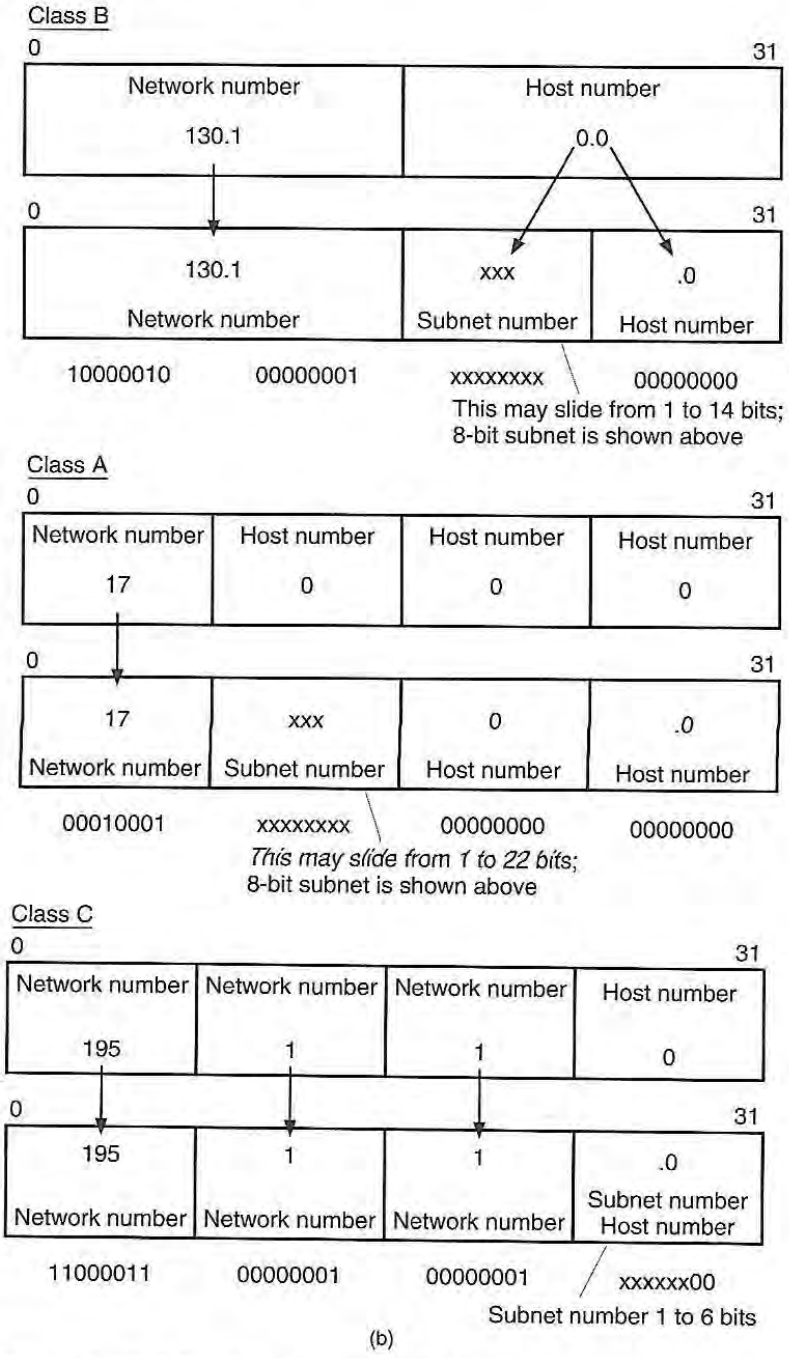


(b)

Figure 6.6 (*b*) Subnetting a network number.

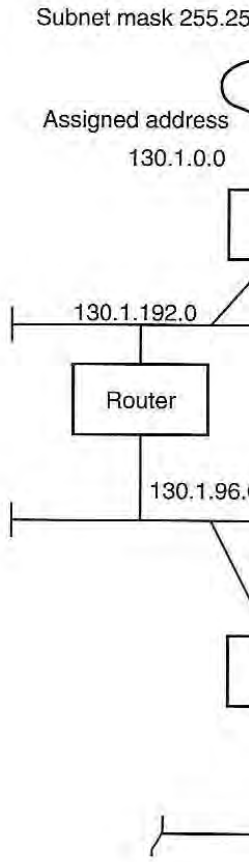


Figure 6.6 (*c*) Subnettir

This takes in all
and all 1s. In the pr
bers, we could possi
a little more realist
have to use many u
work number and h
network. Without s
up to 65535 hosts as
made 30 subnetwor
any of the subnet or

How did we get 3
gives us 32 possible
be used (subtract 2
could have used all
(all 0s and all 1s no

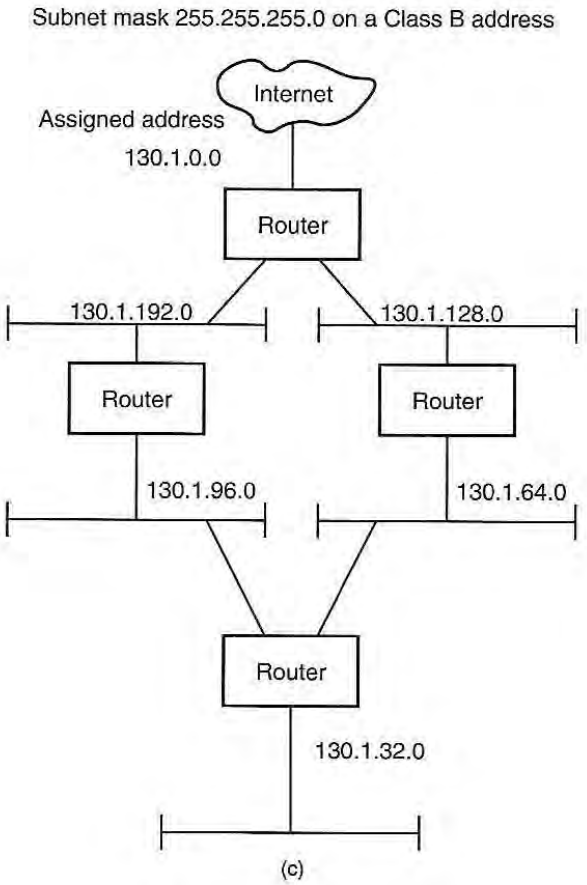


Figure 6.6 (*c*) Subnetting on an internetwork.

This takes in all possible combinations with the exclusion of all 0s and all 1s. In the previous example with each of those subnetwork numbers, we could possibly have 2046 hosts per subnetwork number. This is a little more realistic than not subnetting. With subnetting, we did not have to use many unique class B addresses. We used one class B network number and have 30 subnets available to us from the one class B network. Without subnetting, we would have one network number and up to 65535 hosts assigned to it. Now, we used one network number and made 30 subnetworks from it. All zeros or all ones are not allowed in any of the subnet or host fields.

How did we get 30 possibilities? Using five bits for the subnet mask gives us 32 possible combinations (0 to 31). Since all 1s or all 0s cannot be used (subtract 2), that gives us 30 possibilities. The subnet mask could have used all eight bits, which would give us 254 subnet numbers (all 0s and all 1s not used).

How does a router or workstation know when a subnet is being used. It employs the use of a *subnet mask.* What does a subnet mask look like? It is always written in decimal and shows the number that will be use to mask the bits. For example, let's use the IP address 130.40.132.3. Using the first five bits of the first host field (the third octet) yields 248 (convert the first five bits to binary 11111000). This means the subnet mask for that IP address will be 255.255.248.0 in decimal. This is the mask that we have assigned to the network address of 130.40.132.3. 255 will always be the case for the network number portion of the address. The 248 is used to tell the network station to use the first five bits of the network address, not for a host ID, but for a subnet. It tells a network station which bits to use for a subnet mask. The remaining eleven bits (the remaining three bits of the third octet and eight bits of the fourth octet) should be used for the host ID. This allows for 30 subnets with 2046 (actually 2048, but all 0s and all 1s are not allowed assignable host IDs) hosts on each subnet.

Therefore, the IP address of 130.40.132.3, with a subnet mask of 255.255.248.0, yields the network number of 130.40, subnet number 128, and host ID of 1027. (*Hint:* convert the address to binary, apply the mask in binary, and then convert it back to decimal as shown in Fig. 6.6*a*.

An operation is performed on an IP address. It is called a bitwise AND operation. The IP address is AND'ed with the subnet mask to allow the network station to determine the subnet mask.

Again, Figure 6.6*a* shows the mask operation. At the bottom of the figure is the IP address in binary. This address is logically AND'ed with the mask. The bits that drop out of this operation will indicate to any TCP/IP station the network address. It masks out the host address and leaves the network address.

One last rule: when a network is subnetted, the whole network (all stations assigned to that network number) must be subnetted exactly the same. This is for networks using the IP RIP routing update protocol. When the network number changes (not the subnetwork number), the subnet mask may change. You cannot assign one network number with different subnet masks. The exception to this rule is the routing algorithm of OSPF, which is beyond the scope of this book.

For example, if using the RIP routing protocol (explained later), the subnet mask must remain the same throughout a single class B assignment. For example, if the network assignment is 130.1.0.0 and the subnet mask assigned is 255.255.255.0, this subnet mask must remain the same throughout the 130.1.0.0 network. If the network address changes—for example, to 131.1.0.0—the subnet mask may also change. If the network is using the OSPF routing protocol, the subnet mask may change within one network address.

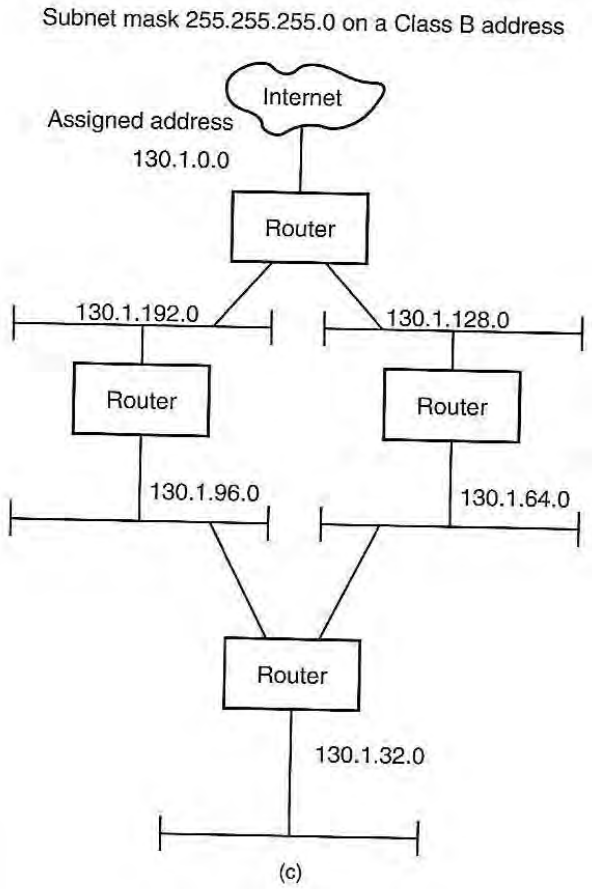


Figure 6.6 (*c*) Subnetting on an internetwork.

This takes in all possible combinations with the exclusion of all 0s and all 1s. In the previous example with each of those subnetwork numbers, we could possibly have 2046 hosts per subnetwork number. This is a little more realistic than not subnetting. With subnetting, we did not have to use many unique class B addresses. We used one class B network number and have 30 subnets available to us from the one class B network. Without subnetting, we would have one network number and up to 65535 hosts assigned to it. Now, we used one network number and made 30 subnetworks from it. All zeros or all ones are not allowed in any of the subnet or host fields.

How did we get 30 possibilities? Using five bits for the subnet mask gives us 32 possible combinations (0 to 31). Since all 1s or all 0s cannot be used (subtract 2), that gives us 30 possibilities. The subnet mask could have used all eight bits, which would give us 254 subnet numbers (all 0s and all 1s not used).

As stated before, the subnet mask for a class B address could extend into the fourth octet. For example, the subnet mask for a given IP address could be 255.255.255.192. This would indicate a 10-bit subnet mask and this would allow for 1022 subnets with 62 hosts on each of the subnets.

Figure 6.6*c* shows a subnetted network topology connected to the Internet. It is assigned a class B address and uses an 8-bit subnet mask. The Internet knows of the IP address 130.1.0.0. It does not know the subnets involved. This allows the Internet address (routing) tables to remain smaller.

Do you need more hosts than networks? Or more networks than hosts? Unless the network employs the routing protocol of OSPF when choosing a subnet mask, a compromise is usually the case in deciding the mask. If the site is assigned only one network number, there can be only one subnet mask assigned to the whole network. Without a routing protocol that supports multiple subnet masks within a single network number, expansion and network design must be thought about. The only other alternative is to assign multiple network numbers to a single site and, unless the network is private, these addresses are hard to come by.

If the network station moves to a new network, does the IP address for that station change? Like the telephone system, IP addresses must change when the network station is moved to a new network that employs a different network number. If the network station is moved on the same logical network, the IP address may remain the same. For example, if a network station moved to a different part of the network, but the network portion of the address (including subnetting) did not change, the whole IP address may stay the same. If the network station was moved to a network that has a different network address, the IP address of the network station must change.

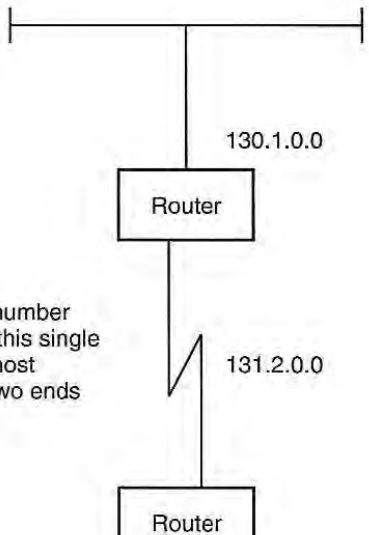
New with TCP/IP is the ability to assign more than one network number to the same network. This means that one network may employ more than one network number on the same physical cable plant. A router will take the steps necessary to allow network stations to converse on the network. Implementations are different; therefore, the amount of network numbers that may be assigned to the same cable plant varies.

In doing this, multiple class C network numbers may be assigned to the same cable plant. Class C addresses allow only for 254 host IDs per network number. This is a rather low number, and some sites will have more than 254 network stations attached to a cable plant. This means that multiple stations on the same cable plant may have different network addresses. A router must be used to translate between two stations that are located on the same cable plant with different network addresses.

An unfortunate circumstance involving IP addressing becomes very visible when an IP network address must be assigned to a serial link

(two routers using a leased phone line to connect). The serial link between two routers has its own network number assigned to it even though the only attachments will be the two routers that are linked together. A serial link will consume a network number and associated *host IDs. Therefore, a unique network number will be assigned and,* instead of being able to use all available hosts IDs, it will be possible to use only two hosts IDs (there will be only two addressable points on that network). Figure 6.7 depicts this situation.

The rest of the host IDs will be lost for that network number and will now be assigned and used for that serial link and will not be able to be assigned to any other links. If you have a large site that will encompass many serial links and you do not have the ability to assign a large number of network numbers, use subnet addressing and the routing protocol of OSPF. OSPF supports variable length subnet masks, which will collapse that serial link into two hosts within a network number and, therefore, no host numbers are wasted on serial links. Variable length subnet masks are beyond the scope of this book, but they allow a single network number to use multiple masks (unlike RIP). This allows more bits to be assigned back to the network, allowing a more efficient use of the address.



A unique network number has to be used on this single interface with two host assignments, the two ends of the serial link.

Figure 6.7 Serial line IP address assignment.

Note:

Class A address
fourth router fi

Class B address
field) field for su

Class C is tricky
ting this is allow
field. You need t

Address resolution
identify those stat
Token Ring, etc.).
works at the netw
not a physical or
layer addressing,
Networks (LANs)
This is known as
addresses identify
scheme used at th
Token Ring, etc.), t
if they know each

An RFC resolved
not affect the alrea
Address Resolution
station-address res
follows: If you are t
ber as the one you
use ARP to find th
ing a remote stati
than yours—is exp

Refer to Fig. 6.8.
network, the sourc
address. Station 12
net addressing is u
class B address is
is 1.1; hence the ad

With ARP, it is a
tion is already kno
vice or file on a r
names, explained
itself. To reduce ov

* Physical- and MAC-

Note:

Class A addresses can use the second, third, or fourth (not the whole fourth router field) field for subnets.

Class B addresses can use the third or fourth (not the whole fourth field) field for subnets.

Class C is tricky. The only field left is the single host field. Subnetting this is allowed, but you must use up to 6 of the bits in the fourth field. You need to have a couple of hosts somewhere!

Address resolution protocol (ARP). The IP address does not physically identify those stations on a high-speed local area network (Ethernet, Token Ring, etc.). IP addresses are meant to identify IP hosts and networks at the network layer of the OSI model. It is an IP address and not a physical or MAC address. (For more information on physical-layer addressing, please refer to Chap. 2). The designers of Local Area Networks (LANs) allotted 48 bits to identify a network attachment. This is known as their *physical address* or *MAC address.** Physical addresses identify stations at their data-link level. IP is an addressing scheme used at the network level. On a local area network (Ethernet, Token Ring, etc.), two communicating stations can set up a session only if they know each other's physical address.

An RFC resolved this problem. The resolution was simple and it did not affect the already established IP addressing scheme. It is known as *Address Resolution Protocol* or *ARP.* This is an IP-address-to-physical-station-address resolution (actual name is *binding*) and is explained as follows: If you are trying to connect to a host on the same network number as the one you are currently residing on, the TCP/IP protocol will use ARP to find the physical address of the destination station. (Finding a remote station—one with a different network number address than yours—is explained in a moment.)

Refer to Fig. 6.8. In order to attach to another station on a TCP/IP network, the source station must know the designation station's IP address. Station 129.1.1.1 wants a connection with 129.1.1.4 (no subnet addressing is used here). Therefore, the network address of this class B address is 129.1.0.0 and the personal computer's host address is 1.1; hence the address 129.1.1.1.

With ARP, it is assumed that the IP address of the destination station is already known either through a name service (a central service or file on a network station that maps IP addresses to host names, explained in more detail later) or by using the IP address itself. To reduce overhead on the network, most TCP network stations

* Physical- and MAC-layer addresses are synonymous throughout this book.

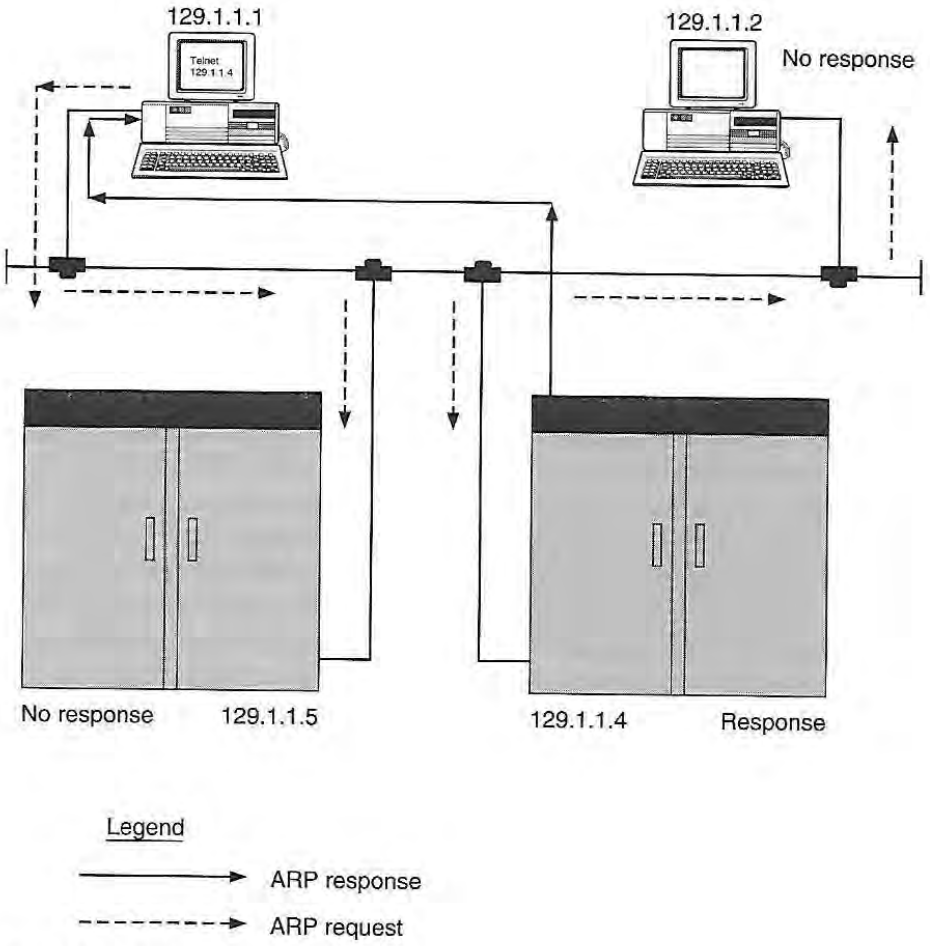


Figure 6.8 ARP request and response.

will maintain a LAN physical-address-to-IP-address table on their host machine. The ARP table is nothing more than a section of RAM memory that will contain data-link physical (or MAC addresses) to IP address mappings that it has learned from the network. Although vendor independent, the first entry in the table should contain the physical address and the IP address of the station on which ARP is currently residing. The second entry in this table may contain the broadcast mapping for the physical address.

Once the IP address is known for the destination station, IP on the source station will first look into its ARP table to find the physical address for that destination IP address. If a mapping was found, no ARP request packet will be transmitted onto the network. IP can bind (place the physical addresses on the data-link headers of the packet) the IP address with the physical address and send the IP datagram to the data link for transmission to the network. (See Table 6.1.)

TABLE 6.1 ARP Table

Physical address
02-60-8C-01-02-03
FF-FF-FF-FF-FF-FF
FF-FF-FF-FF-FF-FF
00-00-A2-05-09-89
08-00-20-67-92-89
08-00-02-90-90-90

If the address is
build an ARP req
broadcast mode
packet is shown in
mode, all stations
only the host with
will reply to the req
physically address
When the host w
will respond with
broadcast but with
inside the ARP rep

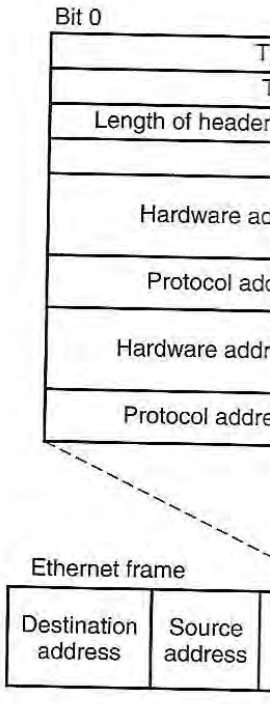


Figure 6.9 (*a*) ARP packe

TABLE 6.1 ARP Table for Station 129.1.1.1

Physical address	IP address
02-60-8C-01-02-03	129.1.1.1
FF-FF-FF-FF-FF-FF	148.9.255.255
FF-FF-FF-FF-FF-FF	255.255.255.255
00-00-A2-05-09-89	129.1.1.4
08-00-20-67-92-89	129.1.1.2
08-00-02-90-90-90	129.1.1.5

If the address is not located in the ARP table, the ARP protocol will build an ARP request packet and send it physically addressed in broadcast mode (destination address FF-FF-FF-FF-FF-FF). This packet is shown in Fig. 6.9*a*. Since the packet is sent out in broadcast mode, all stations on the physical network will receive the packet, but only the host with that IP address will reply. In Fig. 6.8, host 129.1.1.4 will reply to the request packet with an ARP response packet. It will be physically addressed to station 129.1.1.1.

When the host whose IP address is in the request packet responds, it will respond with an ARP reply packet not addressed to destination broadcast but with the source address set to its address (physically and inside the ARP reply packet), and the destination address is the origi-

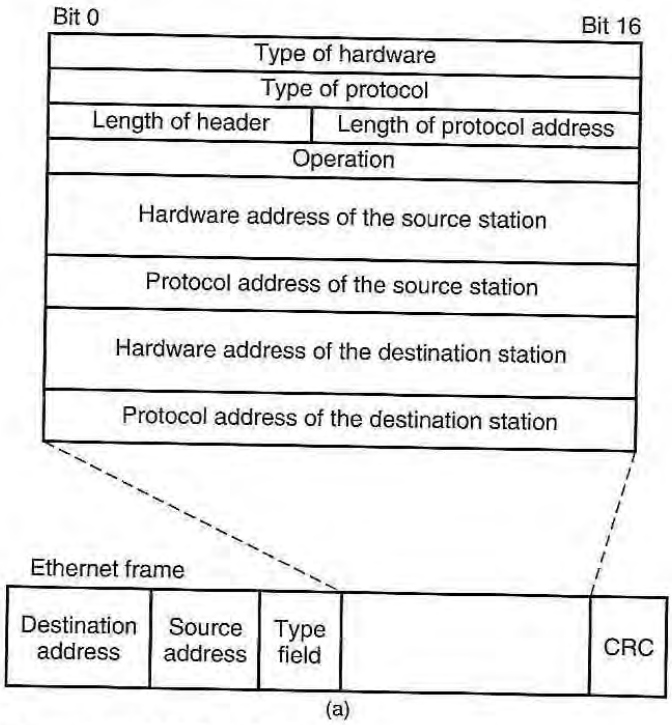


Figure 6.9 (*a*) ARP packet format.

nator. Once the originator of the request receives the response, it will extract the physical address from the source address in the packet and update its ARP table. Now that it has the mapping, it will try to submit its IP datagram to the destination station using the proper addresses (IP and physical address).

This process is completed as an involuntary act to the user. The user will typically be using one of TCP's applications (TELNET for terminal service, SMTP for mail service, or FTP for file transfer service) attempting a connection. This ARP request and reply will happen automatically in the connection. Most TCP vendors supply a utility program that allows a user see the entries in the ARP table.

To improve the efficiency of the protocol, any station on the physical network that received the ARP packet (request packet) can update the ARP cache. In looking at the packet format, the sender's physical and IP addresses will be in the packet. Therefore, all stations can update their ARP tables at the same time.

Figure 6.9*a* shows the ARP packet format. It is encapsulated in an Ethernet packet as shown. (See also Table 6.2. This ARP process works for stations communicating with each other on the same LAN (the

TABLE 6.2 Definition of the ARP Packet

Type of hardware	Normally indicates IEEE 802 network for local area networks. It could also indicate other types of networks.
Type of protocol	Would indicate IP for TCP/IP networks. It could also indicate AppleTalk.
Length of header	Indicates the length of the ARP header.
Length of protocol address	Since this header is used for other types of networks (AppleTalk), this field indicates the length of the protocol address (IP or AppleTalk address, not the physical address).
Operation	Indicates the operation of the header: ARP request or response.
Address of the source station	Physical address of the source station. This would be filled in by the requester.
Protocol address of the source station	IP address of the source station.
Hardware address of the destination station	Physical address of the destination station. This field is usually, but not always, set to 0s if it is a request packet. This field would be set to the physical address of the destination station if it is an ARP reply. This field is filled in by the responding destination station.
Protocol address of the destination station	Set by the source station (ARP requester). This will contain the IP address of the wanted destination station. Only a station whose IP address matches this will respond to the ARP request.

same network nu
cess still works,
explained later.

Rules for ARP

1. ARP is not a p
headers.
2. ARP requests
physical broa
logical networ
3. Since ARP is
assigned to id
0806 is an AR
the 0800 Ethe
request or AR
types. Some s

* All stations broa
by a router.

1. IP requests a MAC
2. Searches ARP cac
3. ARP cache will eit
4. If the address is nc
the network (locali
reply to IP with the
5. Upon an ARP repl
back to IP.

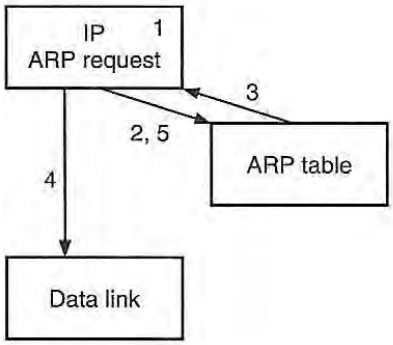
Figure 6.9 (*b*) the A

same network number). If they are not on the same LAN, the ARP process still works, but an address of a router will be found. This is fully explained later. Summarizing the ARP process is shown in Fig. 6.9*b*.

Rules for ARP

1. ARP is not a part of the IP protocol and therefore does not contain IP headers.
2. ARP requests and responses are transmitted with a destination physical broadcast address (all Fs) and therefore never leave their logical network.*
3. Since ARP is not part of the IP protocol, new Ethertypes were assigned to identify this type of packet. 0806 is an ARP request and 0806 is an ARP reply. Some ARP implementations can be assigned the 0800 Ethertype, for IP will be able identify the packet as an ARP request or ARP reply packet. Not all implementors of IP use these types. Some still use the Ethertype of 0800 for ARP.

* All stations broadcasts (all FFs in the MAC destination header) will not be forwarded by a router.



1. IP requests a MAC address to IP address translation.
2. Searches ARP cache table for possible entry.
3. ARP cache will either return the MAC address (if it is in the table) or not.
4. If the address is not in the table, generates an ARP request packet to the network (localized packet). If the address mapping is in the table, reply to IP with the MAC address.
5. Upon an ARP reply, ARP updates its table and reports the address back to IP.

(b)

Figure 6.9 (*b*) the ARP process.

TABLE 6.3 Ethertype Field Entries for ARP

ARP request	0806h
ARP reply	0806h

4. ARP contains an aging entry to delete entries that have not been used for a period of time. This reduces the ARP look-up time and saves memory.
5. If a machine submits an ARP request for itself, it must reply to the request.

Reverse address resolution protocol (RARP). This protocol is used when a network station knows its MAC address but does not know its IP address. When would this happen? This is a common application for diskless workstations (Sun Microsystems, for example).

The requesting client machine will send out a RARP request to a server located on the physical network somewhere that has the RARP server service running on it. This RARP server will respond to the request with that particular station's IP address.

The packet format for a RARP packet is the same for as for ARP. The only difference it that the field that will be filled in will be the senders physical address. The IP address fields will be empty. A RARP server will receive this packet, fill in the IP address fields and reply to the sender. It is the opposite of the ARP process.

Proxy ARP. One last variation on ARP is called Proxy ARP.* Proxy ARP is the capability of a router to be able to respond to an end station (host) that does not support subnet addressing. By the time IP subnet addressing became adopted, there were already a tremendous amount of hosts established with TCP/IP as their networking protocol. Subnetting was implemented later, so if a host did not support subnet addressing, it could incorrectly mistake an IP network number (the subnet portion of the IP address) for a host number. The router tricks the transmitting station into believing that the source station is on the local LAN.

As shown in Fig. 6.10, the host on network A may not attach to other devices on network B since it has no concept of subnetting. When the IP layer does a comparison of its address to the destination IP address, it will think the packet is locally addressed and, therefore, will not transmit it to a router for delivery. Instead it will invoke the algorithm

* This is also known as promiscuous ARP or ARP hack.

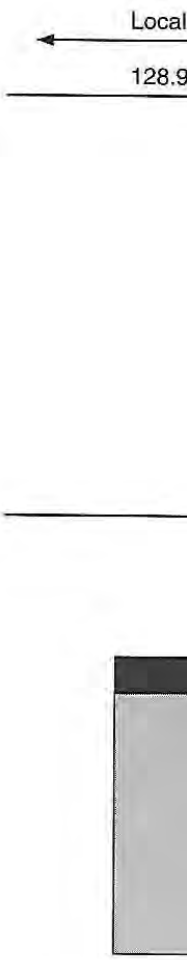


Figure 6.10 Proxy

to deliver it lo
packet will not
tion is not av
allowed within
routing protoc
routing updat

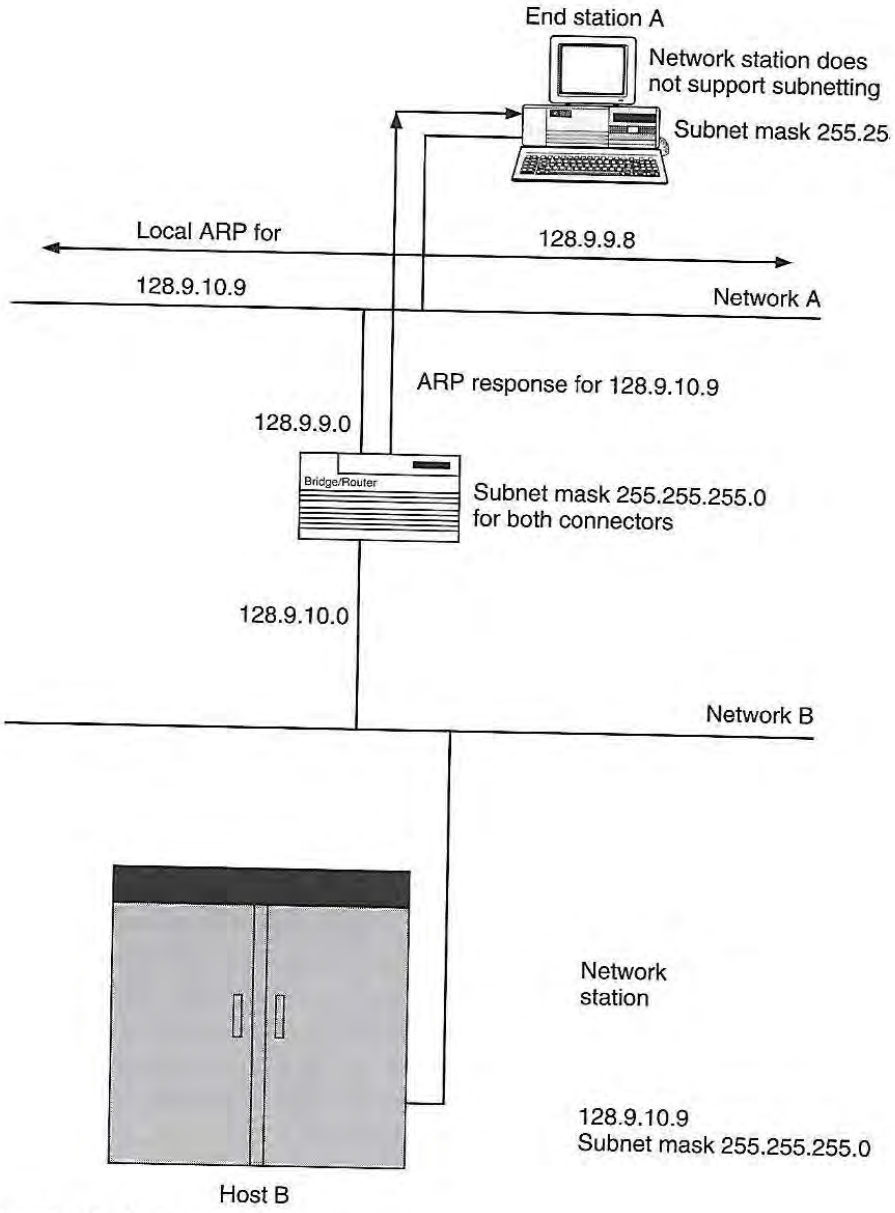


Figure 6.10 Proxy ARP.

to deliver it locally. Unless the local router is running proxy ARP, the packet will not be answered and the host will think the destination station is not available. Furthermore, different subnet masks are not allowed within the same network number (unless you are running a routing protocol that supports the broadcasting of subnet masks in its routing updates, like OSPF).

The problem is that the host is looking on the wrong network. Refer to Fig. 6.10. End station A thinks host B is on the local LAN. By deciphering the IP address, the first two fields are the same. Host B supports subnet addressing and end station A does not. Therefore, end station A will send out a local ARP request packet when it should be submitting the packet to the router so that it can deliver the packet to the end station. The router which supports subnetting will look up the ARP request and then notice that the subnetwork address is in its routing table. If the router has proxy ARP enabled, the router will answer for host B. End station A will receive this response and think it is from host B. There is nothing in the physical address of a packet to indicate where it came from.

The host will then submit all packets to the router and the router will deliver them to end station A. This communication will continue until one end terminates the session.

Proxy ARP is a very useful protocol for those networks that have been using bridges to implement their IP network and are moving to a router environment. Proxy ARP allows the network to migrate to a routed environment. There are other useful situations for proxy ARP, but its use is waning. Today, most hosts on a TCP/IP internet support subnet masking and most IP networks are using routers.

A potential problem in using proxy ARP is for those networks that implement the mechanism to ensure single IP addresses on are each network. Most TCP/IP implementations allow users easy access to their network number (that is, they can change it with a text editor). This allows any hacker to change his or her number to another in order to receive datagrams destined for another host. Some implementations of TCP/IP will detect for this. Routers that implement proxy ARP will get caught, for they will answer for any station on a different network, thereby giving the impression that there is one physical address to multiple IP addresses. There is a trust on any IP network that IP addresses will not be arbitrarily assigned. There should be one IP address for each physical address on a internet.

IP routing

Routing fundamentals: interior gateway protocols. To make any network more manageable, it will be split into many networks. The interconnection of these networks is accomplished by routers. Routers enable data to be forwarded to other networks in a very efficient manner. It will always be easier to manage many smaller networks than it will be to manage one large network. In order for routers to forward data to other networks, they use special protocols to enable them to internally draw a map of the entire internet for the purposes of routing. To accom-

plish this, there ar
tocols (IGPs) and
gateway protocol t
ered in this book.
ing protocol will be
the Routing Inform

In the previous
routers, and packe
own. This section w

Referring to Fig.
ent networks. In o
employ the use of a
ical networks with
can be interconnec

Figure 6.11 A routed env

plish this, there are two types of protocols used. Interior Gateway Protocols (IGPs) and Exterior Gateway Protocols (EGPs). The exterior gateway protocol that is used with IP (known as EGP) will not be covered in this book. For the purposes of this book, only one type of routing protocol will be shown. It is an Interior Gateway Protocol known as the Routing Information Protocol (RIP).

In the previous section, the text referred to network numbers, routers, and packets that must get to a network different from their own. This section will fully explain those items.

Referring to Fig. 6.11, end station B and host A are located on different networks. In order to communicate with one another, they must employ the use of a router. A TCP/IP internet consists of multiple physical networks with network devices attached. All these local networks can be interconnected by special devices known as routers to form an

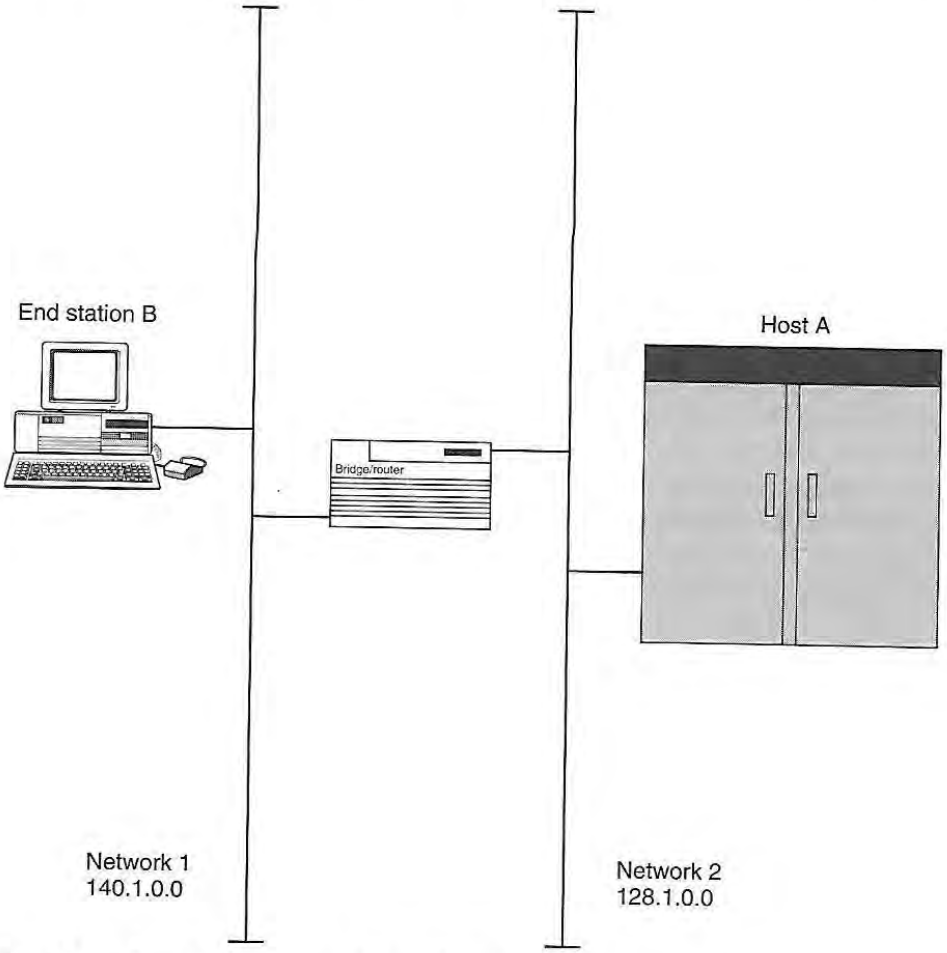


Figure 6.11 A routed environment (no subnets).

internet. These routers are used to connect two or more networks and allow internet traffic to pass through them.

Routing data. *Throughout this section, different* network numbers will be used. The examples will not employ the use of subnets. Subnets effectively act like network numbers. Subnetworks are also separated by a router. For example, in Fig. 6.11, the network numbers could be 140.1.1.1 on the network with end station B and 140.1.2.1 on the network containing host A. Using a subnet mask of 255.255.255.0 would yield two different networks: 140.1.1.0 and 140.1.2.0. For simplicity in explaining routers, I have chosen to use completely different network numbers.

For a packet to be transmitted and received on a local network or an internet, it must be routed to the remote network station by using the IP network layer. Packets may be routed on a local LAN (two stations on the same physical network, i.e., not separated by a router). This is known as *direct routing.* If the packet is destined for another network, it must be routed through a device known as a router. This is known as *indirect routing.* Therefore, data may be transferred using two types of routing: direct and indirect.

How does a network station know whether the packet has to be directly or indirectly routed? For the network station, it is a relatively simple process. The whole basis for routing is in the IP network number assigned to the network station.

Remember from the previous section on addressing that an IP address contains the network number as well as the host number. With the first 1, 2, 3, or 4 bits of the 32-bit IP network address identifying the class of the address, this allows for any network station (workstation or router) to quickly extract the network portion out of the whole IP address.* In other words, by reading up to the first four bits of the IP address, a network station can quick determine how much of the IP address to read to determine the network number of the address. The sending station will compare the packet's destination network number to that of its own network number. If the network number portion of the destination IP address matches it own, the packet can be routed directly on the local LAN, without the use of a router.

Once this determination is made, and the packet is destined for a local route, the network station would check its ARP table to find the IP to physical address mapping. If one is found, the packet is physically addressed and transmitted onto the network. The physical destination

* For a subnetted network, two actions are required: one to read the first bits of the IP address to identify the class and another to check the subnet mask to identify if a subnet is being used.

address (located
station. If the s
request process i
If the host res
(not on the local
the services of a r
ical destination a
if necessary, to fi
packet to the rou
either its locally
address and subn
or to another rou
tion physical addr
station. This type
address is embed
Sending a pack
using both direct
to be delivered ac
it to the router for
No matter wheth
nected to that r
routers to reach i
use *direct routing*
It should be no
alter the original
Live) field and the
is received by a r
the router will de
packet based on
gram's header con
error-detection fie
Since the TTL fiel
all the networks a
the only alteratio
trailers. The IP a
the datagram trav
Routers deliver
antee delivery of a
provides best effo
establish sessions

*ARPs are used only
tion or to find the MA

address (located in the data-link header) will be that of the receiving station. If the station's address is not in the ARP cache, the ARP request process is invoked.

If the host resides on a network with a different network number (not on the local LAN), then the transmitting station will have to use the services of a router. The transmitting station will address the physical destination address of the packet to that of the router (using ARP, if necessary, to find the physical address of the router) and submit the packet to the router.* The router will, in turn, deliver the packet to either its locally attached networks (finding the destination's physical address and submitting the datagram directly to that network station) or to another router for delivery of the data. Notice here, the destination physical address is that of the router and not the final destination station. This type of routing is indirect routing. The destination IP address is embedded in the IP header.

Sending a packet to its final destination may be accomplished by using both direct and indirect routing. For example, when a packet is to be delivered across an internet, the originating station will address it to the router for delivery to its final network. This is *indirect routing.* No matter whether the final destination network ID is directly connected to that router or whether the packet must traverse a few routers to reach its final destination, the last router in the path must use *direct routing* to deliver the packet to its destination host.

It should be noted here that none of the IP routing protocols will alter the original IP datagram, with two exceptions: the TTL (Time-to-Live) field and the Cyclic Redundancy Check fields. If an IP datagram is received by a router and it has not arrived at its final destination, the router will decrement the TTL field. If TTL > 0, it will forward the packet based on routing table information. Otherwise, the IP datagram's header contents will remain the same (with the exception of an error-detection field known as the Cyclic Redundancy Check (CRC). Since the TTL field changed, the CRC must be recalculated throughout all the networks and routers that the datagram traverses. Otherwise, the only alterations that are made are to the data-link headers and trailers. The IP addresses in the IP header will remain the same, as the datagram traverses any routers in the path to its destination.

Routers deliver on a connectionless basis and therefore do not guarantee delivery of any packet. They operate at the network layer which provides best effort or connectionless data transfer. Routers do not establish sessions with other routers on the internet.

---

*ARPs are used only on the local network to either find the MAC address of a local station or to find the MAC address of a router for a nonlocal destination.

These routers forward packets based on the network address of the packet (in the IP header) and *not* on the physical address (the 48-bit address for Ethernet and Token Ring) of the final destination (the receiver) of the packet. When the router receives the packet, it will look at the final network address (embedded in the IP header of the packet) and determine how to route the packet. Routers only route packets that are directly addressed to them. They do not operate in promiscuous mode (watching all LAN traffic).*

For a full explanation of the router forwarding process, refer to the flowchart in Fig. 6.12.

Before complete confusion takes over here, there are some entities that need to be explained about the IP layer that allow the internet to operate. In other words, when a router receives a packet, how does it know where and how to send these packets? In order for a packet to be delivered through a router, the router must know which path to deliver the packet to in order for the packet to reach its final destination. This is accomplished through IP routing algorithms, which involves two steps: maintaining a table of known routes (network numbers) and learning new routes (network numbers) when they become available.

Routing data based on routing tables. Some routing protocols allow routers to determine network paths based on concept known a vector-distance. Distance-vector means that the information sent from router to router is based on an entry in a table consisting of <vector, distance>. Vector means the network number and distance means what it costs to get there. The routers exchange this network reachability information with each other by broadcasting their routing table information consisting of these distance-vector entries.

Each entry in the table is a network number (the vector) and the amount of routers (distance) that are in-between it (the router) and the final network (indicated by the network number). This distance is sometimes referred to as a *metric*. For example, if the source station wants to transmit a packet to a destination station that is four hops away, there are probably four routers separating the two networks.

Any time a datagram must traverse a router (thereby passing through a new network number) it is considered a hop (metric). For RIP, the maximum diameter of the internet is 15 routers (hops). A distance of 16 is an indication that the network is not reachable. In other words, if the network is more than 15 routers away, it is considered unreachable.

As shown in Table 6.4, each router will contain a table with starting entries of those networks that are directly attached to it. For a router

* Multiprotocol routers operate in promiscuous mode, but this concept is beyond the scope of this book.

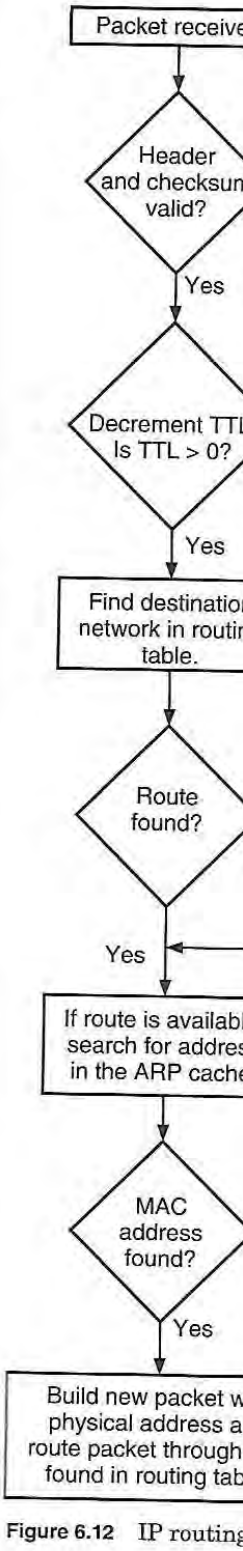


Figure 6.12 IP routing

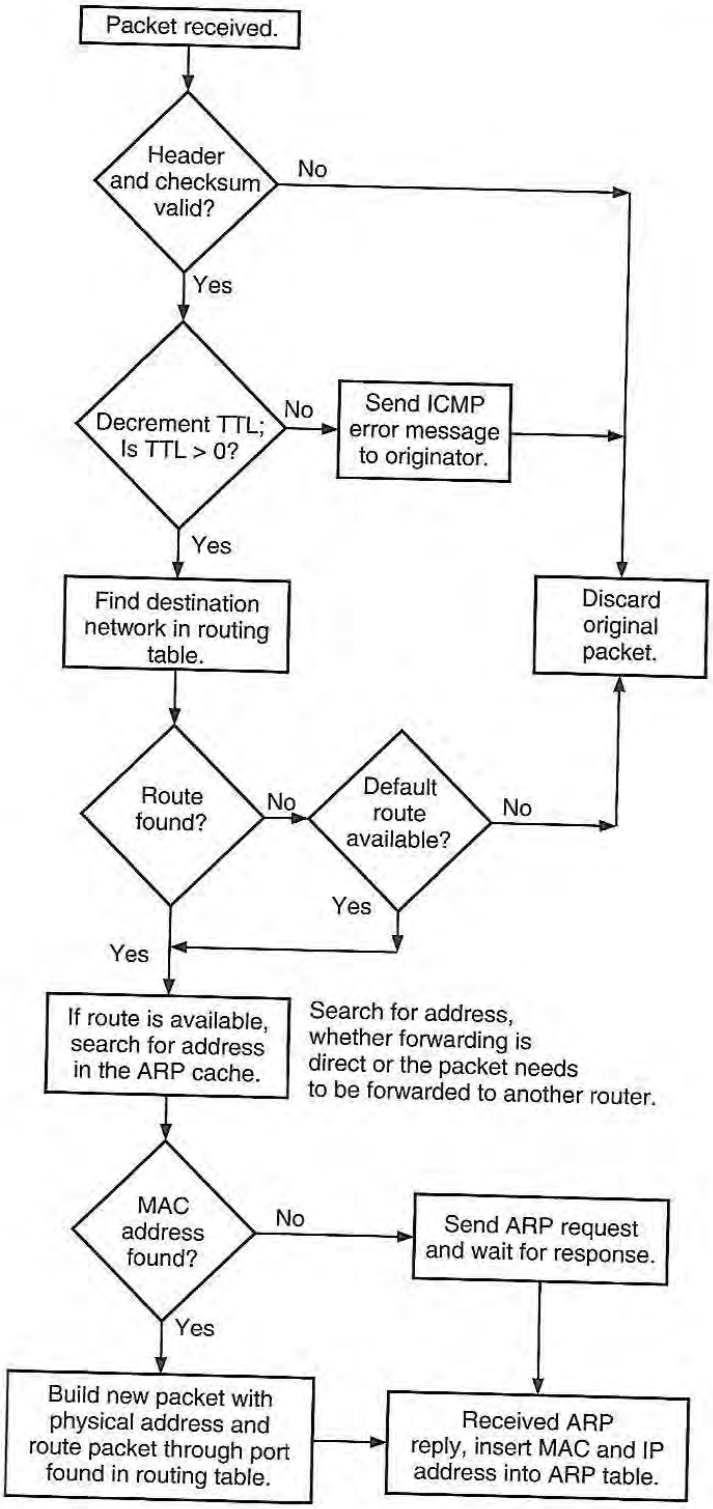


Figure 6.12 IP routing flowchart.

TABLE 6.4 Routing Table Entries*

Network number	Next router to deliver to	Hops	Learned from	Time left before delete	Port
134.4.0.0	Direct route	1	RIP	xxx	1
133.3.0.0	Direct route	1	RIP	xxx	2

* Actual verbiage of the entries in the table may differ from vendor to vendor.

that has only two network connections (there are no other routers on the internet), the initial entries in the table would look like the following:

There are actually more header entries in a routing table, but the significant portions are shown in Table 6.4. From this table, we know that networks 134.4.0.0 and 134.3.0.0 are directly connected to this router. Network 134.4.0.0 is assigned to port 1 of the router. It is running the RIP protocol, and xxx indicates how long the route has before it is deleted from the table.

Refer to Fig. 6.13. This figure shows the updating process of RIP. Parts of a router's table (network number and the hop count) will be

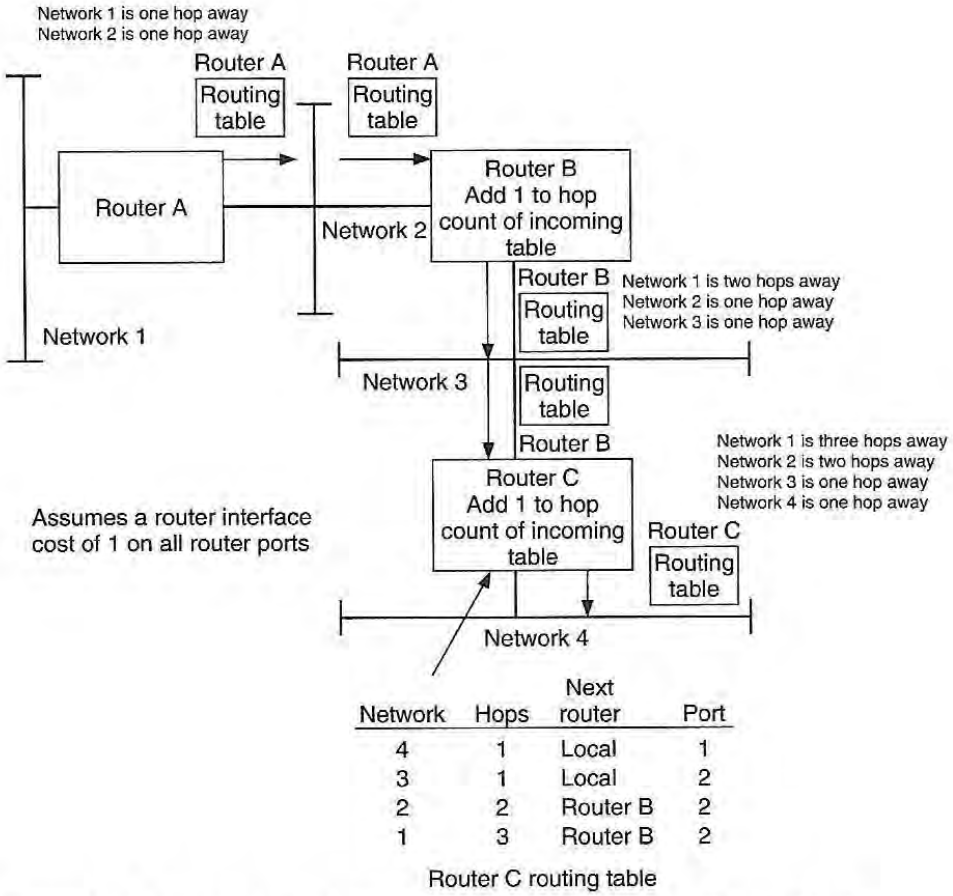


Figure 6.13 Routing table updates (RIP); split horizon is not implemented here.

broadcast to the local networks to which the router has directly attached. There are a few exceptions, which will be explained in a moment. Any router that is located on the same network will receive the packet, read the routing table data, update its table if needed, and then discard the packet. Routers will not forward any update packets they receive. All participating routers will accomplish this. As each table is received, the routers are building a picture of the network. As each broadcast is transmitted, more and more information is being propagated throughout the network. Eventually, all routers will know of all networks in the internet.

The tables are encapsulated in a packet which is physically and logically addressed to broadcast mode. All local routers will receive and interpret the table against their own tables, but they will not forward the received table through their network ports. (Remember that routers will not forward packets addressed to broadcast.) Instead, once a router has updated its table, it will then broadcast its newly updated table out its directly attached ports. All routers on that network will receive the datagram and update their tables. This is how routing table information is propagated throughout a network.

There are three possibilities that can cause a router to update its existing table based on just-received information:

1. If the received table contains an entry to a network with a lower hop count, it will *replace its entry* with the new entry containing the lower hop count.
2. If a network exists in the just-received table that does not exist in its own table, it will *add the new entry.*
3. If the router forwards packets to a particular network through a specified router (indicated by the next hop router address) and that router's hop count to a network destination changes, it will *change its entry.* In other words, if router A normally routes data for a network X through router B, and router B's hop count entry to that network changes, router A *changes its entry.*

Figure 6.13 shows what happens when router A submits it routing table out of its port connected to network 2. (For simplicity, the figure will show the updating through one port only. In reality, routing tables are submitted out all ports of a router, with a few restrictions on which entries of the table get transmitted.)

Router A transmits its table containing two networks: 1 and 2. Each of these networks is one hop away (they are directly connected). Router B will receive this packet and it will add one to each hop count entry in the received table. (This is accomplished assuming the RIP cost assigned to that port of router B is 1. It could be set to something else.)

Router B would examine its table and notice that it does not have an entry for network 1. It will add this entry to its table as: Network 1, available through port 1, two hops away. It will then check the next entry. Network 2 will not be added, for router B already has network 2 in its table with a cost of 1. Since the incoming table reports network 2 has a cost of 2, router B will ignore this entry. (There are rules that will prevent router A from sending out information about network 2 and these rules will be discussed later.)

Once its table is updated, router B will transmit its table out every 30 seconds its ports (again, for simplicity only one port is being shown). Router C will receive this table from router B and will perform the same steps as router B. Eventually, all information about all networks on the internet will be propagated to all routers.

IP routing tables. Referring to Tables 6.5 and 6.6, the significant entries in a routing table for the network shown in Figure 6.14 consist of three elements:

1. Network number
2. Hops to that network number
3. The next router in the path to that network

Routing tables vary, depending on the update mechanism used. The following table is a sample of a routing table used by the routing information protocol (RIP) for the IP protocol.

TABLE 6.5 Router A's Routing Table

Network number	Next router to deliver to	Hops	Learned from	Time left before delete	Port*
132.2.0.0	Direct route	1	RIP	xxx	1
133.3.0.0	Direct route	1	RIP	xxx	2
130.1.0.0	Direct route	1	RIP	xxx	3
134.4.0.0	133.3.0.4	2	RIP	xxx	2

* This indicates the physical port on the router from which the router learned the route and where it will forward a packet to if one is received with the network number.

TABLE 6.6 Router B's Routing Table

Network number	Next router to deliver to	Hops	Learned from	Time left before delete	Port
134.4.0.0	Direct route	1	RIP	xxx	1
133.3.0.0	Direct route	1	RIP	xxx	2
132.2.0.0	133.3.0.3	2	RIP	xxx	2
130.1.0.0	133.3.0.3	2	RIP	xxx	2

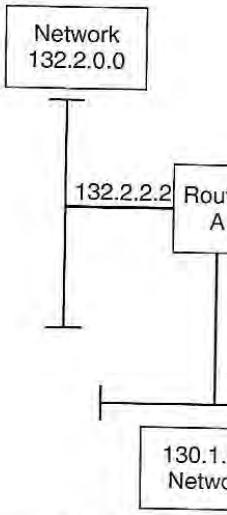


Figure 6.14 IP addre

Tables 6.5 and

Network numb

Next router to be delivered to A directly conn the router, sinc networks.

Hops. This is must traverse a local route.

Learned from. (i.e., RIP, OSPF ally an entry in

Time left to dele deleted from th

Port. The phy received inform

The routing informa

How the table is c

routers update ea advent of dynamic supported manual

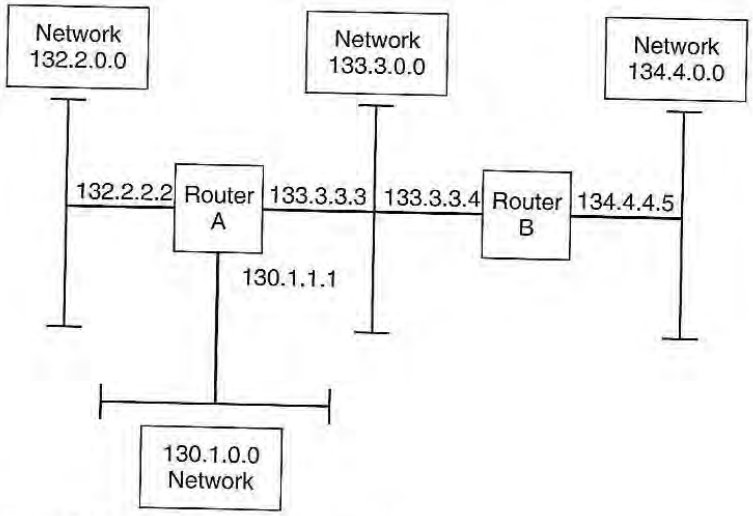


Figure 6.14 IP addressing for routers.

Tables 6.5 and 6.6 are defined as follows:

Network number. A known network ID.

Next router to deliver to. The next router that the packet should be delivered to if the destination network is not directly connected. A directly connected network is one that is physically connected to the router, since most routers today have more than two connected networks.

Hops. This is the metric count of how many routers the packet must traverse before reaching the final destination. A one indicates a local route.

Learned from. Since many routing algorithms may exist in a router (i.e., RIP, OSPF, and EGP may exist in the same router), there is usually an entry in the table to explain how the route was acquired.

Time left to delete. The amount of time left before the route will be deleted from the table.

Port. The physical port on the router from which the router received information about this network.

The routing information protocol (RIP)

How the table is created. Dynamic updating is the process by which routers update each other with reachability information. Before the advent of dynamic updates of routing tables, most commercial vendors supported manual updates for their router table. This meant entering

network numbers, their associated distances, and the port numbers manually into the router table. As networks grew larger, this became a cumbersome way of building tables. RIP is the protocol that enables automatic updates of router tables.

The RIP algorithm is based on the distance-vector algorithms just described. RIP placed the fundamentals of distance-vector in a simple routing algorithm. It was first devised by Xerox Corporation as the routing algorithm used by Internet Datagram Protocol of XNS.

An RFC was developed for the XNS RIP standard to be incorporated into IP, and it was formally adopted by the IAB in 1988. Although it was not primarily intended as *the* routing algorithm for TCP, it gained widespread acceptance when it became embedded into the Berkeley UNIX operating system through a service known as routed* (pronounced "route d"—d is for the daemon process that runs the protocol in UNIX). The protocol was actually in widespread use much before 1988, for the protocol was distributed in Berkeley 4BSD UNIX and gained widespread acceptance through the vast number of installations of this operating system.

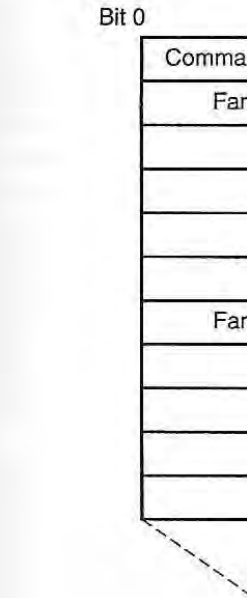
With RIP information, any router knows the length of the shortest path (not necessarily the best) from each of its neighbor routers (routers located on the same network) to any other destination. Keep in mind that RIP understands only the shortest route to a destination. This route may not be the fastest. RIP understands only hop counts. For example, there may be two paths to a destination—one that traverses two T1† lines (three hops) and another that has 2 hops but is a 9600 baud serial line. RIP would pick the 9600 baud line, for it is shorter (two hops). There are variations of RIP that allow the network administrator to assign an arbitrary RIP hop count or cost to a route to disallow for this RIP problem. This solves one problem but creates another. This incremented RIP number adds to the upper limit of a 16-hop diameter in RIP.

As shown in Fig. 6.15*a*, the RIP packet is quite simple. This figure shows the RIP header and data encapsulated in an Ethernet packet.

There are two types of RIP packets that traverse a network (indicated by the command field). One type of RIP packet is to request information and the other is to give information, a response packet. Most RIP packets that traverse a local network will be the periodic RIP table updates. Remember that RIP packets will not leave their local network. All participants in the RIP protocol (for example, routers) will receive the packet, update their tables if necessary, and then discard the packet. They will compute the reachability of net-

* Remember that TCP/IP was also embedded into the Berkeley 4BSD operating system by a research grant provided by DARPA.

† T1 is 1.544 Mbps. It is a serial data line provided by the telephone characters.



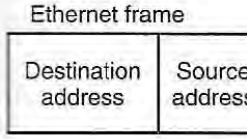


Figure 6.15 (*a*) RIP pa

works based on a
count entry, and th
mindful of a protoc
The fields in the

Command	*L*
1	R
2	R
3–4	T
5	S

Version. Used

Family of net x.
is used to indica
2 for IP. Since X

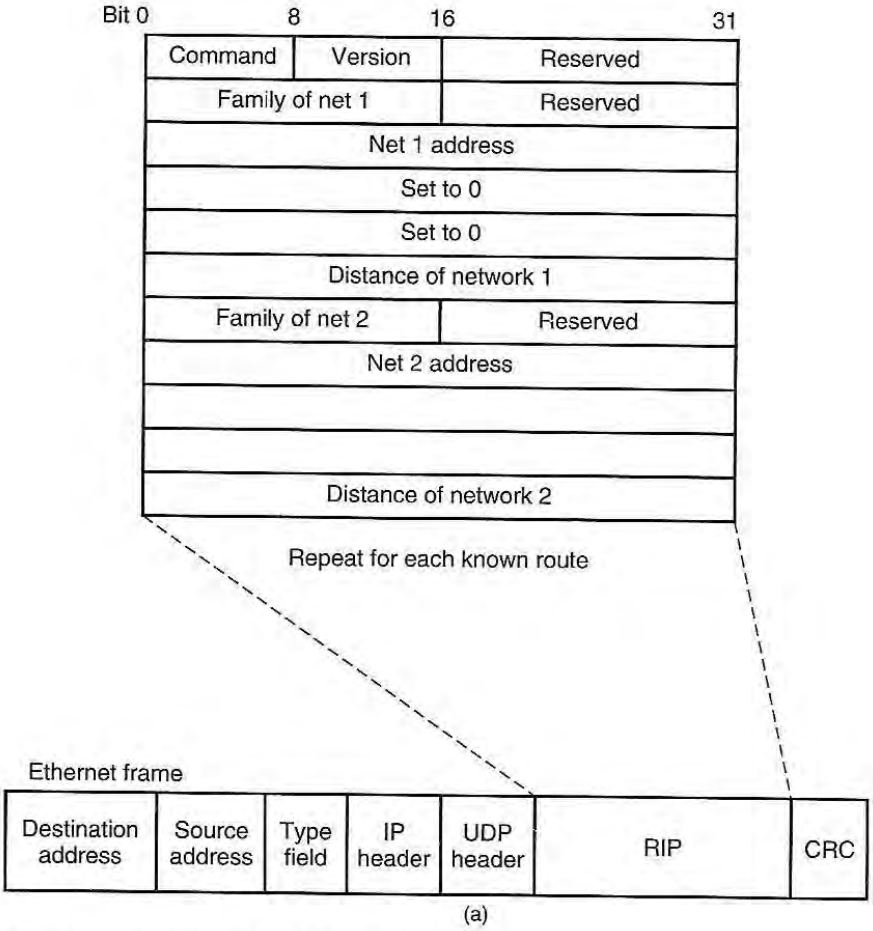


Figure 6.15 (*a*) RIP packet format.

works based on adding a cost (usually 1) to the just-received tables or count entry, and then broadcast their tables out their ports (usually being mindful of a protocol named split horizon, which is explained a little later).

The fields in the RIP packets are:

Command	*Description*
1	Request for partial or full routing table information
2	Response packet containing a routing table
3–4	Turn on (3) or off (4) trace mode (obsolete)
5	Sun Microsystems Internal use

Version. Used to indicate the version of RIP. Currently set to 1.

Family of net x. Used to show the diversity of the RIP protocol. This is used to indicate the protocol that owns the packet. It will be set to 2 for IP. Since XNS could possibly be run on the same network as IP,

the RIP frames would be similar. This shows that the same RIP frame can be used for multiple protocol suites. AppleTalk, Novell NetWare's IPX, XNS, and TCP/IP all use the RIP packet. Each packet is changed a little for each protocol. Refer to Chaps. 7, 5, 4, and 6, respectively, for more information on those protocols.

IP address. Indicates the IP address of a specific destination network. This would be filled in by the requesting station. An address of 0.0.0.0 indicates the default route, explained later. The address field needs only 4 bytes of the available 14 bytes, so all other bytes must be set to 0.

If this is a request packet and there is only one entry, with the address family ID of 0 and a metric of 1, then this is a request for the entire routing table.

As for the distance to network field, only the integers of 1 to 16 are allowed. An entry of 16 in this field indicates that the network is unreachable.

The next entry in the field would start with the IP address field through the metric field. This would be repeated for each table entry of the router to be broadcast. The maximum size of this packet is 512 bytes.

Although not mentioned until later, the RIP protocol relies on the transport-layer protocol of User Datagram Protocol (UDP, discussed in the next section on transport-layer protocols). In this will be the specification for the length of the RIP packet. Also, for those interested, RIP operates on UDP port number 520 (port numbers are discussed in the UDP section). This encapsulation in shown in Fig. 6.15*b*.

Finally, both routers and individual hosts can implement the RIP protocol since RIP has two entities: *active* and *passive*. As active, RIP both listens to RIP broadcasts from other network stations (and builds

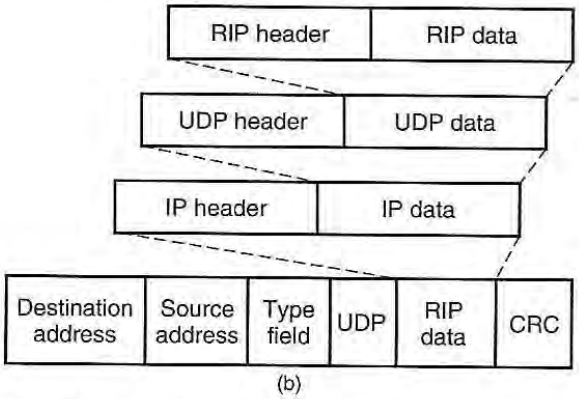


Figure 6.15 (*b*) RIP packet encapsulation.

it own internal ta
requests from oth

In passive mod
own tables or it r
these tables.) It
information from
sive end is used
when using simp
TCP/IP or maybe
broadcast update
Today, most DOS
explained later.

Using the RIP p
the shortest route
the packets to. Th
both the table an
of a default gatew
tined for remote r
fied gateway for i
shortest path to t
overhead to the n
that are on the r
maintain their ow

Default routes. O
default route. This
AppleTalk, IPX,
places: the router

For an end stat
RIP protocol, the
default router (co
The IP layer in th
network is not loc
must be used. Ins
tion may submit
default route num
will reach its fina
route, it would se
ter route. This wil

A router may a
0.0.0.0 in its rout
receives a packet
The router will for
an assigned defau

it own internal tables) and transmits its own broadcasts to respond to requests from other stations.

In passive mode, RIP listens only for RIP updates. (It may build its own tables or it may not. If it does build a table, it will not broadcast these tables.) It will build a table so that it will not have to request information from other routers on the network when it needs to. Passive end is used for nonrouting network stations. This is also useful when using simplex devices such as a DOS personal computer with TCP/IP or maybe a terminal server. These devices have no reason to broadcast updates, but have every reason to listen for the updates. Today, most DOS PC computers will use a concept of a default gateway, explained later.

Using the RIP passive protocol allows the host to maintain a table of the shortest routes to a network and designates which router to send the packets to. This does consume a considerable amount of RAM for both the table and the algorithm. Without it, TCP/IP requires the use of a default gateway entry which specifies that when a packet is destined for remote network, the host must submit the packet to a specified gateway for it to process, even if this gateway does not have the shortest path to that network. Passive implementations add no more overhead to the network, for they listen only to routing table updates that are on the network. Without passive RIP, these devices had to maintain their own tables or implement a default route.

Default routes. On a TCP/IP network, there is a concept known as the *default route.* This is not part of any other network protocol (i.e., XNS, AppleTalk, IPX, etc.). The default route can be maintained in two places: the router and the end station.

For an end station that does not support the passive function of the RIP protocol, thereby allowing it to find a route dynamically, the default router (commonly called a default gateway) is assigned to it. The IP layer in the end station would determine that the destination network is not local (no direct routing) and that the services of a router must be used. Instead of implementing the RIP protocol, the end station may submit the packet to the default router as assigned by the default route number. The router will take care of ensuring the packet will reach its final destination. If that router did not have the best route, it would send a message to the end station to inform it of a better route. This will be explained later.

A router may also be assigned a default route. It is indicated by a 0.0.0.0 in its routing table. This is implemented for when a router receives a packet and does not have the network number in its table. The router will forward the packet to another router for which it has as an assigned default route. This means that when a router has received

a packet to route, and its table does not contain the network number indicated in the received packet, it will forward the packet to another router in hopes that the default router will have the network number in its table and will be able to properly forward the packet. That router will receive the packet and, if the network number is in table, it will forward the packet. If the network number is not in its table, it, too, may have a default router—and it will forward the packet to that router.

The problem with default routes in workstations is that a workstation's default router may go down and the workstation will not know if there is another router on the network. The network number may change or there may be a better path for the workstation to take. The default gateway does allow for the elimination of routing tables in the network station and routers, and allows the routing tables to become small by allowing groups of networks to become available through the default route.

Disadvantages of the RIP protocol. As noted before, the acceptance of RIP in the Internet community was based on its implementation into the popular Berkeley 4BSD UNIX operating system through a process known as routed. Unfortunately, it was implemented before the rapid growth of the TCP/IP. It has many disadvantages that were not considered limiting at the time it became accepted. Before RIP was implemented, most router tables had to be constructed manually (a very tedious and dangerous job). RIP allowed these table to be updated dynamically, which was a real advantage at that time. The disadvantages follow.

Routing table updates received are only as accurate as the router that submitted them. If any router made a computational error in updating its routing table, this error will be received by all other routers.

What may also be apparent is the fact that the routing tables could get very large. If the network consisted of 300 different networks (not uncommon in larger corporations), each routing table of every router would have 300 entries. Since RIP works with UDP (connectionless transport-layer service), the maximum datagram size of a RIP packet is 512 bytes. This allows for a maximum of 24 <network number, distance> tuples in each packet (refer to RIP packet description Fig. 6.15*a*). Therefore, it would take 13 packets from each router to broadcast its routing table to all other routers on all the local networks in the internet. This would be broadcast every 30 seconds by each of the 300 routers. This is an unnecessary consumption of bandwidth, especially over slow-speed serial lines.

This leads to the second disadvantage. RIP broadcasts (data-link physical address of all FFs) to the network, normally every 30 seconds, even across slower-speed serial links. This will make the data link pass

the packet up to the upper-layer protocols on all stations on the network, even if the stations do not support RIP.

The third disadvantage is RIP routes datagrams based on the shortest distance to the destination network. The protocol does not take into effect the throughput associated with any path taken. In the RIP protocol, if the path to a destination network contains three hops and another path contains two hops to the same destination, the router will always select the path with two hops, even if the path with two hops is 56 Kbps serial lines running and the one with three hops is running at Ethernet speeds of 10 Mbps. To overcome this, some RIP implementations provide a mechanism to manually assign an artificial hop count to a path.

But this creates another problem. The limit for the number of hops that a network may be distanced from any network station is 15, for a hop count of 16 is considered unreachable. If you add additional hops to a path, you have decreased the total number of routers allowed in a path.

RIP does not handle growth very well. This problem is twofold. The first limitation is that a destination network may be no more than 15 hops away in diameter (a distance of 16 in any routing table indicates the network is unreachable). This does not allow for large networks to be based on RIP. Careful planning is needed to implement large-scale networks based on the RIP protocol.

The other scale problem is the propagation of routing information. Four terms need to be understood here, for they are used quite frequently: *split horizon, hold-down timer, poison reverse,* and *triggered updates.*

Refer to Fig. 6.16*a*. With router A directly attached to network 1, it will advertise that route through *all* its ports as a distance of 1 (whatever the RIP-assigned cost of that port that attaches to that network is). Router B receives this, updates its table as network 1 with a distance of 2. Router B then broadcasts its table (at the 30-second update timer) and router C will receive this and update its table as network 1 with a distance of 3. Notice that all routers will broadcast all the information in their tables through all ports (even the ports from which they received the update).

Why would router B broadcast a reachability of network 1 when router A already has a direct attachment to it? Wouldn't this confuse router A if network 1 is located? Normally it would, but remember that the only changes that a router will make to its tables is when the hop count distance is lower, is a new entry, or if the next hop router path taken to a network changes its hop count. Since that hop count is higher, router A will simply ignore that particular entry in the update table.

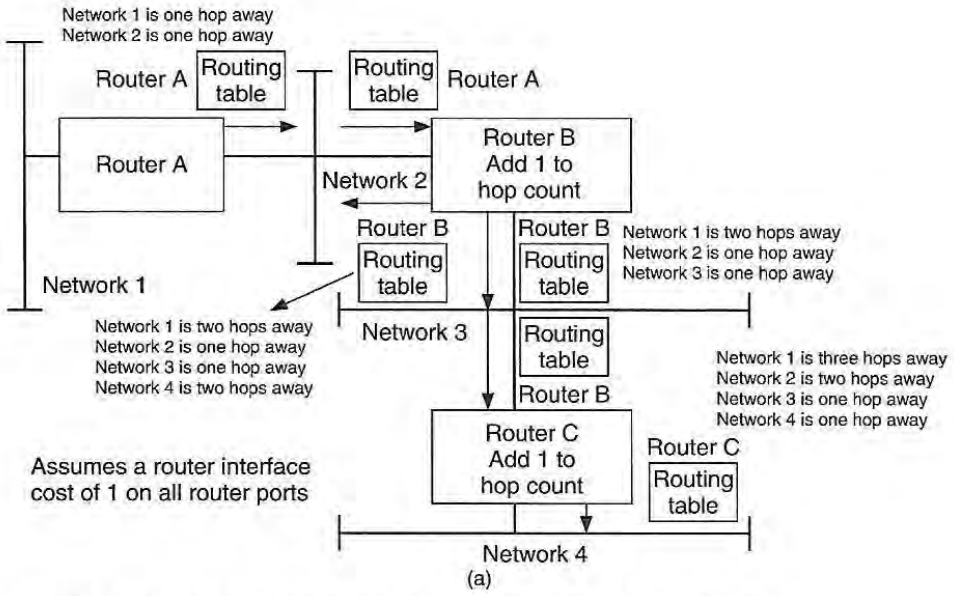


Figure 6.16 Routing table updates. (*a*) Not implementing split horizon.

Using the foregoing original algorithm, a serious problem occurs when router A loses it reachability to network 1. It will update its table entry for that network with a distance of 16 (16 indicates not reachable) but will wait to broadcast this information with the next scheduled RIP update. So far so good, but if router B broadcasts its routing table before router A (notice that not all routers will broadcast their tables at the same time), router A will then see that router B has a shorter path to network 1 than it does (a distance of 2 for router B versus a distance of 16 for router A). Router A will change its entry for network 1. Now, router A, on its next RIP update broadcast, will announce that it has a path to network 1 with a distance of 3 (2 from the table entry received from router B plus 1 to reach router B). There is now a loop between routers A and B. A packet destined for network 1 will be passed between routers A and B until the TTL counter is 0. When router B receives a packet destined for network 1, it will forward the packet to router A; router A will forward it back to router B; and this will continue until the TTL field reaches 0. This is known as *loop*. The RIP protocol works extremely well in a stable environment (an environment where routers and their networks rarely change). The process of clearing out dead routes and providing alternate paths is known as *convergence*.

Even future RIP updates will not quickly fix the convergence in this case. Each update (every 30-second default) will add 1 to the table entry, and it will take a few updates to outdate the entry in these

routers. This is known as *slow convergence,* and it causes errors in routing tables and routing loops to occur.

To overcome this and other problems, a few rules were added to the IP RIP algorithm:

1. *Split horizon.* Implemented by every protocol that uses a variation of RIP (AppleTalk, IPX, XNS, and IP), this states that a router will not broadcast a learned route back through a port from which it was received. Therefore, router B would not broadcast the entry of network 1 back to router A. This would keep router B from broadcasting back to router A the reachability of network 1, thereby eliminating the possibility of a lower hop count being introduced when network 1 became disabled. Figure 6.16*b* shows how the routers would send their tables if split horizon were used. Notice which routes are not included in their updates. Compare Fig. 6.16*b* with Fig. 6.16*a*.
2. *Hold-down timer.* This rule states that once a router receives information about a network that claims a known network is not reachable, it must ignore all future updates that include an entry (a path) to that network, typically for 60 seconds. Not all vendors support this in their routers. If one vendor does support it and another does not, routing loops may occur.
3. *Poison reverse* and *triggered updates.* These are the last two rules which help to eliminate the slow convergence problem. They state that once the router detects a network connection is disabled, the

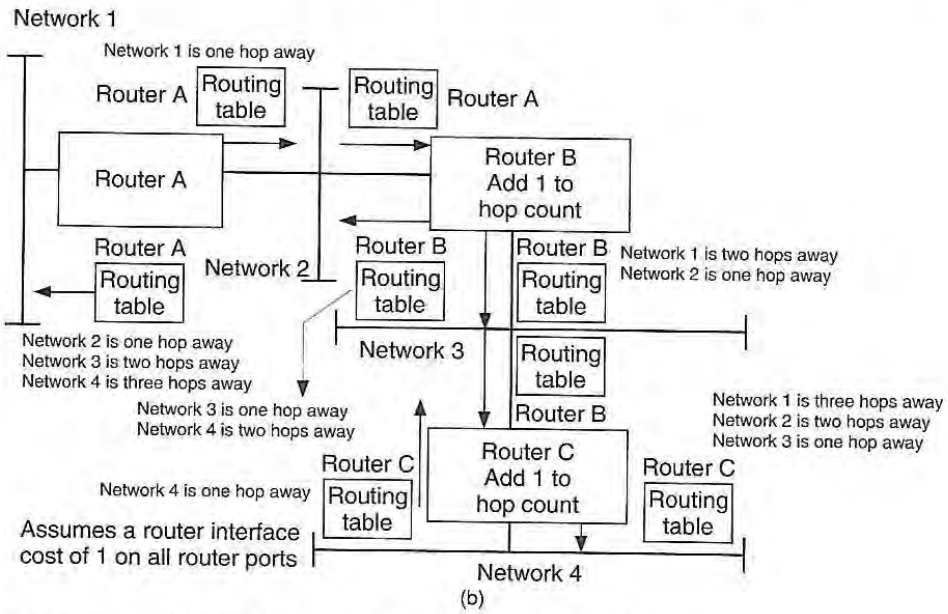


Figure 6.16 Routing table updates. (*b*) Implementing split horizon.

router should keep the present entry in its routing table and then broadcast network unreachable (metric of 16) in its updates. These rules become efficient when all routers in the internet participate using triggered updates. Triggered updates allow a router to broadcast its routing table immediately following receipt of this "network down" information. The two most common are split horizon or poison reverse.

Static routes

Static versus dynamic routing. The last topic of discussion is the capability of routing protocols to accept information for their tables from two sources: the network or a user.

Although the RIP protocol allows for automatic updates for routing tables, manual entries are still allowed and are known as *static entries.* These entries must be entered manually. They can be configured to be included or not included in a dynamic update. Static routes refer to the process of manually updating the tables. For any given port on the router, the network administrator may update that port table with a static route.

Static tables have many disadvantages. First, as discussed before, static tables are not meant for large networks that incur many changes such as growth. As the topology changes, all the tables must be manually reconfigured. Second, in the case of router failure, the tables have no way of updating themselves. Dynamic tables overcome the disadvantages of static entries.

The primary advantage that a static entry may have is for security, for static tables can be configured *not* to broadcast their routes to other routers. In this way, users can customize their routers to become participants on the network without their network identity being broadcast to other routers on the network. Static rates also allow a user to update a routing table with a network entry that will be used in end stations with the dynamic function turned off. This allows the user to maintain the routing table.

Packet routing. Now that routing fundamentals, the RIP protocol, and routing tables have been discussed, the following will show how a datagram is routed via direct routes and indirect routes.

Refer to Fig. 6.17. In this figure we can see that a PC (end station A) is trying to pass a datagram to a host machine, called host B. The host machine is one hop (one router) away. The IP layer of the PC (end station A) knows that it must use a router (the source and destination network addresses are different), and will use RIP or the default gateway to determine the IP address of the router to use. Upon determining the router's physical address, it will physically (MAC layer) address the

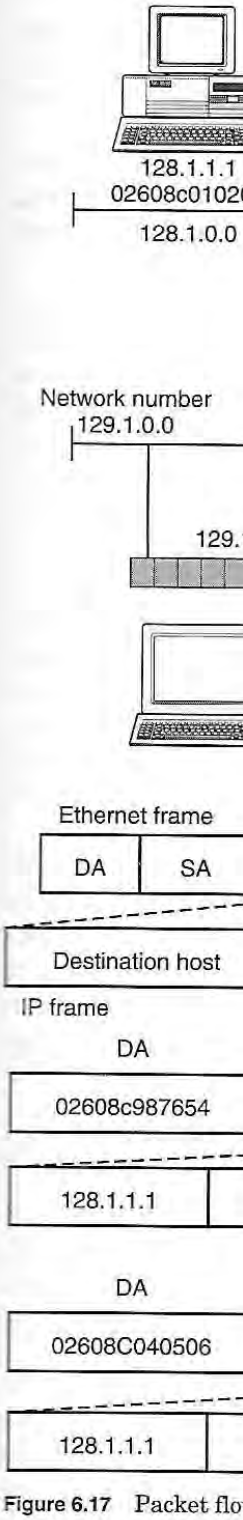


Figure 6.17 Packet flo

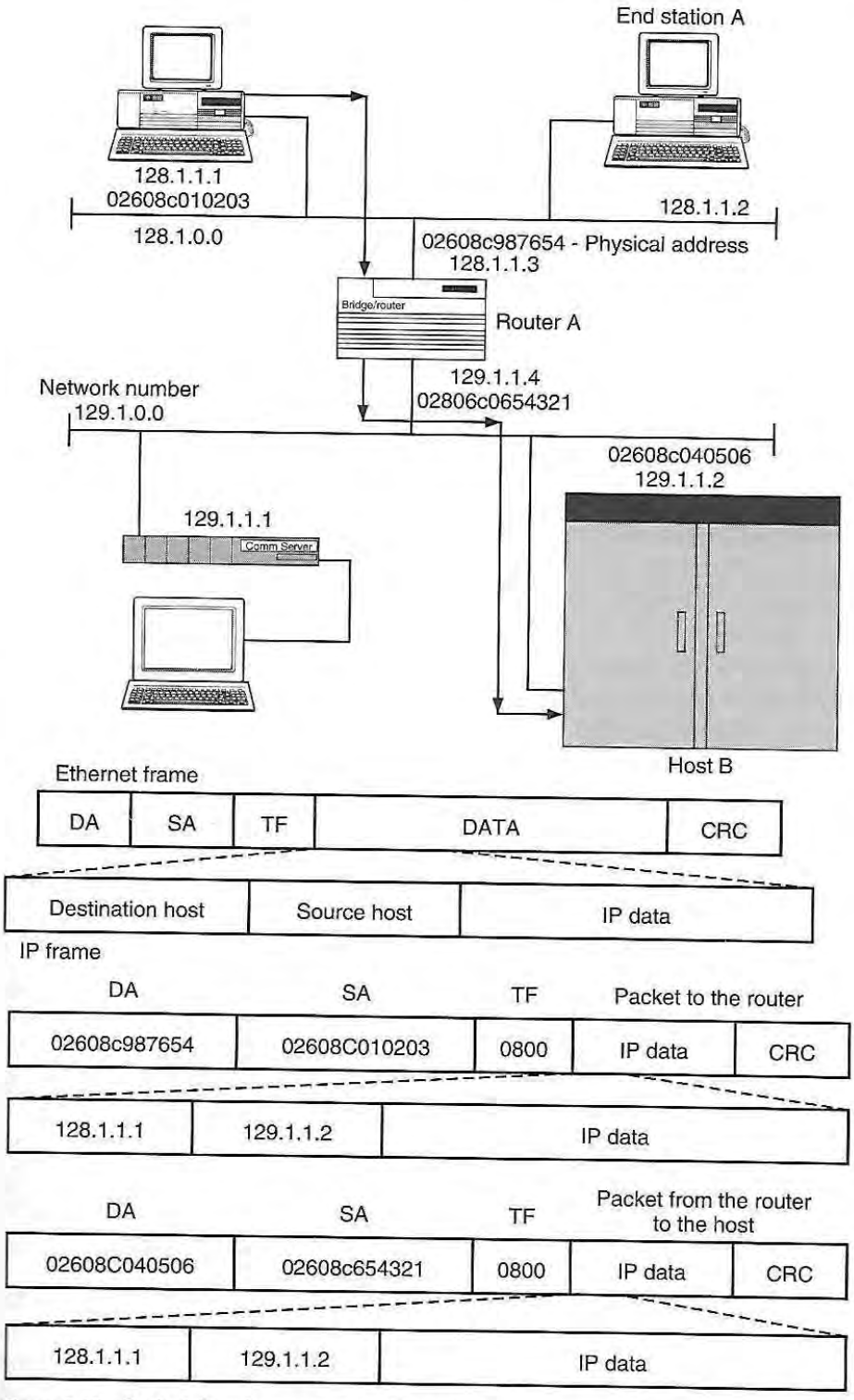


Figure 6.17 Packet flow in a routed environment.

packet to router A. The source and destination IP addresses in the IP header of this datagram will be the PC as the source, and the destination IP address as the host. The source (PC) and final destination (the host) IP addresses will be embedded into the IP header and will not change throughout the routing of this datagram.

Router A will receive this packet and extract the network number from the final destination IP address in the received IP header. The physical address headers will be stripped. The extracted network number will be compared to the router's internal routing table.

Router A will determine that the destination network can be reached directly through one of its ports (the destination network is directly attached). The router will determine the destination station's physical address through its ARP table (or it may request it through the ARP process). Router A will then build a packet with the original datagram sent by end station A to submit to host B. The physical source address will be the router's, the physical destination address will be host B's. The packet is then transmitted to host B.

Refer to Fig. 6.18. This shows a router-to-router packet delivery. End station A is still trying to reach host B, only this time it is two routers away. Router A will determine that the destination can be reached through router B. It will physically address the packet to send to router B, with its physical address as the source address and the physical destination address as that of router B. This is for physical addressing only. It will then submit the packet to the network that has router B attached.

Router B will receive this packet, extract the destination IP network address from the packet, and compare it to its routing table. From its router table, it will determine that the final network is directly connected to port 2 of its router. Router B will perform an ARP lookup in its ARP table to find the physical address of the host. If an address is there, it will physically address the source address of the packet with its own physical address, and the destination address will be that of the final destination station—in this case, the host. This is the only time that the packet will actually carry the physical address for the destination host.

To return the packet, host A will start over again, will notice that the destination network is remote, and will submit the packet to router B.

The numbers 1, 2, and 3 indicate how the packet would look on each network segment.

Remote networking through serial lines. There are times when networks must be connected when they are geographically separated. This means that networks cannot be connected by the conventional means of a LAN interconnect. This could be when a network in New York and

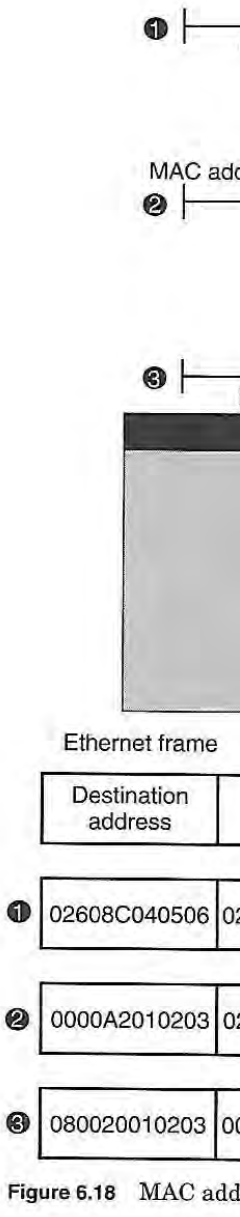
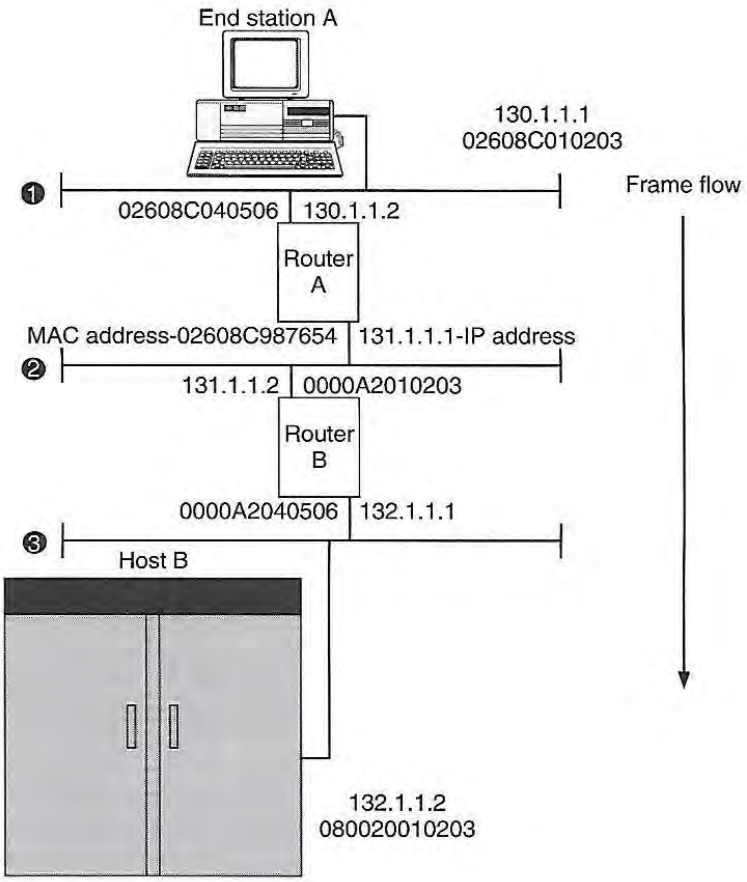


Figure 6.18 MAC add



Ethernet frame

	Destination address	Source address	Type field	IP header		IP data
				Source	Destination	
❶	02608C040506	02608C010203	0800	130.1.1.1	132.1.1.2	IP data
				Source	Destination	
❷	0000A2010203	02608C987654	0800	130.1.1.1	132.1.1.2	IP data
				Source	Destination	
❸	080020010203	0000A2040506	0800	130.1.1.1	132.1.1.2	IP data

Figure 6.18 MAC address assignments.

a network in Virginia need to be connected together. The only feasible way of doing this is by using leased lines from the telephone company. AT&T, MCI, Sprint all provide leased line service for data networks. It comes in many forms, but again, for simplicity, this book will explain point-to-point serial lines. Refer to Fig. 6.19. This figure shows networks connected through serial lines.

Likewise, the router has a connection that enables this type of connection. Instead of the LAN interface on the router, the router will have a serial line interface. The connector for this is usually a V.35, EIA-232 (formerly RS-232-D) or an RS-449 connector.* The connection will then be connected to a device known as a Data Service Unit/Customer Service Unit (DSU/CSU). This is a box that receives the serial signal from the router and repeats it to the telephone switching office.

The leased line is a specially conditioned line that will be provided by the phone company. This line has been conditioned to handle high-speed digital traffic. This line is not the normal line that is used with voice switching. This line is permanently switched to provide a connec-

* The type of connector used will depend on the line speed. For the most part, only two types of connections are used: EIA-232 for lines speeds at 19.2 Kbps (thousand bits per second or less) and V.35 for line speeds above 19.2 Kbps (56K, T1, etc.).

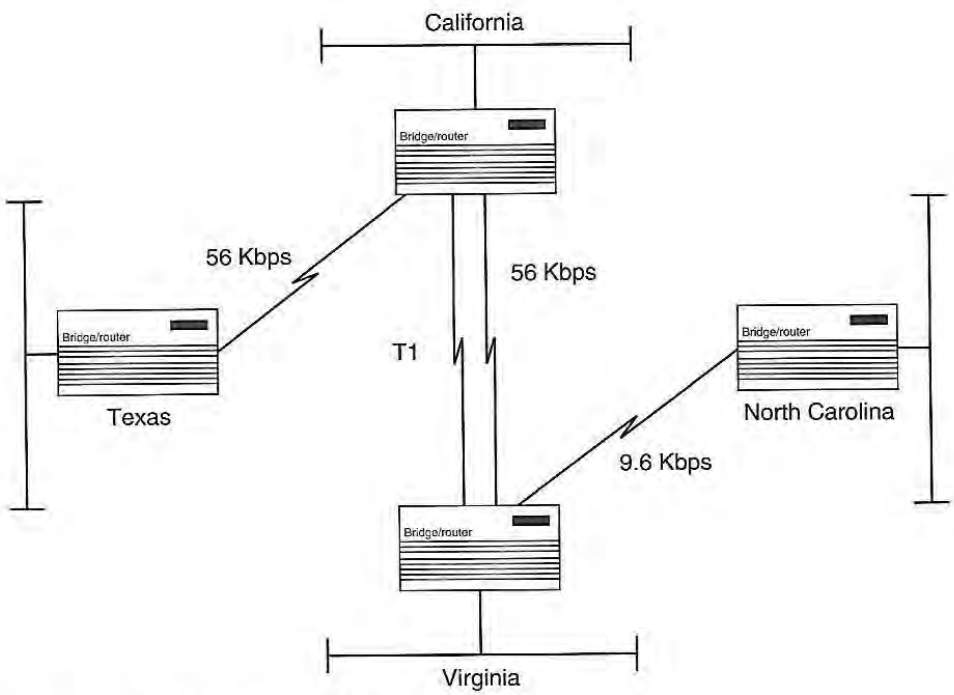


Figure 6.19 Wide area networking with serial lines.

tion between two
point link. It is a
and never hangi

The router at t
will be able to rec
ical speeds at wh
Kbps and T1.* Th
does not own the
vary depending o
short runs (the ot
for longer runs. F

The serial line
that cannot be co
them in an IP int
a subnet number
length subnet m
known as Open
book); otherwise,
are only two poin

Internet control me

Refer to Fig. 6.17
end station (the e
did not respond to
for the ICMP serv
the originator of
node cannot be fo
an error message

Before the dyna
1989, static route
has a packet for
packet. If a defau
the packet, but fo
If the default rou
card the packet
through a control

Since IP is a c
routers and hosts
certain instances
errors could be: a
router is too cong

* Kpbs is thousand
(Mbps).

tion between two points. Therefore, it is sometimes called a point-to-point link. It is analogous to dialing a number, receiving a connection, and never hanging up.

The router at the remote end will also be attached to a DSU/CSU. It will be able to receive the signals generated at the remote end. The typical speeds at which these lines run vary. The most common are: 56 Kbps and T1.* These lines are called leased lines because the customer does not own the line. It is leased from the phone company and the rates vary depending on the length of the line. Rates are usually cheaper for short runs (the other point of the network is a few miles away) and more for longer runs. Rates also vary depending on the speed of the line.

The serial line provides a simple interconnect between two routers that cannot be connected directly by a LAN. The real problem in using them in an IP internet is that they consume a full network number or a subnet number. There have been methods to overcome this (variable length subnet masking available only with the routing algorithm known as Open Shortest Path First—OSPF, not discussed in this book); otherwise, they generally act as a full network even when there are only two points connected.

Internet control message protocol (ICMP)

Refer to Fig. 6.17. What would happen if router A could not find the end station (the end station was not in the router's ARP cache and it did not respond to the router's ARP request)? This is one of the reasons for the ICMP service. The router would send an ICMP message back to the originator of the datagram, end station A, that the destination node cannot be found. This message will be transmitted to the user as an error message on the user's screen.

Before the dynamic routing algorithm RIP was formally accepted in 1989, static routes were fairly common in routers. Therefore, if a router has a packet for which no route can be found, it should discard the packet. If a default route was assigned, the router would not discard the packet, but forward it onto the router assigned in the default route. If the default router could find the route, the default router would discard the packet and notify the originating station of this action through a control protocol known as ICMP.

Since IP is a connectionless, unreliable delivery service, allowing routers and hosts on an internet to operate independently, there are certain instances when errors will occur on the internet. Some of these errors could be: a packet is not routed to the destination network, the router is too congested to handle any more packets, or a host may not

* Kpbs is thousand bits per second and T1 runs at 1.544 million bits per second (Mbps).

be found on the internet. There is no provision in IP to generate error messages or control messages. ICMP is the protocol that handles these instances for IP.

Figure 6.20 shows the packet format for ICMP.

ICMP controls many entities on an internet. As Table 6.7 shows, there are 13 control messages that ICMP uses.

ICMP datagrams are routable since they use IP to deliver their messages. Since they use the IP protocol and not TCP protocol (the transport layer), these messages themselves have the capability of being lost or generating an error. IP is a connectionless protocol and does not have mechanisms to detect whether datagrams were delivered to their proper destination. Therefore, there is not a mechanism in ICMP to detect lost ICMP messages. As a result, no ICMP messages will ever be generated for an erred ICMP message. Furthermore, ICMP messages are not generated and received by a user on the network. The message is intended for the IP software in another network station. A message may be generated to your screen, but this is up to the programmer. ICMP software only talks to other ICMP software.

Figure 6.20 shows how an ICMP message is encapsulated in an IP packet.

One of the most common uses for ICMP is the PING program. PING (packet internet groper) is an ICMP message that tries to locate other stations on the internet to see if they are active or to see if a path is up. It can also be used to test intermediate networks along the way to the destination.

To see if host number 129.1.2.3 is alive, all the user has to do is type: "ping 128.1.2.3" at his or her workstation (provided the TCP/IP proto-

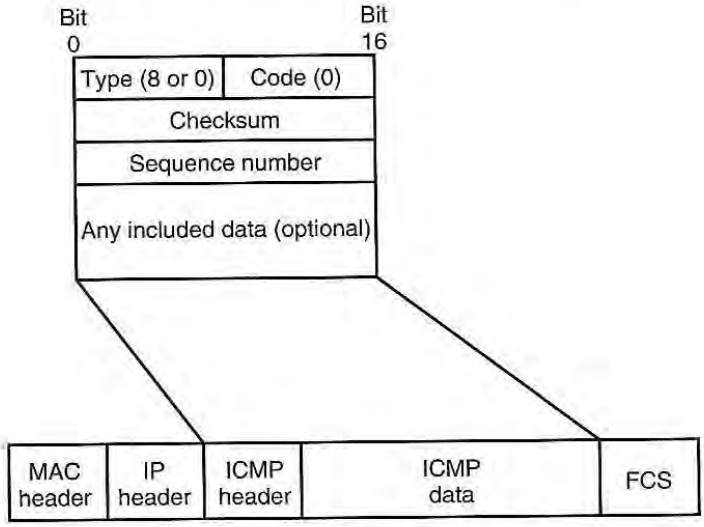


Figure 6.20 ICMP packet format echo request or reply.

TABLE 6.7 ICMP Me

Type field	
0	Ec
3	De
4	So
5	Re
8	Ec
11	Ti
12	Pa
13	Ti
14	Ti
15	In
16	In
17	Ad
18	Ad

col is active on th
will respond with
usually see on the
check for hosts or
will not connect
you can at least c
destination is up.
network delays al
the response dela

A lot of network
mine the status of
build maps to sho
the map. Using co
red for not respon
network. A lot of t

Another use is t
running in a route
address mask for
router the subnet

Source quench is
of a message that
is submitting the
erated to the orig
intended recipient
data rate until it r
that was requeste
rate again. This is
throttle control. Th
increased again. It
to indicate that th

TABLE 6.7 ICMP Message Types

Type field	Message description
0	Echo reply—PING
3	Destination unreachable
4	Source quench—a form of flow control
5	Redirect—there is a better route to take
8	Echo request—PING
11	Time exceeded for a datagram—TTL is zero
12	Parameter problem on a datagram
13	Timestamp request
14	Timestamp reply
15	Information request
16	Information reply
17	Address mask request
18	Address mask reply

col is active on the user's workstation). If the host is active, the host will respond with an ICMP echo reply and the message the user will usually see on the screen is "129.1.2.3 is alive." This is a simple way to check for hosts on the network. There are some times where TELNET will not connect with the destination host; using the ping command, you can at least check to see if the host is active or if the path to the destination is up. Another usage of the PING command is to check for network delays along a path. The response to a ping request can report the response delay. This delay is usually measured in milliseconds.

A lot of network management software uses this command to determine the status of a given station. Network management software will build maps to show the topology and placement of network stations on the map. Using colors (green for active, yellow for possible errors, and red for not responding) a network manager can trace problems on the network. A lot of the work is done through the use of the utility ping.

Another use is to find the address mask of the local network. ICMP running in a router can respond to a host's request to find the subnet address mask for its network. A host, upon start-up, can request of a router the subnet mask assigned to the network.

Source quench is the end station's ability to indicate to the *originator* of a message that the host cannot accept the rate at which the sender is submitting the packets. A source quench packet is continually generated to the originator until the rate of data flow slows down. The intended recipient of a source quench is will continue to slow down its data rate until it receives no more source quench packets. The station that was requested to slow down will then start to increase the data rate again. This is similar to a flow control, except that it is more like throttle control. The data is not stopped, merely slowed down and then increased again. It is generated by any network station on the internet to indicate that the node cannot handle the rate of the incoming data.

There are many other uses of the ICMP protocol. When a router receives a datagram, it may determine a better router that can provide a shorter route to the destination network. This is an ICMP redirect, and this message informs the sender of a better route.

If the TTL field is determined to 0, a router will inform the originator of this through an ICMP message.

A user's workstation can request a time stamp from a router asking it to repeat the time when it received a packet. This is used for measuring delay to a destination.

Section 2: Transport Layer Protocols

User datagram protocol (UDP)

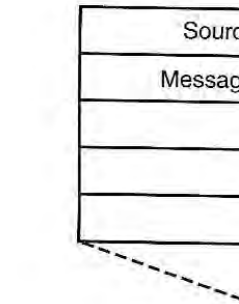
A transport layer allows communication to exist between network stations. Data is handed down to this layer from an upper-level application. The transport layer then envelopes the data with its headers and gives it to the IP layer for transmission onto the network. In TCP/IP there are two transport-layer protocols: UDP and TCP.

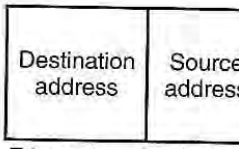
The functionality of UDP should sound familiar. It is a connectionless, unreliable transport service. It does not provide an acknowledgment to the sender upon the receipt of data. It does not provide order to the incoming packets, and may lose packets or duplicate them without issuing an error message to the sender. This should sound like the IP protocol. The only offering that UDP has is the assignment and management of port numbers to uniquely identify the individual applications that run on a network station. UDP tends to run faster than TCP, for its has low overhead involved in the function that it performs. It is used for applications that do not need a reliable transport. Some examples are network management, name server, etc.

Any application program that incorporates the use of UDP as its transport-level service must provide an acknowledgment and sequence system to ensure that packets arrive, and that they arrive in the same order as they were sent. That is, the application program using UDP must provide this type of service.

As shown in Fig. 6.21, an applications data is encapsulated in a UDP header. The transport layer has its own header, independent of all other layers, that it prefaces to the data handed to it from its upper-layer protocol. The UDP header and its data are then encapsulated in an IP header. The IP protocol would then send the datagram to the data-link layer which would then encapsulate the datagram with its headers (and/or trailers) and send the data to the physical layer for actual transmission.

Upon receipt of the packet, the data link would interpret the address as its own, strip off its header (and/or trailers), and submit the packet





Ethernet packet forma

Figure 6.21 UDP hea

to the IP layer. I
address in the IP
the UDP-layer so
has to demultiple
header.

Port numbers. Si
the same machine
these applications
they reside on the
a sending node ad
here in how you
machine that cont

It would not be a
would it be advant
a marker to identi
User Datagram Pr

Ports, along wi
machine on an int

Refer to Table 6.
When a station wis

* Ports are sometime
ports and sockets will b
net address plus the as

There are many other uses of the ICMP protocol. When a router receives a datagram, it may determine a better router that can provide a shorter route to the destination network. This is an ICMP redirect, and this message informs the sender of a better route.

If the TTL field is determined to 0, a router will inform the originator of this through an ICMP message.

A user's workstation can request a time stamp from a router asking it to repeat the time when it received a packet. This is used for measuring delay to a destination.

Section 2: Transport Layer Protocols

User datagram protocol (UDP)

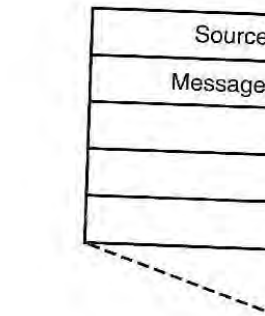
A transport layer allows communication to exist between network stations. Data is handed down to this layer from an upper-level application. The transport layer then envelopes the data with its headers and gives it to the IP layer for transmission onto the network. In TCP/IP there are two transport-layer protocols: UDP and TCP.

The functionality of UDP should sound familiar. It is a connectionless, unreliable transport service. It does not provide an acknowledgment to the sender upon the receipt of data. It does not provide order to the incoming packets, and may lose packets or duplicate them without issuing an error message to the sender. This should sound like the IP protocol. The only offering that UDP has is the assignment and management of port numbers to uniquely identify the individual applications that run on a network station. UDP tends to run faster than TCP, for its has low overhead involved in the function that it performs. It is used for applications that do not need a reliable transport. Some examples are network management, name server, etc.

Any application program that incorporates the use of UDP as its transport-level service must provide an acknowledgment and sequence system to ensure that packets arrive, and that they arrive in the same order as they were sent. That is, the application program using UDP must provide this type of service.

As shown in Fig. 6.21, an applications data is encapsulated in a UDP header. The transport layer has its own header, independent of all other layers, that it prefaces to the data handed to it from its upper-layer protocol. The UDP header and its data are then encapsulated in an IP header. The IP protocol would then send the datagram to the data-link layer which would then encapsulate the datagram with its headers (and/or trailers) and send the data to the physical layer for actual transmission.

Upon receipt of the packet, the data link would interpret the address as its own, strip off its header (and/or trailers), and submit the packet



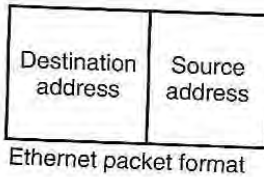


Figure 6.21 UDP header

to the IP layer. IP
address in the IP he
the UDP-layer softv
has to demultiplex
header.

Port numbers. Since
the same machine, a
these applications, e
they reside on the sa
a sending node addre
here in how you di
machine that contain

It would not be adva
would it be advantage
a marker to identify
User Datagram Proto

Ports, along with
machine on an intern

Refer to Table 6.8. T
When a station wishe

* Ports are sometimes in
ports and sockets will be in
net address plus the associa

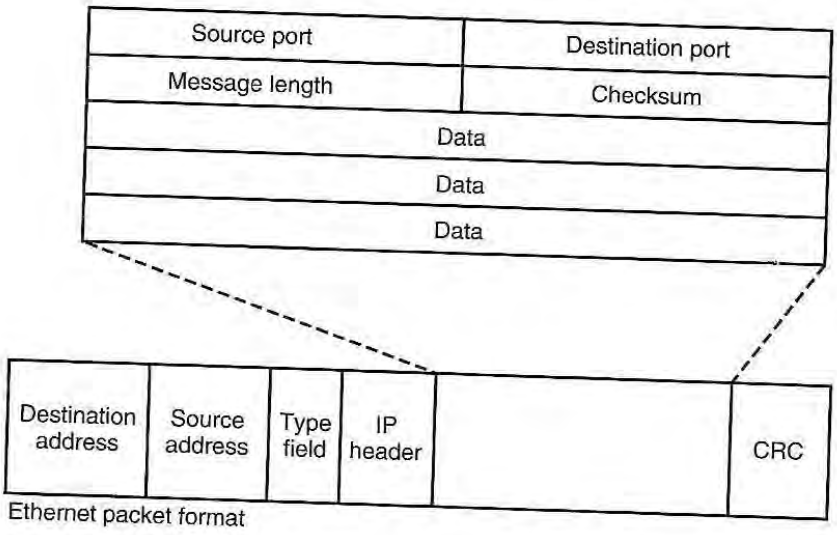


Figure 6.21 UDP header.

to the IP layer. IP would accept the packet based on the correct IP address in the IP header, strip off its header, and submit the packet to the UDP-layer software. The UDP layer accepts the packet and now has to demultiplex the packet based on the port number in the UDP header.

Port numbers. Since many network applications may be running on the same machine, a protocol needed to be developed to allow access to these applications, even though they reside on the same machine. If they reside on the same machine, each application can be accessed by a sending node addressing its packet to that machine. A problem arises here in how you differentiate between all the applications on one machine that contains one IP address.

It would not be advantageous to assign each process an IP address, nor would it be advantageous to change the IP addressing scheme to include a marker to identify a unique application in the machine. Instead, the User Datagram Protocol provides a concept known as *ports.**

Ports, along with an IP address, allow any application in any machine on an internet to be uniquely identified.

Refer to Table 6.8. This table shows the reserved port numbers' UDP. When a station wishes to communicate to a remote application, it must

* Ports are sometimes incorrectly referred to as sockets. For the purposes of this text, ports and sockets will be interchangeable. In actuality, sockets are a station's full internet address plus the associated port number.

TABLE 6.8 UDP Port Assignments*

0	Reserved
7	Echo
9	Discard
11	Active users
13	Daytime
15	Who is up or NETSTAT
17	Quote of the day
19	Character generator
37	Time
42	Name server
43	Who is
53	Domain name server
67	Bootstrap protocol server
68	Bootstrap protocol client
69	Trivial File Transfer Program (TFTP)
111	Sun RPC
123	Network time protocol
161	SNMP net monitor
162	SNMP traps
512	UNIX comsat
513	UNIX rwho process
514	System log
525	Timed

* If you are looking for a port number here and cannot find it, the application may be using TCP as its transport and, therefore, you should look up the port number in the TCP section.

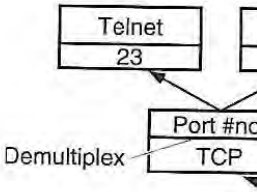
identify that application in the packet. For example, if a station wished to use a simple file transfer protocol known as *trivial file transfer program* (TFTP) on the station 130.1.1.1, it would address the packet to station 130.1.1.1 and insert destination port number 69 in the UDP header. The source port number identifies the application on the local station that requested the file transfer, and all response packets generated by the destination station would be addressed to that port number on the source station. So, when the IP layer demultiplexes the packet and hands it to UDP, UDP will pass the data to the locally assigned port number for it to process the data. Port numbers are explained in detail later.

In looking at Fig. 6.21, the packet header for UDP is small, but functional. The source port indicates the process ID from the sender. This is used for replies from the remote station. If it is not used (broadcast RIP update tables), it should be set to 0. The destination port number indicates the process on the remote station. The message length indicates the size of the UDP header and its data in bytes. The minimum packet size for UDP is 8 bytes (the size of the header). The checksum is used to check for the validity of the UDP header and data. It does not have to be implemented and would be set to 0 if not implemented.

In thinking of t
talk directly to U
the services of IP

UDP accepts d
header) with its i
work delivery. U
depending on the
As shown in Fig.
packet (after strip
to the port numbe
demultiplexing. T
indicate any appl
UDP reads the de
and gives the data
(identified by the
tem works in conj
information may
retrieval by the a
mechanism for por
unreachable messa

Since the TCP/I
cally written to it
cally assigned por



Demultiplex

Et

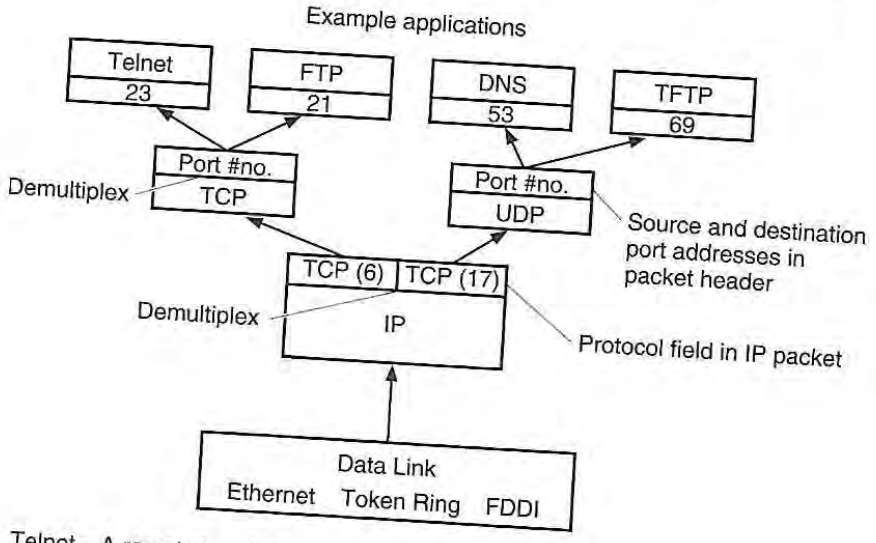
Telnet—A remote term
FTP—File Transfer P
DNS—Domain Name
TFTP—Trivial File Tra

Figure 6.22 IP port assi

In thinking of the OSI model (refer to Fig. 6.2), an application would talk directly to UDP for it to transport the data. UDP would then use the services of IP to actually send the data to the network.

UDP accepts data from the application layer, formats it (UDP header) with its information, and presents it to the IP layer for network delivery. UDP will also accept data from the IP layer and, depending on the port value, present it to the appropriate application. As shown in Fig. 6.22, UDP is responsible for directing the rest of the packet (after stripping off its headers) to the correct process according to the port number assigned in the UDP header. This process is called *demultiplexing*. There are many different types of port numbers to indicate any application program running on the network station. UDP reads the destination port field of the UDP header (demultiplex) and gives the data to the application. When the application program (identified by the port number) initializes, the station's operating system works in conjunction with it and provides a buffer area for which information may be stored. UDP will place the data in this area for retrieval by the application program. UDP does provide one error mechanism for ports which are not valid. It can generate an ICMP port unreachable message to be sent to the originator of the packet.

Since the TCP/IP protocol suite include applications that are specifically written to it (TFTP, Domain Name Service, etc.), there are statically assigned port numbers that identify these applications. Certain



Telnet—A remote terminal program
FTP—File Transfer Program
DNS—Domain Name Server
TFTP—Trivial File Transfer Program

Figure 6.22 IP port assignments.

port numbers are reserved and cannot be used by any unknown application program. They are reserved by the applications that are defined in the RFCs. The reserved port numbers are specified in RFC 1060.

Dynamic and static assigned port numbers. In the TCP/IP protocol, UDP port numbers are both static and dynamic. Table 6.8 lists some of the port numbers that are static. No matter which implementation of TCP/IP (i.e., which vendor's TCP) is in use, those applications listed beside the port number will always be the same. There are known as *globally assigned numbers* or *well-known port numbers.* These are assigned by a central authority. In this case, the RFC 1060 spells out which processes are assigned to which port numbers. If station A wanted to access the TFTP process on station B, it would call it in the UDP header with a destination port number of 69 (decimal). The difference between the two types is: static port numbers are reserved and dynamic port numbers may be used by any network station.

TCP/IP also implements dynamic port numbers. These are available for dynamic use. Since the port number field in the UDP header is 16 bits long, 65535 ports (minus the static port assignments) are available for individual use. One use for a dynamic port is for a source station that is requesting the services of TFTP on a remote station. The source station would dynamically assign itself an available port number (usually above size) to use so that the remote station would know what port to access when it transfers the file. In other words, if a user initiated a trivial file transfer (TFTP), the TFTP request packet sent to the TFTP server would include in its UDP header a dynamic port number of the requesting network station that wanted the TFTP, called the *source port.* Let's say it was assigned port 2000. The destination port number would be 69. In this way, the server will accept the packet, give it to the TFTP process in the host and, when the host responds, it will know how to address the port number in the response packet. In the response packet, the server would fill out the UDP header with a destination port of 2000, source port of 69, and send the packet back to the requesting station.

Another use is when network vendors implement proprietary schemes on their devices—for example, a proprietary scheme for a network station to boot or a proprietary scheme to allow network management statistics to be gathered. All these applications are valid and may run on any TCP environment using a dynamic port assignment.

The disadvantage to dynamic ports occurs when a broadcast IP datagram is transmitted to the network using a dynamic port. This port could be used by another vendor on the network, and another network station may invoke a process to accommodate that request. This is rare, but has been known to happen.

Dynamic port numbers are assigned at the local workstation. They can be duplicated from workstation to workstation. This is because an application on any network station is uniquely identified by the network number, host number, and port number. When taken as a whole number, it is called the socket, and cannot be duplicated on an IP network except by negligence.

Transport control protocol (TCP)

TCP/IP hosts originally were connected via telephone lines (commonly known as serial lines). This communication facility was not the same in the early 1970s as it is today. These line were extremely noisy and were not conditioned to handle data. Therefore, the TCP protocol has strict error detection algorithms built in to ensure the integrity of the data. The following paragraphs explain the TCP protocol and show the strictness with which it is structured to ensure the integrity of the data.

TCP is also a transport-layer protocol. Unlike the UDP protocol, the purpose of the transport-layer software TCP is to allow data to be reliably exchanged with another station on the network. It, too, provides the demultiplexes of port numbers to identify an application in the host, but also provides reliable transport of data.

The analogy is as mentioned before: suppose you are standing next to someone and you are telling a story to this person. If that person stands there and does not acknowledge you and your conversation, you would not be able to tell if the person was understanding what you were saying. If the person acknowledges you with something like a nod of the head or a verbal "yes," then you would know that what you were saying was understood and you could continue with your conversation.

The protocol of TCP uses sequence numbers and acknowledgments to converse with another station on the network. Sequence numbers are used to determine the ordering of the data in the packets and to find missing packets. Since packets on an internet may not arrive in the same sequence as they were sent (for example, a single packet in a series of packets being transmitted was discarded by a router), sequencing the data in the packets ensures that the packets are read in the same order that they were sent. Also, a receiving station may receive two of the same packets. The sequence number with acknowledgments is used to allow a reliable type of communication. This process is called full duplex, for each side of a connection will maintain its own sequence number for the other side.

TCP is a byte-oriented sequencing protocol. This means that every byte in each packet is assigned a sequence number. This does not mean that TCP transmits a packet containing only 1 byte. TCP will transmit data (many bytes) and assign the packet one sequence number. Assigning one sequence number per byte in the packet may sound repetitious,

but remember that TCP/IP was first implemented over noisy serial lines and not reliable high-speed LANs. Packet sequencing, such as that used with IPX and XNS, applies a sequence number to each packet transmitted and not to each data byte in the packet. Refer to Fig. 6.23. In this figure three packets are transmitted. Each packet is assigned one sequence number. Notice how the sequence number jumps by the same amount of bytes that are in each packet. The receiver of these packets will count the amount of bytes received and increment its sequence number of received packets. The first packet received has a sequence number of 40 and contains 4 bytes. The receiver expects the next sequence number to be 44. It is, and that packets also contains 7 bytes of data. The receiver expects the next packet to be sequence number of 51. It is. This is how the byte sequencing of TCP works.

Not all networks use a separate transport-layer software to converse on a network. The best example of this is Novell with their LAN workgroup operating system of NetWare. As you read in Chap. 5, they rely on the network-layer software to transport their data and the NetWare Control Protocol to provide the sequence numbering of the packets. There is nothing wrong with this, and it generally speeds up the communication process between two stations on the network. The overhead of a full transport layer is gone, but without the full transport-layer software there is no way to guarantee the data will be transmitted in a reliable fashion.

Shown in Fig. 6.24 is the TCP header for the TCP. It will be referenced throughout the discussion on TCP. Do not try to fully understand the header now. Most of the fields in the header will be discussed in the text following.

Source port. The port number of the originating station.

Destination port. The port number for the receiving station.

Sequence number. A number assigned to a TCP packet to indicate the beginning byte number of a packet unless the SYN bit is set. If this bit is set, the sequence number is the initial sequence number (ISN) and the first data byte is ISN + 1.

Acknowledgment number. Number sent by the destination station to the source station, indicating an acknowledgment of a previously received packet or packets. This number indicates the next sequence number the destination station expects to receive. Once a connection is established, this field is always set.

Data offset. Indicates how long the TCP header is (i.e., the number of 32-bit words in the TCP header). It indicates where the data begins and the TCP header stops.

Sequence num

Sequence num

Sequence num

Figure 6.23 TCP byte s

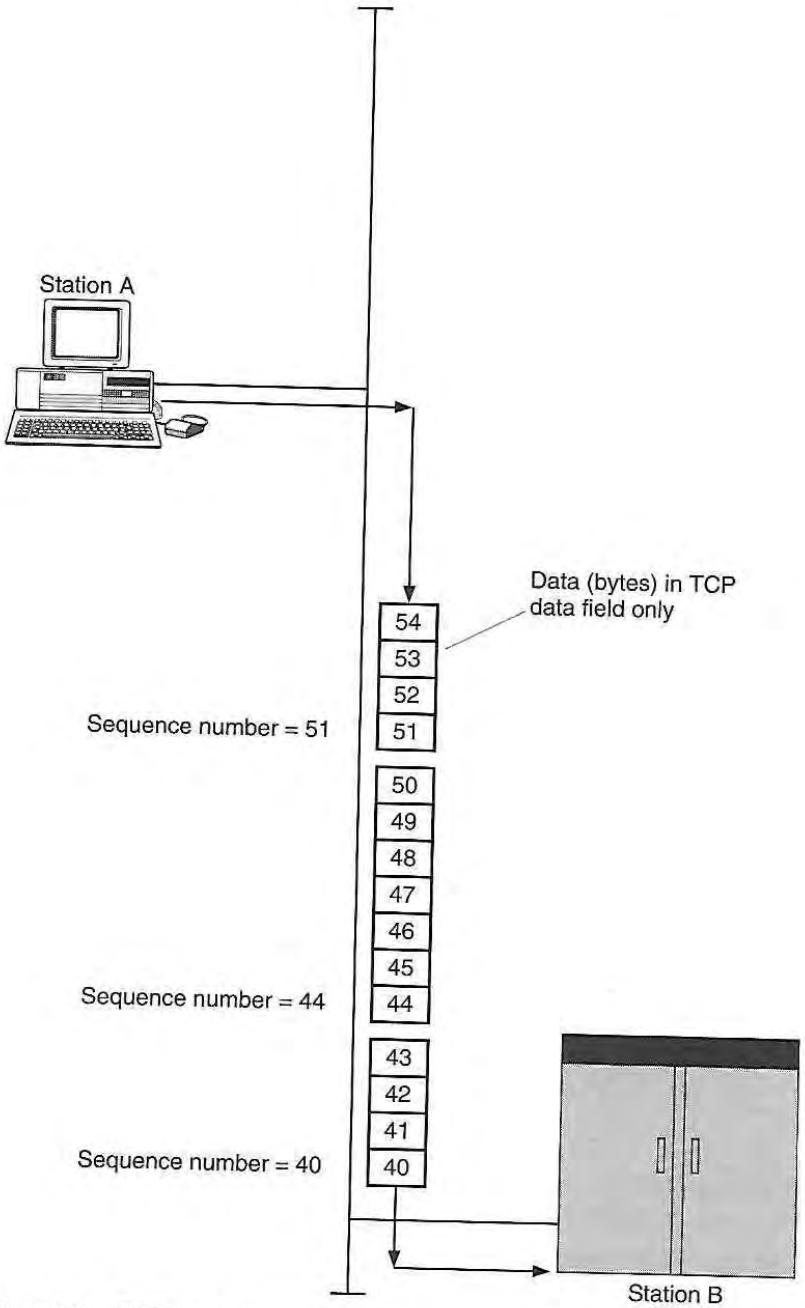


Figure 6.23 TCP byte sequencing.

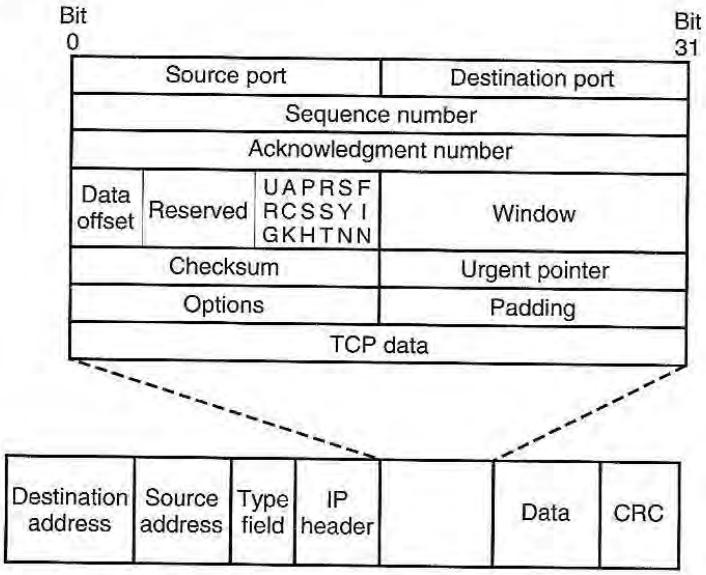


Figure 6.24 TCP header information.

Reserved. Reserved for future use. Must be set to 0.

Control bits

URG Urgent pointer.

ACK If set, this packet contains an acknowledgment.

PSH Push function.

RST Reset the connection. One function for this is to not accept a connection request.

SYN Used to establish sequence number.

FIN No more data is coming from the sender of the connection.

Window. The number of data octets beginning with the one indicated in the acknowledgment field which the sender of this segment is willing to accept.

Checksum. An error detection number.

Urgent pointer. The urgent pointer points to the sequence number of the byte following the urgent data. This field is interpreted only in segments with the URG bit set.

Options. Variable in length, it allows for TCP options to be presented. These are:

End of option list

No operation

Maximum segment size

The main services provided by TCP are:

1. Establishing, maintaining, and terminating connections between two processes
2. Reliable packet delivery—through an acknowledgment process
3. Sequencing of packets—reliable transfer of data
4. Mechanism for controlling errors
5. The ability to allow multiple connections with different processes inside a particular source or destination host through the use of ports
6. Data exchange using full-duplex operations

General TCP data flow. A network application such as TELNET (for remote terminal emulation discussed further under Sec. 3) or a file transfer program such as FTP actually transmits data by placing a call to the TCP software and then passes data to it. TCP encapsulates this data into "segments" and, in turn, places a call to the IP software for it to transmit the data to the destination. The receiving host's TCP software deencapsulates the data from the packet and notifies the appropriate process that it has data to be retrieved.

As we learned previously, the IP layer is a connectionless delivery service and therefore is deemed unreliable. TCP must be a robust and complex protocol to ensure that data is delivered with no errors and also is received in the same order that it was transmitted. TCP allows for this.

TCP functions. Each function will be explained separately and then it will all tied together at the end of this section.

Connection establishment. Unlike the UDP protocol, a TCP connection between two stations on a network must be established before any data is allowed to pass between the two. Applications such as TELNET and FTP communicate using TCP through a series of function calls. These calls include OPEN and CLOSE a connection, SEND and RECEIVE to that connection, and to receive STATUS for a connection.

When a connection to a remote station is needed, an application will request TCP to place an OPEN CALL. There are two types of OPEN CALLS: *passive* and *active*. A passive OPEN is a call to allow connections to be accepted from a remote station. This usually occurs when an application starts on a network station (such as TELNET, FTP, or SMTP), and it will indicate to TCP that it is willing to accept connections from other stations on the network. TCP will note the application through its port assignment and will allow connections to come in. This end of the TCP actions is known as the *responder* TCP. It will open up

connection slots to accept any incoming connection request. This may be thought of as the server end of TCP. These passive open calls do not await for any particular station to attach to it.

An active OPEN is made when a connection attempt to remote network station is needed. Looking at Fig. 6.25, station A wishes to connect to station B. Station A issues an active open call to station B. In order for the connection to be made, station B must already have issued a passive open request to allow incoming connections to be established. In the connection attempt packet is the port number that station A wishes to use on station B. Station B's operating system will spawn a separate process on its system to maintain that connection. *This process will act as if it is running locally on that station. TCP will* then await another incoming connection request. This process is similar to the way a multitasking operating system handles multiple applications.

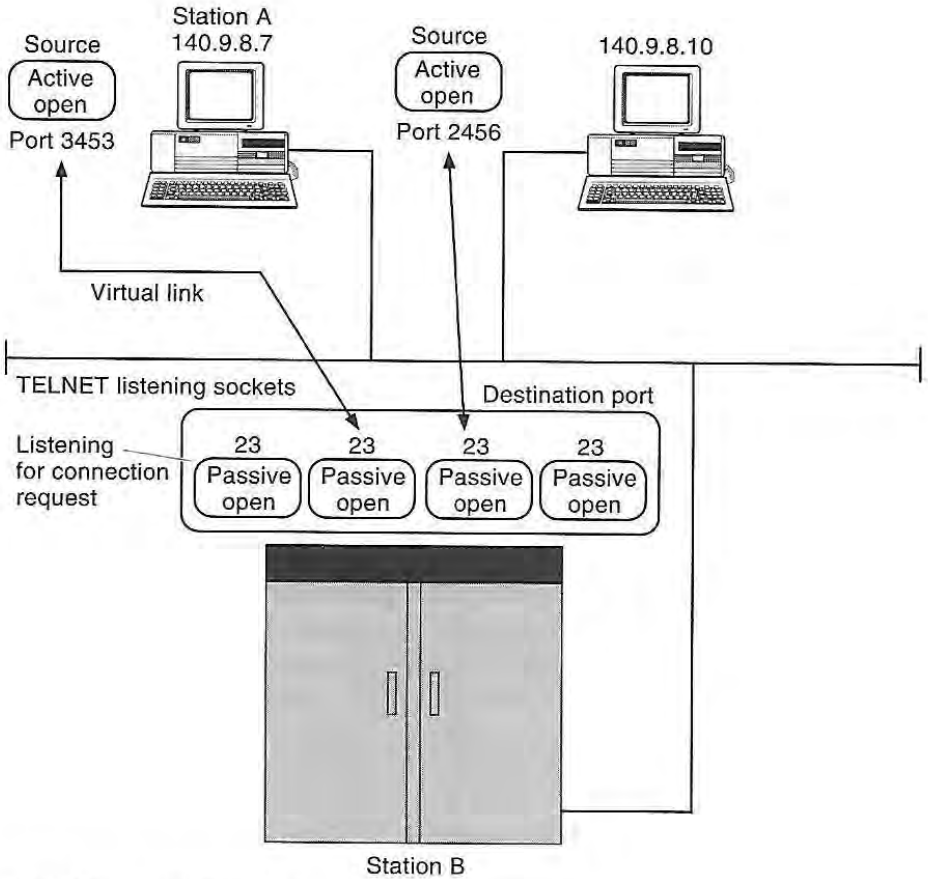


Figure 6.25 Passive and active connection.

A connection will only be an active connection after the sender and receiver exchange a few control packets to establish the connection. This is known as the *three-way handshake.* It is shown in the top portion of Fig. 6.26. Its purpose to synchronize each end of the connection with sequence numbers and acknowledgment numbers.

Refer to the bottom picture on Fig. 6.26. Station A will place an active open call to TCP to request connection to a remote network station's application. Station A will build a TCP header with the SYN (the sync bit shown in Fig. 6.24) bit set and then assign an initial sequence number (say, 100) and place it in the sequence number field. Other fields will be set in the TCP header (not pertinent to us at this time) and the packet will be given to IP for transmission to station B.

Station B will receive this packet and notice it is a connection attempt. If station B can accept a new connection it will acknowledge station A by building a new packet. Station B will set the SYN and the ACK bits in the TCP header shown in Fig. 6.24, place its own initial sequence number (200) in the sequence field of the packet, and the acknowledgment field will be set to 101 (the station A sequence number plus 1, indicating the next expected sequence number).

Station A will receive this response packet and notice it is an acknowledgment to its connection request. Station A will build a new packet, set the ACK bit, fill in the sequence number to 101, fill in the acknowledgment number to 200 + 1, and send the packet to station B. Once this has been established, the connection is active and data and commands from the application (such as TELNET) may pass over the connection. As data and commands pass over the connection, each side of the connection will maintain its own sequence number tables for data being sent and received across the connection. They will always be in ascending order.

Once the connection is established, TCP will allow data to flow between the source and destination station through something known as a *segment.* A TCP segment will contain the TCP header (shown in Fig. 6.24) and its data. The data handed to TCP for transmission is known as a *stream*—more specifically, an *unstructured stream.* A stream is a flow of bytes of data and an unstructured stream is unknown type of data flow of bytes. This means that TCP has no way of marking the data to indicate the ending of a record or the type of data that is in the stream. When TCP receives a data stream from the application, it will divide the data into segments for transmission to the remote network station.

A TCP segment may be as long as 65535 bytes, but is usually much less than that. Ethernet can only handle 1500 bytes of data in the data field of the Ethernet packet (Ethernet V2.0, 1496 bytes for IEEE 802.3 using IEEE 802.2). On the other hand, FDDI can handle a maximum

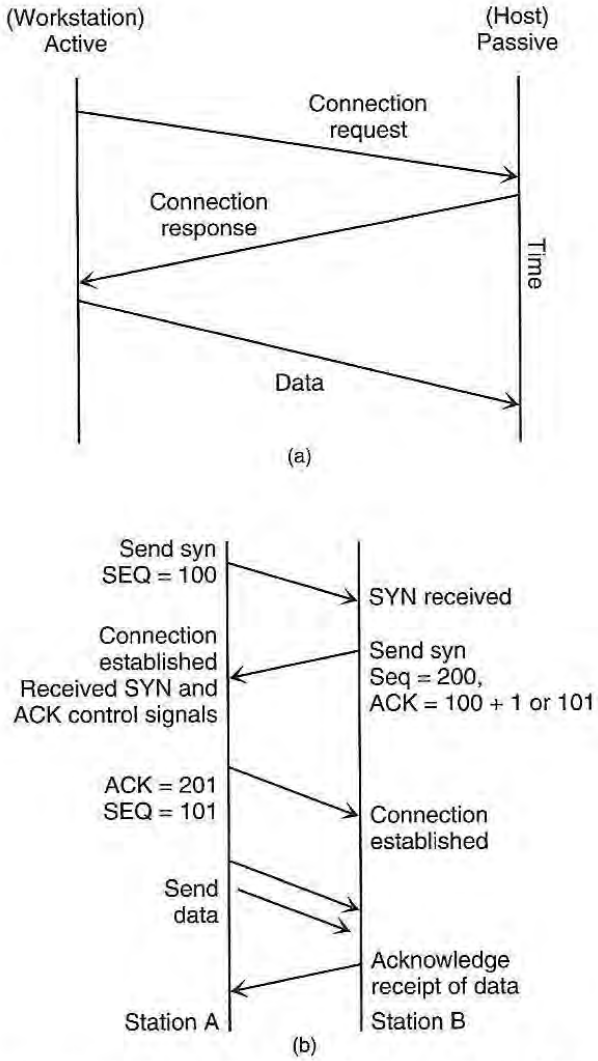


Figure 6.26 Block diagram of the three-way handshake.

of 4472 bytes of data in a packet and Token Ring packet size varies depending on the speed. For 4 Mbps, the maximum size is 4472 bytes. For 16 Mbps, the maximum size of the packet is 17800* bytes, but is usually set to 4472 bytes. To negotiate a segment size, TCP will use one of the options fields located in the TCP header to indicate the largest segment size it can receive, and submit this packet to the remote network station.

TCP does not care what the data is. Once the connection is established, TCP's main job is to maintain the connection (or multiple con-

* Although it may be this large, it is usually set to 4472 or below.

nections if there ar
bers, acknowledgm
management.

Since the connec
(by way of a succes
the connection. The
is sequence numbe

Sequence numbers
number for each by
byte of data that is
by one. Let's say a
Refer to Fig. 6.23
sequence number o
increment its seque
tion B (containing t
the second packet t
mitted. Station A's
for an acknowledgm

Each packet will
sion window (windo
ber is set to the nur
TCP segment (the d

Each packet tran
may be sent with
This is called an inc
across the same co
ACK bit set. With th
field to find the nex
expects. In other w
sequence number of
of the bytes successf
ber is stuffed into a
usually not a separa
ets. All data bytes u
ered good and accep

Since TCP is a b
acknowledgments a
ensure the integrity
that took every byte
destination. Local Ar
Xerox XNS were d
(shielded copper cab
numbers are based
packets. These proto

nections if there are). This is accomplished through the sequence numbers, acknowledgments and retransmissions, flow control, and window management.

Since the connection between stations A and B is now established (by way of a successful three-way handshake), TCP must now manage the connection. The first of the management techniques to be discussed is sequence numbers.

Sequence numbers and acknowledgments. TCP calculates a sequence number for each byte of data in the segment taken as a sum. For each *byte* of data that is to be transmitted, the sequence number *increments by one.* Let's say a connection was made between stations A and B. Refer to Fig. 6.23. Station A sends a packet to station B with a sequence number of 40 and knows the packet contained 4 bytes. It will increment its sequence number to 44. Upon acknowledgment from station B (containing the ACK number of 44), station A will then transmit the second packet to station B and know that 7 bytes are being transmitted. Station A's sequence number increments to 51, and it will wait for an acknowledgment from station B.

Each packet will contain as many bytes as will fit into the transmission window (windows are discussed in a moment). The sequence number is set to the number of the first byte in the packet being sent. The TCP segment (the data) is then given to IP for delivery to the network.

Each packet transmitted must be acknowledged. Multiple packets may be sent with one acknowledgment to all received good packets. This is called an inclusive ACK. TCP accomplishes this bidirectionally across the same connection. Each packet transmitted will have the ACK bit set. With the ACK bit set, TCP will read the acknowledgment field to find the next byte number that the other end of the connection expects. In other words, the number in the ACK field equals the sequence number of the original packet transmitted plus the number of the bytes successfully received in that packet plus 1. The ACK number is stuffed into a data packet to make TCP more efficient. There is usually not a separate packet on the network used just for ACK packets. All data bytes up to but not including the ACK number are considered good and accepted by the receiver.

Since TCP is a byte-oriented transport protocol, sequencing and acknowledgments are accomplished for each byte of TCP data. To ensure the integrity of the data, TCP had to become a robust protocol that took every byte and ensured that it would make it across to the destination. Local Area Network protocols such as Novell NetWare and Xerox XNS were developed to work on high-reliability mediums (shielded copper cable in controlled environments). Their sequence numbers are based not on bytes in the packet but on the number of packets. These protocols are discussed in detail earlier in the book.

As shown in Fig. 6.26, the connection was established using an initial sequence number from the sender and an initial sequence number supplied by the receiver (the destination). Each side will maintain its own sequence number, which may be in the range of 0 to 2,147,483,647. Each side of a TCP connection knows the upper and lower limits of the sequence numbers, and once the limit has been hit, it will roll over to 0. The initialization sequence numbers are selected at random. Each side must ACK each packet's sequence number.

ACK NO = Sequence number + good bytes read in the segment + 1

This is a clean, fast, efficient way of determining which bytes were successfully received and which ones were not. The sender of data must retain a copy of transmitted data until it receives an acknowledgment for those bytes from the remote network station of a connection.

Acknowledgment packets are not necessarily separate packets with only the acknowledgment number in the packet. This would be inefficient. For example, if station A opened a connection to station B and station A and station B were sending data to each other, the ACK packet can be combined with the response data packet. In other words, one packet transmitted contains three things: the data from station B to station A, the acknowledgment from station B of the data previously sent from station A, and the sequence number for the data B is sending to A.

If the sender does not receive an acknowledgment within a specified time, it will resend the data starting from the first unacknowledged byte. TCP will time-out after a number of unsuccessful number of retransmissions. The retransmission of a packet is accomplished using the Go-back-to-N routine. Any number of outstanding packets may be not acknowledged. When the destination station does acknowledge the receipt of a series of packet, the source will look at the ACK number. All sequence numbers up to but not including the ACK number are considered received in good condition. This means that if a source station can start the sequence number with 3 and then send two packets containing 100 bytes each. When it receives an ACK from the destination of 203, it will know that both packets sent previously are considered received in good condition.

The number of outstanding packets allowed is the next topic of discussion.

Control through window management. *Two functions are required of* TCP in order for the protocol to manage the data over a connection. They are called *flow control* and *transmission control.* Do not confuse these functions with the ICMP source quench mechanism. Source quench is for a host to inform the source of transmissions that the host is full and the host would like the sender to slow its rate of transmission.

For those readers who do not understand flow control, it is a mechanism used to control the flow of data. For example, if data is being received at a destination station faster than that station can accept it, it needs to tell the source station to slow down or to stop completely until it can clear out some space (replenish buffers) to receive the data.

There are many methods employed for flow control. The method that is used by TCP is a window and is explained subsequently.

How many packets may be outstanding at any one time? Data management using a "window" is accomplished as shown by the following. Data for TCP to transmit to the remote network station will be accepted by TCP from an application. This data will be placed in memory where it will wait to be sent across a connection to a remote station (for IP to send the packet). TCP places a "window" over this data in which to structure the data: data sent and acknowledged, data sent but not acknowledged, and data waiting to be sent. This is called a *sliding window,* for the window will slide up the data segment as each data packet is sent and that segment acknowledged.

Refer to Fig. 6.27. Sequence numbers 100 to 104 have been transmitted to the destination station and the destination station has acknowledged receipt of these packet. Packets containing sequence numbers 105 to 108 have been transmitted by the source station, but it has not received acknowledgment of these packets. Packets containing sequence numbers 109 to 114 are still in the source station and are waiting to be sent. Packets containing 115 to 118 are not yet in the window.

The important thing to notice is the black box covering the sequence numbers. This is the window. It will constantly move in ascending sequence order upon receipt of acknowledgments from the destination station.

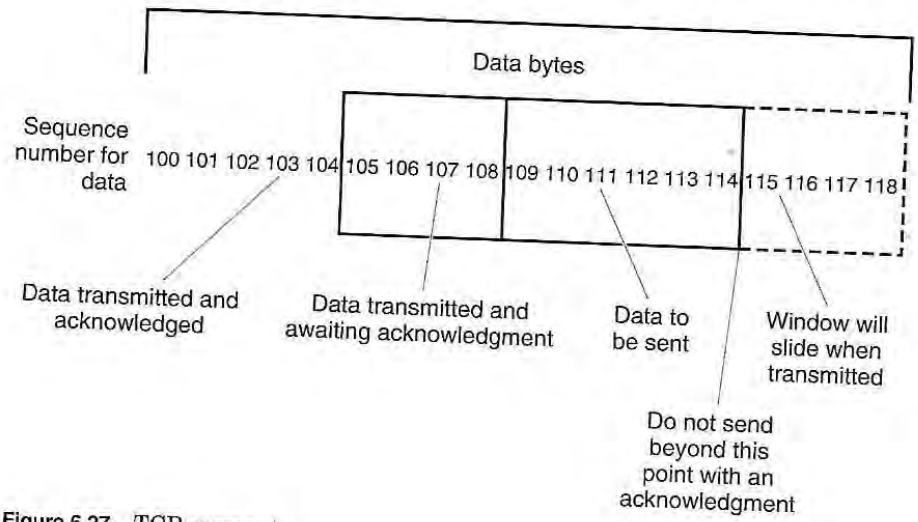


Figure 6.27 TCP segment.

For those readers who do not understand flow control, it is a mechanism used to control the flow of data. For example, if data is being received at a destination station faster than that station can accept it, it needs to tell the source station to slow down or to stop completely until it can clear out some space (replenish buffers) to receive the data.

There are many methods employed for flow control. The method that is used by TCP is a window and is explained subsequently.

How many packets may be outstanding at any one time? Data management using a "window" is accomplished as shown by the following. Data for TCP to transmit to the remote network station will be accepted by TCP from an application. This data will be placed in memory where it will wait to be sent across a connection to a remote station (for IP to send the packet). TCP places a "window" over this data in which to structure the data: data sent and acknowledged, data sent but not acknowledged, and data waiting to be sent. This is called a *sliding window,* for the window will slide up the data segment as each data packet is sent and that segment acknowledged.

Refer to Fig. 6.27. Sequence numbers 100 to 104 have been transmitted to the destination station and the destination station has acknowledged receipt of these packet. Packets containing sequence numbers 105 to 108 have been transmitted by the source station, but it has not received acknowledgment of these packets. Packets containing sequence numbers 109 to 114 are still in the source station and are waiting to be sent. Packets containing 115 to 118 are not yet in the window.

The important thing to notice is the black box covering the sequence numbers. This is the window. It will constantly move in ascending sequence order upon receipt of acknowledgments from the destination station.

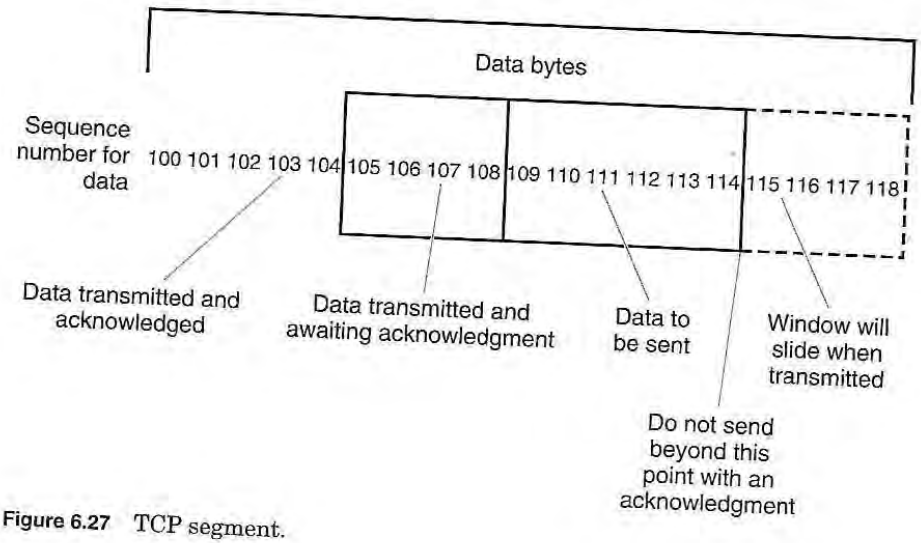


Figure 6.27 TCP segment.

When the receiving station (the destination station) is running low on buffer space (an area of memory to store incoming data), the receiver of this data has a capability to inform the sender to slow its transmission rate. This is accomplished through the window field in the TCP header packet. The destination station will inform the source station as to the amount of data it can accept. This is indicated by the window field of the TCP header. This field will contain the number of bytes (by indicating the range of sequence numbers) that the destination station is willing to accept. Figure 6.24 shows the TCP header, specifically the window field in the TCP header.

When the remote network station cannot accept any more data, it may set this window field to a 0. It will continue to submit these zero packets until it can again accept data (that is, the sender can send data to a host, and this host should respond with the ACK set to the previous ACK sent and a window set to 0). When buffer space is freed up, it can again submit a packet with the window size set to a nonzero number to indicate it can again accept data.

The one instance it may accept data, even with the window set to 0, is when it receives a packet with the urgent bit set, indicating the packet cannot wait. Any packet with the urgent bit set means that the packet must be received immediately. This is indicated by the URG bit in Fig. 6.24.

This connection management technique allows TCP to maintain control of the data transfer over a connection by informing TCP on the sending side how much data the receiver is willing to accept. This enables a sender and receiver of data on a connection to maintain consistent data flow over the connection.

Differentiating connections through PORTS. Like the UDP transport protocol, TCP uses ports to enable access to an application program. The well-known (also known as reserved or static) ports are shown in Table 6.9. When a connection is established between two network stations, the originating station must be able to tell the receiving station which application it would like to use. Likewise, the originating station must indicate to the receiving station where to send data back. As shown in Fig. 6.25, ports are assigned to allow this communication.

Refer to Fig. 6.28. Two personal computers have connections established to a host computer. They each indicated to the host station that they wanted to establish TELNET communications with it (TELNET, a remote terminal program will be explained later). The TCP/IP software will know this, for the incoming connection packet stated that the port is number 23. This is a well-known port assignment reserved for the TELNET application. The personal computer's port assignments (source port indicated in the TCP header, see Fig. 6.28) were chosen at random by the TCP software running in the personal computer. Any

TABLE 6.9 TCP Por

0	R
1	T
5	R
7	E
9	D
11	A
13	D
15	N
17	Q
19	C
20	F
21	F
23	T
25	S
37	T
42	N
43	W
53	D
77	A
79	F
93	D
95	S
101	N
102	O
103	X.
104	X.
111	S
113	A
117	U
119	U
129	P
139	N
160–223	R

* Some port number
UDP. This means that
its transport service.

random port assi
software as long
header being 16
reserved ports) m

One final note
lot of the structu
protocol of OSI k

As stated befor
number is known
nection point (ap
a port number is
it becomes the so

TABLE 6.9 TCP Port Assignments*

0	Reserved
1	TCP multiplexor
5	RJE
7	Echo
9	Discard
11	Active users
13	Daytime
15	Network status program
17	Quote of the day
19	Character generator
20	FTP—data connection
21	FTP—command connection
23	TELNET
25	Simple mail transport protocol
37	Time
42	Name server
43	Who is
53	Domain name server
77	Any private RJE service
79	Finger—find a active user
93	Device control protocol
95	SUPDUP protocol
101	Network info. center host name server
102	OSI-transport service access point
103	X.400 mail service
104	X.400 mail sending
111	Sun Microsystems remote procedural call
113	Authentication service
117	UNIX to UNIX copy (UUCP) path service
119	Usenet news transfer protocol
129	Password generator protocol
139	NetBIOS session service
160–223	Reserved

* Some port numbers are assigned with the same numbers as UDP. This means that the application may use TCP or UDP as its transport service.

random port assignment can be issued by the personal computer's TCP software as long as it is above 512. With the field for ports in the TCP header being 16 bits wide, up to 65535 ports (minus the first 512 reserved ports) may be dynamically assigned.

One final note on TCP. The merits of this protocol are far reaching. A lot of the structures used to build TCP were incorporated into the OSI protocol of OSI known as TP4.

As stated before, the combination of the internet address and its port number is known as a *socket.* Port numbers indicate only the end connection point (application program identifier) of the connection. When a port number is used with the internet address of a network station, it becomes the socket number.

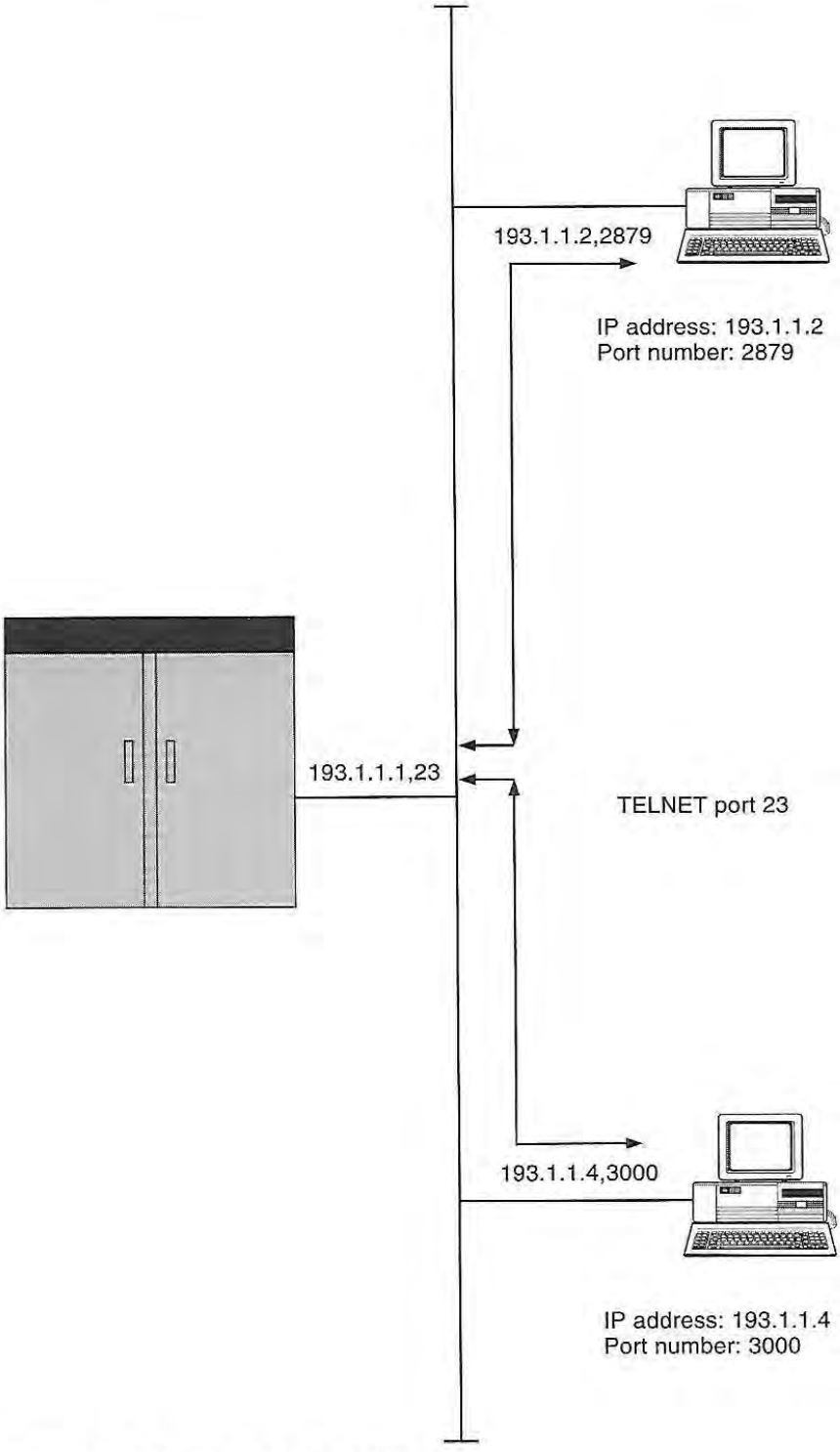


Figure 6.28 Telneting to the same host.

The second thir
server has accept
ers gained access
connection unique
personal compute
computer is using

Each *active* call
bers above 512. T
the port 23 on the
connection as a pr
be uniquely ident
then return to wai
many *passive* oper

Connection stat
tion. This could
closed, etc.

Local address.
tion. If this conn
0.0.0.0.

Local port. Co

Remote address.

Remote port. c

Termination of a co
minate a connectio
header. Since TCP
nection must close
cating devices, en
running on end sta
nection by sending
station B will ackn
data from end stat
cation to send to e

TABLE 6.10 TCP Conn

	Conne sta
Connection 1	
Connection 2	
Connection 3	
Connection x	

The second thing to notice about Fig. 6.28 is that the host TELNET server has accepted two TELNET connections. Both personal computers gained access to the host computer using port 23. What makes each connection unique in this case is the dynamic port assignment on the personal computer. Each TELNET connection initialed on the personal computer is using a different port number.

Each *active* call on a TCP/IP internet will assign its own port numbers above 512. The host has two TELNET connections connected to the port 23 on the host computer. The host computer will spawn each connection as a process to the operating system. Each connection will be uniquely identified by the socket address. The host computer will then return to waiting for any incoming connections, depending on how many *passive* open calls it has allowed. (See Table 6.10.)

Connection state. This describes the current status of the connection. This could be in the listen mode, in the process of termination, closed, etc.

Local address. Contains the local IP address for the TCP connection. If this connection is in the listen state, it should be filled in with 0.0.0.0.

Local port. Contains the local port number for the connection

Remote address. contains the remote IP address for the connection.

Remote port. contains the remote port number for the connection.

Termination of a connection. Finally, TCP must be able to gracefully terminate a connection. This is accomplished using the FIN bit in the TCP header. Since TCP offers a full-duplex connection, each side of the connection must close the connection. Refer to Fig. 6.29 for two communicating devices, end station A and host station B. The application running on end station A indicates to host B that it wishes to close a connection by sending a packet to host station B with the FIN bit set. Host station B will acknowledge that packet and will now no longer accept data from end station A. Host station B will accept data from its application to send to end station A, though. End station A will continue to

TABLE 6.10 TCP Connections

	Connection state	Local address	Local port	Remote address	Remote port
Connection 1					
Connection 2					
Connection 3					
Connection x					

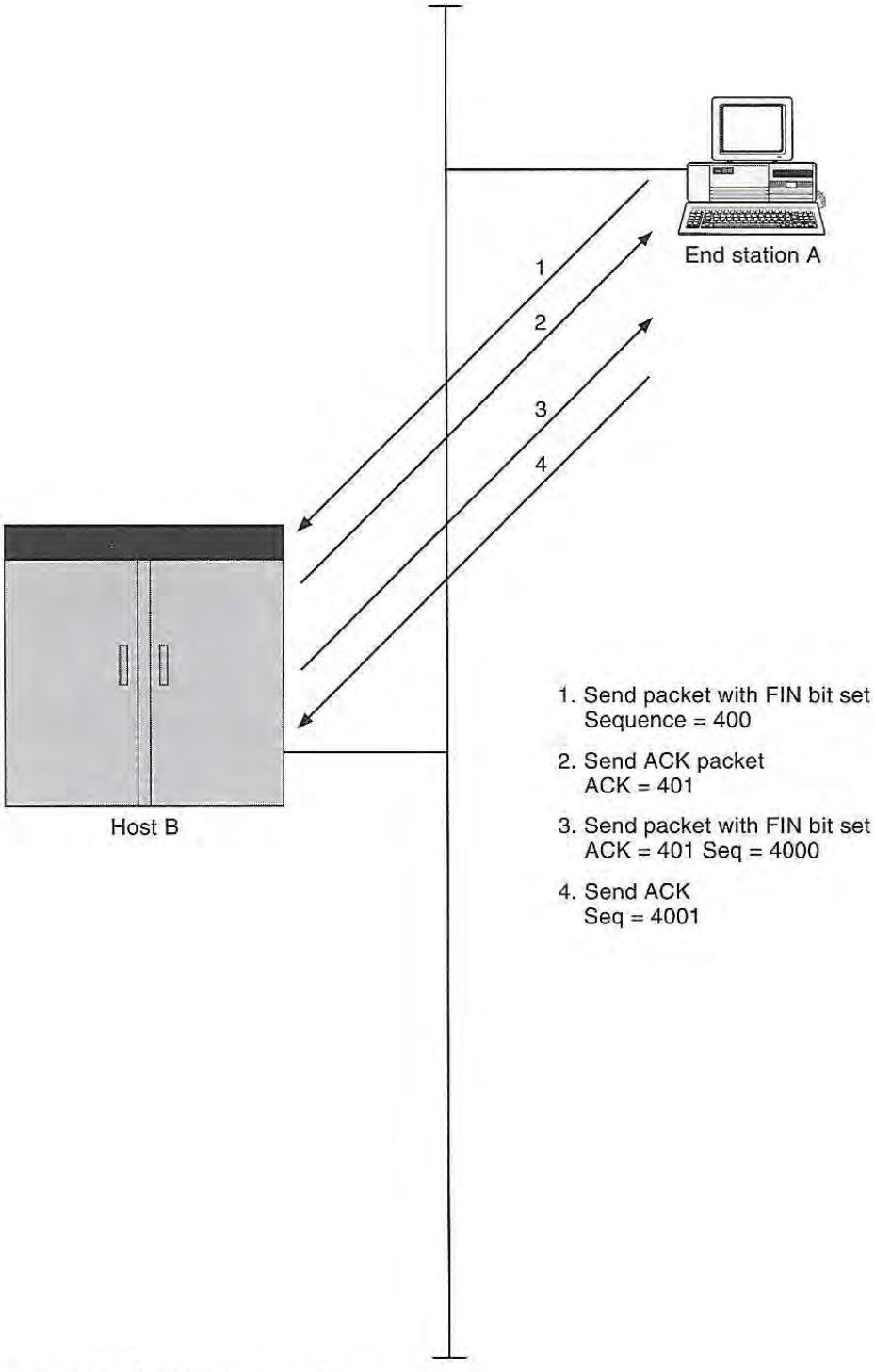


Figure 6.29 Terminating a session.

accept data from host station B. This way, station A can, at a minimum, accept a FIN packet from host station B to completely close the connection. To finalize the closing of this connection, host station B will send a packet to end station A with the FIN bit set. End station A will ACK this packet and the connection is closed. If no ACK is received, FINs will be retransmitted and will eventually time-out if there is no response.

Retransmission timer. One last function to discuss is TCP's capability to know when to send a retransmission of a packet. Retransmissions occur when a segment of data has not be acknowledged within an *allotted time.* This allotted time is dynamic (not held to one number) and is accomplished as follows: When TCP submits a packet to be sent, TCP records the time of the transmission and the sequence number of the packet. When TCP receives an acknowledgment to that sequence number, it will again record the time. TCP actually records the difference in time between the packet sent and the acknowledgment that was received. TCP uses this time to build an average time for a packet to be sent and an acknowledgment to be received.

This calculation of time used for TCP is a transport-layer protocol that was devised to run on multiple network-layer protocols. TCP knows that the packet to be transmitted may run through long delays on multiple types of networks. Therefore, it is impossible to accurately (in advance of sending the packet) gauge the amount of time it will take to receive an acknowledgment. TCP uses this timer to determine when to send a retransmission. The time is dynamic and has the capability to constantly change.

Section 3: Selected TCP/IP Applications

This section will give you an introduction to TELNET, FTP, SMTP, and DNS.

There can be many applications written for the TCP/IP environment, and many exist today. There are word processing systems, CAD/CAM systems, mail systems, and so on. The four most common applications that run on the TCP/IP network system are:

1. TELNET
2. File Transfer Protocol (FTP)
3. Simple Mail Transfer Protocol (SMTP)
4. Domain Name Service (DNS)

These applications are fully documented in the RFCs and almost always will be delivered with any TCP/IP protocol suite in the market today. These applications are usually included with every vendor's

implementation of TCP/IP. These means that you can switch to almost any type of computer using TCP applications software and the commands and functions of these programs will be the same.

The applications were specifically written for TCP/IP and basically provide almost all the applications that users need to access on any network. Database programs, word processing programs, and so forth, are all viable programs, but are not pertinent to the operation of a TCP/IP network. Using the foregoing application, a user can find any other needed application on the internet. The ones listed previously are the bare minimum needed to create a networked user environment in which all users can actively communicate and share data with each other across the network.

One nice thing about these available network applications is that they run on TCP/IP no matter which operating system is being used. The commands, their connection techniques, the commands that control the application, and the interface to the user almost always will be the same. So, if you normally work with UNIX and then switch for a day to DOS, the same FTP commands that operated on the UNIX machine will be there in the DOS machine. It is hard to say that with most applications that run on any system today. The discussion starts with the TELNET protocol.

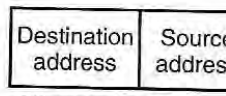
The protocols are covered briefly. Please refer to the TCP/IP books at the back of this book for more information about these protocols.

Telnet

This is pronounced TELNET and not Telenet. Telenet is a wide-area packet-switching technology based on the CCITT X.25 recommendation. TELNET is a TCP/IP application that provides remote terminal services.

TELNET allows a user from his or her network workstation to log in to a remote network station across the network and act as if the terminal were directly connected. The network may be local or it may be geographically distant (i.e., Virginia to Texas). TELNET is a relatively simple protocol compared to the complex terminal emulators that are represented in the personal computer arena today. In effect, it is a completely different application, for those emulators usually provide asynchronous terminal emulation connection, whereas TELNET provides network terminal emulation. The main reason for its popularity is that it is an open specification (in the public domain) and is widely available throughout all the TCP/IP company product offerings. There is even a version which provides 3270 emulation across TELNET connections. It is called TN3270. It does require a TN3270 server program at the host end.

As shown in Fig. 6.30, the TELNET protocol is encapsulated in a TCP header. The user starts the TELNET protocol at his or her work-



Ethernet frame

Figure 6.30 Telnet e

station, usually
TELNET applica
The argument a
TELNET process
fied by the argun
and attempt to e
ing the services
address; DNS w
plied, TELNET
the DNS or an IP

Refer to Fig. 6
association with
tocol was written
tems. Therefore,
TELNET protoco
connection with
system for person
line feed) be used
UNIX require a
example is the e
TELNET protocol
do the echoing of

During the con
station, the two s
cate how each end
nection. These op

* An argument is a
on a workstation. For
wanted a TELNET co
ment to the TELNET

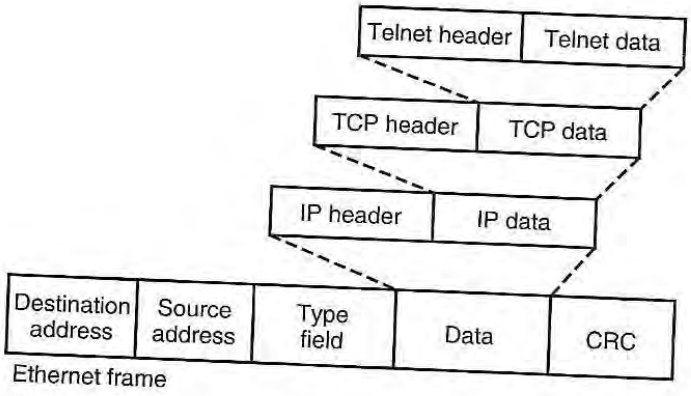


Figure 6.30 Telnet encapsulation.

station, usually by typing *TELNET <domain name or IP address>*. The TELNET application may be started with or without an argument.* The argument allows a simpler procedure to be invoked so that the TELNET process would automatically try to connect to the host signified by the argument statement. The TELNET application would start and attempt to establish a connection to the remote device (by accessing the services of the domain name server or directly with the IP address; DNS will be discussed later). If an argument was not supplied, TELNET would wait for the user to OPEN a connection using the DNS or an IP address.

Refer to Fig. 6.31. This figure shows the TELNET process and its association with a computer and operating system. The TELNET protocol was written so that it would work on a variety of operating systems. Therefore, before a connection is made to the remote device, the TELNET protocol has some work to do in order to synchronize the connection with the remote device. For example, the DOS operating system for personal computers requires that a CR-LF (carriage return-line feed) be used to terminate a line of text. Other systems such as UNIX require a line of text to be terminated with an LF. Another example is the echoing of characters. Upon connection attempt, the TELNET protocol will negotiate with the remote device as to who will do the echoing of typed characters to the initiator of a connection.

During the connection attempt between a source and destination station, the two stations will communicate options. These options indicate how each end of the connection will respond on the TELNET connection. These options include:

---

* An argument is a parameter that is typed in after the application name is typed in on a workstation. For example, if you wanted to invoke the TELNET application and wanted a TELNET connection to destination host 129.1.1.1, then 129.1.1.1 is the argument to the TELNET application program.

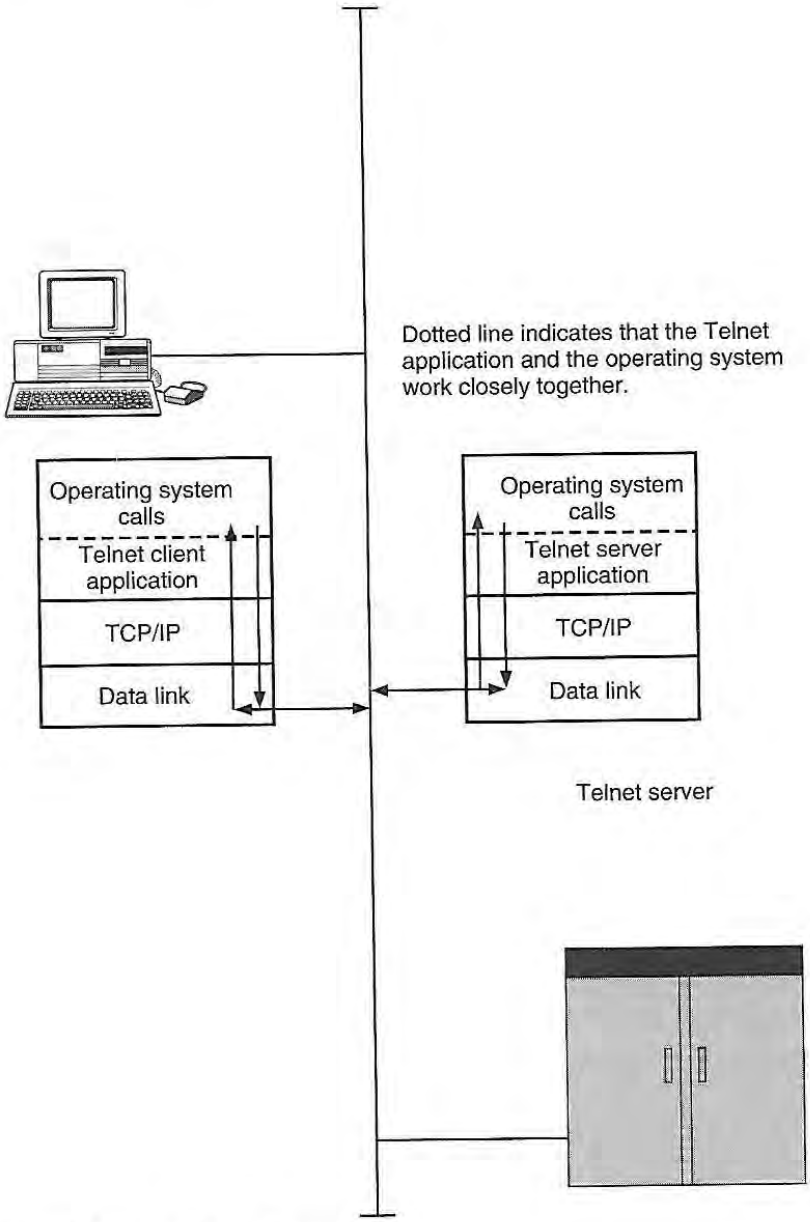


Figure 6.31 The Telnet process.

1. The ability to
2. Allowing one s
3. Specifying a te
4. Requesting th
   nection
5. Setting a timi
6. The ability to
7. Setting line m
   a character-at
8. Stopping the g

The options ar
following manne

Request

Will <option>

For example, will
provide the echoi
meaning the rem
will not allow sta

Agreement bet
<option> will be r

An example of
was running on a
echo, upon the c
would be WILL E
PC had been set
remote end to pr
ECHO and the re

Using the WILL
side of the connec
side provides serv

The TELNET c
TCP/IP network.
nals were conne
attached to the ho
nal service for the
minals, dependin
There are TELNE
minals, IBM3270

1. The ability to change from 7-bit text to 8-bit binary
2. Allowing one side or the other to echo characters
3. Specifying a terminal type
4. Requesting the status of a TELNET option from the remote connection
5. Setting a timing mark to synchronize two ends of a connection
6. The ability to terminate a record with an EOR code
7. Setting line mode so that strings of characters may be sent instead a character-at-a-time transmit
8. Stopping the go-ahead signal after data

The options are negotiated between the two network stations in the following manner:

Request	**Response**
Will <option>	Do or Don't <option>

For example, will echo from station A is requesting that station A will provide the echoing of characters. The response will either be DO echo, meaning the remote end agrees, or Don't echo, meaning the remote end will not allow station A to echo.

Agreement between the two TELNET ends communicated for a DO <option> will be responded to with a Will <option> or Won't <option>.

An example of this option negotiation: If the TELNET application was running on a DOS personal computer and it was set up for local echo, upon the connection setup, the TELNET option from the PC would be WILL ECHO and the response should be DO ECHO. If the PC had been set up without the local echo option and you wish the remote end to provide echo, the PC should negotiate echo with DO ECHO and the response would be WILL ECHO.

Using the WILL, WON'T and DO, DON'T provides symmetry. Either side of the connection can provide the command or the response. One side provides services in exactly the same manner as the other side.

The TELNET connection simply allows a terminal service over the TCP/IP network. Remember from Chap. 1 that computers and terminals were connected by a cable and the terminals were directly attached to the host computer. The TELNET service provides a terminal service for the network. It can emulate many different types of terminals, depending on the manufacturer of the TELNET program. There are TELNET programs that emulate DEC VTxxx series of terminals, IBM3270 terminals, etc.

The advantage to the TELNET program is that, with access to this program, a user may log on to any host on the TCP/IP internet (providing security options are allowed). Sessions are set up over the TCP/IP network.

Figure 6.32 shows a typical TELNET connection on a TCP/IP network.

File transfer protocol (FTP)

TELNET provides users with the ability to act as a local terminal even though users are not directly attached to the host. One other TCP/IP application that provides network services for users on a network is a file transfer protocol. With TCP/IP, there are three types of file access protocols in use: FTP, Trivial File Transfer Protocol TFTP, and Network File System (NFS). This FTP protocol provides for files to be transferred reliably across the network.

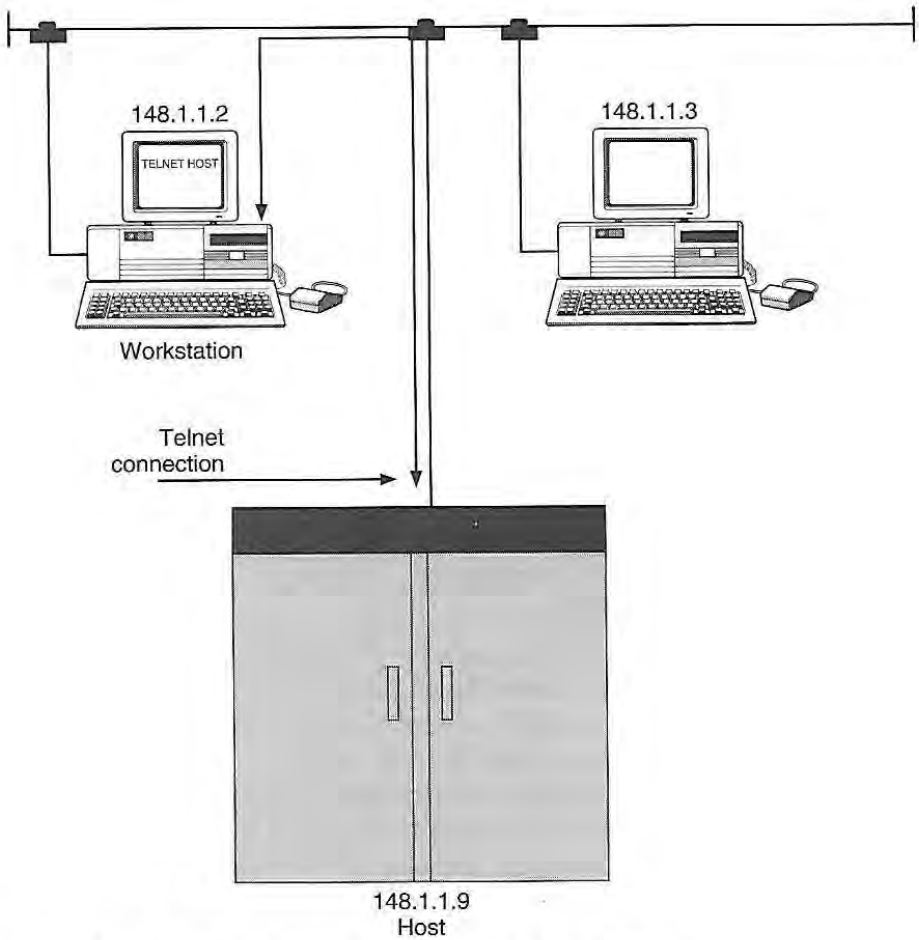


Figure 6.32 Remote terminal emulation over the network.

FTP is very ro
port and sockets
col actually use
tions): 20 and
network station
port. A network
tion must connec
for FTP to work.

Port 20 is used
the control conne
trol information.
ferred) to pass ov

Once the conne
mand from the cli
user would type
over to the remot
established betwe
the *data connecti*
file transfer proc
port is closed.

All data connec
known (or static)
a connection betw
ilar to TELNET;
type in FTP and
would use the op
lists the available

There are a lot
used. They are:

TABLE 6.11 FTP Com

!	cr
$	delete
account	debug
append	dir
ascii	disconn
bell	form
binary	get
bye	glob
case	hash
cd	help
cdup	lcd
close	ls