using a connection less datagram service.
- RIP - Routing Information Protocol is the default routing protocol for IPX/SPX networks which operates at the network layer. A distance-vector algorithm is used to calculate the best route for a packet.
- ODI - Open Data-link Interface operates at the data link layer allowing IPX to work with any network interface card.

NetWare frame types

Novell NetWare 2.x and 3.x use Ethernet 802.3 as their default frame type. Novell NetWare 4.x networks use Ethernet 802.2 as their default frame type. If communication does not occur between two NetWare computers it is a good idea to check the netware versions of the two computers to be sure their frame types match. If the frame types do not match on an ethernet network, the computers cannot communicate.

NetBEUI

In order to properly describe NetBEUI, the transport protocol sometimes used for Microsoft networking, it is necessary to describe Microsoft networking in some detail and the various protocols used and what network layers they support.

NetBIOS, NetBEUI, and SMB are Microsoft Protocols used to support Microsoft Networking. The NetBIOS stack includes SMB, NetBIOS, and NetBEUI which are described in the table below. The following are parts of the Microsoft networking stack:

Name	Network Layer	Description
Redirector	Application	Directs requests for network resources to the appropriate server and makes network resources seem to be local resources.
SMB	Presentation	Server Message Block provides redirector client to server communication
NetBIOS	Session	Controls the sessions between computers and maintains connections.
NetBEUI	Transport, Network	Provides data transportation. It is not a routable transport protocol which is why NBT exists on large networks to use routable TCP protocol on large networks. This protocol may sometimes be called the NetBIOS frame (NBF) protocol.
NDIS and NIC driver	Data Link	NDIS allows several adapter drivers to use any number of transport protocols. The NIC driver is the driver software for the network card.

NetBIOS Extended User Interface (NetBEUI)

This is a separate protocol from NetBIOS. It supports small to medium networks providing transport and network layer support. It is fast and small and works well for the DOS operating system but NetBEUI **is not a routable protocol**.

Name Resolution

There are three methods of mapping NetBIOS names to IP addresses on small networks that don't perform routing:

1. IP broadcasting - A data packet with the NetBIOS computer name is broadcast when an associated address is not in the local cache. The host who has that name returns its address.

2. The lmhosts file - This is a file that maps IP addresses and NetBIOS computer names.
3. NBNS - NetBIOS Name Server. A server that maps NetBIOS names to IP addresses. This service is provided by the nmbd daemon on Linux.

System wide methods of resolving NetBIOS names to IP addresses are:

1. b-node - Broadcast node
2. p-node - Point-to-point node queries an NBNS name server to resolve addresses.
3. m-node - First uses broadcasts, then falls back to querying an NBNS name server.
4. h-node - The system first attempts to query an NBNS name server, then falls back to broadcasts if the nameserver fails. As a last resort, it will look for the lmhosts file locally.

NetBIOS name services use port 137 and NetBIOS session services use port 139. NetBIOS datagram service uses port 138.

To resolve addresses from names, a computer on a Microsoft network will check its cache to see if the address of the computer it wants to connect to is listed there. If not it sends a NetBIOS broadcast requesting the computer with the name to respond with its hardware address. When the address is received, NetBIOS will start a session between the computers. On larger networks that use routers, this is a problem since routers do not forward broadcasts, nor is NetBEUI a routable protocol. Therefore Microsoft implemented another method of resolving names with the Windows Internet Name Service (WINS). The following steps are taken to resolve NetBIOS names to IP addresses for H-node resolution on larger networks using TCP/IP (NBT):

1. NetBIOS name cache
2. WINS Server
3. NetBIOS broadcast
4. lmhosts file
5. hosts file
6. DNS server

For a more complete explanation of NetBIOS name resolution, WINS, and Windows networking in general, see the manuals in the Windows operating system section such as the "Windows TCP/IP Reference." Also a Windows Networking manual will be written for this section.

NetBIOS over TCP/IP (NBT)

Since NetBEUI is not a routable protocol, Microsoft implemented NBT for larger networks. NetBIOS messages are normally encapsulated in NetBEUI datagrams, but when using NBT, they are encapsulated in TCP/IP datagrams. The NBT protocol is defined by RFC 1001 and RFC 1002.

NWLink

NWLink is Microsoft's implementation of IPX/SPX. NWLink will act as a transport mechanism for NetBIOS similar to the use of TCP/IP described in the NBT section above. NWLink is normally used to support medium networks and may be used where NetWare servers are present.

Windows Internet Name Service (WINS)

WINS is the Microsoft implementation of NetBIOS name service. Samba on Linux can be used as a WINS server.

Computers configured to use WINS, when booted, contact the WINS name server and give the server their NetBIOS name and IP address. The WINS server adds the information to its database and it may send the information to other WINS servers on your network. When a computer that is configured to use WINS needs to get an address of another computer, it will contact the WINS server for the information. Without the use of a WINS server, NetBIOS will only be able to see computers on the unrouted sections of the local network. Does this mean a WINS server must exist in each routed section of the network? The answer is no. This is because WINS uses TCP/IP which is routable. Only one WINS server needs to exist on the network.

The Windows Networking Environment

A domain in a Microsoft networking environment refers to a collection of computers using user level security. It is not the same as the term domain used with regard to the domain name system (DNS). Domain related terms are:

- BDC - Backup Domain Controller is a backup for a PDC
- TLD - Top Level domain
- PDC - Primary Domain Controller is an NT server providing central control of user access permissions and accounts on a network.

AppleTalk Protocols

AppleTalk is the architecture used on with Apple brand computers and is a suite of protocols for networking Apple computers. Some of the protocols are:

- AppleShare - Works at the application layer to provide services.
- AFP - AppleTalk Filing protocol - Makes network files appear local by managing file sharing at the presentation layer.
- ATP - AppleTalk Transaction Protocol provides a Transport Layer connection between computers. Three transaction layers:
    - transaction requires (TREQ)
    - transaction response (TRESP)
    - transaction release (TREL)
- DDP - Datagram Delivery Protocol is a routable protocol that provides for data packet transportation. It operates at the network layer at the same level of the IP protocol.

The AppleTalk networking scheme puts computers into groups called zones. This is similar to workgroups on a Windows network.

Four Session layer protocols

- ASP - AppleTalk session protocol controls the starting and ending of sessions between computers called nodes. It works at the session level. The NBP, described below is used to get addresses from computer names. ATP is used at the transport level.
- ADSP - AppleTalk data stream protocol manages the flow of data between two established socket connections.
- ZIP - Zone information protocol used with RTMP to map zones. Routers use zone information tables (ZITs) to define network addresses and zone names.
- PAP - Printer access protocol manages information between workstations and printers.

Other Protocols

- NBP - Name-binding protocol translates addresses into names.
- AEP - AppleTalk echo protocol uses echoes to tell if a computer, or node, is available.
- RTMP - Routing table maintenance protocol is used to update routers with information about network status and address tables. The whole address table is sent across the network.
- ARUP - AppleTalk update routing is a newer version of RTMP.

System Network Architecture

System Network Architecture (SNA) by IBM is a suite of protocols mainly used with IBM mainframe and AS/400 computers. Two SNA protocols are:

- APPC - Advanced Peer-to-Peer Communications provides peer to peer services at the transport and session layer.
- APPN - Advanced Peer-to-Peer Networking supports the computer connections at the network and transport layers.

Microsoft produced the SNA Server so PC networks could connect with SNA networks.

SNA Layers

SNA has its own network model which is:

- Physical
- Data link - Uses protocols such as token-ring or Synchronous Data Link Control (SDLC).
- Path Control - Performs routing, division, and re-assembly of data packets.
- Transmission - Connection software
- Data flow - Prevents data overflows by monitoring and handling traffic
- Presentation - Handles interfaces to applications
- Transaction - Provides an interface for applications to use network services

SNA Network Devices

- host systems
- terminals
- Output devices
- Communications controllers
- Cluster controllers - Allow many devices to connect through them. They connect ot a host or communications controller.

SNA Network Categories

- Nodes
    - Type 2 - PCs, terminals and printers
    - Type 4 - Communications controllers
    - type 5 - Host computers used to manage the network
- Data links - Connection between combinations of hosts, cluster controllers, or nodes.

Possible SNA communications architectures

- SDLS - Synchronous Data Link Control
- BSC - Binary Synchronous Communication sends bits in frames which are timed sequences of data.
- Token-ring
- X.25
- Ethernet
- FDDI

SNA units

NAU - Network Addressable Units

- LU - Logical Units are ports that users use to access network resources
    - Type 1 - An interactive batch session
    - Type 2 - An IBM 3270 terminal
    - Type 3 - An IBM 3270 printer
    - Type 6.2 - A program to program session
    - Type 7 - An IBM 5250 family session
- PU - Physical Units are a network device used to communicate with hosts.
    - Type 2 - Cluster controllers
    - Type 3 - Front end process
    - Type 5 - Host communications software

SNA software components

- SSCP - Systems Services Control Point manages all resources in the host's domain.
- NCP - Network Control Program performs routing, session management tasks. It runs in the communications controller.

Other Transport Protocols

DECnet

DECnet from Digital Equipment Corporation is a suite of protocols which may be used on large networks that integrate mainframe and minicomputer systems. It is a routable protocol. DNA - Digital Network Architecture.

Data Link Control (DLC)

This protocol operates at the data link layer and is designed for communications between Hewlett-Packard network printers and IBM mainframe computers. This protocol is not routable.

Open Systems Interconnect (OSI)

A suite of protocols developed by the International Standards Organization (ISO) which corresponds with the layers of the OSI model. These protocols provide a number of application protocols for various functions. The OSI protocol stack may be used to connect large systems. OSI is a routable transport protocol.

Network Routing

Simple Networking Routing and Routers

This section will explain routing in simple terms with some simple standard rules. There may be exceptions to these rules, but for introductory purposes we will keep the first example simple. Please be aware, that the examples in this section are working examples, but more complexity may be added when a larger network is considered, and multiple data routes become available.

Each network interface card (NIC) has a specific address which is an IP address or number. When data is sent between two computers, the data must be sent in a package that has the address of the intended receiver (IP) on it. It is like an envelope (ethernet) with the sender's and recipient's address on it. There is somewhat of a difference, however. When the computer intends to send a packet, it first checks its routing table to see if the intended data must be sent through a gateway. Many computers only have a simple routing table, which is built from the network mask and the gateway information entered, when you set your computer up to do networking. The computer, when set up for networking, must be assigned an IP address, netmask, and default gateway. This may be done manually or done automatically using Dynamic Host Configuration Protocol (DHCP) to assign this information to the computer when it boots. DCHP is described in another section. If the computer determines that the packet must be sent to a gateway, it puts it in a special packet (ethernet) for that gateway, with the actual recipient's address wrapped inside.

In the above paragraph, data packets are equated to a letter with an envelope. For this type of thinking, the envelope would be similar to the ethernet, SLIP, or PPP packet which encapsulates the IP packet. The IP packet and its encapsulated data would similar to a letter. Here's generally what happens when a package is sent:

> The sending computer checks the IP part of the package to see the sender's IP address, and based on the address and instructions in its routing table will do one of the following:

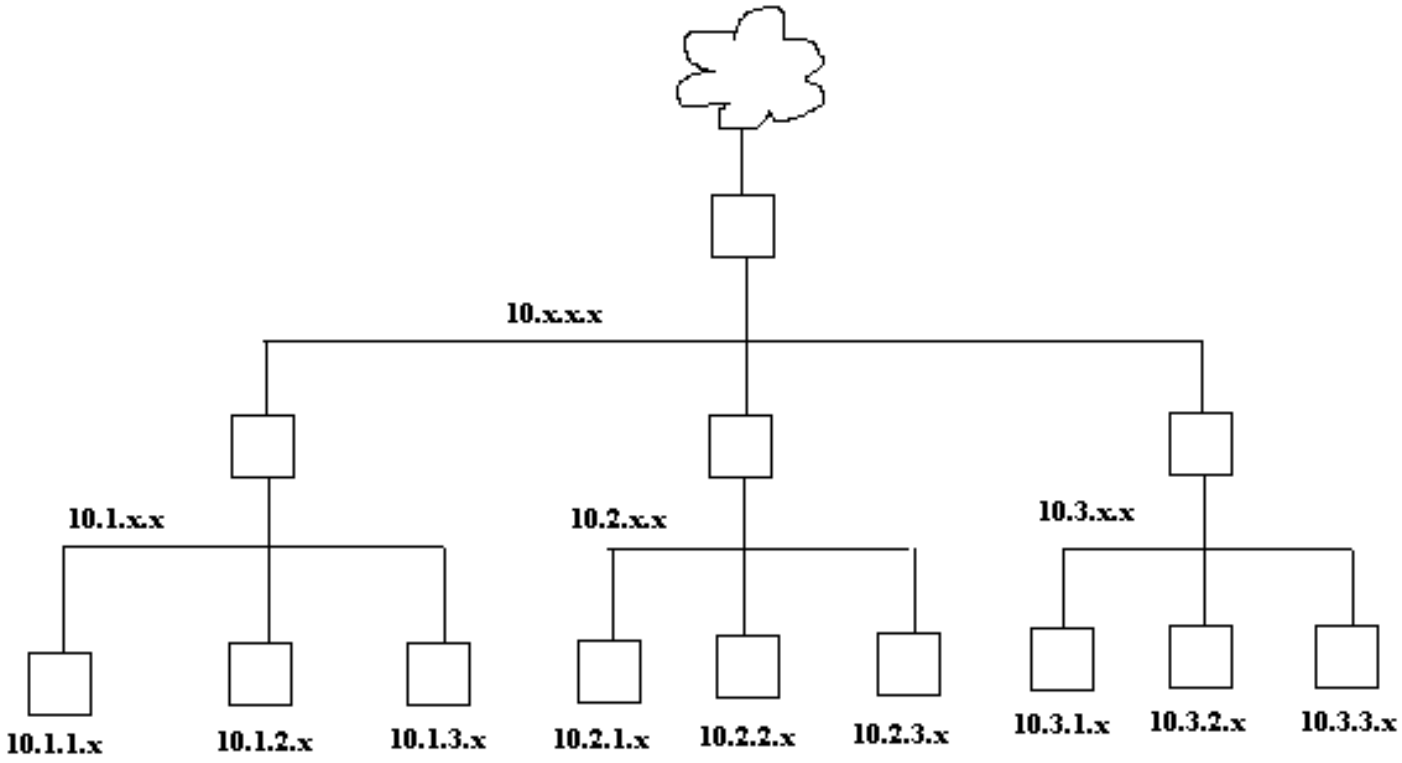
1. Send the packet to the ethernet address of the intended recipient. The following will happen:
    1. The ethernet card on the receiving computer will accept the packet.
    2. The other network levels (IP, TCP) will open the packet and use it according to filtering and other programming instructions.
2. Send the packet to the ethernet address of a router, depending on the instructions in the routing table.
    1. The ethernet card on the router will accept the packet.
    2. The IP level of the router will look at the packet's IP address and determine according to its routing table where to send the packet next. It should send it to another router or to the actual recipient.
    3. The router will encapsulate the IP packet in another ethernet packet with the ethernet address of the next router or the intended recipient.
    4. Router hops will continue until the packet is sent on a network where the intended recipient is physically located unless the packet expires.
    5. The ethernet card on the receiving computer will accept the packet.
    6. The other network levels (IP, TCP) will open the packet and use it according to filtering and other programming instructions.

Lets say you enter an IP address of 10.1.20.45 and a netmask of 255.255.0.0. This means you are on the network 10.1.0.0 (I show it as 10.1.x.x, the X's mean don't care conditions). The machine's IP address and netmask, together define the network, that it's NIC is on. Therefore any machine that fits in the address range provided under 10.1.x.x can be accessed directly from your NIC, and any that are not in this number range, such as 10.3.34.67 cannot be accessed directly and must be sent to a gateway machine since it is on another network. Typically most machines will use their netmask to make this determination which means if the address does not match their known network, the package will be sent to that machine's default gateway in a special package meant for a router. It works similar to a post office. When you send a letter in your town, you put it in the local slot. It can be delivered to someone else in your town (network), but if you are sending to another town (network), you put the letter in the out of town slot (default gateway), then the mail personnel put it in a special container or box and send it to a main town (gateway), which then decides where to send it based on its address. Although this simple network and default gateway may be common, specific computers or gateways can have much more complex rules for routing that allow exceptions to this example.

Please be aware that in order to be forwarded, data packets must be addressed to a router. They cannot just be sent to the recipient's address out to a network. The router does not pick packets off the network and forward them. If a packet is sent on a network and a valid recipient is not on that network, there will be no response. This will be demonstrated in the next section where a subnetwork will be described.

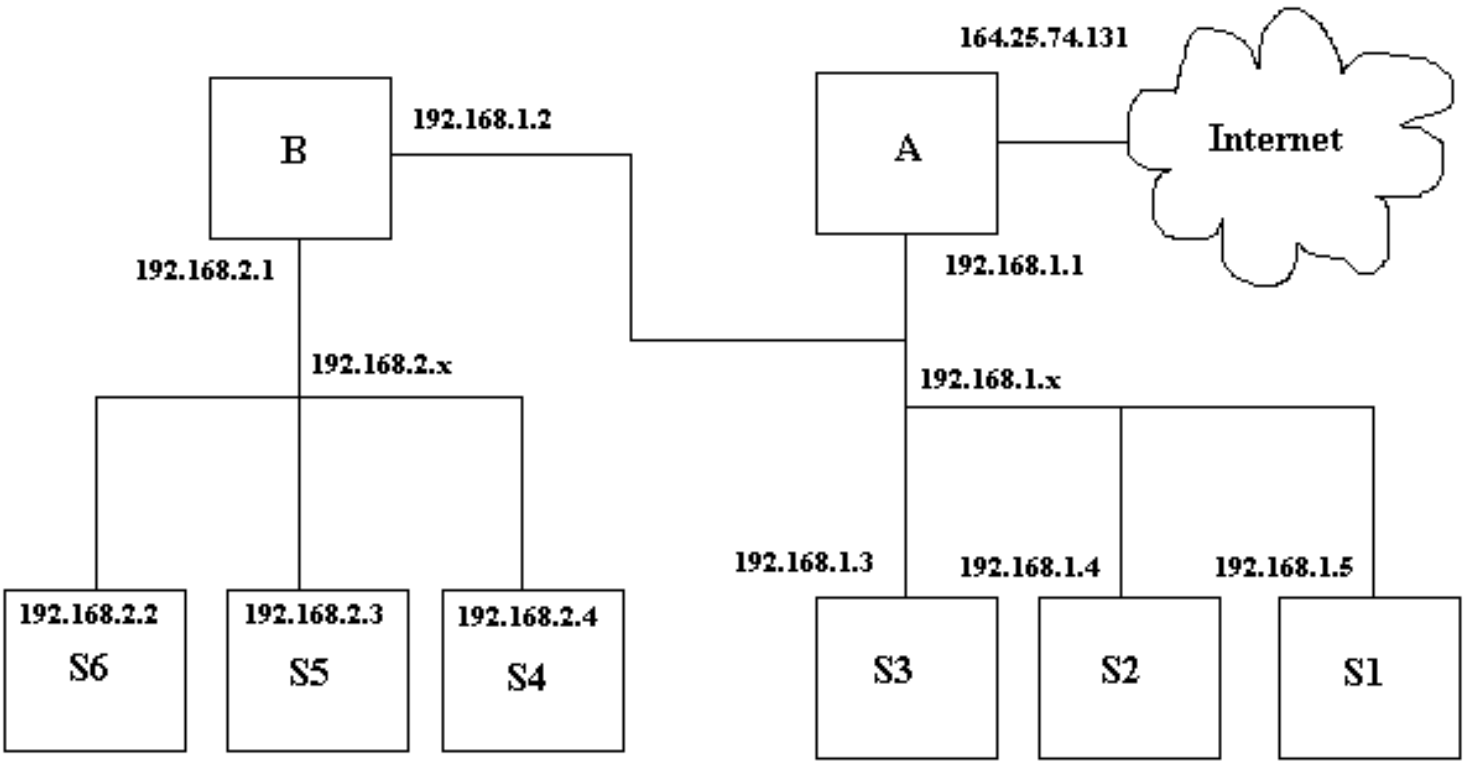
To keep routing simple, most networks are structured as shown below. Generally, the higher networks are 10.x.x.x, then the next are 10.0-254.x.x, then 10.0-254.0-254.x. The number 10 is used as an example Class A network. This numbering scheme keeps routing simple and is the least confusing but networks can be set up in other ways. In the diagram below, only gateways and their networks are shown.

Typical Network Structured Addressing Scheme



In my simple network example below I vary from convention and make network 192.168.2.x be below network 192.168.1.x. causing traffic between the internet and 192.168.2.x to go through the network 192.168.1.x. Normally the network 192.168.1.x would be 192.168.x.x, but this will show you that there can be many variants that will work as long as you have thought your layout through well, and set your routing tables up in your gateways correctly.

A Small Network with Two Gateways



The boxes labeled A and B must be gateways or routers in order for anyone on networks 192.168.2.x or 192.168.1.x to talk to any other network or internet. The boxes labeled S1 through S6 are stations which could be workstations or servers providing services like BOOTP, DHCP, DNS, HTTP, and/or file sharing such as NFS or Samba. The gateways may also provide these services. These stations may combine any combination of server or workstation function. The reasons for putting the various services on separate machines is because of security concerns and the ability of a given machine to handle specific demand. Typically, the computer that is connected directly to the internet, would be a firewall and provide no other services for security reasons. For example, it is not a good idea to provide TFTP services on a machine that you want to have high security. This is why, depending on the security needs of the company or individual along with the relative amount of each service to be provided, various servers are set up with limited functionality.

The machine S6 in the diagram above has the following characteristics:

```
IP Address:   192.168.2.2
Network:      192.168.2.0
Netmask:      255.255.255.0
Gateway:      192.168.2.1
```

In Linux, the "ifconfig" command is used to configure the NIC and the command "route" is used to set up routing tables for that machine. Please note that in Redhat Linux, the GUI interface programs "netconf" and "linuxconf" may be used to set this up also. These GUI interface programs will set these changes up to be permanent by

writing them to files that are used to configure network information. Changes made with "route" without adding the changes to permanent files will no longer be valid when you reboot the machine. The command "ifconfig eth0 192.168.2.2 netmask 255.255.255.0" will set the NIC card up with its address and network number. You can type "netconfig", then select "basic host information" and do the same thing. The command "route add -net default gw 192.168.2.1 dev eth0" will add the route required for this computer for its gateway. This can be done using "ifconf" by selecting "routing and gateways" and "defaults", then setting the address of the default gateway, and enabling routing. Please be aware that various versions of Linux have different means of storing and retrieving network and routing information and you must use the tools that come with your system or learn it well enough to determine what files to modify. On Redhat 6.1 the file "/etc/sysconfig/static-routes" can be modified to make your route changes permanent, but this does not apply to your default route. Other files are "/etc/sysconfig/routed" and "/etc/sysconfig/network". Other files include "/etc/gateways", "/etc/networks", "/proc/net/route", "/proc/net/rt_cache", and "/proc/net/ipv6_route". The file "/etc/sysconfig/network-scripts" is a script file that controls the network setup when the system is booted.

If you type "route" for this machine, the routing table below will be displayed:

Destination	Gateway	Genmask	Flags	Metric	Ref	Use	Iface
192.168.2.2	*	255.255.255.255	UH	0	0	0	eth0
192.168.2.0	*	255.255.255.0	U	0	0	0	eth0
127.0.0.0	*	255.0.0.0	U	0	0	0	lo
default	192.168.2.1	0.0.0.0	UG	0	0	0	eth0

Here is a simple explanation of routing tables and their purpose. All computers that are networked have a routing table in one form or another. A routing table is a simple set of rules that tell what will be done with network packets. In programming language it is easiest to think of it as a set of instructions, very similar to a case statement which has a "default" at its end. If can also be thought of as a series of if..then..elseif..then..else statements. If the lines above are labeled A through C and a default (the last line), an appropriate case statement is: (Don't count the header line)

```
switch(address){
    case A: send to me;break;
    case B: send to my network;break;
    case C: send to my local interface;break;
    default: send to gateway 192.168.2.1
```

An appropriate if statement is:

```
if (address=me) then send to me;
elseif (address=my network) then send to my network;
elseif (address=my local) then send to my local interface;
else send to my gateway 192.168.2.1;
```

In everyday terms this is similar to a basic decision process. Imagine you are holding a letter. If it is addressed to

you, you keep it, if it is addressed to someone in your town, you drop it in the local slot at the post office, but if it is addressed to someone out of town, you would drop it in the out of town slot.

Note how the routing table is arranged. It is arranged from the most specific to the least specific. Therefore as you go down the table, more possibilities are covered. You will notice the first Genmask is 255.255.255.255 and the last is 0.0.0.0. There can be no doubt that the last line is the default. The genmasks between the start and the end have a decreasing number of least significant bits set.

The above default routing table may be added manually with the command:

```
route add -net default gw 192.168.2.1 dev eth0
```

The routing table for machine B, the gateway for the network 192.168.2.0 is as follows.

Destination	Gateway	Genmask	Flags	Metric	Ref	Use	Iface
192.168.2.1	*	255.255.255.255	UH	0	0	0	eth0
192.168.1.2	*	255.255.255.255	UH	0	0	0	eth1
192.168.2.0	192.168.2.1	255.255.255.0	UG	0	0	0	eth0
192.168.2.0	*	255.255.255.0	U	0	0	0	eth0
192.168.1.0	192.168.1.2	255.255.255.0	UG	0	0	0	eth1
192.168.1.0	*	255.255.255.0	U	0	0	0	eth1
127.0.0.0	*	255.0.0.0	U	0	0	0	lo
default	192.168.1.1	0.0.0.0	UG	0	0	0	eth0

The Iface specifies the card where packets for this route will be sent. The address of eth1 is 192.168.1.2 and eth0 is 192.168.2.1. The NIC card addresses could have easily been switched. Line 1 (above) provides for the eth0 address, while line 2 provides for the address of eth1. Lines 3 and 4 are the rules for traffic going from network 192.168.1.0 to network 192.168.2.0 which will be sent out on NIC eth0. Lines 5 and 6 are the rules for traffic going from network 192.168.2.0 to network 192.168.1.0 which will be sent out NIC eth1. This may seem confusing, but please note the first value on lines 3 and 4 is 192.168.2.0 which the header indicates as the destination of the packet. Don't think of it as source! The last line is the default line which specifies that any packet not on one of the networks 192.168.1.0 or 192.168.2.0 will be sent to the gateway 192.168.1.1. This is how the internet access can be attained, though IP masquerading will probably be used. The flags above mean the following:

- U - Route is up
- H - Target is a host
- G - Use gateway

There are other flags, you can look up by typing "man route". Also the metric value above, indicating the distance to the target, is not used by current Linux kernels but may be needed by some routing daemons. Please note that if route knows the name of the gateway machine, it may list its name rather than the IP address. The same is true for defined networks. Networks may be defined in the file "/etc/networks" as in the example:

```
net1 192.168.1.0
net2 192.168.2.0
```

The routing table above can be set up with the following commands.

```
route add -net 192.168.2.0 netmask 255.255.255.0 gw 192.168.2.1 dev eth0
route add -net 192.168.1.0 netmask 255.255.255.0 gw 192.168.1.2 dev eth1
```

Again be aware that you are specifying destination networks here and the ethernet device and address the data is to be sent on.
In Redhat Linux this can be specified using "netconf" by selecting "routing and gateways" and "other routes to networks" and entering the following:

Network	Netmask	Gateway
192.168.2.0	255.255.255.0	192.168.2.1
192.168.1.0	255.255.255.0	192.168.1.2

Alternatively in Redhat Linux, you can add the following two lines to the file "/etc/sysconfig/static-routes":

```
eth0 net 192.168.2.0 netmask 255.255.255.0 gw 192.168.2.1
eth1 net 192.168.1.0 netmask 255.255.255.0 gw 192.168.1.2
```

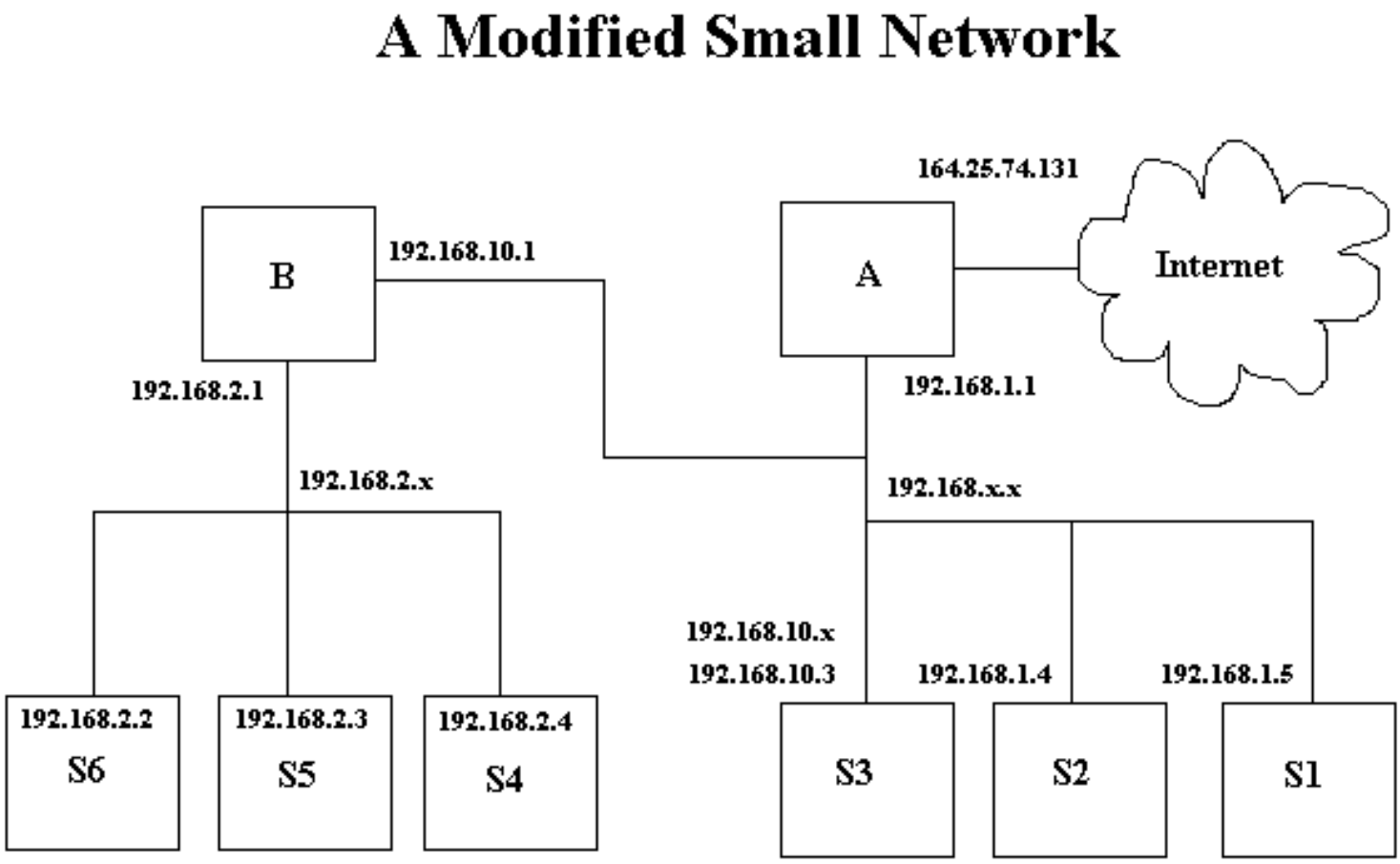
The commands to delete the above routes with route are:

```
route del -net 192.168.2.0 netmask 255.255.255.0 gw 192.168.2.1 dev eth0 route del -net 192.168.1.0
netmask 255.255.255.0 gw 192.168.1.2 dev eth1
```

Be aware, the program route is very particular on how the commands are entered. Even though it may seem that you entered them as the man page specifies, it will not always accept the commands. I don't know if this is a bug or not, but if you enter them as described here with the network, netmask, gateway, and device specified, it should work. The slightest misnomer in network name, netmask, gateway, device, or command syntax and the effort will fail.

More Complex Networking Routing

Now let's modify the small network in the example in the previous section. The 192.168.1.x network is changed to 192.168.x.x and gateway B's address is changed to 192.168.10.1. All the netmasks on the computers on the 192.168.x.x network are modified to 255.255.0.0 to accommodate the change, except machine S3 which keeps the netmask 255.255.255.0 and changes its address to 192.168.10.3. This effectively puts S3 on a different network than S2 and S1, it no longer believes it can talk directly to them and must talk to gateway B to talk to them. It can't even talk to gateway A anymore since it can't address it directly. Machines S1, S2, and A are not on network 192.168.10.0, their addresses are 192.168.1.*. S1 and S2 can talk to S3, but S3 will not be able to respond unless it utilizes gateway B.



Please be aware, in the example in the previous section, that gateway A was aware of gateway B. If it were not, no messages could have been transmitted from the internet to the 192.168.2.0 network. In this example, gateway A knows nothing about gateway B, and as far as it's concerned, the network 192,168.2.0 is part of 192.168.0.0 and there is no gateway between them. Gateway B, does know about gateway A and is using that gateway as its default gateway. Therefore if S1 and S2 use gateway A for their default gateway, they will not be able to talk to S4, 5, or 6 unless their routing table is modified. S1 and S2 will be able to talk to S3, however, assuming S3 is using gateway B.

Here is a listing of machine S1's routing table, using gateway A as default and no other routes.

Destination	Gateway	Genmask	Flags	Metric	Ref	Use	Iface
192.168.1.5	*	255.255.255.255	UH	0	0	0	eth0
192.168.0.0	*	255.255.0.0	U	0	0	0	eth0
127.0.0.0	*	255.0.0.0	U	0	0	0	lo
default	192.168.1.1	0.0.0.0	UG	0	0	0	eth0

Here it is modified to let it use network 192.168.2.0.

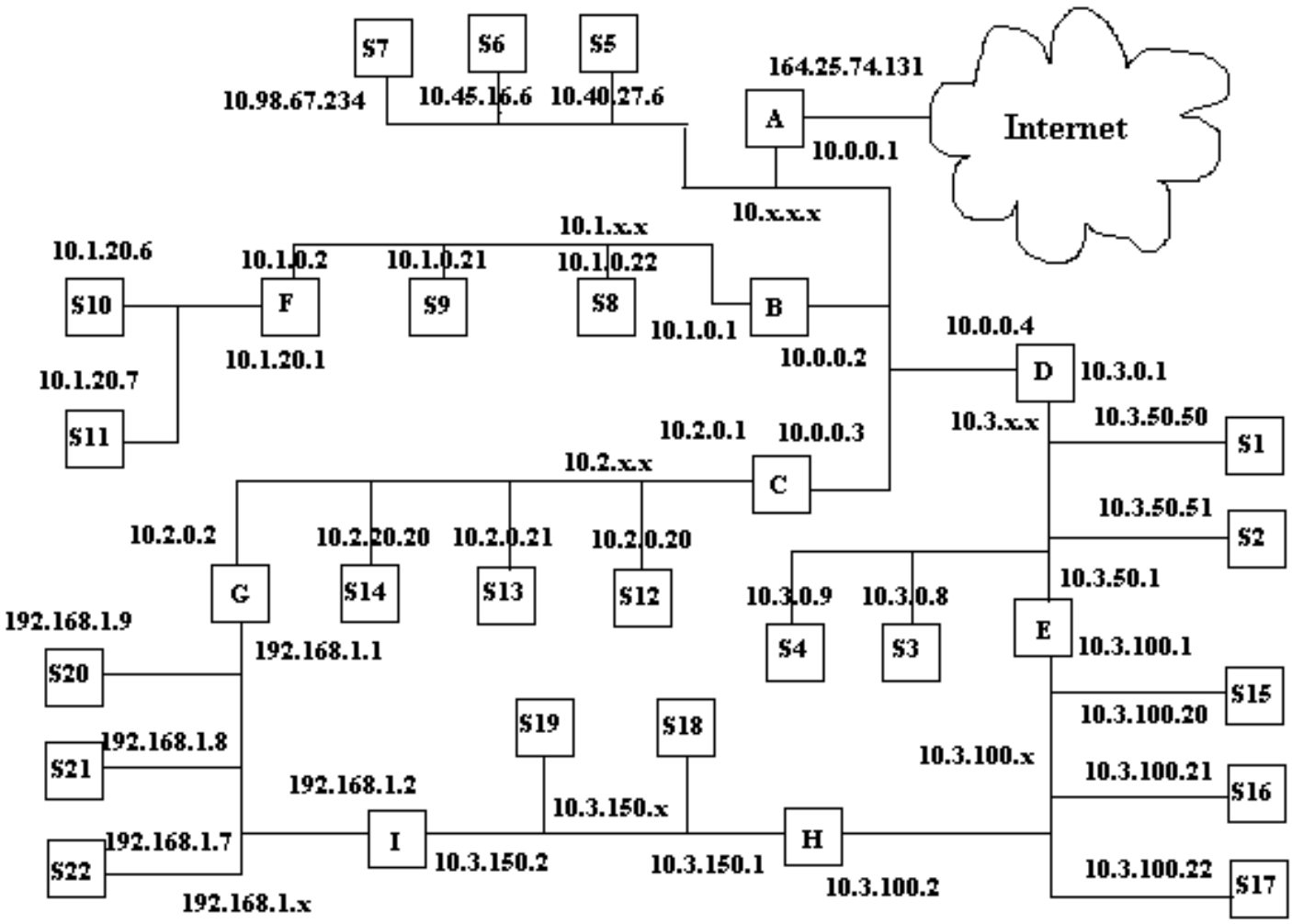
Destination	Gateway	Genmask	Flags	Metric	Ref	Use	Iface
192.168.1.5	*	255.255.255.255	UH	0	0	0	eth0
192.168.0.0	*	255.255.0.0	U	0	0	0	eth0
192.168.2.0	192.168.10.1	255.255.255.0	UG	0	0	0	eth0
192.168.2.0	*	255.255.255.0	U	0	0	0	eth0
127.0.0.0	*	255.0.0.0	U	0	0	0	lo
default	192.168.1.1	0.0.0.0	UG	0	0	0	eth0

It specifies the gateway B, 192,168.10.1 to be used if the destination is 192.168.2.x.

The figure below shows an ethernet network with bus topology excluding the hubs. It is a large Class A network with many subnetworks. The machines labeled A through D are routers or potential routers and each have two network interface cards(NIC). These machines may be called gateways since their function is to be a gate to another location. Each card has a valid address on its own network or subnetwork. The table below lists each gateway, and each NIC address and associated network.

Gateway	eth0	eth0 network	eth1	eth1 network
A	10.0.0.1	10.x.x.x	164.25.74.131	Internet
B	10.0.0.2	10.x.x.x	10.1.0.1	10.1.x.x.
C	10.0.0.3	10.x.x.x	10.2.0.1	10.2.x.x.
D	10.0.0.4	10.x.x.x	10.3.0.1	10.3.x.x.
E	10.3.50.1	10.3.x.x	10.3.100.1	10.3.100.x.
F	10.1.0.2	10.1.x.x	10.1.20.1	10.1.20.x.
G	10.2.0.2	10.2.x.x	192.168.1.1	192.168.1.x.
H	10.3.100.2	10.3.100.x	10.3.150.1	10.3.150.x.
I	10.3.150.2	10.3.150.x	192.168.1.2	192.168.1.x.

Large Class A Network



In this figure, there are 9 gateways. which are labeled A through I. There are multiple paths between several networks. The possible paths between networks 10.1.100.x and 192.168.1.x can be through gateways E, D, C, then G (E-D-C-G) or through gateways H-I. The path from 10.3.100.x ot 10.1.20.x can be E-D-B-F or H-I-G-C-B-F. Obviously there are ways to set the routing paths up that may not be fully efficient. In this type of network, the administrator must give careful thought to the setup of the routing tables in their gateways. It would be easy to set up an infinite packet route loop in this network where some packets may go in circles from router to router. Here's how I would route for this network.

The below table lists each network and their default router.

Network	Default Router
10.3.100.x	E
10.3.150.x	H
192.168.1.x	G
10.1.20.x	F

10.1.x.x	B
10.2.x.x	C
10.3.x.x	D
10.x.x.x	A

The router, I, is not used as a default router for any network.

The table below lists an abbreviated route table for each gateway.

Router	Destination	Gateway
A	192.168.1.x	C
	10.1.x.x	B
	10.2.x.x	C
	10.3.x.x	D
	10.x.x.x	10.0.0.1
	default	internet
B	10.1.20.x	F
	10.1.x.x	10.1.0.1
	default	A
C	192.168.1.x	G
	10.2.x.x	10.2.0.1
	default	A
D	10.3.150.x	E
	10.3.100.x	E
	10.3.x.x	10.3.0.1
	default	A
E	192.168.1.x *	H
	10.3.150.x	H
	10.3.100.x	10.3.100.1
	default	D
F	10.1.20.x	10.1.20.1
	default	B
G	10.3.100.x *	I
	192.168.1.x	192.168.1.1
	10.3.150.x *	I
	default	C
H	192.168.1.x	I
	10.3.100.x	10.3.100.2

	10.3.150.x	10.3.150.1
	default	E
I	10.3.100.x	H
	192.168.1.x	192.168.1.2
	10.3.150.x	10.3.150.2
	default	G

The destinations with '*' indicate destinations that shorten the normal route path through network 10.3.150.x.

Also in this network since there are multiple possible paths, dynamic routing can be used to provide alternate routing, if one router goes down.

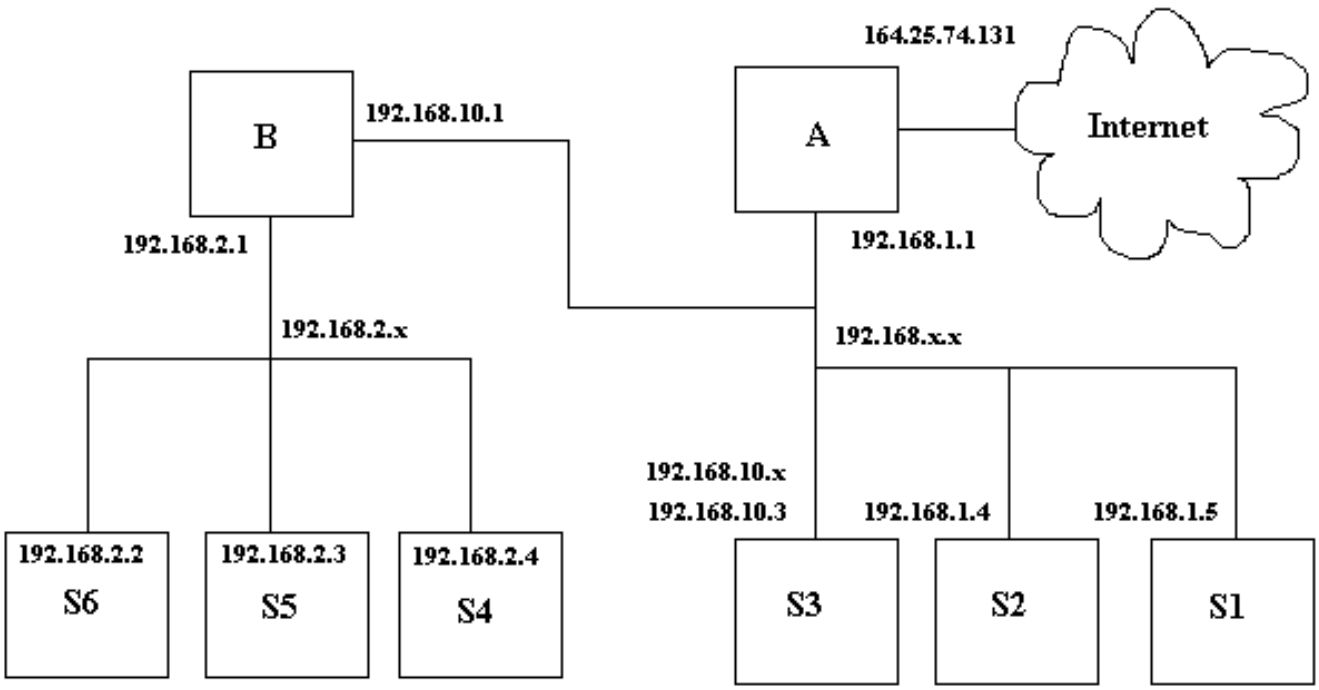
IP Masquerading

IP masquerading is a form of network address translation (NAT) which allows internal computers with no known address outside their network, to communicate to the outside. It allows one machine to act on behalf of other machines. It's similar to someone buying stocks through a broker (without considering the monetary transaction). The person buying stocks, tells the broker to buy the stocks, the broker gets the stocks and passes them to the person who made the purchase. The broker acts on behalf of the stock purchaser as though he was the one buying the stock. No one who sold the stock knew or cared about whether the broker was buying for himself or someone else.

Please DO NOT confuse routers with firewalls and the performance of IP masquerading. The commands that allow IP masquerading are a simple form of a firewall, however routing is a completely different function, as described previously. Setting a computer up to act as a router is completely different than setting up a computer to act as a firewall. Although the two functions are similar in that the router or firewall will act as a communication mechanism between two networks or subnets, the similarity ends there. A computer can be either a router or a firewall, but not both. If you set up a computer to act as both a router and a firewall, you have defeated the purpose of your firewall!

If you refer to the diagram below, the machines on network 192.168.2.x will obtain services through gateway B using IP masquerading, when gateway B is setup properly. What basically happens when IP masquerading is set up on gateway B is described in the following example. If machine S6 tries to ping S2, its ping packages will be wrapped in a package for its default gateway, gateway B, because S6 knows by its netmask that S2 in on another network. When gateway B receives the packages from S6, it converts them to ping packages as though they were sent from itself and sends them to S2. As far as S2 can tell, gateway B has pinged it. S2 receives the packages and responds to gateway B. Gateway B then converts the packages to be addressed to S6 and sends them. This is why it is called IP masquerading, since gateway B masquerades for machines S4, S5, and S6. Machines S1 through S3 and gateway A cannot initiate any communication with S4 through S6. In fact they have no way to know that those machines even exist!

A Modified Small Network



IP masquerading allows internal machines that don't have an officially assigned IP addresses to communicate to other networks and especially the internet. In Linux, IP masquerading support is provided by the kernel. To get it to work you must do essentially three things:

1. Be sure the kernel has support for IP masquerading.
2. Be sure modules needed for support are loaded into the kernel.
3. Set up the firewall rules.

For complete information on the setup of IP masquerading, see the following Linux how-tos:

- IPCHAINS-HOWTO
- Firewall-HOWTO
- IP-Masquerade-HOWTO

Some of the information in this section is based on these how-tos. This section summarizes and puts in simple steps some of the items you will be required to perform to set up IP masquerading. It is not a replacement for the Linux how to documents, but a complement to them by giving an overview of what must be done. You may access the howtos from one of the websites listed in the Linux websites section. The Linux Documentation Project or Metalab's Index of Linux publications will have copies if these howtos.

To set up IP masquerading in Linux you must first be sure your kernel supports IP masquerading with the following options set (This is for a 2.2.x kernel or higher):
Prompt for development and/or incomplete code/drivers (CONFIG_EXPERIMENTAL) [Y/n/?]- YES
Enable loadable module support (CONFIG_MODULES) [Y/n/?] - YES
Networking support (CONFIG_NET) [Y/n/?] - YES
Packet socket (CONFIG_PACKET) [Y/m/n/?] - YES
Kernel/User netlink socket (CONFIG_NETLINK) [Y/n/?] - YES
Routing messages (CONFIG_RTNETLINK) [Y/n/?] - NO
Network firewalls (CONFIG_FIREWALL) [Y/n/?] - YES
TCP/IP networking (CONFIG_INET) - YES
IP: advanced router (CONFIG_IP_ADVANCED_ROUTER) [Y/n/?] - NO
IP: verbose route monitoring (CONFIG_IP_ROUTE_VERBOSE) [Y/n/?] - YES
IP: firewalling (CONFIG_IP_FIREWALL) [Y/n/?] - YES
IP: firewall packet netlink device (CONFIG_IP_FIREWALL_NETLINK) [Y/n/?] - YES
IP: always defragment (required for masquerading) (CONFIG_IP_ALWAYS_DEFRAG) [Y/n/?] - YES
IP: masquerading (CONFIG_IP_MASQUERADE [Y/n/?] - YES
IP: ICMP masquerading (CONFIG_IP_MASQUERADE_ICMP) [Y/n/?] - YES
IP: masquerading special modules support (CONFIG_IP_MASQUERADE_MOD) [Y/n/?] - YES
IP: ipautofw masquerade support (EXPERIMENTAL) (CONFIG_IP_MASQUERADE_IPAUTOFW) [Y/n/?] - NO
IP: ipportfw masq support (EXPERIMENTAL) (CONFIG_IP_MASQUERADE_IPPORTFW) [Y/n/?] - YES
IP: ip fwmark masq-forwarding support (EXPERIMENTAL) (CONFIG_IP_MASQUERADE_MFW) [Y/m/n/?] - NO
IP: optimize as router not host (CONFIG_IP_ROUTER) [Y/n/?] - YES
IP: GRE tunnels over IP (CONFIG_NET_IPGRE) [N/y/m/?] - NO
IP: TCP syncookie support (not enabled per default) (CONFIG_SYN_COOKIES) [Y/n/?] - YES
Network device support (CONFIG_NETDEVICES) [Y/n/?] - YES
Dummy net driver support (CONFIG_DUMMY) [M/n/y/?] - YES
/proc filesystem support (CONFIG_PROC_FS) [Y/n/?] - YES

These are the kernel options you need for IP Masquerade. You will need to select other options for your specific hardware and network setup. Read the IP masquerade and kernel howtos for more information. You may also want the section about how to compile the Linux kernel on the Linux User's Guide in the Linux section of this documentation.

Create the following text and place it in a file "/etc/rc.d/rc.firewall". This will load your needed modules into your kernel and set up your basic firewall rules. If you copy the file from this page, be sure to remove carriage returns when you get it into Linux or it may not work properly.

```
# rc.firewall - Initial SIMPLE IP Masquerade setup for 2.0.x kernels using IPFWADM
#
# Load all required IP MASQ modules
#
#   NOTE:  Only load the IP MASQ modules you need.  All current available IP MASQ
modules
#          are shown below but are commented out from loading.

# Needed to initially load modules
#
/sbin/depmod -a

# Supports the proper masquerading of FTP file transfers using the PORT method
#
/sbin/modprobe ip_masq_ftp

# Supports the masquerading of RealAudio over UDP.  Without this module,
#  RealAudio WILL function but in TCP mode.  This can cause a reduction
#  in sound quality
#
#/sbin/modprobe ip_masq_raudio

# Supports the masquerading of IRC DCC file transfers
#
/sbin/modprobe ip_masq_irc

# Supports the masquerading of Quake and QuakeWorld by default.  This modules is
#   for for multiple users behind the Linux MASQ server.  If you are going to play
#   Quake I, II, and III, use the second example.
#
#Quake I / QuakeWorld (ports 26000 and 27000)
#/sbin/modprobe ip_masq_quake
#
#Quake I/II/III / QuakeWorld (ports 26000, 27000, 27910, 27960)
# /sbin/modprobe ip_masq_quake ports=26000,27000,27910,27960

# Supports the masquerading of the CuSeeme video conferencing software
#
#/sbin/modprobe ip_masq_cuseeme

#Supports the masquerading of the VDO-live video conferencing software
#
#/sbin/modprobe ip_masq_vdolive

#CRITICAL:  Enable IP forwarding since it is disabled by default since
#
#          Redhat Users:  you may try changing the options in /etc/sysconfig/network
from:
```

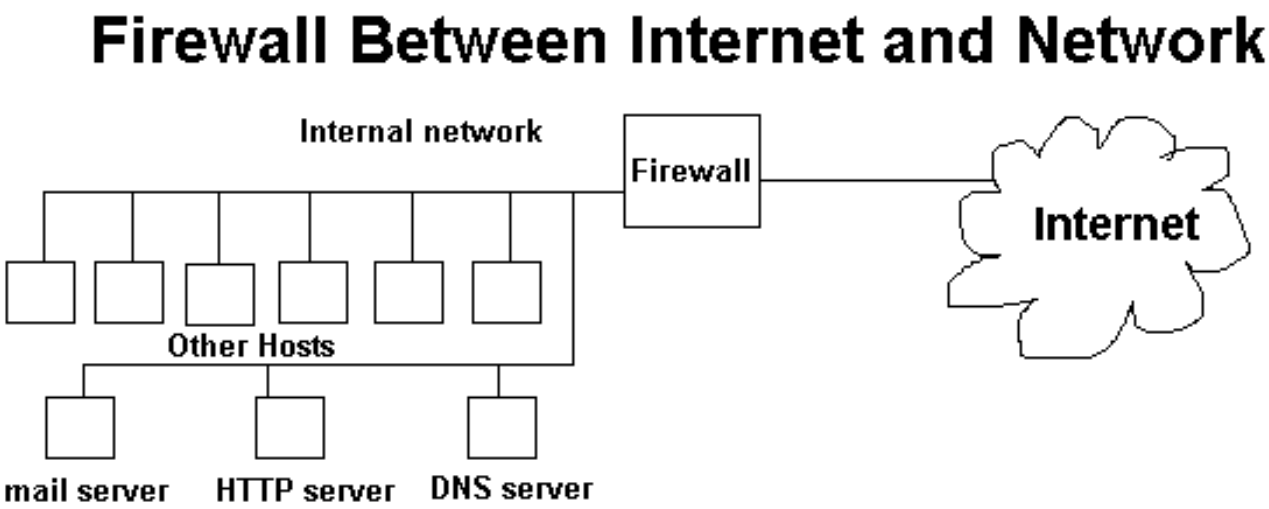
```
#
#                       FORWARD_IPV4=false
#                             to
#                       FORWARD_IPV4=true
#
echo "1" > /proc/sys/net/ipv4/ip_forward

# Dynamic IP users:
#
#   If you get your Internet IP address dynamically from SLIP, PPP, or DHCP, enable
this following
#       option.  This enables dynamic-ip address hacking in IP MASQ, making the life
#       with DialD, PPPd, and similar programs much easier.
#
echo "1" > /proc/sys/net/ipv4/ip_dynaddr

# MASQ timeouts
#
#   2 hrs timeout for TCP session timeouts
#  10 sec timeout for traffic after the TCP/IP "FIN" packet is received
#  160 sec timeout for UDP traffic (Important for MASQ'ed ICQ users)
#
/sbin/ipchains -M -S 7200 10 160

# DHCP:  For people who receive their external IP address from either DHCP or BOOTP
#        such as ADSL or Cablemodem users, it is necessary to use the following
#        before the deny command.  The "bootp_client_net_if_name" should be replaced
#        the name of the link that the DHCP/BOOTP server will put an address on to?
#        This will be something like "eth0", "eth1", etc.
#
#        This example is currently commented out.
#
#
/sbin/ipchains -A input -j ACCEPT -i eth1 -s 0/0 67 -d 0/0 68 -p udp

# Enable simple IP forwarding and Masquerading
#
#  NOTE:  The following is an example for an internal LAN address in the 192.168.0.x
#         network with a 255.255.255.0 or a "24" bit subnet mask.
#
#         Please change this network number and subnet mask to match your internal
LAN setup
#
/sbin/ipchains -P forward DENY
/sbin/ipchains -A forward -s 10.1.199.0/24 -j MASQ
```

Add the following line to the "/etc/rc.d/rc.local" file:
/etc/rc.d/rc.firewall

Of course the machines that you are configuring to be behind the machine providing the masquerading service should be configured to use that as their gateway. In this case S4 through S6 should use gateway B as their default gateway.

Firewalls

Firewalls are mainly used as a means to protect an organization's internal network from those on the outside (internet). It is used to keep outsiders from gaining information to secrets or from doing damage to internal computer systems. Firewalls are also used to limit the access of individuals on the internal network to services on the internet along with keeping track of what is done through the firewall. Please note the difference between firewalls and routers as described in the second paragraph in the IP Masquerading section.



Types of Firewalls

1. Packet Filtering - Blocks selected network packets.
2. Circuit Level Relay - SOCKS is an example of this type of firewall. This type of proxy is not aware of applications but just cross links your connects to another outside connection. It can log activity, but not as detailed as an application proxy. It only works with TCP connections, and doesn't provide for user authentication.
3. Application Proxy Gateway - The users connect to the outside using the proxy. The proxy gets the information and returns it to the user. The proxy can record everything that is done. This type of proxy may require a user login to use it. Rules may be set to allow some functions of an application to be done and other functions denied. The "get" function may be allowed in the FTP application, but the "put" function may not.

Proxy Servers can be used to perform the following functions.

- Control outbound connections and data.
- Monitor outbound connections and data.
- Cache requested data which can increase system bandwidth performance and decrease the time it takes for other users to read the same data.

Application proxy servers can perform the following additional functions:

- Provide for user authentication.
- Allow and deny application specific functions.
- Apply stronger authentication mechanisms to some applications.

Packet Filtering Firewalls

In a packet filtering firewall, data is forwarded based on a set of firewall rules. This firewall works at the network level. Packets are filtered by type, source address, destination address, and port information. These rules are similar to the routing rules explained in an earlier section and may be thought of as a set of instructions similar to a case statement or if statement. This type of firewall is fast, but cannot allow access to a particular user since there is no way to identify the user except by using the IP address of the user's computer, which may be an unreliable method. Also the user does not need to configure any software to use a packet filtering firewall such as setting a web browser to use a proxy for access to the web. The user may be unaware of the firewall. This means the firewall is transparent to the client.

Circuit Level Relay Firewall

A circuit level relay firewall is also transparent to the client. It listens on a port such as port 80 for http requests and redirect the request to a proxy server running on the machine. Basically, the redirect function is set up using ipchains then the proxy will filter the package at the port that received the redirect.

Configuring a Proxy Server

The following packages are available in Linux:

- Ipchains soon to be replaced by netfilter (Packet filtering supported by the Linux kernel). It comes with Linux and is used to modify the kernel packet routing tables.
- SOCKS - Circuit Switching firewall. Normally doesn't come with Linux, but is free.
- Squid - A circuit switching proxy. Normally comes with Linux.
- Juniper Firewall Toolkit - A firewall toolkit product used to build a firewall. It uses transparent filtering, and is circuit switching. It is available as open source.
- The TIS Firewall Toolkit (FWTK). A toolkit that comes with application level proxies. The applications include Telnet, Rlogin, SMTP mail, FTP, http, and X windows. it can also perform as a transparent proxy for other services.

Ipchains and Linux Packet filtering

For complete information on the use of IP chains and setting up a firewall, see the following Linux how-tos:

- IPCHAINS-HOWTO
- Firewall-HOWTO
- IP-Masquerade-HOWTO

Some of the information in this section is based on these how-tos. This section summarizes and puts in simple steps some of the items you will be required to perform to set up a firewall. It is not meant as a replacement for the Linux how to documents, but a complement to them by giving an overview of what must be done. You may access the howtos from one of the websites listed in the Linux websites section. The Linux Documentation Project or Metalab's Index of Linux publications will have copies if these howtos.

The administration of data packet management is controlled by the kernel. Therefore to provide support for things like IP masquerading, packet forwarding, and port redirects, the support must be compiled into the kernel. The kernel contains a series of tables that each contain 0 or more rules. Each table is called a chain. A chain is a sequence of rules. Each rule

contains two items.

1. Characteristics - Characteristics such as source address, destination address, protocol type (UDP, TCP, ICMP), and port numbers.
2. Instructions - Instructions are carried out if the rule characteristics match the data packet.

The kernel filters each data packet for a specific chain. For instance when a data packet is received, the "input" chain rules are checked to determine the acceptance policy for the data packet. The rules are checked starting with the first rule (rule 1). If the rule characteristics match the data packet, the associated rule instruction is carried out. If they don't match, the next rule is checked. The rules are sequentially checked, and if the end of the chain is reached, the default policy for the chain is returned.

Chains are specified by name. There are three chains that are available and can't be deleted. They are:

1. Input - Regulates acceptance of incoming data packets.
2. Forward - Defines permissions to forward packets that have another host as a destination.
3. Output - Permissions for sending packets.

Each rule has a branch name or policy. Policies are listed below:

- ACCEPT - Accept the data packet.
- REJECT - Drop and the packet but send a ICMP message indicating the packet was refused.
- DENY - Drop and ignore the packet.
- REDIRECT - Redirect to a local socket with input rules only even if the packet is for a remote host. This applies to TCP or UDP packets.
- MASQ - Sets up IP masquerading. Works on TCP or UDP packets.
- RETURN - The next rule in the previous calling chain is examined.

You can create more chains then add rules to them. The commands used to modify chains are as follows:

- -N Create a new chain
- -X Delete an empty chain
- -L List the rules in the chain
- -P Change the policy for a chain
- -F Flush=Delete all the rules in a chain
- -Z Zero the packet and byte counters in all chains

Commands to manipulate rules inside the chain are:

- -A Append a new rule to a chain.
- -I Insert a new rule at some position in a chain.
- -R Replace a rule at some position in a chain.
- -D Delete a rule at some position in a chain.
- Options for masquerading:
    - -M with -L to list the currently masqueraded connection.
    - -M with -S to set the masquerading timeout values.

IPchains Options for setting rule specifications:

- -s Source
- -d Destination
- -p Protocol=tcp, upd, icmp, all or a name from /etc/protocols
- -j Jump target, Specifies the target of the rule. The target can be a user defined chain, but not the one this rule is in.
- -i Interface=Name of the interface the packet is received on or the interface where the packet will be sent
- -t Mask used to modify the type of service (TOS) field in the IP header. This option is followed by two values, the first one is and'ed with the TOS field, and the second is exclusive or'ed. The masks are eight bit hexadecimal values. An example of use is "ipchains -A output -p tcp -d 0.0.0.0/0 telnet -t 0x01 0x10" These bits are used to set priority. See the section on IP message formats.
- -f Fragment

When making changes to firewall rules, it is a good idea to deny all packages prior to making changes with the following three commands:

```
ipchains -I input 1 -j DENY
ipchains -I output 1 -j DENY
ipchains -I forward 1 -j DENY
```

These commands inserts a rule at location 1 that denies all packages for input, output, or forwarding. This is done so no unauthorized packets are not let through while doing the changes. When your changes have been completed, you need to remove the rules at position 1 with the following commands:

```
ipchains -D input 1
ipchains -D output 1
ipchains -D forward 1
```

Examples of the use of ipchains to allow various services

Create a new chain:

```
ipchains -N chainame
```

The option "-N" creates the chain.

Add the chain to the input chain:

```
ipchains -A input -j chainame
```

Allow connections to outside http servers from inside our network:

```
ipchains -A chainame -s 10.1.0.0/16 1024: -d 0.0.0.0/0 www -j ACCEPT
```

The "-A chainame" adds a rule to the chain called "chainame". The "-s 10.1.0.0/16 1024:" specifies any traffic on network 10.1.0.0 at port 1024 or above. The "-d 0.0.0.0/0 www" specifies any destination for www service (in the /etc/services file) and the "-j ACCEPT" sets the rule to accept the traffic.

Allow connections from the internet to connect with your http server:

```
ipchains -A chainame -s 0.0.0.0/0 www -d 10.1.1.36 1024: -j ACCEPT
```

The "-A chainame" adds a rule to the chain called "chainame". The "-s 0.0.0.0/0 www" specifies traffic from any source for www service. The "-d 10.1.1.36 1024:" specifies the http server at IP address 10.1.1.36 at ports above 1024 and the "-j ACCEPT" sets the rule to accept the traffic.

Allow DNS to go through the firewall:

```
ipchains -A chainame -p UDP -s 0/0 dns -d 10.1.0.0/16 -j ACCEPT
```

The "-A chainame" adds a rule to the chain called "chainame". The "-p UDP" specifies UDP protocol. The "-s 0/0 dns" specifies any dns traffic from any location. The "-d 10.1.0.0/16" specifies our network and the "-j ACCEPT" sets the rule to accept the traffic. This allows DNS queries from computers inside our network to be received.

Allow e-mail to go from our internal mail server to mailservers outside the network.

```
ipchains -A chainame -s 10.1.1.24 -d 0/0 smtp -j ACCEPT
```

The "-A chainame" adds a rule to the chain called "chainame". The "-s 10.1.1.24" specifies any traffic from 10.1.1.24 IP address. The "-d 0/0 smtp" specifies any smtp type of service going anywhere and the "-j ACCEPT" sets the rule to accept the traffic.

Allow e-mail to come from any location to our mail server:

```
ipchains -A chainame -s 0/0 smtp -d 10.1.1.24 smtp -j ACCEPT
```

The "-A chainame" adds a rule to the chain called "chainame". The "-s 0/0 smtp" specifies mail traffic from anywhere. The "-d 10.1.1.24 smtp" specifies mail traffic going to our mail server and the "-j ACCEPT" sets the rule to accept the traffic.

Perform a HTTP port redirect for a transparent proxy server:

```
ipchains -A input -p tcp -s 10.1.0.0/16 -d 0/0 80 -j REDIRECT 8080
```

The "-A input" adds a rule to the input chain. The "-p tcp" specifies the protocol TCP. The "-s 10.1.0.0/16" specifies the source as a network with netmask 255.255.0.0. The "-d 0/0" specifies a destination of anywhere. The number 80 is the HTTP port number, and the command "-j REDIRECT 8080" redirects the traffic to port 8080.

Give telnet transmissions a higher priority

```
ipchains -A output -p tcp -d 0.0.0.0/0 telnet -t 0x01 0x10"
```

The bits at the end of the line specified in hexadecimal format are used to set the priority of the IP message on the network. The first value is and'ed with the TOS field in the IP message header, and the second value is exclusive or'ed. See the section on IP message formats for more information.

Using ipchains-save and ipchains-restore to make rules permanent

When you are done setting your ipchains rules, use the following procedure while logged on as root to make them permanent:

1. Type the command "ipchains-save > /etc/iprules.save".
2. Create the following script named "packetfw":

```
#! /bin/sh
# Packet filtering firewall script to be used turn the firewall on or off

if [ -f /etc/iprules.save ]
then
   case "$1" in
      start)
          echo -n "Turning on packet filtering firewall:"
          /sbin/ipchains-restore < /etc/iprules.save
          echo 1 > /proc/sys/net/ipv4/ip_forward
          echo "."
          ;;
       stop)
          echo -n "Turning off packet filtering:"
          echo 0 > /proc/sys/net/ipv4/ip_forward
          /sbin/ipchains -X
          /sbin/ipchains -F
          /sbin/ipchains -P input ACCEPT
          /sbin/ipchains -P output ACCEPT
          /sbin/ipchains -P forward ACCEPT
          echo "."
          ;;
       *)
          echo "Usage: /etc/init.d/packetfw {start|stop}"
          exit 1
          ;;
   esac
   exit 0
else
   echo the /etc/iprules.save file does not exist.
   exit 1
fi
```

3. Save the file in the /etc/rc.d/init.d directory.
4. In the /etc/rc.d/rc3.d and the /etc/rc.d/rc5.d directories make a symbolic link called S07packetfw to the /etc/rc.d/init.d/packetfw file with the command "ln -s /etc/rc.d/rc3/S07packetfw /etc/rc.d/init.d/packetfw". This applies to runlevel 3. Do the same for the runlevel 5 initialization directory. Note: You may need to use a different number than the "S07" string to number your link file. Look in your /etc/rc.d/rc3.d and /etc/rc.d/rc5.d directories to determine what number is available to give this file. Try to give it a number just below your network number file. On my system the S10network file is used to start my network.

Domain Name Service

Host Names

Domain Name Service (DNS) is the service used to convert human readable names of hosts to IP addresses. Host names are not case sensitive and can contain alphabetic or numeric letters or the hyphen. Avoid the underscore. A fully qualified domain name (FQDN) consists of the host name plus domain name as in the following example:

computername.domain.com

The part of the system sending the queries is called the resolver and is the client side of the configuration. The nameserver answers the queries. Read RFCs 1034 and 1035. These contain the bulk of the DNS information and are superceded by RFCs 1535-1537. Naming is in RFC 1591. The main function of DNS is the mapping of IP addresses to human readable names.

Three main components of DNS

1. resolver
2. name server
3. database of resource records(RRs)

Domain Name System

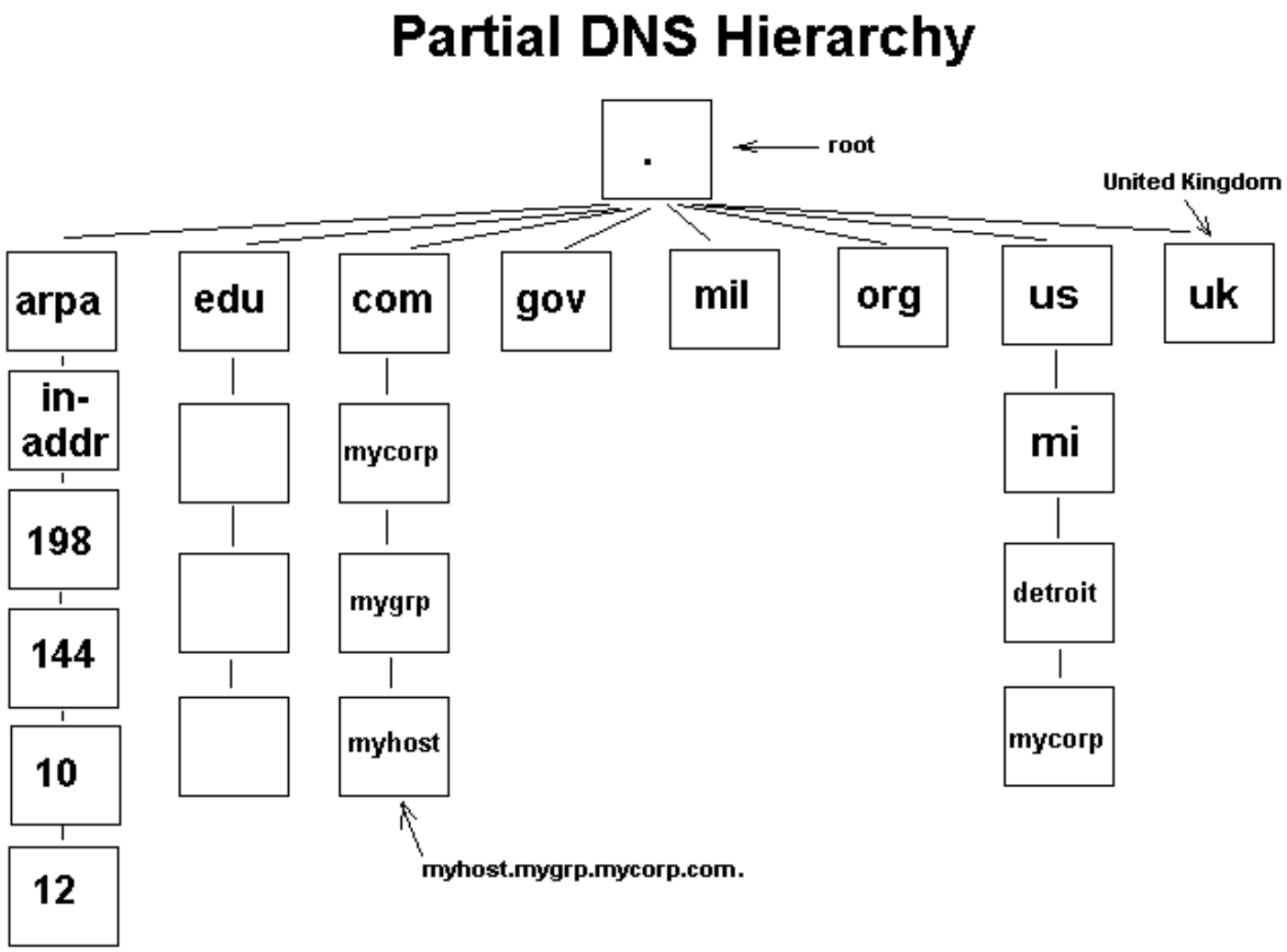
The Domain Name System (DNS) is basically a large database which resides on various computers and it contains the names and IP addresses of various hosts on the internet and various domains. The Domain Name System is used to provide information to the Domain Name Service to use when queries are made. The service is the act of querying the database, and the system is the data structure and data itself. The Domain Name System is similar to a file system in Unix or DOS starting with a root. Branches attach to the root to create a huge set of paths. Each branch in the DNS is called a label. Each label can be 63 characters long, but most are less. Each text word between the dots can be 63 characters in length, with the total domain name (all the labels) limited to 255 bytes in overall length. The domain name system database is divided into sections called **zones**. The name servers in their respective zones are responsible for answering queries for their zones. A zone is a subtree of DNS and is administered separately. There are multiple name servers for a zone. There is usually one primary nameserver and one or more secondary name servers. A name server may be authoritative for more than one zone.

DNS names are assigned through the Internet Registries by the Internet Assigned Number Authority (IANA). The domain name is a name assigned to an internet domain. For example, mycollege.edu represents the domain name of an educational institution. The names microsoft.com and 3Com.com represent the domain names at those commercial companies. Naming hosts within the domain is up to individuals administer their domain.

Access to the Domain name database is through a resolver which may be a program or part of an operating system that resides on users workstations. In Unix the resolver is accessed by using the library functions "gethostbyname" and "gethostbyaddr". The resolver will send requests to the name servers to return information requested by the user. The requesting computer tries to connect to the name server using its IP address rather than the name.

Structure and message format

The drawing below shows a partial DNS hierarchy. At the top is what is called the root and it is the start of all other branches in the DNS tree. It is designated with a period. Each branch moves down from level to level. When referring to DNS addresses, they are referred to from the bottom up with the root designator (period) at the far right. Example: "myhost.mycompany.com.".



DNS is hierarchical in structure. A domain is a subtree of the domain name space. From the root, the assigned top-level domains in the U.S. are:

- GOV - Government body.
- EDU - Educational body.
- INT - International organization
- NET - Networks
- COM - Commercial entity.
- MIL - U. S. Military.
- ORG - Any other organization not previously listed.

Outside this list are top level domains for various countries.

Each node on the domain name system is separated by a ".". Example: "mymachine.mycompany.com.". Note that any name ending in a "." is an absolute domain name since it goes back to root.

DNS Message format:

Bits	Name	Description

0-15	Identification	Used to match responses to requests. Set by client and returned by server.
16-31	Flags	Tells if query or response, type of query, if authoritative answer, if truncated, if recursion desired, and if recursion is available.
32-47	Number of questions	
48-63	Number of answer RRs	
64-79	Number of authority RRs	
80-95	Number of additional RRs	
96-??	Questions - variable lengths	There can be variable numbers of questions sent.
??-??	Answers - variable lengths	Answers are variable numbers of resource records.
??-??	Authority - variable lengths	
??-??	Additional Information - variable lengths	

Question format includes query name, query type and query class. The query name is the name being looked up. The query class is normally 1 for internet address. The query types are listed in the table below. They include NS, CNAME, A, etc.

The answers, authority and additional information are in resource record (RR) format which contains the following.

1. Domain name
2. Type - One of the RR codes listed below.
3. Class - Normally indicates internet data which is a 1.
4. Time to live field - The number of seconds the RR is saved by the client.
5. Resource data length specifies the amount of data. The data is dependent on its type such as CNAME, A, NS or others as shown in the table below. If the type is "A" the data is a 4 byte IP address.

The table below shows resource record types:

Type	RR value	Description
A	1	Host's IP address
NS	2	Host's or domain's name server(s)
CNAME	5	Host's canonical name, host identified by an alias domain name
PTR	12	Host's domain name, host identified by its IP address
HINFO	13	Host information
MX	15	Host's or domain's mail exchanger
AXFR	252	Request for zone transfer
ANY	255	Request for all records

Usage and file formats

If a domain name is not found when a query is made, the server may search for the name elsewhere and return the information to the requesting workstation, or return the address of a name server that the workstation can query to get more information. There are special servers on the Internet that provide guidance to all name servers. These are known as root name servers. They do not contain all information about every host on the Internet, but they do provide direction as to where domains are located (the IP address of the name server for the uppermost domain a server is requesting). The root name server is the starting point to find any domain on the Internet.

Name Server Types

There are three types of name servers:

1. The primary master builds its database from files that were preconfigured on its hosts, called zone or database files. The name server reads these files and builds a database for the zone it is authoritative for.
2. Secondary masters can provide information to resolvers just like the primary masters, but they get their information from the primary. Any updates to the database are provided by the primary.
3. Caching name server - It gets all its answers to queries from other name servers and saves (caches) the answers. It is a non-authoritative server.

The caching only name server generates no zone transfer traffic. A DNS Server that can communicate outside of the private network to resolve a DNS name query is referred to as **forwarder**.

DNS Query Types

There are two types of queries issued:

1. **Recursive** queries received by a server forces that server to find the information requested or post a message back to the querier that the information cannot be found.
2. **Iterative** queries allow the server to search for the information and pass back the best information it knows about. This is the type that is used between servers. Clients used the recursive query.
3. **Reverse** - The client provides the IP address and asks for the name. In other queries the name is provided, and the IP address is returned to the client. Reverse lookup entries for a network 192.168.100.0 is "100.168.192.in-addr arpa".

Generally (but not always), a server-to-server query is iterative and a client-resolver-to-server query is recursive. You should also note that a server can be queried or it can be the person placing a query. Therefore, a server contains both the server and client functions. A server can transmit either type of query. If it is handed a recursive query from a remote source, it must transmit other queries to find the specified name, or send a message back to the originator of the query that the name could not be found.

DNS Transport protocol

DNS resolvers first attempt to use UDP for transport, then use TCP if UDP fails.

The DNS Database

A database is made up of records and the DNS is a database. Therefore, common resource record types in the DNS database are:

- A - Host's IP address. Address record allowing a computer name to be translated into an IP address. Each computer must have this record for its IP address to be located. These names are not assigned for clients that have dynamically assigned IP addresses, but are a must for locating servers with static IP addresses.
- PTR - Host’s domain name, host identified by its IP address
- CNAME - Host’s canonical name allows additional names or aliases to be used to locate a computer.
- MX - Host’s or domain’s mail exchanger.
- NS - Host’s or domain’s name server(s).
- SOA - Indicates authority for the domain

- TXT - Generic text record
- SRV - Service location record
- RP - Responsible person
- HINFO - Host information record with CPU type and operating system.

When a resolver requests information from the server, the DNS query message indicates one of the preceding types.

DNS Files

- CACHE.DNS - The DNS Cache file. **This file is used to resolve internet DNS queries**. On Windows systems, it is located in the WINNTROOT\system32\DNS directory and is used to configure a DNS server to use a DNS server on the internet to resolve names not in the local domain.

Example Files

Below is a partial explanation of some records in the database on a Linux based system. The reader should view this information because it explains some important DNS settings that are common to all DNS servers. An example /var/named/db.mycompany.com.hosts file is listed below.

```
mycompany.com.          IN      SOA     mymachine.mycompany.com.
root.mymachine.mycompany.com. (
                1999112701          ; Serial number as date and two digit number
YYMMDDXX
                10800               ; Refresh in seconds 28800=8H
                3600                ; Retry in seconds 7200=2H
                604800              ; Expire 3600000=1 week
                86400 )             ; Minimum TTL 86400=24Hours
mycompany.com.                  IN      NS      mymachine.mycompany.com.
mycompany.com.                  IN      MX              10
mailmachine.mycompany.com.
mymachine.mycompany.com.        IN      A       10.1.0.100
mailmachine.mycompany.com.      IN      A       10.1.0.4
george.mycompany.com.           IN      A       10.1.3.16
```

A Line by line description is as follows:

1. The entries on this line are:
    1. mycompany.com. - Indicates this server is for the domain mycompany.com.
    2. IN - Indicates Internet Name.
    3. SOA - Indicates this server is the authority for its domain, mycompany.com.
    4. mymachine.mycompany.com. - The primary nameserver for this domain.
    5. root.mymachine.mycompany.com. - The person to contact for more information.

    The lines in the parenthesis, listed below, are for the secondary nameserver(s) which run as slave(s) to this one (since it is the master).
2. 1999112701 - Serial number - If less than master's SN, the slave will get a new copy of this file from the master.
3. 10800 - Refresh - The time in seconds between when the slave compares this file's SN with the master.
4. 3600 - Retry - The time the server should wait before asking again if the master fails to respond to a file update (SOA request).
5. 604800 - Expire - Time in seconds the slave server can respond even though it cannot get an updated zone file.
6. 86400 - TTL - The time to live (TTL) in seconds that a resolver will use data received from a nameserver before it will

ask for the same data again.

7. This line is the nameserver resource record. There may be several of these if there are slave name servers.

```
mycompany.com.                  IN        NS      mymachine.mycompany.com.
```

   Add any slave server entries below this like:

```
mycompany.com.                  IN        NS      ournamesv1.mycompany.com.
mycompany.com.                  IN        NS      ournamesv2.mycompany.com.
mycompany.com.                  IN        NS      ournamesv3.mycompany.com.
```

8. This line indicates the mailserver record.

```
mycompany.com.                  IN        MX              10
mailmachine.mycompany.com.
```

   There can be several mailservers. The numeric value on the line indicates the preference or precedence for the use of that mail server. A lower number indicates a higher preference. The range of values is from 0 to 65535. To enter more mailservers, enter a new line for each one similar to the nameserver entries above, but be sure to set the preferences value correctly, at different values for each mailserver.

9. The rest of the lines are the name to IP mappings for the machines in the organization. Note that the nameserver and mailserver are listed here with IP addresses along with any other server machines required for your network.

```
mymachine.mycompany.com.        IN        A       10.1.0.100
mailmachine.mycompany.com.      IN        A       10.1.0.4
george.mycompany.com.           IN        A       10.1.3.16
```

Domain names written with a dot on the end are absolute names which specify a domain name exactly as it exists in the DNS hierarchy from the root. Names not ending with a dot may be a subdomain to some other domain.

Aliases are specified in lines like the following:

```
mymachine.mycompany.com          IN      CNAME   nameserver.mycompany.com.
george.mycompany.com    IN       CNAME   dataserver.mycompany.com.
Linux1.mycompany.com    IN       CNAME   engserver.mycompany.com.
Linux2.mycompany.com    IN       CNAME   mailserver.mycompany.com.
```

When a client (resolver) sends a request, if the nameserver finds a CNAME record, it replaces the requested name with the CNAME, then finds the address of the CNAME value, and return this value to the client.

A host that has more than one network card which is set to address two different subnets can have more than one address for a name.

```
mymachine.mycompany.com          IN      A       10.1.0.100
                                 IN      A       10.1.1.100
```

When a client queries the nameserver for the address of a multi homed host, the nameserver will return the address that is closest to the client address. If the client is on a different network than both the subnet addresses of the multi homed host, the server will return both addresses.

For more information on practical application of DNS, read the DNS section of the Linux User's Guide.

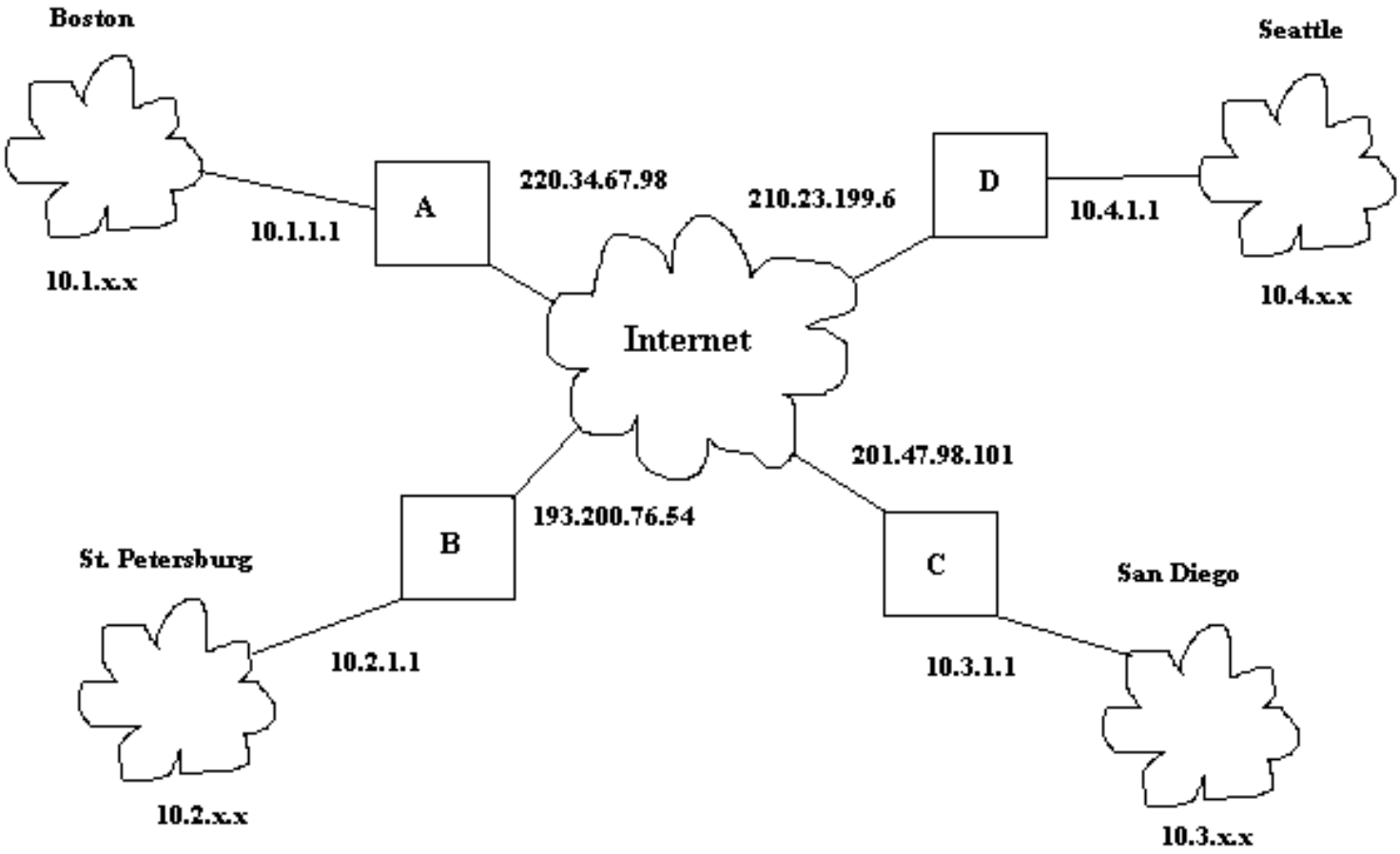
Virtual Private Networking

If you've understood most of this document so far, the principles of Virtual private networking (VPN) will be easy to understand. The most confusing part of VPN is that many acronyms show up. This is partly because VPN requires data encryption to be "private" and there are many encryption techniques and terms. Also there are many complicated security issues relating to VPN concerning encryption and user authentication. This section will first explain the concept and methodology behind VPN, then explain some of the acronyms. I can't explain them all, there will be more tomorrow.

Purpose of VPN

The function of VPN is to allow two computers or networks to talk to each other over a transport media that is not secure. To do this VPN uses a computer at each of the two or more points on the various ends of the transport media such as the internet. Each point at the end of the transport media (internet) is called a point of presence (POP). In this example, the transport media is the internet. In the example below our company "Boats and More, Inc." has four offices. One in Boston, St Petersburg, Seattle, and San Diego. The owner wants a networking setup so he can access any of the 4 network locations at any time through the internet. He wants his data secure since some of it is confidential. His offices are set up on networks 10.1.x.x, 10.2.x.x, 10.3.x.x, and 10.4.x.x. Each of the four networks, when they need to send a data packet to one of the other networks, will route its data packet to its respective router, A, B, C, or D. For example if a computer on the 10.1.x.x network in Boston needs to send a packet to a computer with address 10.3.6.1 on the network in San Diego at 10.3.x.x, it will send its packet to its router, A. Since the network number, 10.x.x.x, is reserved for private use, the packet can't be sent going from computer A with 10.3.6.1 as its intended address. This is because the routers on the internet will not recognize this address as a valid destination. IP masquerading won't solve this problem since the computer on the other end would have no way of knowing that a packet that it didn't send was a masqueraded packet. Tunneling is the technique used to solve this problem.

VPN Setup for Multiple Points of Presence



Tunneling means that the complete IP packet to be sent from Boston to San Diego must be encapsulated into another IP packet. This new packet will have a legal internet IP address. Therefore, machine A will take the packet it needs to route (already has destination address 10.3.6.1) and roughly the following will happen:

1. Machine A will extract the IP packet.
2. Machine A will encrypt the packet.
3. Machine A will wrap the original IP packet in a new IP packet with destination address 201.47.98.101, which is machine C's true internet address.
4. Machine A will wrap the new IP packet in an ethernet packet and send it to the network.
5. The packet will be routed through the internet until it reaches machine C.
6. Machine C will extract the outer IP packet.
7. Machine C will determine that the IP packet contains another IP packet and extract it.
8. Machine C will decrypt the packet.
9. Machine C will examine the destination address of the inner IP packet, wrap it in an ethernet packet with the correct ethernet address, and send it to the internal network on its port 10.3.1.1.

This description is simplistic, but it is essentially what happens. This did not account for authentication and being sure machine C had the authority or ability to decrypt the packet. Therefore VPN can be examined in two main functional areas which are the tunneling mechanism and the security mechanisms.

VPN tunneling Protocols

The list below describes the tunneling protocols which may be used for VPN.

- L2F - Layer2 Forwarding, works at the link layer of the OSI model. It has no encryption. Being replaced by L2TP.
- PPTP - Point-to-Point Tunneling Protocol (RFC 2637) works at the link layer. No encryption or key management included in specifications.
- L2TP - Layer2 Tunneling Protocol. (RFC 2661) Combines features of L2F and PPTP and works at the link layer. No encryption or key management included in specifications.
- IPSec - Internet protocol security, developed by IETF, implemented at layer 3. it is a collection of security measures that address data privacy, integrity, authentication, and key management, in addition to tunneling. Does not cover key management.
- Socks - handled at the application layer

VPN Security

In addition ot tunneling, VPN needs to provide for authentification, confidentiality, data integrity and key management. This is important if you need to keep your data going across the transmission media, secret. The capability of sending the data is easy, but the security measures necessary make VPN a much more complex subject. Security functions that must be covered are:

- Authentification - Making sure the data is from where it is supposed to be from.
- Confidentiality - Keeping any third parties from reading or understanding the data.
- Data integrity - Being sure the data received was not changed by a third party and that it is correct.
- Access control - Keeping third parties without authorization from getting access to your data or network.

Essentially the part of the system that must make the data secure, must encrypt the data and provide a method to decrypt the data. There are many different encryption formulas, but typically handling of decryption is usually done by providing a "key" to the party that must decrypt the data. Keys are secrets shared between two parties, that allow one party to pass encrypted information from one to the other without third parties being able to read it. It is similar to a house or car key that allows only members of your family to enter the house or use the car. Keys are a digital code that will allow the second party to decrypt the data. The digital code must be long enough to keep any third parties from being able to break the code by guessing. Key management can be a complex subject since there are many ways to implement it, but it needs to be secure so no third party gets, intercepts, or guesses the key.

There are many different protocols used to support each of the above functions. Each have various advantages and disadvantages including the fact that some are more secure than others. If you are going to use VPN as a data exchange method, and you want secure data, you or someone on your staff had better know what they're doing (Knowledge of the strengths and weaknesses of the protocols and how to implement them properly), or sooner or later, you may get burned.

Managing user access rights and Key Management or Authentification

Systems

Two key management protocols are:

1. RADIUS - Remote Authentication Dial-In User Service is used for dial in clients to connect to other computers or a network. It provides authentication and accounting when using PPTP or L2TP tunneling.
2. ISAKMP/Oakley - Internet Security Association and Key Management Protocol Authentication uses one of the following three attributes to authenticate users.
    1. Something you have such as a key.
    2. Something you know such as a secret.
    3. Something you are such as your fingerprint.

    More than one means of authentification is recommended for stronger security.

VPN terms

VPN Protocols:

- PPTP - Point to point tunneling protocol (RFC 2637)
- L2TP - Layer 2 tunneling protocol (RFC 2661)
- IPIP tunneling - Tunneling IP packets in IP packets.

Encryption protocols, methods and terms:

- CIPE - Crypto IP Encapsulation
- SSL - Secure sockets layer
- IPSEC - Internet protocol security

Authentication Protocols:

- PAP - Password Authentification Protocol is a two way handshake protocol designed for use with PPP.
- CHAP - Challenge Handshake Authentication Protocol is a three way handshake protocol which is considered more secure than PAP.
- TACACS - Offers authentication, accounting, and authorization.
- S/Key - A one time password system, secure against replays. RFC 2289.

Projects and software:

- SWAN - Secure wide area network
- PoPToP Point to point tunneling protocol server.

DHCP

Dynamic Host Configuration Protocol (DHCP)

This protocol is used to assign IP addresses to hosts or workstations on the network. Usually a DHCP server on the network performs this function. Basically it "leases" out address for specific times to the various hosts. If a host does not use a given address for some period of time, that IP address can then be assigned to another machine by the DHCP server. When assignments are made or changed, the DHCP server must update the information in the DNS server.

As with BOOTP, DHCP uses the machine's or NIC ethernet (MAC) or hardware address to determine IP address assignments. The DHCP protocol is built on BOOTP and replaces BOOTP. DHCP extends the vendor specific area in BOOTP to 312 bytes from 64. RFC 1541 defines DHCP.

DHCP RFCs

DHCP RFCs are 1533, 1534, 1541, and 1542. Sent from DHCP server:

- IP address
- Netmask
- Default Gateway address
- DNS server addresse(s)
- NetBIOS Name server (NBNS) address(es).
- Lease period in hours
- IP address of DHCP server.

DHCP Lease Stages

1. Lease Request - The client sends a broadcast requesting an IP address
2. Lease Offer - The server sends the above information and marks the offered address as unavailable. The message sent is a DHCPOFFER broadcast message.
3. Lease Acceptance - The first offer received by the client is accepted. The acceptance is sent from the client as a broadcast (DHCPREQUEST message) including the IP address of the DNS server that sent the accepted offer. Other DHCP servers retract their offers and mark the offered address as available and the accepted address as unavailable.
4. Server lease acknowledgement - The server sends a DHCPACK or a DHCPNACK if an unavailable address was requested.

DHCP discover message - The initial broadcast sent by the client to obtain a DHCP lease. It contains the client MAC address and computer name. This is a broadcast using 255.255.255.255 as the destination address and 0.0.0.0 as the source address. The request is sent, then the client waits one second for an offer. The request is repeated at 9, 13, and 16 second intervals with additional 0 to 1000 milliseconds of randomness. The attempt is repeated every 5 minutes thereafter. The client uses port 67 and the server uses port 68.

DHCP Lease Renewal

After 50% of the lease time has passed, the client will attempt to renew the lease with the original DHCP server that it obtained the lease from using a DHCPREQUEST message. Any time the client boots and the lease is 50% or more passed,

the client will attempt to renew the lease. At 87.5% of the lease completion, the client will attempt to contact any DHCP server for a new lease. If the lease expires, the client will send a request as in the initial boot when the client had no IP address. If this fails, the client TCP/IP stack will cease functioning.

DHCP Scope and Subnets

One DHCP scope is required for each subnet.

DHCP Relay Agents

May be placed in two places:

- Routers
- Subnets that don't have a DHCP server to forward DHCP requests.

Client Reservation

Client Reservation is used to be sure a computer gets the same IP address all the time. Therefore since DHCP IP address assignments use MAC addresses to control assignments, the following are required for client reservation:

- MAC (hardware) address
- IP address

Exclusion Range

Exclusion range is used to reserve a bank of IP addresses so computers with static IP addresses, such as servers may use the assigned addresses in this range. These addresses are not assigned by the DHCP server.

Sample DCHP Configuration File

In Linux, a sample configuration file is:

```
subnet 192.168.199.0 netmask 255.255.255.0 {
# --- default gateway
        option routers                  192.168.199.1;
        option subnet-mask              255.255.255.0;

        option nis-domain               "mynet.net";
        option domain-name              "mynet.net";
        option domain-name-servers      192.168.199.1;

        option time-offset              -5;     # Eastern Standard Time
#       option ntp-servers              192.168.199.1;
#       option netbios-name-servers     192.168.199.1;
# --- Selects point-to-point node (default is hybrid). Don't change this unless
# -- you understand Netbios very well
#       option netbios-node-type 2;
```

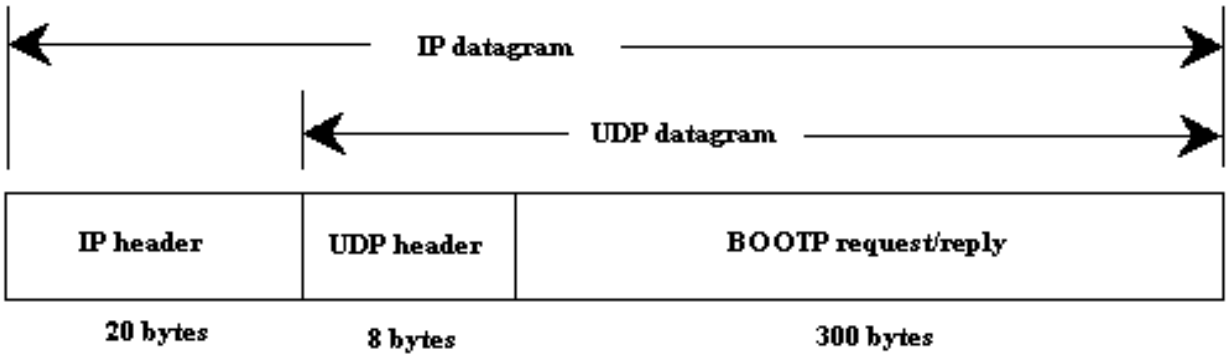
```
        default-lease-time 1209600;  # 2 weeks
        max-lease-time 1814400;          # 3 weeks

        range 192.168.199.10 192.168.199.250;

        # we want the nameserver to appear at a fixed address
        host nameserver {
                next-server nameserver.mynet.net;
                hardware ethernet 00:10:4b:ca:db:b5;
                fixed-address 192.168.199.1;
        }
}
```

This demonstrates that the IP addresses are based on lease times to the various clients. If they are not used within the period of their lease time by the client, those IP addresses are freed up for use by other clients.

BOOTP

BOOTP (Boot Protocol) may be used to boot remote computers over a network. BOOTP messages are encapsulated inside UDP messages and therefore it's requests and replies are forwarded by routers. BOOTP is defined by RFCs 951 and 1542. The drawing below illustrates the data encapsulation:

BOOTP Data Encapsulation



The diskless system reads its unique hardware address from its network interface card then sends a BOOTP request. The table below shows the BOOTP package format from most significant bit to least significant bit.

Bit range	# of Bits	Name	Description
0-7	8	Op code	Tells if the message is a BOOTP request or reply. Request=1, reply=2
8-15	8	Hardware type	Indicates the type of hardware (link level). A value of 6 indicates ethernet
16-23	8	Hardware address length	Tells the length in bytes of the hardware address number. Ethernet addresses are 6 bytes long.
23-31	8	Hop count	Initially set to 0. Incremented each time it is forwarded.
32-63	32	Transaction ID	A random number set by the client and returned by the server. Used to match replies with requests
64-79	16	Number of seconds	The time since the client started trying to bootstrap. Used to tell if a backup BOOTP server should respond.
80-95	16	unused	not used
96-127	32	Clients IP address	The clients IP address. If a request, it is normally 0.0.0.0
128-159	32	IP address for client	The server sets this in the reply message.

160-191	32	Server IP address	Filled in by the server.
192-223	32	Gateway IP address	Returned by the server.
224-351	128	Clients hardware address	Provided by the client.
352-1375	1024	Server hostname	A null terminated string optionally filled in by the server.
1376-3423	2048	Boot filename	A fully qualified boot file name with path information, terminated with a null. Supplied by the server.
3424-4447	1024	Vendor information	Used for various options to BOOTP including the subnet mask to the client.

The BOOTP server uses port 67 and the BOOTP client uses port 68. The following is a brief explanation of what happens when a remote client boots:

1. BOOTP request. The client sends a BOOTP request from 0.0.0.0.68 to 255.255.255.255.67 with its ethernet address and number of second's fields filled in.
2. BOOTP reply. The server responds with the client's IP address, the server's IP address (it's own), and the IP address of a default gateway.
3. ARP request. The client issues an ARP to tell if the IP address it just received is being used. It uses 0.0.0.0 as it's own address
4. ARP request. The client waits 0.5 seconds and repeats the same ARP request.
5. ARP request. The client waits another 0.5 seconds and repeats the ARP request with it's own address as the senders address.
6. BOOTP request. The client waits 0.5 seconds and sends another BOOTP request with its own IP address in the IP header
7. BOOTP reply. The server sends the same BOOTP reply it sent the last time.
8. ARP request. The client outputs an ARP request for the server hardware address
9. ARP reply. The server replies with its own ethernet address.
10. TFTP read request. The client sends a TFTP read request asking for its specified boot file.

RPC and NFS

Network File System (NFS)

NFS, defined by RFC 1094, is a method for client systems to use a filesystem on a remote host computer. NFS uses the UDP protocol and is supported by RPC.

Remote Procedure Call (RPC)

RPC, defined by RFC 1057, is a set of function calls used by a client program to call functions in a remote server program. The port mapper program is the program used to keep track of which ports programs supporting RPC functions use. The port mappers port is 111. In Redhat Linux the portmapper daemon is started in the /etc/rc.d/init.d/portmap and the daemon program is called "portmap".

The rpcinfo command

The command "rpcinfo -p" will show the port numbers that are assigned to the RPC services.

```
   program vers proto   port
    100000    2   tcp    111  portmapper
    100000    2   udp    111  portmapper
    100011    1   udp    747  rquotad
    100011    2   udp    747  rquotad
    100005    1   udp    757  mountd
    100005    1   tcp    759  mountd
    100005    2   udp    762  mountd
    100005    2   tcp    764  mountd
    100003    2   udp   2049  nfs
```

Services that may be listed include:

- rquotad - Enforces the set quotas for remote mounted NFS systems.
- mountd - Performs the requested mounts.
- nfs - Handles the user interface to the kernel module that performs NFS.

NFS related services in Linux include:

- amd - Runs the automount daemon for automatic remote filesystem mounting such as nfs. It is especially worthwhile for working with removeable media such as floppies or CD ROM disks.
- autofs - This is the startup, stop, and status script for the automount program used to configure

mount points for automatic mounting of file systems.

- nfs - Provides Network File System server services.
- netfs - Mounts and unmounts Network Fils System (NFS), Windows (SMB), and Netware (NCP) file systems. The mount command is used to perform this operation and no daemon is run in the background.

The /etc/exports file is used to configure exported filesystems.