Configuring and Managing TCP/IP

Order No. 008543
Revision 01

Apollo Computer Inc.
330 Billerica Road
Chelmsford, MA 01824

First Printing: March, 1986.
Latest Printing: January, 1987.

This document was produced using the Interleaf Workstation Publishing Software (WPS). Interleaf and WPS are trademarks of Interleaf, Inc.

Preface

Configuring and Managing TCP/IP describes the configuration, management, and troubleshooting procedures for DOMAIN TCP/IP and DOMAIN®/IX™ BSD4.2 TCP/IP.

The Organization of this Manual

We've organized the information in this manual as follows:

Chapter 1	Provides an overview of DOMAIN TCP/IP and DOMAIN/IX BSD4.2 TCP/IP products and explains the relationships between them. It also provides an overview of TCP/IP communication concepts and sample configurations.
Chapter 2	Describes how to select TCP/IP Internet addresses for host and gateway nodes on your network.
Chapter 3	Describes the TCP/IP files that you must edit when configuring TCP/IP.
Chapter 4	Describes the DOMAIN TCP/IP servers and DOMAIN/IX BSD4.2 daemons that must be running to support TCP/IP communications.
Chapter 5	Describes how to configure each TCP/IP host and gateway node on your network. It provides procedures for configuring DOMAIN TCP/IP and DOMAIN/IX BSD4.2 on a DARPA Internet and within a DOMAIN network or internet.
Chapter 6	Describes how to manage TCP/IP software on a DOMAIN node. It includes procedures for updating hosts and TCP/IP information files.
Chapter 7	Describes how to troubleshoot TCP/IP. It includes information on how to locate causes of problems and how to monitor TCP/IP performance.
Appendix A	Describes how TCP/IP routes messages through a network.
Appendix B	Contains reference descriptions, in alphabetical order, of the TCP/IP management commands and utilities.
Appendix C	Lists error messages returned by TCP/IP software and indicates possible causes and corrective actions.
Glossary	Defines terms that are used in this manual.

Audience

This manual is written for system administrators who are responsible for establishing and maintaining TCP/IP communications. This manual describes how to set up and maintain a TCP/IP configuration. It does not contain complete information on system administration in DOMAIN networks or internets. (For details on these topics, see the "Related Manuals" section below.)

Related Manuals

This manual describes how to establish TCP/IP communications; it does not describe how to use the TCP/IP applications such as the Telnet terminal emulator or the file transfer program (FTP). For information on these applications, see *Using telnet and ftp* (008667).

For information about DOMAIN Internets see *Planning DOMAIN Internets* (009745) and *Managing DOMAIN Internets* (005694). For complete information on system administration, see *Administering Your DOMAIN System* (001746).

For detailed information on TCP/IP protocols, including Request for Comment (RFC) documentation, inquire with the Network Information Center, SRI International, Menlo Park, California 94025.

Problems, Questions, and Suggestions

We appreciate comments from the people who use our system. In order to make it easy for you to communicate with us, we provide the User Change Request (UCR) system for software-related comments, and the Reader's Response form for documentation comments. By using these formal channels you make it easy for us to respond to your comments.

You can get more information about how to submit a UCR by consulting the *DOMAIN System Command Reference*. Refer to the CRUCR (CREATE_USER_CHANGE_REQUEST) Shell command description. You can view the same description on-line by typing:

```
$ HELP CRUCR <RETURN>
```

For your documentation comments, we've included a Reader's Response form at the back of each manual.

Documentation Conventions

Unless otherwise noted in the text, this manual uses the following symbolic conventions.

italics	Italics in pathnames indicate DOMAIN System files or directory names that you must enter literally, that is, without any change; for example: *'node_data/startup.19l.*
bold	Bold words or characters in formats and command descriptions represent keywords that you must use literally. (In case-sensitive Shells, use the case shown in the command description.) Bold characters in text indicate program names or new concepts.
non-bold	Non-bold words or characters in formats and command descriptions represent values that you must supply. (In case-sensitive Shells, use the case shown in the command description.)
example	Bold or color words in command examples represent literal user keyboard input, that is, input you must enter exactly as it is indicated.
`output`	Typewriter font words in command examples represent system output. Typewriter font pathnames in text represent example, not actual pathnames. Here is an example pathname: `//node_1/pete_g`.

[ ]	Square brackets enclose optional items in formats and command descriptions. In sample Pascal statements, square brackets assume their Pascal meanings.
{ }	Braces enclose a list from which you must choose an item in formats and command descriptions.
|	A vertical bar separates items in a list of choices where only one choice is allowed.
< >	Angle brackets enclose the name of a key on the keyboard.

Contents

Procedures

Illustrations

Tables

Chapter 1

Overview of DOMAIN TCP/IP and DOMAIN/IX BSD4.2 TCP/IP

Transmission Control Protocol (TCP) and Internet Protocol (IP), commonly referred to as TCP/IP, provide services that allow different computers to communicate with each other. This chapter provides an overview of our available TCP/IP products, DOMAIN TCP/IP and DOMAIN/IX BSD4.2 TCP/IP. It also provides an introduction to TCP/IP communications concepts. While it is not an introduction to networking, it does explain some of the concepts required to understand TCP/IP. This chapter concludes with some sample TCP/IP configurations of DOMAIN TCP/IP and DOMAIN/IX BSD4.2 TCP/IP.

1.1. Overview of TCP/IP Products

TCP/IP is a standard protocol, defined by the Defense Advance Research Projects Agency (DARPA). It works on various types of computers so users can share the resources among many different machines. The most common applications that use TCP/IP communications are remote log in and file transfer.

The TCP/IP protocols were designed to provide communication services over a variety of physical networks — from computer networks to radio networks. TCP/IP can provide this broad communications service by defining protocols for *how* to send and receive messages, but does not define *what* the physical devices must do to send and receive the messages. By leaving the device details to those who want to implement TCP/IP, computers on numerous types of networks can use TCP/IP to communicate with each other.

Many computer manufacturers use TCP/IP as a way to communicate with competitors' computer systems because it is an industry–wide protocol.

We provide two TCP/IP products, DOMAIN TCP/IP and DOMAIN/IX BSD4.2 TCP/IP. These products perform similar functions, so which TCP/IP product you use depends primarily on which operating system you are most accustomed to using, AEGIS or DOMAIN/IX. Both products require similar TCP/IP information, but in a different format. Also, the contents of the two products differ as follows:

DOMAIN TCP/IP	Provides TCP/IP communications in a DOMAIN distributed environment. Nodes can run either or both AEGIS or DOMAIN/IX operating systems. The product contains DOMAIN Telnet for remote log in, and DOMAIN FTP for file transfer. It also contains the **makehost.sh** Shell script to help you build the files in the TCP/IP format required for *each* product, and provides more diagnostic tools than DOMAIN/IX BSD4.2 TCP/IP. DOMAIN TCP/IP is an optional product sold separately.
DOMAIN/IX BSD4.2 TCP/IP	Provides TCP/IP communications in a DOMAIN distributed environment where all nodes run the DOMAIN/IX operating system. This product does not contain DOMAIN Telnet or DOMAIN FTP; however, similar programs are available with DOMAIN/IX BSD4.2. The product also contains a variety of system utilities for DOMAIN/IX BSD4.2 users. DOMAIN/IX BSD4.2 TCP/IP comes with the standard DOMAIN/IX product, but must be installed separately.

DOMAIN/IX BSD4.2, like all standard BSD4.2 UNIX® systems, uses TCP/IP as its communication link between various operating system utilities. You can install DOMAIN/IX BSD4.2 TCP/IP to perform the following functions on your DOMAIN network:

BSD4.2 Utility	Allows users to:
ftp	Access the File Transfer Protocol, which lets you transfer files to and from a remote network site.
lpr	Queue and print files.
rcp	Copy files between machines.
rexec	Return a stream to a remote command.
rlogin	Connect your terminal to a remote host network.
rsh	Execute a shell command on a remote host.
ruptime	Get status of host on local machine. Status information includes the current time; the length of time the system has been up; and the average number of jobs in the run queue over the last one, five, and 15 minutes.
rwho	Determine who is logged in on all machines in the local network.
telnet	Access the Telnet Protocol, which lets you communicate with another host.

Both DOMAIN TCP/IP and DOMAIN/IX BSD4.2 TCP/IP enable you to communicate with computers on many other types of computer networks as long as they also have TCP/IP. The networks can be:

- Your local DOMAIN network.
- Another DOMAIN network that is part of a DOMAIN internet. (A DOMAIN Internet consists of two or more networks connected by a DOMAIN bridge product.)
- Another manufacturer's local area network (LAN) physically connected to your DOMAIN network or to another DOMAIN network within your internet.

Both TCP/IP products provide the TCP/IP *software* required to communicate within your DOMAIN network. However, to communicate within a DOMAIN internet, you must first install the appropriate bridge product to physically connect the DOMAIN networks. To communicate with other manufacturers' networks, one node must contain one of our DOMAIN network controller products, which makes the physical connection to each network. Contact your Sales Representative for the most current hardware products.

If your network contains nodes that run *both* DOMAIN and DOMAIN/IX BSD4.2, you can use either of the two TCP/IP products. Regardless of which TCP/IP product you install, you can use the procedures in this book to establish a TCP/IP network. The procedures explain how to:

- Select unique addresses for each node in your TCP/IP configuration.
- Determine which files TCP/IP requires to maintain routing information.
- Determine which server and daemon processes are required to run TCP/IP. (Server processes in the BSD4.2 UNIX environment are called **daemons**.)
- Set up the files and servers on *each* node that will be part of the TCP/IP configuration.
- Update and maintain the TCP/IP configuration once you've installed it.
- Locate problems with your TCP/IP communications.

Table 1–1 summarizes the available TCP/IP products. The next section provides more details on the TCP/IP protocols.

Table 1–1. TCP/IP Products

Product	Operating System	Description
DOMAIN TCP/IP	AEGIS and DOMAIN/IX	Provides communication to computers on foreign networks with the appropriate hardware.
DOMAIN/IX BSD4.2 TCP/IP	DOMAIN/IX	Provides BSD4.2 communication services in a DOMAIN network or internet. It requires no additional hardware. It also provides communication to computers on foreign networks with the appropriate hardware.

NOTE: Consult your sales or service representative for the most current information on DOMAIN network hardware products.

1.2. TCP/IP Gateways and Hosts

A node containing the TCP/IP software and appropriate hardware is called a TCP/IP **gateway**. It provides the physical link between different networks so that computers in each network can communicate with each other via TCP/IP. Computers that communicate with one another via TCP/IP communications are called TCP/IP **hosts**.

With this physical link, the gateway can route a message from one network to the other. Figure 1–1 shows how a gateway routes information. For a computer in Network A to communicate with a computer in Network B via TCP/IP, data from Network A must pass through the gateway on its way to Network B.

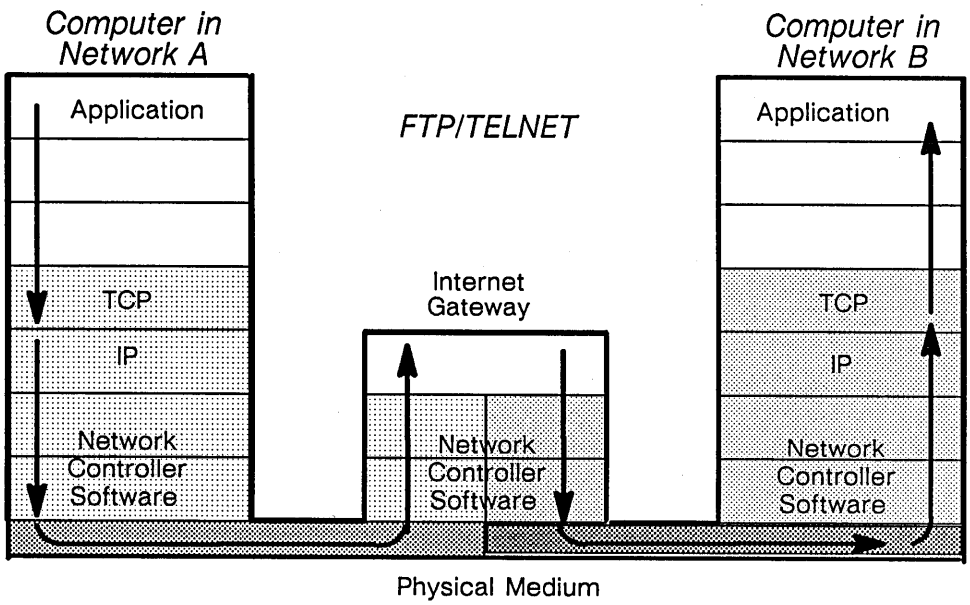

Figure 1–1. Internet Gateway Layers

Note that for our TCP/IP products, a TCP/IP gateway node is the node on which the hardware (either a DOMAIN bridge product or network controller) is located. A TCP/IP host is any node that has TCP/IP software to communicate with hosts on other computer networks. That is, the gateway performs the physical and routing functions to connect the networks, the host provides the applications such as FTP or Telnet. A node can be both a gateway and a node as long as it provides the hardware and software for both.

1.3. TCP/IP and the DARPA Internet

In addition to *physically* connecting two different networks with a TCP/IP gateway, both networks must follow some protocols to communicate. These protocols, defined by DARPA, control how networks assign addresses and route messages through the gateways. You can connect your network with any other network that conforms to the DARPA TCP/IP standards. By connecting your network to other DARPA–conforming networks, you are creating a **DARPA Internet**.

Note the difference between a DOMAIN internet and a DARPA Internet: A DOMAIN internet refers to a network of networks all running the DOMAIN distributed environment; nodes within the internet can also be running TCP/IP. A DARPA Internet is a network of *physically different* networks; computers within the DARPA Internet communicate via TCP/IP. You might also see the DARPA Internet called a TCP/IP internet, or simply Internet.

One of the largest DARPA internets that you can connect your network to is the nationwide network called the ARPANET. The ARPANET, the first large scale network using TCP/IP, was developed by the Department of Defense (DoD) and Bolt Baranek and Newman (BBN).

We refer to computers on other DARPA networks as **remote hosts**. We refer to other DARPA networks as **foreign** networks.

DOMAIN/IX BSD4.2 TCP/IP conforms to DARPA Internet standards. It also supports utilities defined by standard BSD4.2 UNIX®. So, you can connect your network with nodes running DOMAIN/IX BSD4.2 TCP/IP with other networks with nodes running the TCP/IP utilities.

We describe the DARPA standards in detail in Chapter 2, "Selecting Internet Addresses." For specific information about DARPA Internets, contact the Network Information Center (NIC) at SRI International. The NIC maintains specifications and detailed information about all the DARPA Internet protocols.

1.4. Examples of TCP/IP Configurations

Figure 1–2 illustrates a sample TCP/IP implementation, in which a DOMAIN ring network communicates with a network of computers that use the ETHERNET Local Area Network (LAN).

In this figure, nodes 1, 3, and 4 run TCP/IP software. Node 4 contains the ETHERNET controller and is a gateway between the DOMAIN network and the ETHERNET LAN. With this TCP/IP configuration, users at nodes 1 and 3 in Network A can communicate with systems 4 and 6 in Network B via the TCP/IP gateway (node 4). A user at node 4 could also communicate with systems 4 and 6 if node 4 also contains TCP/IP host software in addition to its gateway software.

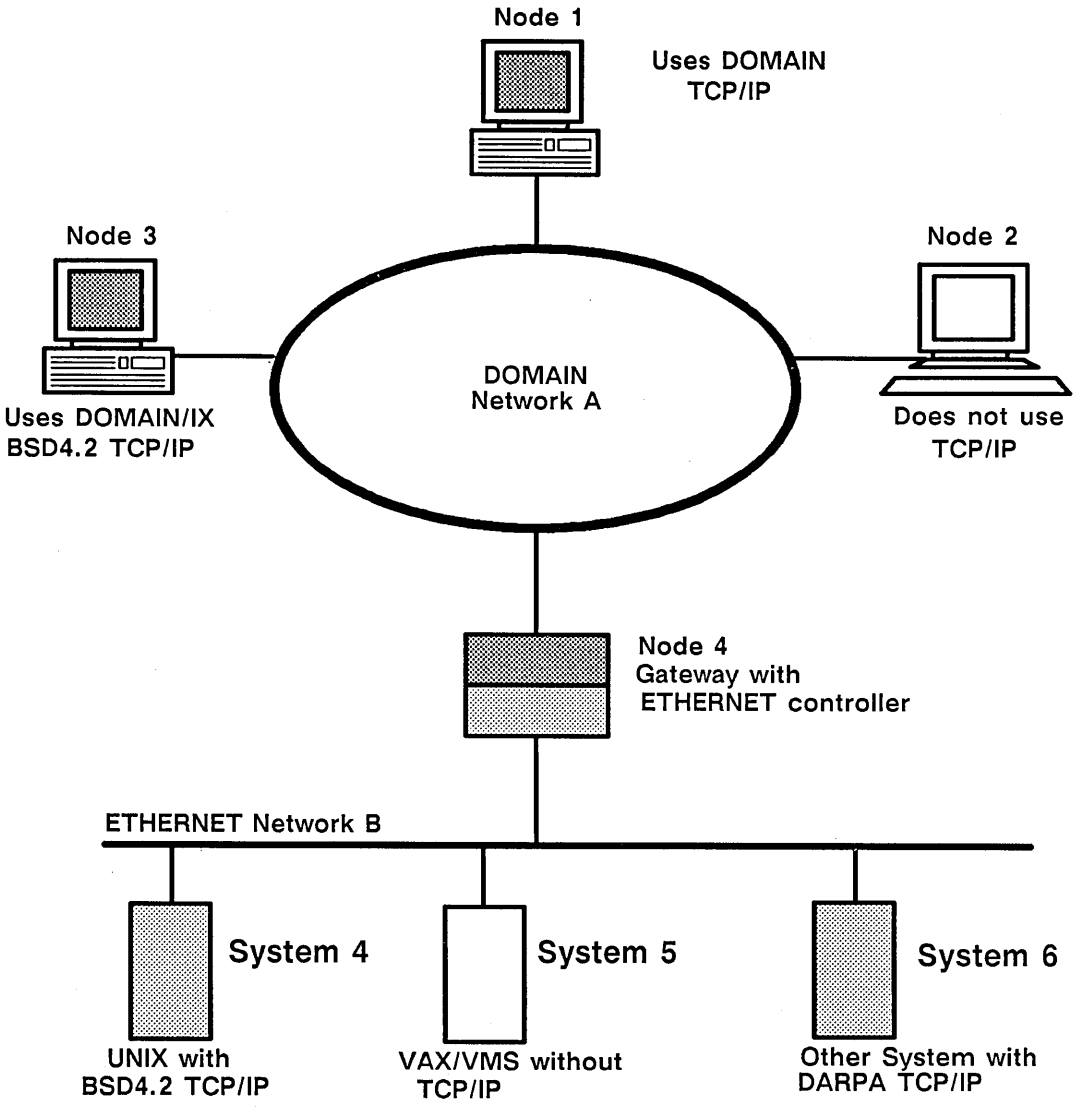

Figure 1–2. Sample Internet Configuration

A more complicated example appears in Figure 1–3, in which the TCP/IP DARPA Internet consists of seven networks, plus an interface to the ARPANET. The DARPA internet consists of: three DOMAIN networks, an ETHERNET LANs, and Network E. Also, the link between the bridge and the network is considered to be a network, so this example consists of two internet link networks.

Network E may or may not be an ETHERNET LAN, which illustrates that DOMAIN nodes can communicate with any remote computers that understand the Internet model; they are not limited to computers that run on an ETHERNET LAN.

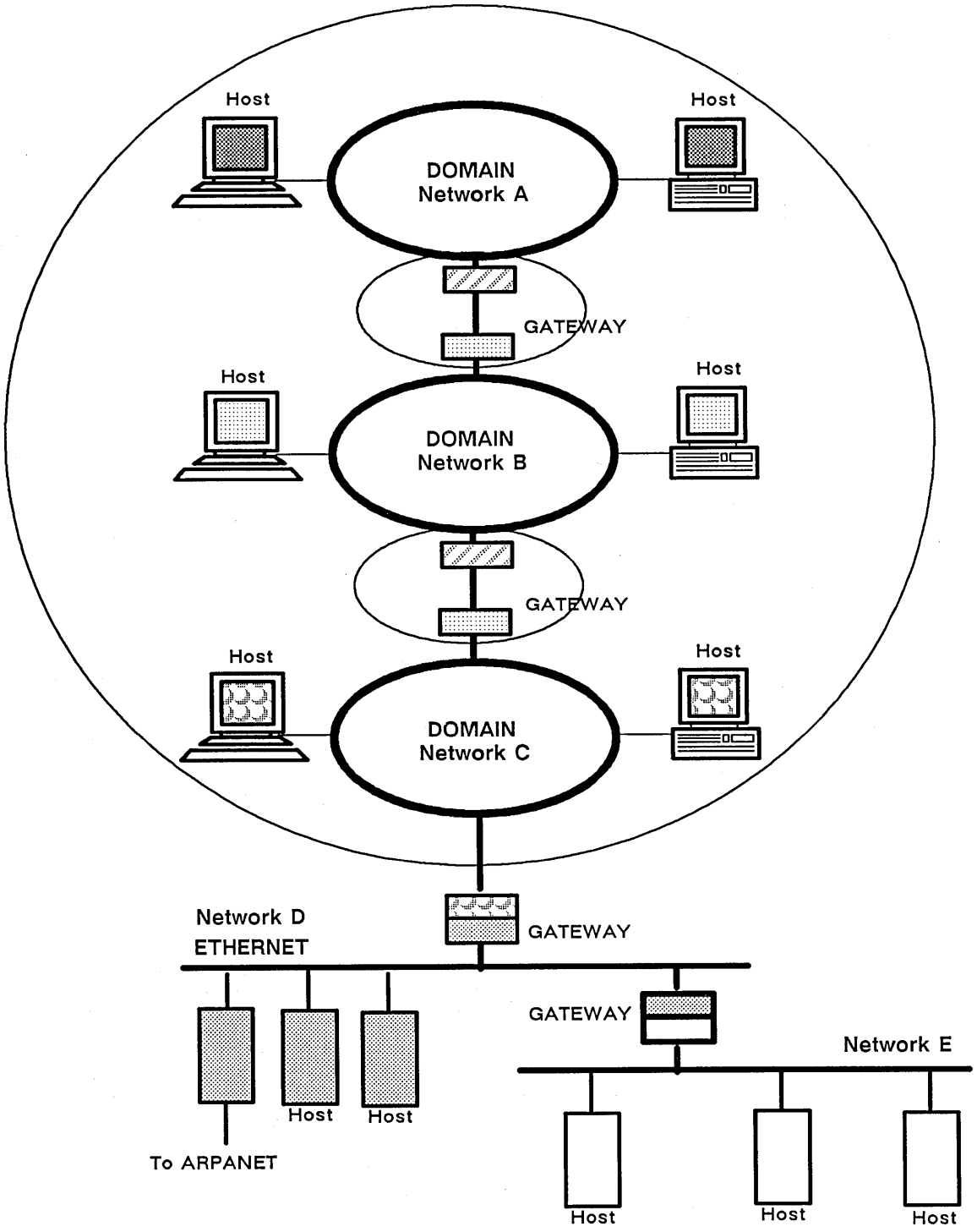

Figure 1–3. Multi–Network Internet Configuration

Chapter 2

Selecting Internet Addresses

To begin configuring TCP/IP, you must first determine which nodes in your network will use TCP/IP. Then you must select Internet addresses for each node. The **Internet address** allows communication between computers on different physical networks by providing a standard addressing mechanism that all the computers can understand.

This chapter describes the information that you must provide to select Internet addresses for all the nodes that will be part of the TCP/IP configuration.

Note that most of the information in this chapter is helpful if you are configuring TCP/IP on an entire network for the first time. If you're familiar with TCP/IP, or if you're updating an established TCP/IP configuration, you can skip this information and go to the configuration procedures in Chapter 5, "Configuring TCP/IP."

2.1. Drawing the Internet

To begin, draw a picture of the network (or internet) you're configuring. (See Figure 2–1 for an example.) Decide which nodes will be running TCP/IP as hosts and gateways. Desginate any node that users will use to run TCP/IP applications to be a TCP/IP host. To designate a node to be a gateway, it must have necessary hardware (either a network controller to communicate to a foreign host, or a DOMAIN Bridge product to communicate to another DOMAIN network in an internet). In addition to helping you configure TCP/IP, drawing this picture will help you later when you're trying to locate communication problems within your network.

Figure 2–1 shows the relationship between the nodes on the DOMAIN network and the other network. In this case, the other network is an ETHERNET LAN. In Figure 2–1, the TCP/IP hostname is PARIS, the gateway is BERLIN and the foreign hostname is MOSCOW.

After drawing the picture of your network and deciding which nodes are hosts and which nodes are gateways, you can assign addresses to each host and gateway. These addresses must follow the standard addressing format, as defined by DARPA. We describe this standard, called the **DARPA Internet address** format, in the next section.

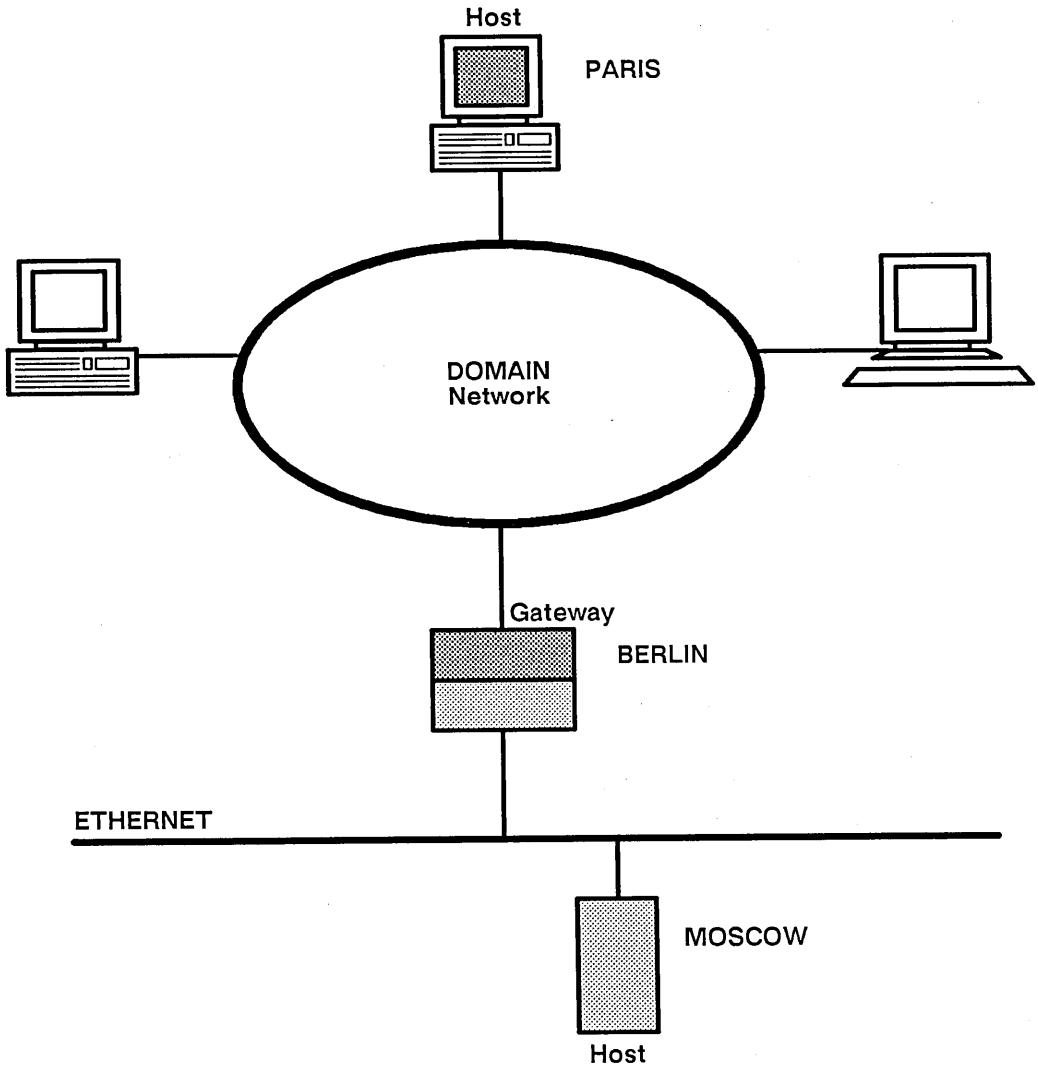

Figure 2–1. Drawing a Network

2.2. Selecting Internet Addresses

Whenever you refer to an object, be it a host or a file, you usually use a name that is easy to remember, but the operating system converts this name to an address value that's more meaningful to it. For example, you refer to your DOMAIN node by a name such as //PARIS while the operating system refers to it by an address such as 06d49.

When referring to a TCP/IP host, you can still use a mnemonic name. For simplicity, DOMAIN TCP/IP highly recommends (though it's not required) that the host name be the same as the node name without the slashes. For example, the TCP/IP host name for the node //PARIS can be PARIS. (Note that you can't include the slashes in the TCP/IP host name.)

Within the DOMAIN network, you can transfer messages simply by specifying the local name. However, to communicate with a different network, you need an additional addressing layer. For TCP/IP, you must also supply an **Internet address.**

2.2.1. Format of the Internet Address

A typical Internet address consists of two fields; the left field (or the **network number**) identifies the network, and the right field (or the **host number**) identifies the particular host within the network.

The DARPA Internet address is 32-bits long and can be interpreted differently to accommodate networks of varying sizes. The Type A address allows you to have one network with many hosts (up to 16,777,214). Type B only allows a network to have up to 65,534 hosts, but it allows you to have multiple networks. Type C allows you to have millions of networks with up to 254 hosts on each.

The size of network and host numbers depends on the address type. You can recognize a type by the value of the Most Significant Bit (MSB) or the leftmost bits in the address. For example:

- Type A addresses have a 7-bit network number, a 24-bit host number, and the value of the MSB is 0.
- Type B addresses have a 14-bit network number, a 16-bit host number, and the value of the two MSB's is 10 (in hexadecimal).
- Type C addresses have a 21-bit network number, an 8-bit host number, and the value of the three MSB's is 110 (in hexadecimal).

Figure 2-2 shows how a 32-bit Internet address is divided into network and host numbers. It also shows how the most significant bits (MSB) in each network number identify the address type.

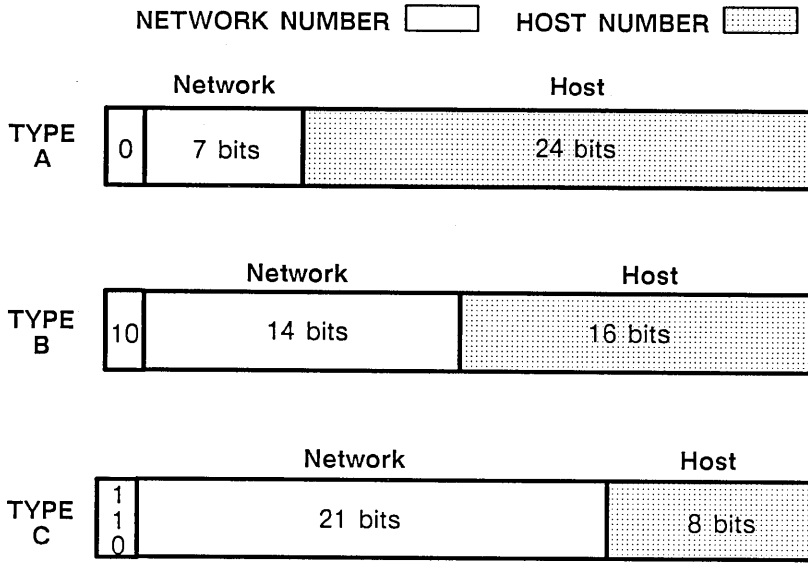

Figure 2-2. Type A, B, and C Internet Addresses

When selecting Internet addresses for your network, you don't need to calculate the size of the network and host fields. Instead, after choosing the type of address you want to use, you simply supply decimal numbers within a specific range. You must supply *decimal numbers* to conform to the DARPA Internet addressing standard format.

The standard DARPA Internet addressing format is:

W.X.Y.Z

where W, X, Y, and Z are decimal numbers between 0 and 255. Each of these decimal numbers represents one byte of the Internet address. The four bytes together represent both the network and host address. However, which numbers refer to the network and which numbers refer to the host number depends on the Internet address type (Type A, B, or C).

For example, Type C addresses have a one-byte host address so your host number can be any number within the range of 1 and 254. (DARPA reserves numbers 0 and 255.) Type C addresses have a 21-bit network address and a 2-bit MSB, so the network number will be 3 bytes long; and can fall within the range of 192.0.1 and 223.255.254. The number starts after 192 because the first three bits (0 through 192 in decimal) are reserved to signify the MSB.

Table 2-1 summarizes the ranges you can specify for network and host numbers of each type. By using this table to select numbers, you also avoid using the Internet addresses that DARPA reserves for its own use. For example, DARPA reserves network and host numbers that have a value of zero (all four numbers are 0) and a value of one (all four numbers have the decimal value of 255). It also reserves Type C network numbers greater than 223.255.254. If you use reserved numbers, TCP/IP might generate errors.

Table 2-1. Ranges of Values for Type A, B, and C Internet Addresses

	Format		Range of Values	
Type	Network	Host	Network	Host
A	1	3	1 - 126	0.0.1 - 255.255.254
B	2	2	128.1 - 191.254	0.1 - 255.254
C	1	3	192.0.1 - 223.255.254	1 - 254

Note that you must assign *two* Internet addresses to the gateway node since it belongs to two different networks. For example, the node BERLIN (host address 3.23) is on the DOMAIN network (network number 129.9) and the ETHERNET network (network number 149.8). The two Internet addresses can be 129.9.3.23 (DOMAIN Internet address) and 149.8.3.23 (ETHERNET address).

Returning to our configuration in Figure 2-1, we can assign Internet addresses. Table 2-2 lists the Internet addresses for the configuration in Figure 2-1. Note that we list BERLIN as a gateway for each network, and then as a host on the DOMAIN network because we want to use the DOMAIN gateway nodes as hosts too. (In general, gateway nodes are also TCP/IP hosts if users will be using the node to run TCP/IP applications.) We inserted question marks in places where we need to get information from the system administrator for the foreign network. To specify a **network address**, we simply specify the network number followed by an all-zero host number.

Table 2–2. Internet Addresses for our Sample Configuration

Type	Node Name	Internet Name	Internet Number: Network	Host
DOMAIN Network			129.9.	0.0
ETHERNET Network			149.8.	0.0
Gateway (DOMAIN)	//BERLIN	BERLIN	129.9.	3.23
Gateway (ETHERNET)	ask foreign net?	BERLIN	149.8.	3.23
Host (DOMAIN)	//PARIS	PARIS	129.9.	3.21
Host (ETHERNET)	ask foreign net?	MOSCOW	149.08.	5.30
Host (DOMAIN)	//BERLIN	BERLIN	129.9.	3.23

2.2.2. Creating Internet Addresses with Subnet Numbers

In addition to selecting Internet addresses that consist of your network and host numbers, you can also designate an intermediate number called a **subnet** number. You can use subnet numbers when you want to set up a hierarchy of Internet addresses within your network. That is, using subnets allows you to have one network number for your entire internet, and various subnet numbers for each network within your internet.

The following example illustrates the advantage of having subnet numbers. Consider two hosts on the ARPANET — one at the University of Southern California (USC) and the other at Massachusetts Institute of Technology (MIT). Since both hosts are part of a large campus internet that consists of numerous networks, sending messages is complicated when you don't have subnet numbers. To send a message from the USC host to the MIT host, the USC sender must know the specific network within the internet at MIT. That is, the sender at USC must know the network topology of the receiver at MIT. Moreover, if the MIT network changes, the USC sender might need to learn a *new* network address.

If the two colleges assign subnet numbers, sending messages between them is easy. The USC host sends a message to the MIT host simply by specifying an Internet address whose network number represents the entire MIT internet. When the message reaches the MIT gateway, the gateway checks whether subnets are implemented, and if so, relays the message to the appropriate network within the MIT internet.

To create subnets on your internet, you use the same Internet address format but it's interpreted differently. Rather than representing the network and host number, the 4-byte Internet address represents a network, subnet, and host number. Note that the size of the network number remains the same. You create a subnet by dividing the host number into a subnet and host number.

Figure 2–3 shows some possible ways you can subdivide an Internet address into network, subnet, and host numbers. You can actually sudivide it any way you want depending on the number of subnets (networks within the internet) and hosts you have.

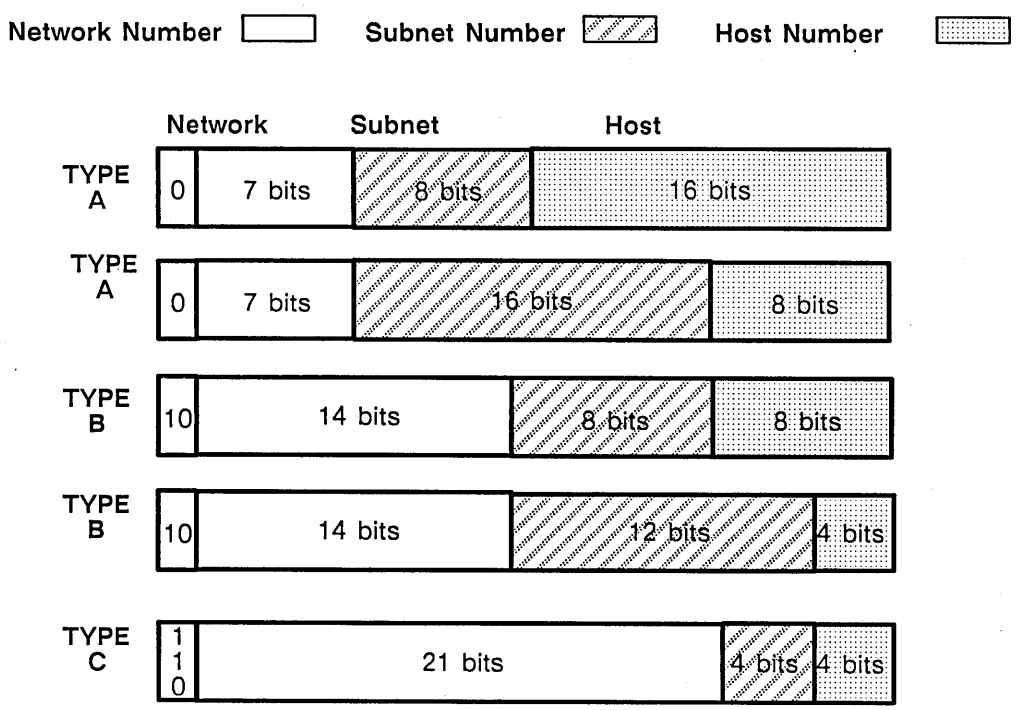

Figure 2–3. Internet Addresses with Subnet Numbers

To create a subnet, you subdivide the host portion of your Internet address. Table 2–3 lists the range of subnet and host values for each type. Note that since Type C host numbers are only 8–bits long, you're limited to 15 subnets and 14 hosts. For this reason, most users implement subnets with Type A or B addresses.

Table 2–3. Range of Subnet and Host Values for Type A, B, and C Addresses

Type	Size in Bits		Range of Values	
	Subnet	Host	Subnet	Host
A	16	8	0.1–255.255	1 – 254
A	8	16	1–255	0.1–255.254
B	8	8	1 – 255	1 – 254
C	4	4	1 – 15	1 – 14

As we stated earlier, using subnets does not change the Internet address format. Instead, you are changing how TCP/IP *interprets* the Internet address. You do so by supplying a bit mask or *subnet mask* in the */sys/node_data/networks* file for each node. By adding a subnet mask to the node's *networks* file, you're telling TCP/IP that your network uses subnets, and which part of the Internet address corresponds to the subnet numbers. We describe this file in Chapter 3, "Editing TCP/IP Files."

To understand how to divide an internet into TCP/IP subnets, refer to Figure 2–4 and Table 2–4. The figure is a drawing of a DOMAIN internet connected to a foreign network via TCP/IP. The table lists the Internet addresses corresponding to the figure. To assign the Internet addresses, we first assigned a network number to correspond to the entire DOMAIN internet and to the ETHERNET LAN. Then we assigned a subnet number to each network within our DOMAIN internet.

Note that in DOMAIN internets, the bridge connecting the two physical rings is actually a network itself, so that we must also assign network numbers to the links. You can think of a bridge between two DOMAIN networks as a special network that consists of only two gateways — each gateway connects the network on which it resides to the physical link (such as a T1 line or coaxial cable). Figure 2–4 illustrates this. Table 2–4 lists these bridge addresses as Type C to distinguish them from the other addresses.

Also in Table 2–4, we list the gateways BERLIN, LONDON and NYC as hosts, so the gateway and host entries are the same for these nodes. We inserted question marks for information that we would need to get from the system administrator of the foreign host.

To ensure that you understand how to assign these addresses, pencil in the addresses from Table 2–4 onto Figure 2–4.

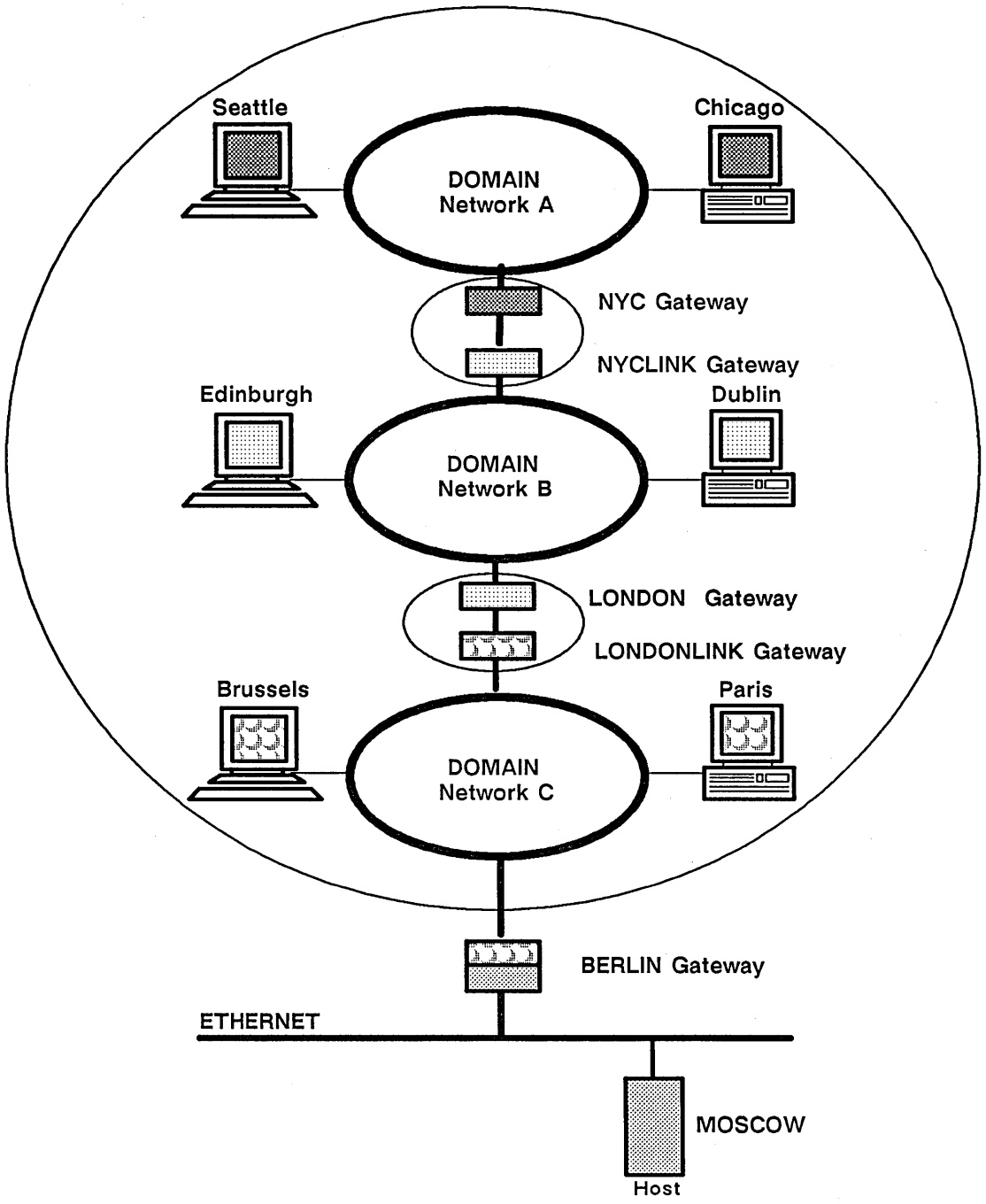

Figure 2–4. TCP/IP Configuration for DOMAIN Internet with Subnet Numbers

Table 2-4. Internet Addresses for Sample Subnet Configuration

Type	Node Name	Internet Name	Internet Number: Network	Subnet	Host
DOMAIN Internet			129.9.	0.	0
ETHERNET Network			149.8.	0.	0
Subnet DOMAIN Network A			129.9.	1.	0
Subnet DOMAIN Network B			129.9.	2.	0
Subnet DOMAIN Network C			129.9.	3.	0
Link from DOMAIN Network A to Network B			192.9.1.		0
Link from DOMAIN Network B to Network C			192.9.2.		0
Gateway Net A to Link	//NYC	NYC	129.9. 192.9.1.	1.	1 1
Gateway NYCLNK to Network B		NYCLNK	129.9. 192.9.1.	2.	1 2
Gateway Net B to Link	//LONDON	LONDON	129.9. 192.9.2.	2.	15 1
Gateway LONDONLNK to Network C		LONDONLNK	129.9. 192.9.2.	3.	1 2
Gateway Net C to ETHERNET	//BERLIN Ask foreign net?	BERLIN BERLIN	129.9. 149.8.	3. 5.23*	23
Host (DOMAIN)	//SEATTLE	SEATTLE	129.9.	1.	3
Host (DOMAIN)	//CHICAGO	CHICAGO	129.9.	1.	2
Host (DOMAIN) and	//NYC	NYC	129.9. 192.9.1.	1.	1 1
Host (DOMAIN)	//DUBLIN	DUBLIN	129.9.	2.	12
Host (DOMAIN)	//EDINBURGH	EDINBURGH	129.9.	2.	13
Host (DOMAIN) and	//LONDON	LONDON	129.9. 192.9.2	2.	15 1
Host (DOMAIN)	//PARIS	PARIS	129.9.	3.	21
Host (DOMAIN)	//BRUSSELS	BRUSSELS	129.9.	3.	22
Host (DOMAIN) and	//BERLIN	BERLIN	129.9. 149.8.5.23	3.	23
Host (ETHERNET)	Ask foreign net?	MOSCOW	149.8.	5.30*	

* We don't know if the foreign host has implemented subnets.

2.2.3. Creating Internet Addresses for Internets without Subnet Numbers

You can configure TCP/IP for your internet *without* implementing subnets. Table 2–5 lists the Internet addresses that correspond to the internet configuration in Figure 2–4 *without* subnet numbers.

Note that we must specify addresses for DOMAIN internets the same way — the bridge connecting the two physical rings is actually a network itself. We must also assign network numbers to the links. You can think of a bridge between two DOMAIN networks as a special network that consists of only two gateways — each gateway connects the network on which it resides to the physical link.

In Table 2–5, we list the gateways BERLIN, LONDON and NYC also as hosts, so the gateway and host entries are the same for these nodes. We inserted question marks for information that we would need to get from the system administrator of the foreign host.

To ensure that you understand how to assign these addresses, pencil in the addresses from Table 2–5 onto Figure 2–4.

Table 2–5. Internet Addresses for Sample Internet Configuration without Subnets

Type	Node Name	Internet Name	Internet Number: Network	Host
ETHERNET network			196.6.6.	0
DOMAIN Network A			198.8.8.	0
DOMAIN Network B			198.8.6.	0
DOMAIN Network C			198.8.4.	0
Link from DOMAIN Network A to Network B			192.9.1.	0
Link from DOMAIN Network B to Network C			192.9.2.	0
Gateway Net A	//NYC	NYC	198.8.8.	253
to Link			192.9.1.	1
Gateway NYCLNK		NYCLNK	198.8.6.	253
to Network B			192.9.1.	2
Gateway Net B	//LONDON	LONDON	198.8.6.	241
to Link			192.9.2.	1
Gateway LONDONLNK		LONDONLNK	198.8.4.	241
to Network C			192.9.2.	2
Gateway Net C	//BERLIN	BERLIN	198.8.4.	231
to ETHERNET	Ask foreign net?	BERLIN	196.6.6.	231
Host (DOMAIN)	//SEATTLE	SEATTLE	198.8.8.	251
Host (DOMAIN)	//CHICAGO	CHICAGO	198.8.8.	252
Host (DOMAIN)	//NYC	NYC	198.8.8	253
			192.9.1.	1

Table 2–5. Internet Addresses for Sample Internet Configuration without Subnets (Cont.)

Type	Node Name	Internet Name	Internet Number Network	Host
Host (DOMAIN)	//DUBLIN	DUBLIN	198.8.6	244
Host (DOMAIN)	//EDINBURGH	EDINBURGH	198.8.6	242
Host (DOMAIN)	//LONDON	LONDON	198.8.6 192.9.2.	241 1
Host (DOMAIN)	//PARIS	PARIS	198.8.4.	235
Host (DOMAIN)	//BRUSSELS	BRUSSELS	198.8.4.	233
Host (DOMAIN)	//BERLIN	BERLIN	198.8.4. 196.6.6.	231 231
Host (ETHERNET)	Ask foreign net?	MOSCOW	196.6.6.	222

2.3. Procedure 2–1: Selecting Internet Addresses

Now that you know about Internet numbers, you can select them for *each* host and gateway in your TCP/IP configuration. This procedure provides step–by–step instructions to tell you how.

Task 1: Make a List of Host Names

Decide on the host name for each host on the network. Use the node name without the slashes; for example, PARIS. If you have a diskless host that doesn't have a name, choose any mnemonic name you want.

Note that you can't include the slashes in the host name because they're not part of the standard; Internet names must start with an alphabetical character and can include any of the following: A–Z, 0–9, a period (.), minus sign (–). They can have up to 24 characters.

You can assign more than one name to a single host or gateway. These additional names are called **aliases.** You might want to use aliases when a node serves as a gateway as well as a host. Or you might want to identify a host according to the network on which it's located. For example, you can assign the node //BERLIN to have the Internet name BERLIN and the aliases, BERLIN.GATE and BERLIN.NETWORK.

Task 2: Decide on Type A, B, or C Internet Address Format

Decide on the type of Internet address you want, Type A, B, or C. If you have a large number of hosts and a few networks, select Type A or B. If you have many networks and fewer hosts, select Type C. Refer to Section 2.2., "Selecting Internet Addresses" for details.

If you ever plan to use TCP/IP within a DARPA Internet, you'll want to choose an address type that the National Information Center (NIC) is likely to assign. They'll usually assign a Type B address if you plan to implement subnets. Otherwise, they'll provide you with a Type C address.

Task 3: Select a Network Number

Select a network number that will be unique across all interconnected networks. Note that the size of the network number depends on the type of Internet address format you selected. Refer to Section 2.2., "Selecting Internet Addresses" for details.

If you are implementing subnets within a DOMAIN internet, choose a network number to represent the internet as a whole. Individual networks within the internet share the same network number. They'll have a different subnet and host number.

Note that if you plan to use DOMAIN TCP/IP to communicate within a DARPA Internet such as ARPANET, you must apply for a network number from the Network Information Center (NIC). Until you receive your official network number, choose your own temporary number to set up your network.

You should apply to NIC for a network number if you *ever* intend to attach your network to the DARPA Internet, even if you do not initially intend to do so. This way, you won't have to change your host addresses when you start using the DARPA Internet.

Task 4: Select Subnet Numbers

If you are implementing subnets within a DOMAIN internet, select a unique subnet number for *each* network within the internet.

Task 5: Assign Internet Addresses for Each Gateway

Assign an Internet address for each gateway in the network. For TCP/IP purposes, a gateway connects two networks within a DOMAIN internet (using a routing server), or two different networks (using DOMAIN gateway hardware products).

Note that you must assign *two* Internet addresses to the gateway node since it belongs to two different networks. For example, the node BERLIN is on the DOMAIN network (network number 129.9) and the ETHERNET network (network number 149.8). The two Internet addresses can be 129.9.5.023 (DOMAIN Internet address) and 149.8.5.023 (ETHERNET address).

Task 6: Assign Internet Addresses for Each Host

Assign an Internet address for each host on the network. Each host must have the same network number, but a different host number. Note that if you want a node to be both a gateway and a host, you must list the node as a host entry as well as a gateway entry.

Before you go on, record the Internet address you have selected for each host and for the gateway. You'll use these addresses in subsequent chapters. When you record the Internet addresses, remember that they are in *decimal*, not *hexadecimal*.

END OF PROCEDURE 2-1.

Chapter 3

Editing TCP/IP Files

Once you've selected Internet addresses, you must determine which TCP/IP files you must edit to establish connections between hosts across connected networks. This chapter describes these files.

3.1. Overview of TCP/IP Files

During the configuration process, you provide the information about a host or gateway's names, addresses, physical interfaces by editing several files. However, before you edit the files, you should define the contents of each file. This section describes the files, and describes how to edit the files.

Note that DOMAIN TCP/IP and DOMAIN/IX BSD4.2 TCP/IP require similar files but in different formats. So, the files that you edit are different depending on whether you are running AEGIS or DOMAIN/IX or both operating systems. Table 3–1 lists the files for DOMAIN TCP/IP. Table 3–2 lists the files required for DOMAIN/IX BSD4.2 TCP/IP. While this chapter describes the TCP/IP information files in detail, Chapter 5, "Configuring TCP/IP" provides step–by–step procedures for editing them.

3.1.1. Administrative Nodes

Before we can describe TCP/IP information files in detail, we must introduce an additional concept that is *not* part of standard TCP/IP. It reflects the distributed nature of the DOMAIN system. The TCP/IP protocol provides for communication between gateways and hosts. (A TCP/IP host node has the appropriate TCP/IP services needed for users to communicate via TCP/IP. A TCP/IP gateway node has the appropriate TCP/IP information needed to route messages between hosts on different networks.)

Our TCP/IP products define another concept, the **TCP/IP administrative node**, which maintains configuration information. Defined broadly, an administrative node is any node that provides consistent information for multiple users.

The TCP/IP administrative node contains files of TCP/IP names and Internet addresses. A TCP/IP administrative node does not necessarily have to use TCP/IP communications itself; it can provide this information to hosts and gateways through file links. Also, a TCP/IP administrative node can be a host and a gateway.

You can have a single TCP/IP administrative node on a network, or you can have multiple administrative nodes. However, you must make sure that all administrative nodes on the network always have identical information.

The advantage of having a single administrative node is that you need to maintain only one database. The advantage of having more administrative nodes is to provide an alternate database in case the main node crashes, or to provide additional databases in a large network.

If you're using multiple administrative nodes, note that a host can be linked to only one administrative node, so you would have to change the links manually to change the host's administrative node. Also, you must update *each* TCP/IP administrative node whenever you change your network configuration.

DOMAIN/IX installations normally have an administrative node, which contains the */etc* directory. We refer to this node as the **DOMAIN/IX administrative node.** For networks on which nodes are running DOMAIN/IX, you must have additional DOMAIN/IX BSD4.2 TCP/IP files. For simplicity, you can have one node be both the DOMAIN/IX and DOMAIN/IX BSD4.2 TCP/IP administrative node. However, this isn't required.

Table 3–1 lists the TCP/IP files that you must edit when configuring DOMAIN TCP/IP. Table 3–2 lists the TCP/IP files required for DOMAIN/IX BSD4.2 TCP/IP. Note that all the BSD4.2 files must be stored on the DOMAIN/IX TCP/IP administrative node.

Table 3–1. DOMAIN TCP/IP Information Files

File	Location	Description
/sys/node_data[.nodeid]*/thishost*	All hosts	Lists the Internet name of the local host.
/sys/node_data[.nodeid]*/networks*	All hosts	Lists the local host's Internet addresses and their physical interfaces.
/sys/tcp/hostmap/local.txt	TCP/IP Administrative	Contains information on locally defined Internet addresses and names.
/sys/tcp/hostmap/hosts.txt	TCP/IP Administrative	Contains information on Internet addresses for networks belonging to a DARPA Internet and administered by the NIC.
/sys/tcp/host_addr	All hosts not supporting ARP	Associates Internet and local addresses of foreign hosts that don't use the Address Resolution Protocol (ARP).

Table 3–2. DOMAIN/IX BSD4.2 TCP/IP Information Files

File	Description
/etc/hosts.equiv	Contains the names of all hosts that you can access using **rlogin, rsh** and **rcp** without having to specify a password.
/etc/gateways	Contains routing information for remote destinations that do not use the **routed** daemon. Note that this file is *different* from the */sys/tcp/gateways*.
/etc/hosts	Contains the names and Internet addresses of all hosts on the DOMAIN network or internet.
/etc/networks	Contains the Internet network numbers and names of all accessible networks. Note that this file is *different* from the */sys/node_data/networks*. (You must edit both these files when configuring TCP/IP.)

3.1.2. Links and File Locations

Since most of the TCP/IP information files reside on the TCP/IP administrative node (or nodes), each gateway and host accesses the files through links. The TCP/IP installation procedure creates the appropriate links for each node. To check these links for yourself, you can refer to list of the links in Table 3–3.

Note that the installation procedure also creates links from each node's */sys/tcp* directory to files in its *'node_data* directory. For details on the TCP/IP installation procedure, see the Release Notes document supplied with the latest version of TCP/IP. (An on–line version of the TCP/IP Release Notes document is located in the */doc* directory.)

Table 3–3. TCP/IP DOMAIN and DOMAIN/IX Links

Pathname on host	Location	Link to
/sys/tcp/hostmap (directory)	All hosts	//tcp_admin_node/*sys/tcp/hostmap*
/sys/tcp/gateways	All hosts	//tcp_admin_node/*sys/tcp/gateways*
/sys/tcp/hosts.hst	All hosts	//tcp_admin_node/*sys/tcp/hosts.hst*
/sys/tcp/networks	All hosts	*'node_data/networks*
/sys/tcp/thishost	All hosts	*'node_data/thishost*
/etc (directory)	BSD4.2 hosts	//domain_ix_admin_node/*etc*
/etc/rc	BSD4.2 hosts	*'node_data/etc.rc*
/etc/inetd.conf	BSD4.2 hosts	*'node_data/etc.inetd.conf*

3.2. Differences between DOMAIN and DOMAIN/IX TCP/IP

As the previous tables indicate, DOMAIN and DOMAIN/IX BSD4.2 TCP/IP require different files because the two products require information in a different format. However, if you are running DOMAIN/IX BSD4.2 TCP/IP *along with* DOMAIN TCP/IP, the Shell script **/sys/tcp/makehost.sh** takes care of these differences.

The **makehost.sh** script combines the two TCP/IP information files, *local.txt and hosts.txt*, and creates the two files required for DOMAIN TCP/IP, */sys/tcp/hosts.hst* and */sys/tcp/gateways*. In addition, if DOMAIN/IX is running on the node on which **makehost.sh** executes, it creates three additional files required for DOMAIN/IX BSD4.2, */etc/hosts*, */etc/gateways*, and */etc/networks*. Figure 3–1 shows which files **makehost.sh** creates depending on the operating system in use.

Note that if you're running DOMAIN/IX BSD4.2 TCP/IP only, you must edit the three BSD4.2 files yourself because the **makehost.sh** Shell script is not available. For details of these files, see Section 3.4., "DOMAIN/IX BSD4.2 Files" later in this chapter.

DOMAIN TCP/IP uses the files created by **makehost.sh**. The file, */sys/tcp/hosts.hst* provides information required to convert Internet names and addresses for hosts, gateways, and networks. DOMAIN FTP and Telnet use this file. It cannot be read by the DM because it is a hashed table. The */sys/tcp/gateways* file lists gateways between the DOMAIN network and other networks. Each entry consists of the gateway's addresses for *all* networks to which the gateway is connected.

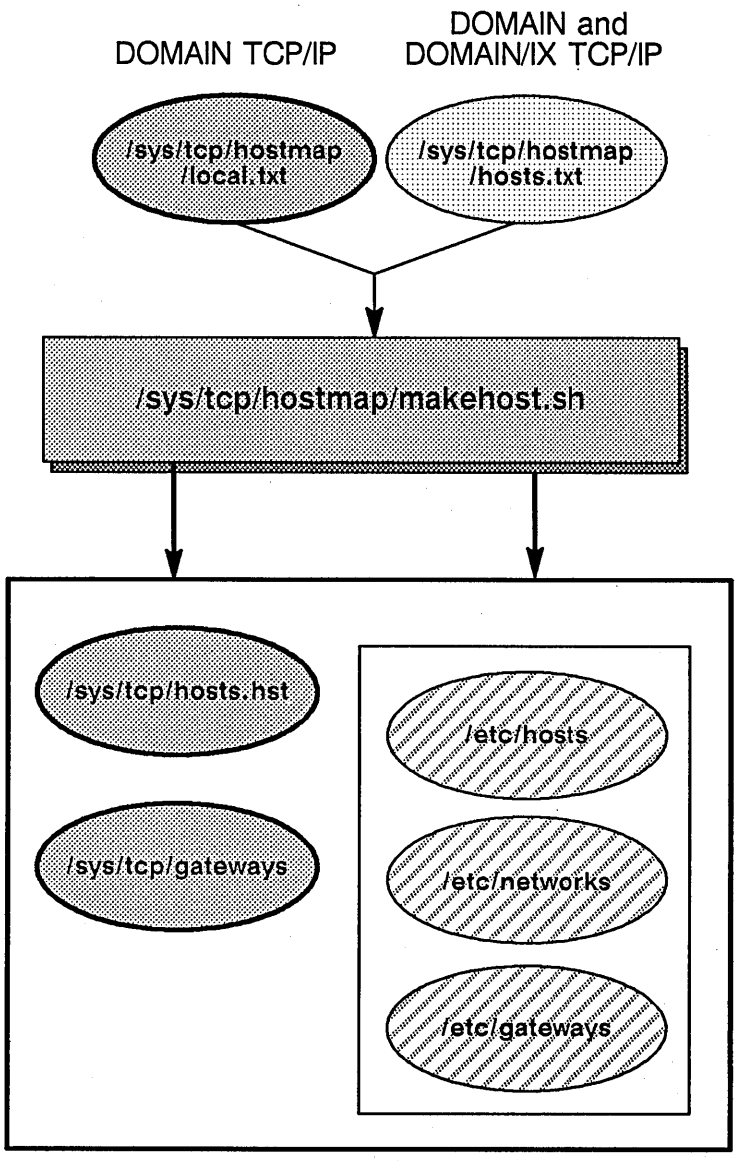

Figure 3–1. makehost.sh and the Mapping Files

3.3. DOMAIN TCP/IP Files

The following sections describe the DOMAIN TCP/IP files that must exist on each TCP/IP host and gateway. They also describe the files located on the DOMAIN TCP/IP administrative node.

3.3.1. /sys/node_data[.node_id]/thishost

The */sys/node_data*[.node_id]*/thishost* file lists the **Internet name** of the local host. You must have this file on each node that serves as a TCP/IP host and/or gateway. The file consists of the host's Internet name on a single line. For example, the *sys/node_data*[.node_id]*/thishost* file for //PARIS is simply:

```
paris
```

3.3.2. /sys/node_data[.node_id]/networks

The */sys/node_data*[.node_id]*/networks* file defines the **Internet addresses** and the **physical interface names** of the local host. The physical interface specifies the physical medium that supports DOMAIN TCP/IP and translates protocols between networks using different protocols. Each host must have a networks file that contains the physical interface information. Also each gateway and routing server must list all the physical interfaces they are connected to.

To specify this information in the networks file, you list all the physical interfaces on which the node is connected in the following format:

internet_address on physical_interface_symbol

Currently, you can specify the following physical interfaces:

Symbol	Physical Interface
dr0	DOMAIN network interface for hosts on a single DOMAIN network.
dr1	DOMAIN routing service interface for the internetwork link that connects two DOMAIN networks in an internet.
eth0[1, 2, 3]	ETHERNET network interfaces for gateways that link a DOMAIN network to an ETHERNET LAN.

Gateways between two networks have two physical interfaces. So, the gateway BERLIN between a DOMAIN network and an ETHERNET LAN has the following networks file:

```
129.9.3.023 on dr0
149.8.5.023 on eth0
```

Gateways between two DOMAIN networks also have two physical interfaces: one for the network (dr0) and one for the internetwork link (dr1). So, the gateway between a DOMAIN network and a T1 link has the following networks file:

```
197.9.8.11 on dr0  ; DOMAIN ring
197.9.12.3 on dr1  ; T1 link
```

The networks file also provides information about subnets. If you implement a TCP/IP configuration with subnets, you supply a bit mask or *subnet mask* in this file to indicate how the gateway should interpret the Internet address.

This mask identifies which bits of the Internet address correspond to a subnet number, and which bits correspond to the host number.

To supply the mask, you must edit the */sys/node_data*[.node_id]*/networks* file to add the following information to *each* physical interface line.

internet_address on physical_interface [; subnet mask W.X.Y.Z [; comment]]

where **W.X.Y.Z** fields can contain either a one (255) to denote the network or subnet field, or a zero (0) to denote the host field.

For example, the following is a network entry without a subnet mask. It indicates that you have a Type A internet address on an ETHERNET physical interface. We know this is a Type A address because the first number is within the range of 1 and 126.

```
10.9.9.7. on eth0
```

The following is a network entry with a subnet mask. Given that this is a Type A address, we know that the first field is the network number. The next field is the subnet number because it is all one's, and the host number corresponds to the last two bytes, as indicated by the zeros.

```
10.9.9.7  on eth0 ;  mask 255.255.0.0
```

The following is a two-byte subnet mask for a Type A address:

```
10.9.9.7  on eth0 ;  mask 255.255.255.0
```

The following is a one-byte subnet mask for a Type B address where the first two bytes indicate the network number, the third byte is the subnet number, and the fourth byte is the host number.

```
129.9.9.9 on eth0 ;  mask 255.255.255.0
```

For a more complete example of physical network files we can return to our network configuration example from Chapter 2, "Selecting Internet Addresses." Table 3-4 lists the contents of the networks file for *each* of the gateways and hosts shown in Figure 2-4. Note that you'll have a entry in the networks file for each network that your're connected to; so, in this case, the networks files for the gateway nodes have two entries.

Table 3-4. Network File Entries for Sample Subnet Configuration

Type		Internet Name	Networks File Entry or Entries
Gateway between Net A and link		NYCA	129.9.1. 1 on dr0; 255.255.255.0 192.9.1.1 on dr1
Gateway NYCLNK to Network B		NYCLNK NYCB	192.9.1.2 on dr1 129.9.2.1 on dr0; 255.255.255.0
Gateway Net B to Link	//LONDON	LONDON LONDONLNK	129.9.2.15 on dr0; 255.255.255.0 192.9.2.1 on dr1
Gateway LONDONLNK to Network C		LONDONLNK LONDONC	192.9.2.2 on dr1 129.9.3.1 on dr0; 255.255.255.0
Gateway Net C to ETHERNET	//BERLIN Ask foreign net?	BERLIN BERLIN	129.9.3.23 on dr0; 255.255.255.0 149.8. 5.23 on eth0
Host (DOMAIN)	//SEATTLE	SEATTLE	129.9.1.3 on dr0; 255.255.255.0
Host (DOMAIN)	//CHICAGO	CHICAGO	129.9.1.2 on dr0; 255.255.255.0
Host (DOMAIN) and	//NYC	NYC	129.9.1.1 on dr0; 255.255.255.0 192.9.1.1 on dr1
Host (DOMAIN)	//DUBLIN	DUBLIN	129.9.2.12 on dr0; 255.255.255.0
Host (DOMAIN)	//EDINBURGH	EDINBURGH	129.9.2.13 on dr0; 255.255.255.0
Host (DOMAIN) and	//LONDON	LONDON	129.9.2.15 on dr0; 255.255.255.0 192.9.21 on dr1
Host (DOMAIN)	//PARIS	PARIS	129.9.3.21 on dr0; 255.255.255.0
Host (DOMAIN)	//BRUSSELS	BRUSSELS	129.9.3.22 on dr0; 255.255.255.0
Host (DOMAIN) and	//BERLIN	BERLIN	129.9.3. 23 on dr0; 255.255.255.0 149.8.5.23 on eth0
Host (ETHERNET)	Ask foreign net?	MOSCOW	149.8. 5.30 on eth0

3.3.3. /sys/tcp/hostmap/hosts.txt

The */sys/tcp/hostmap/hosts.txt* file is the Department of Defense (DoD) Internet Host Table from the Network Information Center (NIC). This file contains the names and addresses of all the hosts on the ARPANET, as well as many other networks that conform to the DoD DARPA Internet standard. You must have a copy of this file on your TCP/IP administrative node if you plan to communicate over ARPANET or any other network listed by the NIC.

If do *not* plan to connect your DOMAIN network to the DoD Internet, you should replace */sys/tcp/hostmap/hosts.txt* with an empty file. If you want to save the contents of the file, you can rename it. By creating an empty file, TCP/IP takes less time to translate between host names and Internet addresses. It also improves the performance of **makehost.sh**, the Shell script that you use to configure TCP/IP.

We include a copy of */sys/tcp/hostmap/hosts.txt* with the DOMAIN TCP/IP software. However, the copy we send may not be the most current copy available. We suggest that you use the copy we ship you initially to create your host table, and then copy a new version from NIC. You should copy this file periodically to keep this file current. To copy this file, refer to Section 6.5, "Getting the Official hosts.txt File from the NIC."

3.3.4. /sys/tcp/hostmap/local.txt

The */sys/tcp/hostmap/local.txt* file contains network information for your own network numbers and addresses that aren't part of a DARPA Internet.

If you use a DARPA network, you can use *local.txt* to temporarily list new computers and networks that are not yet part of the NIC-supplied version of *hosts.txt*. You can have identical entries in both the *hosts.txt* and *local.txt* files, but it's less efficient.

> NOTE: Both the *local.txt* and the *hosts.txt* files must follow the format defined by RFC 810, "DoD Internet Host Table Specification."

The */sys/tcp/local.txt* file has three categories of entries called **NET**, **GATEWAY**, and **HOST**:

- The **NET** entries define the networks that you can access. You list the network number and a name for each network that you can access with your TCP/IP network configuration.
- The **GATEWAY** entries specify the gateways between the networks that you can access. You list both Internet addresses of each gateway node, its name, and information about the protocols it uses.
- The **HOST** entries specify the TCP/IP hosts that you can access. You list each host, its name, and information about the TCP/IP applications it's running. You have a HOST entry for each host that you want to access, including remote hosts and all gateways. You should also have an entry for the **tcp_server** software loopback interface, which has an address of **127.0.0.1**; by convention it has the name **localhost.**

Each entry begins with a keyword, **NET**, **GATEWAY**, or **HOST** followed by pertinent address information. Note that the order in which you list the entries is important. That is,

1. List all the **NET** entries first.
2. List all the **GATEWAY** entries in the order in which you want TCP/IP to use them when establishing connections. If you are connecting the DOMAIN network to an Internet that consists of more than one foreign host, TCP/IP tries to use the first gateway on the list that has an address on the destination network. If that gateway fails to make the connection, TCP/IP tries to find another gateway to the network. If this fails, it tries, in turn, each gateway on the list that is marked as a PRIME gateway.
3. List all the **HOST** entries last.

To list the names and addresses in *local.txt*, edit the file that resides on the TCP/IP administrative node. Use the information you gathered from selecting the Internet addresses in Chapter 2, "Selecting Internet Addresses." Figure 3-2 shows a sample *local.txt* file using the Internet addresses from Table 3-5. Note that colons (;) begin comments.

```
; NET : NET-ADDR : NETNAME :
; GATEWAY : ADDR, ADDR : NAME : CPUTYPE : OPSYS : PROTOCOLS:
; HOST : ADDR, ALTERNATE-ADDR (if any): HOSTNAME,NICKNAME:
;   CPUTYPE : OPSYS : PROTOCOLS :
;
NET : 149.8.0.0 : FOREIGN-ETHER :
NET : 129.9.0.0  : DOMAIN-RING :
;
GATEWAY : 129.9.1.1, 192.9.1.1 : NYC, NETA-GATE: DN460 :
   AEGIS : IP/GW, GW/PRIME :
GATEWAY : 129.9.2.1, 192.9.1.2 : NYCLINK, NETA-GATE: DN460 :
   AEGIS : IP/GW, GW/PRIME :
GATEWAY : 129.9.2.15, 192.9.2.1 : LONDON, NETB-GATE: DN550 :
   AEGIS/DOMAIN/IX : IP/GW, GW/PRIME :
GATEWAY : 129.9.3.1, 192.9.2.2 : LONDONLNK, NETB-GATE: DN550 :
   AEGIS/DOMAIN/IX : IP/GW, GW/PRIME :
GATEWAY : 129.9.3.23, 149.8.5.23 : BERLIN, FOREIGN-GATE: DSP90 :
   AEGIS/DOMAIN/IX : IP/GW, GW/PRIME :
; The tcp_server software loopback interface;
HOST : 127.0.0.1 : LOCALHOST : : :
;
; The following are hosts on the DOMAIN network.
HOST : 129.9.1.3 : SEATTLE: DN550 : AEGIS/DOMAIN/IX : TCP/TELNET, TCP/FTP:
HOST : 129.9.1.2 : CHICAGO: DN460 : AEGIS : TCP/TELNET,TCP/FTP :
HOST : 129.9.1.1, 192.9.1.1 : NYC : DN460 : AEGIS : TCP/TELNET, TCP/FTP :
HOST : 129.9.2.1, 192.9.1.2 : NYCLINK : DN460 : AEGIS : TCP/TELNET, TCP/FTP :
HOST : 129.9.2.13 : EDINBURGH : DN3000 : AEGIS : TCP/TELNET,TCP/FTP :
HOST : 129.9.2.12 : DUBLIN : DN3000 : AEGIS/DOMAIN/IX : TCP/TELNET,TCP/FTP :
HOST : 129.9.2.15, 192.9.2.1 : LONDON : DN550 : AEGIS/DOMAIN/IX : TCP/TELNET, TCP/FTP :
HOST : 129.9.3.22 : BRUSSELS : DN460 : AEGIS/DOMAIN/IX : TCP/TELNET,TCP/FTP :
HOST : 129.9.3.21 : PARIS : DN3000 : AEGIS/DOMAIN/IX : TCP/TELNET,TCP/FTP :
HOST : 129.9.3.23, 149.8.5.23 : BERLIN : DN3000 : AEGIS/DOMAIN/IX : TCP/TELNET,TCP/FTP :
;
; The following is a remote host on the ETHERNET
HOST : 149.8.5.30 : MOSCOW: VAX/11-750 : VMS : TCP/TELNET, TCP/FTP :
```

Figure 3–2. A Sample /sys/tcp/hostmap/local.txt File

We describe the format of each entry below. Table 3–5 lists the punctuation you can use within these entries. Note that entries are not case sensitive, so you can specify the entries in either uppercase or lowercase.

Table 3–5. Punctuation Meaning for local.txt File

Punctuation	Symbol	Rule
Colon	:	Terminates each field.
Double colons	::	Indicates a null field.
Comma	,	Separates values within a field.
Semicolon	;	Begins a comment; it's not part of the table.

Each **NET** entry in the *local.txt* file has the following format:

NET : net-addr : netname :

Where:

net-addr — is the network's Internet network number, followed by one or more zeros to make a full Internet address; for example, 129.9.0.0 or 21.0.0.0.

netname — is the name of the network; for example, BERLIN-ETHER. The name may have up to 24 characters and must start with an alphabetic character. Valid name characters are A–Z, 0–9, period (.), and minus sign (–).

Each **GATEWAY** entry in the *local.txt* file has the following format:

GATEWAY : addr1,addr2 : name : [cputype] : [opsys] : [protocols] :

Where:

addr1 — is the address of the gateway on one of the networks that it connects; for example, 129.08.5.23

addr2 — is the address of the gateway on the other network that it connects; for example, 149.9.3.15

name — is the Internet name of the gateway; for example, BERLIN. The name may have up to 24 characters; it must start with an alphabetic character. Valid characters are A–Z, 0–9, period (.), and minus sign (–).

cputype — describes the gateway processor. This value is optional; it provides you with information for troubleshooting, but TCP/IP does not use it. To ensure consistency, use workstation model numbers for DOMAIN nodes, for example DN3000.

opsys — describes the gateway processor's operating system. This value is optional; it provides you with information for troubleshooting, but TCP/IP does not use it. To ensure consistency, use **AEGIS** for nodes that run AEGIS, **DOMAIN/IX** for nodes that run DOMAIN/IX and **AEGIS/DOMAIN/IX** for nodes that run both.

protocols — describes the Internet protocols that the gateway supports. For all DOMAIN gateways and routing servers, specify *both* of the following, separated by a comma:

IP/GW Internet gateway

GW/PRIME Prime routing, gateway

Specify **IP/GW, GW/DUMB** for non-DOMAIN gateways that do not maintain current routing information. See RFC 810 for a list of gateway protocols.

Each **HOST** entry in the *local.txt* file has the following format:

HOST : addr [,alt-addr] : name [,alias...] : [cputype] : [opsys] : [protocols] :

Where:

addr — is the Internet address of the host; for example, 129.9.3.021

alt-addr — is one or more alternate Internet addresses for the host, separated by commas; for example a gateway would have two addresses, 129.9.1.1, 129.9. 2.1

name — is the host's Internet name; for example, PARIS. The name may have up to 24 characters and must start with an alphabetic character. Valid name characters are A–Z, 0–9, period (.), and minus sign (–).

alias — is one or more alternate names that you can use to access the host. All names must be separated by commas; for example, DOMAIN–HOST, NETC–HOST.

cputype — describes the host processor. This value is optional; it provides you with information for troubleshooting, but TCP/IP does not use it. To ensure consistency use workstation model numbers for DOMAIN nodes, for example DN3000.

opsys — describes the host operating system. This value is optional; it provides you with information for troubleshooting, but TCP/IP does not use it. To ensure consistency, use **AEGIS** for nodes that run AEGIS, **DOMAIN/IX** for nodes that run DOMAIN/IX and **AEGIS/DOMAIN/IX** for nodes that run both.

protocols — describes the Internet protocols that the host or gateway supports. This value is optional; it provides you with information for troubleshooting, but TCP/IP does not use it. For DOMAIN hosts, including gateways, specify both of the following, separated by commas. If your node or a remote host supports other protocols, specify them as defined in RFC 810.

TCP/FTP FTP file transfer protocol

TCP/TELNET Telnet terminal emulator protocol

3.3.5. /sys/tcp/host_addr

Most TCP/IP hosts, including all DOMAIN nodes, support the Address Resolution Protocol (ARP). However, some foreign hosts might not support the protocol. If any node is connected by some network to a host that does *not* support ARP, the node's */sys/tcp/host_addr* file must list the address of the non-ARP hosts. Each non-ARP host must be listed in the following format:

internet_address , local_address

Currently, you may run across a non-ARP host on an ETHERNET LAN. In this case, you specify its Internet address, and the machine's local address, separated by a comma. The ETHERNET local address is a hexadecimal number. Specify one entry on each line of the file. For example, enter:

```
192.9.9.5   ,   2.7.1.0.3.a4
192.9.9.6   ,   9.4.c3.5.6.8
```

Normally, you'd store this file on the gateway to the foreign host that does not support ARP. When you configure the gateway, you must use the **maphost** command to put the information from the */sys/tcp/host_addr* file in the gateway's address mapping tables. Section 6.5.2, "Address Mapping Files" describes ARP and routing in greater detail, and Appendix B, "TCP/IP Reference" includes a reference description of **maphost**.

3.4. DOMAIN/IX BSD4.2 TCP/IP Files

If you use DOMAIN/IX BSD4.2 TCP/IP on a network that runs DOMAIN/IX, you must edit additional files. These are the files that DOMAIN/IX BSD4.2 uses for TCP/IP communication:

- */etc/hosts.equiv*
- */etc/gateways*
- */etc/networks*
- */etc/hosts*

The following sections describe these files in detail.

3.4.1. /etc/hosts.equiv

The */etc/hosts.equiv* file lists hosts that are equivalent to your host for log-in purposes. That is, if a host is listed in your node's */etc/hosts.equiv* file, that host can execute any of the following programs or functions on your node:

- lpr(1)
- lprm(1)
- lpq(1)
- rcmd(3X)
- rcp(1)
- rlogin(1)
- rsh(1)

NOTE: All nodes that use **lpr**, including the node that runs the line printer daemon **lpd**, must be configured for TCP/IP communications. The names of all nodes that will print files using **lpr** must be in the **lpd** node's */etc/hosts.equiv* file.

The */etc/hosts.equiv* file contains the name of each equivalent TCP/IP host, one name per line. For example:

```
paris
brussels
berlin
nyc
seattle
```

3.4.2. /etc/gateways

The */etc/gateways* file contains static routing information that the **routed**(8) daemon uses to manage network routing tables. Use this on BSD4.2 gateways to include information about gateways on your DARPA Internet that do *not* support Routing Information Protocol (RIP) protocol. This file associates a destination network with the next gateway in the route to the destination. The **tcp_server** will then send messages for all hosts on that network to the required gateway.

You should not need to edit this file if your network runs DOMAIN TCP/IP on a gateway because DOMAIN TCP/IP's **makehost.sh** Shell script automatically generates the */etc/gateways* file from information in the */sys/tcplocal.txt* file.

Each entry in the file is a single line in the following format:

dest-type name1 gateway name2 metric hops gate-type

Where:

dest-type	Type of routing destination; this must always be **network**.
name1	Name or Internet address of the destination, or a symbolic name located in */etc/networks*.
gateway	The gateway to which the packets should be addressed.
name2	Name or Internet address of the next gateway in the route to the destination (the next hop). The name2 gateway must be on the same network as the gateway that uses this file.
metric	An optional count indicating the number of hops to the destination. If no metric is specified, the value is 0.
hops	Number of gateways between the current gateway and the destination (the hop count).
gate-type	Indicates whether the name2 gateway uses **routed**. active specifies that the gateway is running **routed** to exchange routing information. passive indicates a gateway that does not use **routed**, and is not exchanging routing information.

In the following example, this */etc/gateways* file provides routing information about two networks. The first is connected to one of this gateway's networks, the second is another gateway away. The next gateway in the route to both of these networks (that is the gateway to the network next–net) does not use **routed**.

```
network next-net gateway gate1 metric 1 passive
network third-net gateway gate1 metric 2 passive
```

3.4.3. /etc/networks

The */etc/networks* file contains the names and addresses of the networks that you can access. Each entry is a single line of the format:

network–name **network–number**

Where network–name and network–number are the Internet network name and number of the network, and the two values are separated by one or more blanks or TAB characters.

You do not have to create these files on DOMAIN networks or internets running DOMAIN TCP/IP on the gateway because the **makehost.sh** Shell script automatically generates the */etc/networks* file from information in the */sys/tcp/local.txt* file. You must create and edit the file only if you use BSD4.2 TCP/IP on a single DOMAIN network or internet. You must have these files if you use **lpr**(1), **rcmd**(3X), **rcp**(1), **rlogin**(1), **rsh**(1), and **rexec**(3X), as well as **ftp** and **telnet**.

In a DOMAIN network or internet where there is no gateway hardware to communicate with foreign networks, this file consists of a single line, and the network name must be **domain–ring**. For example:

```
domain-ring  129.9.0.0
```

3.4.4. /etc/hosts

The */etc/hosts* file contains the name and Internet address of each TCP/IP host you can access. Each line has the following format:

Internet–address **host–name**

The address and name must be separated by one or more blanks or TAB characters.

For example:

```
127.0.0.1          localhost
197.9.8.1          timeix
197.9.8.3          unclex
196.6.3.8          berkowix
197.6.3.15         felix
```

You must edit this file if you're running DOMAIN/IX BSD4.2 TCP/IP on a DOMAIN network or internet and you want to run the following: **lpr**(1), **rcmd**(3X), **rcp**(1), **rlogin**(1), **rsh**(1), **rexec**(3X), **ftp** or **telnet**. You don't need to edit this file if you're running DOMAIN TCP/IP on a gateway because DOMAIN/TCP's **makehost.sh** script automatically generates */etc/hosts*.

Note that you must include the **localhost 127.0.0.1** entry so you can access the software loopback interface on each host by using the **localhost** host name.

3.5. Procedure 3–1: Deciding Where to Store TCP/IP Files

Once you have selected the Internet addresses (as described in Chapter 2, "Selecting Internet Addresses"), you can determine which nodes will store the network–wide mapping files that TCP/IP needs. To do so, follow these steps:

1. Decide which node (or nodes) will serve as the TCP/IP administrative node and store the */sys/tcp/hostmap* directory.

2. If nodes on the network are running BSD4.2, determine which node is the DOMAIN/IX administrative node (or nodes) that contain the */etc* directory.

3. If you are configuring TCP/IP on a network that uses TCP/IP over a gateway or routing server, define the contents of the //tcp_admin_node*/sys/tcp/hostmap/local.txt* file.

   If you are configuring BSD4.2 TCP/IP on a network that does *not* have a gateway or routing server, define the contents of the */etc/networks* and */etc/hosts* files.

The configuration procedures in Chapter 5, "Configuring TCP/IP" provide step–by–step instructions for defining the contents of these files.

Chapter 4

Starting TCP/IP Servers and Daemons

DOMAIN TCP/IP and DOMAIN/IX BSD4.2 TCP/IP require the support of several processes. These processes respond to requests for some form of service. They are generally called **servers**. Processes in the UNIX environment are called **daemons**. Tables 4–1 and 4–2 list the processes, describe their purposes, and indicate the nodes on which they must be running. Table 4–1 lists the servers for nodes running the AEGIS operating system. Table 4–2 lists the servers and daemons for nodes running DOMAIN/IX BSD4.2. The following sections describe the servers and daemons in more detail.

Table 4–1. TCP/IP Server Processes for AEGIS Nodes

Server	Location	Description
tcp_server	All AEGIS and DOMAIN/IX host and gateway nodes	Enables TCP/IP communications on the node.
rip_server	AEGIS gateway nodes	Manages the gateway routing tables.
ftp_server	Host nodes running AEGIS that accept FTP connections	Enables direct FTP access to the host.
telnet_server	Host nodes running AEGIS accept inbound Telnet	Enables inbound Telnet access to the host.

Table 4-2. TCP/IP Server and Daemon Processes for DOMAIN/IX Nodes

Server	Location	Description
tcp_server	All AEGIS and DOMAIN/IX nodes	Enables TCP/IP communications on the node.
routed	DOMAIN/IX gateway	Manages gateway network routing tables.
rwhod	DOMAIN/IX gateway	Maintains database used by **rwho** and **ruptime** to provide system status.
sendmail	DOMAIN/IX gateway	Handles mail received over the Internet.
tftpd	Hosts that accept **tftp** connections	Enables TFTP access to the host.
inetd	All DOMAIN/IX BSD4.2 nodes	Starts the daemons listed in the file as needed.
ftpd	Hosts that accept FTP connections	Enables direct FTP access to the host.
telnetd	Hosts that accept inbound **telnet**	Enables inbound Telnet to the host.
rexecd	Hosts that accept **rexec** routine	Enables remote execution of commands on this node.
rlogind	Hosts that accept **rlogin** program	Enables remote login to this node.
rshd	Hosts that accept **rsh** program	Enables remote execution of commands on this node with user authentication.

4.1. Running the tcp_server on All Nodes

The **tcp_server** must run on *every* node that uses TCP/IP, the nodes can be running either the AEGIS or DOMAIN/IX operating system. The **tcp_server** process ensures that all TCP/IP data is transmitted reliably between end-user processes such as FTP and Telnet. It also performs routing services and maintains mapping tables that relate Internet addresses to local addresses. The **tcp_server** maintains two internal routing and mapping tables, a gateways table, and an Internet address to local address mapping table. Section 6.5., "Maintaining Internal Tables" describes these tables in detail.

NOTE: The **tcp_server** must have delete access control list (ACL) rights to the */sys/node_data*[.nodeid] directory.

4.2. Running DOMAIN TCP/IP Servers

Use the following DOMAIN TCP/IP servers in networks that *do not* use DOMAIN/IX BSD4.2. The **rip_server** provides routing services on gateway nodes that do not use BSD4.2. The **ftp_server** and **telnet_server** support DOMAIN FTP and Telnet, respectively. If you are running both AEGIS and DOMAIN/IX BSD4.2 TCP/IP, and want to use the DOMAIN/IX BSD4.2 **telnet** and **ftp** utilities, you don't need to run the DOMAIN **telnet_server** or **ftp_server**.

4.2.1. rip_server

You must run the **rip_server** on *each* gateway that runs AEGIS and not DOMAIN/IX BSD4.2, including all routing servers in a DOMAIN internet that use TCP/IP. The **rip_server** uses the Routing Information Protocol (RIP) to maintain the gateway's network routing tables. It performs the same functions on DOMAIN gateways as the **routed** daemon does on DOMAIN/IX BSD4.2 TCP/IP gateways.

The **rip_server** maintains the **routing table**, which is an internal database of destinations and routes from the local gateway. It broadcasts its routing tables every 30 seconds, and deletes table entries that are not refreshed by broadcasts from other hosts and gateways that use RIP. This dynamic operation eliminates the need to update static gateway tables each time the network configuration changes.

If **rip_server** is *not* running on the DOMAIN gateway; the gateway does not transmit packets with routing information to the connected network. Systems that use **rip_server** purge their databases of table entries for gateways that have not sent a routing information packet within three minutes. As a result, remote destinations that use **rip_server** may lose knowledge of any DOMAIN gateway that does not use either **rip_server** or **routed**. For this reason, you must run **rip_server** on each gateway.

Some remote destinations do not support **rip_server** (that is, they are **dumb gateways**) so the remote destination could potentially lose knowledge of the DOMAIN gateway. To prevent this, you can put the DOMAIN gateway entry into the remote gateway's routing table. This is considered a permanent or *static* entry because you must manually edit the routing table files. For details, see Section 6.5.1., "Maintaining the Internal Routing Table."

NOTE: If the gateway runs DOMAIN/IX BSD4.2; use the DOMAIN/IX BSD4.2 equivalent, **routed** instead. You *cannot* run both processes on the same node.

To put entries permanently in the **rip_server** routing table, use the **setroute** command, or list them as gateways in the */sys/tcp/hostmap/local.txt* file and run the **makehost.sh** Shell script.

4.2.2. ftp_server

You should have one or more **ftp_server** processes per DOMAIN network. The **ftp_server** listens for and accepts FTP connections. It then services the FTP request. You can establish an FTP connection only to a node that runs the **ftp_server**. That is, you must specify a node that runs **ftp_server** in the Shell **ftp** command. The **ftp_server** provides access to files on all nodes on the DOMAIN network. You do *not* need the **ftp_server** to issue the **ftp** command.

NOTE: **ftp_server** and **ftpd** (the DOMAIN/IX BSD4.2 equivalent) *cannot* execute on the same node.

4.2.3. telnet_server

Run the **telnet_server** on each node that accepts remote log-in using DOMAIN Telnet. **telnet_server** listens for and accepts Telnet connections. You must run the **telnet_server** on each node that accepts inbound Telnet sessions; that is, on each node that allows you to use Telnet to log in from a remote host. You do *not* need the **telnet_server** to issue the **telnet** command.

NOTE: **telnet_server** and **telnetd** (the DOMAIN/IX BSD4.2 equivalent) *cannot* execute on the same node.

4.3. Running DOMAIN/IX BSD4.2 Daemons

You must run the BSD4.2 daemons to enable various BSD4.2 commands and utilities. The following sections describe these daemons and indicate the nodes on which they must execute. See the *DOMAIN/IX Programmer's Reference for BSD4.2* for detailed descriptions of each daemon.

4.3.1. routed

Run **routed** on each gateway that runs DOMAIN/IX, including all routing servers in DOMAIN internets that use DOMAIN/IX TCP/IP. The **routed**(8) daemon uses the Routing Information Protocol (RIP) to maintain the gateway's network routing tables. It performs the same functions on DOMAIN/IX BSD4.2 gateways as the **rip_server** does on DOMAIN gateways. See Section 4.2.1., "rip_server" for more information. You can also see the *DOMAIN/IX Programmer's Reference for BSD4.2* for more information on **routed**.

NOTE: If the gateway runs DOMAIN and not DOMAIN/IX BSD4.2; use **rip_server** instead of **routed**. You *cannot* run both processes on the same node.

If you're running DOMAIN/IX BSD4.2 TCP/IP on a single DOMAIN network, do not use **routed**.

To put entries permanently in the **routed** table, either enter them in the */etc/gateways* file (described later in this chapter) or use the DOMAIN/IX **route**(8) command.

4.3.2. rwhod

Run **rwhod** on each gateway so it can provide information on both networks that it connects. The **rwhod**(8) daemon is the Internet system status server. It maintains the database of status information that the **rwho**(1) and **ruptime**(1) programs use.

4.3.3. sendmail

The **sendmail**(8) program routes mail messages that you send using BSD4.2 or DARPA mail commands over the Internet. When it runs as a daemon on a gateway, it enables the two networks to send and receive mail messages to and from each other. **sendmail** is *not* a user-level interface.

NOTE: If you use mail between the DOMAIN network and a non-DOMAIN network, **sendmail** must run as a daemon on the gateway between the networks. You can do this by including the **-bd** flag in the **sendmail** command.

4.3.4. tftpd

You should have one or more **tftpd** processes per DOMAIN network if you want to support the TFTP protocol. The **tftpd**(8) daemon is a server that supports the DARPA Trivial File Transfer Protocol (TFTP). It listens for and accepts TFTP requests. You can establish a TFTP request *only* to a node that runs the **tftpd**. That is, you must specify a host that runs **tftpd** in the Shell **tftp**(1) command. However, the **tftpd** daemon provides access to files on all nodes on the DOMAIN network. You do *not* need to run **tftpd** to issue the **tftp** command.

4.3.5. inetd

The **inetd**(8C) daemon is a server-manager (in essence, a daemon daemon) that invokes Internet services such as **ftpd**(8) or **rlogind**(8) as necessary. Since it is a single process, **inetd** can efficiently manage many types of Internet connections.

Note that you must have an **inetd** process on each node that requires any of the following servers and daemons:

- **ftpd**(8)
- **rexecd**(8)
- **rlogind**(8)
- **rshd**(8)
- **telnetd**(8)

The file */etc/inetd.conf*(4) lists the daemons that **inetd** invokes. The DOMAIN/IX BSD4.2 installation procedure automatically creates a template file in */sys/node_data/etc/inetd.conf*. To include only those daemons for the services that the node supports, remove the comment marks (#) at the beginning of the lines that start the required daemons.

4.3.6. ftpd

You should have one or more **ftpd** processes per DOMAIN network. The **ftpd**(8) daemon accepts FTP connections and services FTP requests. You can establish an FTP connection only to a node that can run the **ftpd**. That is, you must specify a node that can run **ftpd** in the Shell **ftp**(1) command or in response to the **ftp** Host: prompt. However, the **ftpd** provides access to files on all nodes on the DOMAIN network. You do *not* need the **ftpd** to issue the FTP command.

NOTE: You must use the **inetd** daemon to invoke **ftpd**.

ftpd and **ftp_server** *cannot* execute on the same node.

4.3.7. rexecd

Run **rexecd** on each node that supports invocation of commands from a remote host that uses **rexec**. The **rexecd**(8) daemon services requests from the **rexec**(3X) library function. It allows you to execute UNIX commands remotely on the server node. **rexecd** must receive a valid user ID and password from **rexec**.

NOTE: You must use the **inetd** daemon to invoke **rexecd**.

4.3.8. rlogind

Run **rlogind** on each node that supports log-in from a remote host using **rlogin**. The **rlogind**(8) daemon services requests from the **rlogin**(1) program. It allows you to log in remotely on the server node. **rlogind** requires pseudo-ttys. These pseudo-ttys are normally created when you install DOMAIN/IX, but can be created by the **/etc/crpty**(8) program. **rlogind** does not request a password if the remote host is listed in the daemon node's */etc/hosts.equiv* file.

NOTE: You must use the **inetd** daemon to invoke **rlogind**.

4.3.9. rshd

Run **rshd** on each node that supports invocation of commands from remote hosts using **rsh** or **rcmd**. The **rshd**(8) daemon is the remote shell server. It services requests from the **rsh**(1) program and **rcmd**(3X) library function. It allows you to execute UNIX commands remotely on the server node. **rshd** does not request a password, but the remote host must be listed in the server's */etc/hosts.equiv* file.

NOTE: You must use the **inetd** daemon to invoke **rshd**.

4.3.10. telnetd

The **telnetd**(8) daemon accepts Telnet connections. You must run the **telnetd** daemon on each node that accepts inbound Telnet sessions. You do *not* need the **telnetd** daemon to issue the **telnet**(1) command.

NOTE: **telnetd** must execute on each node that accepts remote log-in using Telnet.

You must use the **inetd** daemon to invoke **telnetd**.

telnetd and **telnet_server** cannot execute on the same node.

4.4. Starting Server and Daemon Processes

To enable TCP/IP, you must make sure that the TCP/IP servers are always running. To do this, you must include commands to start the appropriate processes in the node start up files. This way, whenever you reboot your node, you can be sure that all the appropriate TCP/IP processes are running.

The name and location of the node startup file depends on the type of node that you are using. If you are configuring a disked node, use the */sys/node_data/startup*[.node_type] file. If you are configuring a diskless node, use the //partner_node/*sys/node_data*.nodeid/*startup*[.node_type] file. The .node_type indicates the model node you are using; and .nodeid is the node ID number. For details on the startup files, see the *Administering Your DOMAIN System* manual.

Which processes you start in this node startup file depend on which processes you want the node to run, and whether you are running TCP/IP on an AEGIS or DOMAIN/IX BSD4.2 node.

For nodes running the AEGIS operating system, you can start all the servers that you want the node to run by editing the node's start-up file. You include the commands to start each server. However, you start daemons on nodes running DOMAIN/IX differently. Rather than starting *all* the daemons in the node startup file on DOMAIN/IX nodes, you start the **/sys/tcp/tcp_server** and **/etc/run_rc** processes. Then **/etc/run_rc** starts all the processes specified in the */etc/rc* file.

Note that since most DOMAIN/IX installations have a single */etc* directory on the DOMAIN/IX administrative node, all other nodes gain access to this directory through links. However, each node must have the *rc* and *inetd.conf* files in its node startup directory. DOMAIN/IX BSD4.2 TCP/IP should set these links up automatically. However, you can check to make sure the following links are set up:

The Link	**Must resolve to this file:**
/etc/rc	*'node_data/etc.rc*
/etc/inetd.conf	*'node_data/etc.inetd.conf*

Procedures in Chapter 5, "Configuring TCP/IP" and Chapter 6, "Managing TCP/IP" provide step-by-step instructions for starting servers and daemons.

4.5. Procedure 4–1: Determining Server Processes

Before you begin configuring a network that uses TCP/IP, you should determine which nodes will run the servers and daemons. Refer to your drawing of the network when deciding which nodes require which servers or daemons.

Table 4–3 lists the servers required for each node in our sample network configuration as shown in Chapter 2, "Selecting Internet Addresses," Figure 2–4. Note that DOMAIN Networks A and C are running the AEGIS operating system. While DOMAIN Network B is running DOMAIN/IX. Table 4–3 lists the servers running on each node of our sample TCP/IP network configuration.

Table 4–3. TCP/IP Server Processes Running on DOMAIN Internet

Node	Node Type	Servers Running
SEATTLE	DOMAIN host	**tcp_server, telnet_server**
CHICAGO	DOMAIN host	**tcp_server**
NYC	DOMAIN host/gateway	**tcp_server, rip_server, telnet_server, ftp_server**
NYCLINK	DOMAIN host/gateway	**tcp_server, rip_server, telnet_server, ftp_server**
EDINBURGH	DOMAIN/IX host	**tcp_server, inetd, ftpd, telnetd, recec, rlogind, rshd**
DUBLIN	DOMAIN/IX host	**tcp_server, inetd, ftpd, telnetd, rexec, rlogind, rshd**
LONDON	DOMAIN/IX host and gateway	**tcp_server, inetd, routed, rwho, tftpd, ftpd, telnetd, rlogind, rshd**
LONDONLNK	DOMAIN/IX host and gateway	**tcp_server, inetd, routed, rwho, tftpd, ftpd, telnetd, rlogind, rshd**
BRUSSELS	DOMAIN host	**tcp_server, telnet_server, ftp_server**
PARIS	DOMAIN host	**tcp_server, telnet_server, ftp_server**
BERLIN	DOMAIN host/gateway	**tcp_server, rip_server, telnet_server, ftp_server**
MOSCOW	ETHERNET host	**tcp_server, telnet_server**

Chapter 5

Configuring TCP/IP

This chapter describes how to configure DOMAIN and DOMAIN/IX TCP/IP for each node when configuring a network for the first time. You can also follow these procedures to add a node within an existing network. It briefly describes how to configure TCP/IP on foreign hosts.

NOTE: Before you configure a node, you *must* be familiar with the terms and concepts that are described in Chapter 1, "Overview of TCP/IP." Also, if you are configuring a network for the first time, read and follow the procedures in Chapter 2, "Selecting Internet Addresses"; Chapter 3, "Editing TCP/IP Information Files"; and Chapter 4, "Starting TCP/IP Servers and Daemons."

5.1. Configuring a DOMAIN Network

If you are configuring TCP/IP on a DOMAIN network for the first time, you must configure *all* of the nodes that use or support TCP/IP — that is, each host and gateway and administrative node.

Briefly, we define these types of nodes as follows:

Host nodes — Nodes that are attached to a network that run TCP/IP and use TCP/IP utilities such as FTP or Telnet to communicate with each other.

Gateway nodes — Nodes with network hardware to connect networks. A gateway node can connect DOMAIN ring networks to create a DOMAIN internet (in which case the node is also a DOMAIN internet routing node). It can also connect a DOMAIN ring network to a different network (in which case the node contains network controller hardware). A gateway can also be a host.

Administrative nodes — Nodes that *assist* TCP/IP communications by providing host mapping tables. TCP/IP administrative nodes do not necessarily act as hosts or gateways.

This chapter provides different configuration procedures for you to follow. Which procedures you follow depends on whether you're configuring a node that:

- Has a disk or is diskless.
- Runs the AEGIS or DOMAIN/IX operating system.
- Runs DOMAIN/IX BSD4.2 TCP/IP in a DOMAIN ring network or internet. That is, the network does not run DOMAIN TCP/IP to connect the network to any different networks via gateway hardware.

5.1.1. Configuring TCP/IP on a Network That Contains Foreign Hosts

If you are configuring a DOMAIN network that uses gateway hardware to communicate with other foreign networks (including DARPA Internets), use the procedures described in this chapter in the following order:

1. Use Procedure 5-1 to set up TCP/IP information files on the TCP/IP administrative node (or nodes).
2. Use Procedure 5-2 or 5-3 to configure *each* host or gateway node that uses TCP/IP. Use Procedure 5-2 if the node is running DOMAIN; use Procedure 5-3 if the node is running DOMAIN/IX.
3. Use Procedure 5-5 to configure the remote (non-DOMAIN) hosts that will communicate with the DOMAIN nodes.
4. Use Procedure 5-6 to verify that TCP/IP is running on your network.

5.1.2. Configuring TCP/IP on a DOMAIN Internet

If you are configuring a DOMAIN network that uses DOMAIN routing service to communicate with other DOMAIN networks, use the procedures described in this chapter in the following order:

1. Use Procedure 5-1 to set up TCP/IP information files on the TCP/IP administrative node or nodes.
2. Use Procedure 5-2 or 5-3 to configure *each* host or gateway node that uses TCP/IP. Use Procedure 5-2 if the node is running DOMAIN; use Procedure 5-3 if the node is running DOMAIN/IX.
3. Use Procedure 5-6 to verify that TCP/IP is running on your network.

5.1.3. Configuring DOMAIN/IX BSD4.2 TCP/IP on a DOMAIN Network or Internet

When you're running DOMAIN/IX BSD4.2 TCP/IP in a single DOMAIN network or internet (that is, you don't have gateway hardware to communicate with different networks), you can use a simplified configuration procedure. So use the procedures in the following order:

1. Use Procedure 5-1 to set up TCP/IP information files for the TCP/IP administrative node(s).
2. Use Procedure 5-4 to configure *each* host or gateway node that uses TCP/IP.
3. Use Procedure 5-6 to verify that TCP/IP is running on your network.

If the network contains gateway hardware to communicate with different networks, follow Procedure 5–3 to configure the nodes running DOMAIN/IX.

Table 5–1 summarizes the procedures in this chapter you would use to configure each type of node on your network.

Table 5–1. Node Configuration Procedures

Use Procedure:	On System:	To Configure this node type:
5–1	Both	TCP/IP administrative node
5–2	DOMAIN	Internet host or gateway
5–3	DOMAIN/IX BSD4.2	Internet host or gateway
5–4	DOMAIN/IX BSD4.2	Host or routing node on a single DOMAIN network or a DOMAIN internet
5–5	Both	Foreign host

5.2. Before You Begin

Before you begin to configure your nodes, make sure you've performed all the required preliminary steps. Table 5–2 lists the steps you must take and the appropriate procedure to follow.

Table 5–2. Preliminary Configuration Procedures

Use Procedure:	To:
2–1	Select Internet addresses for each host and gateway that you want to configure.
3–1	Decide which nodes will serve as TCP/IP administrative nodes, and define the contents of the */sys/tcp/hostmap/local.txt* file (for DOMAIN TCP/IP) or the */etc/networks* and */etc/hosts* files (for a single ring network running DOMAIN/IX BSD4.2 TCP/IP).
4–1	Decide which nodes will run which servers and daemons.

5.3. Procedures for Configuring DOMAIN Nodes

The following sections contain the various procedures to configure your nodes with TCP/IP. To begin, follow Procedure 5–1 to configure the TCP/IP administrative node. Then follow the appropriate procedures to configure each gateway and host on the network. By configuring the administrative node first you ensure that all the TCP/IP mapping files are current, and it prevents you from duplicating files if you configure your administrative node to also be a gateway or host.

Most of the following procedures describe the steps required for either a disked or a diskless node since you can configure TCP/IP as a host or gateway on either type of node. However, Procedure 5–1 describes steps for disked nodes only because you should have a disked node for a TCP/IP administrative node. (The node's main function is to provide mapping files information for all nodes on the network.) Some steps in these procedures may differ depending on whether the node is disked, diskless, or used as a gateway. We mark such steps for your convenience.

> NOTE: These procedures refer to the */sys/node_data*[.nodeid] and *'node_data* directories. These two pathnames are *not* interchangeable. Using the wrong name in a link can result in circular file references.

5.3.1. Procedure 5–1. Configuring the TCP/IP Administrative Node

Use the following procedure to configure the node that you've selected to provide the TCP/IP configuration files. The administrative node contains the TCP/IP name and address mapping files. Since these files must be the same for all users on the network, we recommend that you have a single administrative node for the entire network. If you have more, you must be sure that they contain identical information.

The TCP/IP administrative node can be running either DOMAIN or DOMAIN/IX. If it's running DOMAIN/IX, it can also contain (but is not required to) the DOMAIN/IX administrative information contained in the */etc* directory. As a general rule, the */etc* directory resides on the DOMAIN/IX administrative node and all other nodes access it through links.

PROCEDURE 5-1. Configuring the TCP/IP Administrative Node

Task 1: Select the Internet Addresses

You should have a list of the Internet addresses for *all* of the nodes that will run TCP/IP. Include each host, gateway and the remote hosts that you can access across the gateway. Refer to Chapter 2, "Selecting Internet Addresses" for details on how to select these addresses.

Task 2: Install the Software

Install the TCP/IP software as described in the Release Notes of your TCP/IP product. (A hardcopy of the Release Notes is shipped with the product; an online version is available in the */doc* directory.) The installation procedure in this document tells you which software revision level of the operating system is required. If you are installing the operating system software at this time, you must install the software in the following order:

1. DOMAIN

2. DOMAIN/IX

NOTE: The DOMAIN/IX installation procedure instructs you to give */etc/run_rc* root ownership. If */etc/run_rc* does not have root ownership, processes required for TCP/IP will not run properly. If this was not done as part of the installation procedure; use the **chown** command to do it now.

3. TCP/IP

   When you install TCP/IP according to the Release Notes, the install procedure asks if you are installing software for an administrative, gateway, and host node. Answer YES to all three questions to get all the necessary TCP/IP files. (The installation procedure installs certain files depending on which node type you select. For a list of these files, see the Release Notes. After the install, use the DOMAIN **ld** command or DOMAIN/IX **ls** command to make sure that you have the required files.)

Task 3: Configure /sys/tcp/hostmap/hosts.txt

Configure the */sys/tcp/hostmap/hosts.txt* file as follows, depending upon whether you are connecting to the DARPA Internet.

Connecting to DARPA Internet?	Then:
YES	Use */sys/tcp/hostmap/hosts.txt*, which contains the DARPA Internet mapping information for all the hosts, gateways, and networks on the DARPA Internet.*
NO	Replace */sys/tcp/hostmap/hosts.txt* with an empty file to eliminate this information from your mapping files. Note that you *must* keep the name of this file.

* The copy of *hosts.txt* that we supply might not be the most recent version. After configuring your network, you can update this file by following Procedure 6-4 (for a DOMAIN administrative node) or Procedure 6-5 (for a DOMAIN/IX administration node).

Note that if you're installing TCP/IP for the first time, the TCP/IP installation procedure creates a */sys/tcp/hostmap/hosts.txt* template file that contains sample entries. If you're updating TCP/IP, the installation procedure preserves your edited file.

Task 4: Configure /sys/tcp/hostmap/local.txt

Edit the */sys/tcp/hostmap/local.txt* host mapping file to include the network numbers and addresses for each network, gateway, and host that is *not* already included in the */sys/tcp/hostmap/hosts.txt* file. Note that if you're installing TCP/IP for the first time, the TCP/IP installation procedure creates a */sys/tcp/hostmap/local.txt* template file that contains sample entries. If you're updating TCP/IP, the installation procedure preserves your edited file.

1. Add a **NET** entry to the file for *each* network that you want to access, including the DOMAIN network you're on. The **NET** entries should be the first entries in the file.

2. Add a **GATEWAY** entry to the file for *each* gateway between networks that you can access. All **GATEWAY** entries should follow the **NET** entries in the file. If your *local.txt* file lists multiple gateways to the same destination, list the **GATEWAY** entries in the order in which you want TCP/IP to use them when establishing connections.

3. Add a **HOST** entry to the file for each host on *all* networks that you can access. All **HOST** entries should follow the **GATEWAY** entries. If the node will be both a gateway and a host be sure to list it as both **GATEWAY** and **HOST** entries.

NOTE: If a node is a gateway and a host, you must list it separately as a **GATEWAY** and a **HOST** entry.

For details on */sys/tcp/hostmap/local.txt*, see Chapter 3, "Editing TCP/IP Information Files."

Task 5: Create the Host Tables

Run the **/sys/tcp/hostmap/makehost.sh** Shell script to convert the *local.txt* file into a format that TCP/IP software can use. Enter the following AEGIS Shell command:

```
$ /sys/tcp/hostmap/makehost.sh
```

NOTE: If you're running DOMAIN/IX, you must be in superuser mode to run this script.

If **makehost.sh** displays error messages, you can see the program as it is running by using the AEGIS **von** command. By invoking **von** before invoking **makehost.sh**, the script displays its program output line-by-line.

Task 6: Edit /sys/tcp/host_addr

If any host is connected by some network to a foreign host that does not support the Address Resolution Protocol (ARP), edit this node's */sys/tcp/host_addr* file to include each non-ARP host's Internet address physical address. You might have to create this file. Each non-ARP machine must be listed in the following format:

```
internet_address , local_address
```

The following example lists an Internet address and local address for a non-ARP ETHERNET host.

```
192.9.9.5   ,  2.7.1.0.3.a4
192.9.9.6   ,  9.4.c3.5.6.8
```

For details on */sys/tcp/host_addr*, see Section 3.3.5., "/sys/tcp/host_addr."

Task 7: Edit /etc/hosts.equiv if your network contains nodes using DOMAIN/IX BSD4.2

Use this file only if your network contains nodes running DOMAIN/IX BSD4.2. Otherwise, skip this step.

The */etc/hosts.equiv* file contains the names of hosts that are equivalent to this node for login purposes. That is, any host listed in this file does not have to provide a password when executing certain programs that require one; for example, **lpr**(1), **lprm**(1), **rcmd**(3X), **rcp**(1), **rlogin**(1), **rsh**(1).

You must list all nodes that use **lpr**, including the **lpd** printer daemon, in the **lpd** node's */etc/hosts.equiv* file.

The */etc/hosts.equiv* file contains the name of each equivalent host on a separate line. For example:

```
paris
brussels
berlin
nyc
seattle
```

We suggest that you store the */etc/hosts.equiv* file on the TCP/IP administrative node because it performs an administrative function. By editing this file now (when configuring TCP/IP for the first time), you don't have to edit it each time you configure a host.

For details, see Section 3.4.1., "/etc/hosts.equiv."

END OF PROCEDURE 5-1.

5.3.2. Procedure 5-2. Configuring a DOMAIN Host or Gateway Node

Use the following procedure to configure *each* DOMAIN node in your network that will run TCP/IP either as a host or gateway (or both).

If you're configuring TCP/IP for the first time, use Procedure 5-1 to configure the administrative node before configuring any hosts or gateways.

Task 1: Select an Internet Address

You should have a list of the Internet addresses for *all* of the nodes that will run TCP/IP. Include each host, gateway and the remote hosts that you can access across the gateway. Refer to Chapter 2, "Selecting Internet Addresses" for details on how to select these addresses.

Task 2: Stop tcp_server

If you are already using TCP/IP on the node, stop the **tcp_server** process by entering the following DOMAIN command:

```
$sigp tcp_server
```

Make sure you stop **tcp_server** on the gateway node as well as the node you're configuring so that **tcp_server** updates its routing table to include this new information.

Diskless Nodes: If you are configuring a diskless node for TCP/IP, and the partner node also uses TCP/IP, stop the **tcp_servers** on *both* nodes.

Task 3: Install the Software

Install the TCP/IP software as described in the Release Notes of your TCP/IP product. (A hardcopy of the Release Notes is shipped with the product; an online version is available in the */doc* directory.) The installation procedure in this document tells you which software revision level of the operating system is required. If you are installing the operating system software at this time, you must install the software in the following order:

1. DOMAIN
2. TCP/IP

   When you install TCP/IP according to the Release Notes, the install procedure asks if you are installing software for an administrative, gateway, and host node. Answer YES accordingly. (The installation procedure installs certain files depending on which node type you select. For a list of these files, see the Release Notes. After the install, use the DOMAIN **ld** command to make sure that you have the required files.)

Task 4: Update the Administrative Node Host Tables

If you are updating your TCP/IP configuration by adding a new node, or changing a node's Internet address, use the following steps to update the TCP/IP administrative node's host mapping tables. Note that the install procedure should have automatically created a link to the */sys/tcp/hostmap/local.txt* on the TCP/IP administrative node.

1. **Gateway nodes:** If you are configuring this node as a gateway, you must list it as a **GATEWAY** entry in the */sys/tcp/hostmap/local.txt* file. All **GATEWAY** entries must precede the **HOST** entries in the file.

2. **Gateway and host nodes:** If you are configuring this node as a host, list it as a **HOST** entry in the */sys/tcp/hostmap/local.txt* file. If the node will be both a gateway and a host be sure to list it as both **GATEWAY** and **HOST** entries.

3. Run the **/sys/tcp/hostmap/makehost.sh** Shell script to convert the *local.txt* file into a format that TCP/IP can use. Enter the following AEGIS Shell command:

   ```
   $ /sys/tcp/hostmap/makehost.sh
   ```

If **makehost.sh** displays error messages, you can see the program as it is running with the AEGIS **von** command. By invoking **von** before invoking **makehost.sh**, the script displays its program output line–by–line. For details on */sys/tcp/hostmap/local.txt*, see Chapter 3, "Editing TCP/IP Information Files."

Task 5: Edit the Node Start-up File

Edit the */sys/node_data*[.node_id]*/startup*[.type] file to include the following commands. See Chapter 4, "Starting TCP/IP Servers and Daemons" for a description of the processes that you can start using each of these files.

Server Process Command:	Where to Run:	Action:
cps /sys/tcp/tcp_server -n tcp_server	All hosts	Enables TCP/IP on node.
cps /sys/tcp/rip_server -n rip_server	All gateways	Manages TCP/IP routing tables.
cps /sys/tcp/ftp_server -n ftp_server	Hosts accepting **ftp** connections	Enables direct FTP access to service FTP requests.*
cps /sys/tcp/telnet_server -n telnet_server	Hosts accepting **telnet** connections	Enables inbound **telnet** access to accept **telnet** connections.*

* You don't need these servers to execute the corresponding commands.

Task 6: Edit /sys/node_data/thishost

The */sys/node_data*[.node_id]*/thishost* file lists the Internet name of the local host. You must have this file on each node that serves as a TCP/IP host or gateway. The file consists of the host's Internet name on a single line. Supply the node's name *without* the slashes. For example, the *thishost* file for //PARIS is simply `paris`.

Note that the TCP/IP installation procedure creates a template version of *thishost* in the */sys/node_data* directory and it creates a link in */sys/tcp* to *'node_data/thishost*.

Disked Nodes: If your node has a disk, add the host name to the */sys/node_data/thishost* file.

Diskless Nodes: If your node is diskless, copy the partner node's */sys/node_data/thishost* file to */sys/node_data*.node_id/*thishost*. Then edit this file to replace your partner's host name with your host's name.

Task 7: Edit /sys/node_data/networks

The */sys/node_data*[.node_id]*/networks* file defines the Internet addresses and the physical interface names (which identify the physical medium) of the local host. Also, if this node is part of a DOMAIN internet that's subdivided into subnets, you must supply the subnet mask in this file.

Specify this information in the networks file in the following format:

internet_address on physical_interface_symbol ; [subnet mask W.X.Y.Z ; comment]

where **W.X.Y.Z** fields can contain either a one (255) to denote the network or subnet field, or a zero (0) to denote the host field.

For example, the following is a two-byte subnet mask for a Type A address:

```
10.9.9.7  on eth0 ;  mask 255.255.255.0
```

For gateways, list *each* physical medium to which the node is connected. Note that you don't have to specify the loopback interface in this file. TCP/IP automatically assigns this interface (**lo0**) the Internet address 127.0.0.1.

PROCEDURE 5-2

Disked Nodes: If your node has a disk, edit the */sys/node_data/networks* file to specify the host's associations between Internet addresses and physical interfaces.

Diskless Nodes: If your node is diskless, copy the partner node's */sys/node_data/networks* file to */sys/node_data.*node_id*/networks*. Then edit this file to specify the host's associations between Internet addresses and physical interfaces.

The TCP/IP installation procedure creates template versions of the */sys/node_data/networks*. This file should be a link to *'node_data/networks*. For details on this file, see Chapter 3, "Editing TCP/IP Information Files."

Task 8: Initialize TCP/IP

Initialize TCP/IP by starting the node's **tcp_server** process and any other server processes required for TCP/IP communications. You can do this in either of two ways:

- Restart the Display Manager (DM) to start all the servers automatically. (Since you've included the appropriate commands in the node start-up file, the servers initialize when you reboot your node).
- Start each server manually with the **cps** command. Do this if you don't want to shut down your node.

When the **tcp_server** initializes, it automatically runs the following programs:

/sys/tcp/tcpinit
/sys/tcp/makegate

NOTE: The **tcp_server** might not be able to initialize these programs. If not, you can execute these programs yourself. Appendix B, "TCP/IP Reference" describes these programs.

To restart the DM, do the following:

1. Type the DM **ex** command to exit the DM:

```
Command: ex
```

   All current processes stop executing, the Display Manager exits, and the node enters the bootshell, which prompts you with a parenthesis.

2. Enter the **go** command to restart the DM:

```
) go
```

   The DM restarts and returns you to the **login** message. You can now log in and use TCP/IP.

To start the server(s) manually use the **cps** command. Start the **tcp_server** first, then you can start any other TCP/IP servers the node might want to run.

Start each server process by executing the following DM command, substituting **server_name** for the appropriate server: **rip_server, telnet_server,** or **ftp_server.**

```
Command: cps /sys/tcp/server_name -n server_name
```

To start a server remotely, execute the following DM command, substituting **server_name** for the appropriate server: **rip_server, telnet_server,** or **ftp_server.**

```
Command: crp -on //another-node -cps /sys/tcp/server_name -n server_name
```

Task 9: For Non-RIP Gateway Support, Use setroute

If you are configuring a gateway node, and some of the gateway nodes in the DARPA Internet do not support the Routing Information Protocol (RIP), use the **/sys/tcp/setroute** program to add each non-RIP gateway to the internal routing tables. Appendix B describes the **setroute** program.

Task 10: For Non-ARP Support, Run maphost

If you are configuring a gateway node that is connected to a network that contains a host that does not support the ARP protocol, run the **/sys/tcp/maphost** program. This utility updates the **tcp_server** address mapping tables to include the ETHERNET addresses from the */sys/tcp/host_addr* file. Appendix B describes the **maphost** program.

END OF PROCEDURE 5-2.

5.3.3. Procedure 5-3. Configuring a DOMAIN/IX BSD4.2 Host or Gateway Node

Use the following procedure to configure *each* DOMAIN/IX BSD4.2 node in your network that will run TCP/IP either as a host or gateway (or both).

If you're configuring TCP/IP for the first time, use Procedure 5-1 to configure the administrative node before configuring any hosts or gateways.

PROCEDURE 5-3. Configuring a DOMAIN/IX BSD4.2 Host or Gateway Node

Task 1: Select an Internet Address

You should have a list of the Internet addresses for *all* of the nodes that will run TCP/IP. Include each host, gateway and the remote hosts that you can access across the gateway. Refer to Chapter 2, "Selecting Internet Addresses" for details on how to select these addresses.

Task 2: Stop tcp_server

If you are already using TCP/IP on the node, stop the **tcp_server** process as follows:

1. Enter the following command to list all processes:

   ```
   % ps ax
   ```

2. Find the process number that corresponds to the **tcp_server** process and enter the number in the **kill**(1) command:

   ```
   % kill process_number
   ```

Make sure you stop **tcp_server** on the gateway node as well as the host so that **tcp_server** updates its routing table to include this new information.

Diskless Nodes: If you are configuring a diskless node for TCP/IP, and the partner node also uses TCP/IP, stop the **tcp_servers** on *both* nodes.

Task 3: Install the Software

Install the TCP/IP software as described in the Release Notes of your TCP/IP product. (A hardcopy of the Release Notes is shipped with the product; an online version is available in the */doc* directory.) The installation procedure in this document tells you which software revision level of the operating system is required. If you are installing the operating system software at this time, you must install the software in the following order:

1. DOMAIN

2. DOMAIN/IX

NOTE: The DOMAIN/IX installation procedure instructs you to give */etc/run_rc* root ownership. If */etc/run_rc* does not have root ownership, processes required for TCP/IP will not run properly. If this was not done as part of the installation procedure; use the **chown** command to do it now.

3. TCP/IP

   When you install TCP/IP according to the Release Notes, the install procedure asks if you are installing software for an administrative, gateway, and host node. Answer YES accordingly. (The installation procedure installs certain files depending on which node type you select. For a list of these files, see the Release Notes. After the install, use the DOMAIN/IX ls command to make sure that you have the required files.)

Task 4: Update the Administrative Node Host Tables

If you are updating your TCP/IP configuration by adding a new node, or changing a node's Internet address, use the following steps to update the TCP/IP administrative node's host mapping tables. Note that the install procedure should have automatically created a link to the */sys/tcp/hostmap/local.txt* on the TCP/IP administrative node.

1. **Gateway nodes:** If you are configuring this node as a gateway, you must list it as a **GATEWAY** entry in the */sys/tcp/hostmap/local.txt* file. All **GATEWAY** entries must precede the **HOST** entries in the file.

2. **Gateway and host nodes:** If you are configuring this node as a host, list it as a **HOST** entry in the */sys/tcp/hostmap/local.txt* file. If the node will be both a gateway and a host be sure to list it as both **GATEWAY** and **HOST** entries.

3. Run the **/sys/tcp/hostmap/makehost.sh** Shell script to convert the *local.txt* file into a format that TCP/IP can use. Enter the following Shell command:

```
$ /sys/tcp/hostmap/makehost.sh
```

If **makehost.sh** displays error messages, you can see the program as it is running by using the DOMAIN **von** command. By invoking **von** before invoking **makehost.sh**, the script displays its program output line-by-line.

For details on */sys/tcp/hostmap/local.txt*, see Chapter 3, "Editing TCP/IP Information Files."

Task 5: Update /etc/hosts.equiv

The */etc/hosts.equiv* file contains the names of hosts that are equivalent to this node for login purposes. That is, any host listed in this file does not have to provide a password when executing certain programs that require one; for example, **lpr**(1), **lprm**(1), **rcmd**(3X), **rcp**(1), **rlogin**(1), **rsh**(1).

You must list all nodes that use **lpr**, including the **lpd** printer daemon, in the **lpd** node's */etc/hosts.equiv* file.

The */etc/hosts.equiv* file lists the names of all equivalent hosts on a separate line. For example:

```
paris
brussels
berlin
nyc
seattle
```

As a general rule, the */etc* directory resides on the DOMAIN/IX administrative node and all other nodes access it through links. In most cases, only the system administrator has access rights to edit this file. For details, see Section 3.4.1., "/etc/hosts.equiv."

Task 6: Edit /etc/gateways

If you are configuring a gateway and some of the gateways in the Internet do *not* support the Routing Information Protocol (RIP), create or edit the */etc/gateways* file to include each non-RIP gateway.

The */etc/gateways* file associates a destination network with the next gateway in the route to the destination. The **tcp_server** will then send messages for all hosts on that network to the required gateway.

Each entry in the file is a single line in the following format:

dest-type name1 gateway name2 metric hops gate-type

For example, the following lines from a */etc/gateways* file provide routing information about two networks.

```
network next-net gateway gate1 metric 1 passive
network third-net gateway gate1 metric 2 passive
```

See Section 3.4.2., "/etc/gateways" for details.

Task 7: Edit the Node Start-up Files

Use the following steps to update your node start-up files. See Chapter 4, "Starting TCP/IP Servers and Daemons" for a description of the processes that you can start using each of these files.

1. Edit the */sys/node_data*[.node_id]*/startup*[.type] file to include the following commands. You *must* start **tcp_server** first; otherwise processes that */etc/run_rc* initializes will not run.

   ```
   env SYSTYPE 'bsd4.2'
   cps /sys/tcp/tcp_server -n tcp_server
   cps /etc/run_rc
   ```

2. Edit the */etc/rc* file (which is a link to *'node_data/etc.rc*) by removing the comment character (#) from the lines that contain processes that you want to run *on this host*. You can start the processes **ined**(8), **routed**(8), **rwhod**(8), **sendmail**(8), **tftpd**(8). For example, to specify **inetd** uncomment the following lines in */etc/rc*:

   ```
   if [-f /etc/inetd ]; then
      /etc/inetd &
   fi
   ```

   NOTE: You must give *etc.rc* root ownership with the **chown** command. If you do not give this file root ownership, TCP/IP won't run properly. To give root ownership, log in as root, and type the following command:

   ```
   % chown root etc.rc
   % chmod 4755 etc.rc
   ```

3. If you specified **inetd** in Step 2, edit the */etc/inetd.conf* file (which is a link to *'node_data/etc.inetd.conf*) by removing the comment character (#) from the lines that contain processes that you want to run *on this host*. You can start the processes **ftpd**(8), **rexecd**(8), **rlogind**(8), **rsh**(8), **telnetd**(8).

For details, see Section 4.3, "Running DOMAIN/IX BSD4.2 Daemons."

Task 8: Edit the Shell Login Files

Edit your *.cshrc* file (if you use the C shell) or your *.profile* file (if you use the Bourne shell) to include the */com* directory in the search path. The */com* directory includes the **tcpstat**, **host**, and **net** commands that you use to monitor and manage TCP/IP communications. By including the */com* directory in the search path you eliminate the need to specify the directory in these commands.

The */com* directory also includes the DOMAIN versions (as opposed to BSD4.2 versions) of **telnet** and **ftp**. Therefore, the */com* directory must *follow* the */usr/ucb* directory in the search path (unless you want to use the DOMAIN Telnet and FTP as the default versions).

For example, include the following line in your *.cshrc* file:

```
set path=(. /bin /usr/bin /usr/ucb /com ~/com)
```

Include the following lines in your *.profile* file:

```
PATH=:.:/bin:/usr/bin:/usr/ucb:/com:~/com
export PATH
```

Task 9: Edit /sys/node_data/thishost

The */sys/node_data*[.node_id]*/thishost* file lists the Internet name of the local host. You must have this file on each node that serves as a TCP/IP host or gateway. The file consists of the host's Internet name on a single line. Supply the node's name *without* the slashes. For example, the *thishost* file for //PARIS is simply `paris`.

Note that the TCP/IP installation procedure creates template versions of */sys/node_data/thishost*. Also, this file should be a link to *'node_data/thishost*.

Disked Nodes: If your node has a disk, add the host name to the */sys/node_data/thishost* file.

Diskless Nodes: If your node is diskless, copy the partner node's */sys/node_data/thishost* file to */sys/node_data*.node_id*/thishost*. Then edit this file to replace your partner's host name with your host's name.

Task 10: Edit /sys/node_data/networks

The */sys/node_data*[.node_id]*/networks* file defines the Internet addresses and the physical interface names (which identify the physical medium) of the local host. Also, if this node is part of a DOMAIN internet that's subdivided into subnets, you must supply the subnet mask in this file.

Specify this information in the networks file in the following format:

internet_address on physical_interface_symbol ; [subnet mask W.X.Y.Z ; comment]

where **W.X.Y.Z** fields can contain either a one (255) to denote the network or subnet field, or a zero (0) to denote the host field.

For example, the following is a two-byte subnet mask for a Type A address:

```
10.9.9.7  on eth0 ;  mask 255.255.255.0
```

For gateways, list *each* physical medium to which the node is connected. Note that you don't have to specify the loopback interface in this file. TCP/IP automatically assigns this interface (**lo0**) the Internet address 127.0.0.1.

Disked Nodes: If your node has a disk, edit */sys/node_data/networks* file to specify the host's associations between Internet addresses and physical interfaces.

Diskless Nodes: If your node is diskless, copy the partner node's */sys/node_data/networks* file to */sys/node_data*.node_id*/networks*. Then edit this file to specify the host's associations between Internet addresses and physical interfaces.

The TCP/IP installation procedure creates template versions of */sys/node_data/networks*. This file should be a link to *'node_data/networks*. For details on this file, see Chapter 3, "Editing TCP/IP Information Files."

PROCEDURE 5-3

Task 11: Initialize TCP/IP

Initialize TCP/IP by starting the node's **tcp_server** process and any other server processes required for TCP/IP communications. You can do this in either of two ways:

- Restart the Display Manager (DM) to start all the servers and daemons automatically. Since you've included the appropriate commands in the node start-up file, the servers initialize when you reboot your node.
- Start each server manually with the **cps** command. Do this if you don't want to shut down your node.

When the **tcp_server** initializes, it automatically runs the following programs:

/sys/tcp/tcpinit
/sys/tcp/makegate

NOTE: The **tcp_server** might not be able to initialize these programs. If not, you can try executing these programs yourself. Appendix B, "TCP/IP Reference" describes these programs.

To restart the DM, do the following:

1. Type the DM **ex** command to exit the DM:

   ```
   Command: ex
   ```

   All current processes stop executing, the Display Manager exits, and the node enters the bootshell, which prompts you with a parenthesis.

2. Enter the **go** command to restart the DM:

   ```
   ) go
   ```

   The DM restarts and returns you to the **login** message. You can now log in and use TCP/IP.

To start the server and daemon processes manually use the **cps** command. Start the **tcp_server** first, then you can start any other TCP/IP servers the node might want to run. To start **tcp_server**:

```
Command: cps /sys/tcp/tcp_server -n tcp_server
```

Start any daemons, such as **inetd**(8), **lpd**(8), **routed**(8), **rwhod**(8), **sendmail**(8), or **tftpd**(8), by running the **/etc/run_rc** program. To do so, enter the following DM command:

```
Command: cps /etc/run_rc
```

Task 12: For Non-ARP Support, Run maphost

If you are configuring a gateway that is connected to a foreign host that does not support the ARP protocol run the **/sys/tcp/maphost** program. This utility updates the **tcp_server** address mapping tables to include the ETHERNET addresses from the */sys/tcp/host_addr* file. Appendix B describes the **maphost** program.

END OF PROCEDURE 5-3.

5.3.4. Procedure 5–4. Configuring a DOMAIN/IX BSD4.2 Host that Uses Only BSD4.2 TCP/IP

Use the following procedure to configure each node running DOMAIN/IX BSD4.2 TCP/IP on your network or internet. Note that since BSD4.2 TCP/IP does not provide a way to run on foreign networks, you can use it only within a DOMAIN network or internet. If the node runs gateway hardware to communicate with other networks, follow Procedure 5–3.

If you're configuring TCP/IP for the first time, use Procedure 5–1 to configure the administrative node before configuring all hosts.

PROCEDURE 5-4. Configuring a DOMAIN/IX BSD4.2 Host that Uses BSD4.2 TCP/IP to Communicate Only on DOMAIN Networks or Internets

Task 1: Select an Internet Address

You should have a list of the Internet addresses for *all* of the nodes that will run TCP/IP. Include each host, gateway and the remote hosts that you can access across the gateway. Refer to Chapter 2, "Selecting Internet Addresses" for details on how to select these addresses.

Task 2: Stop tcp_server

If you are already using TCP/IP on the node, stop the **tcp_server** process as follows:

1. Enter the following command to list all processes:

   % ps ax

2. Find the process number that corresponds to the **tcp_server** process and enter the number in the **kill**(1) command:

   % kill process_number

Diskless Nodes: If you are configuring a diskless node for TCP/IP, and the partner node also uses TCP/IP, stop the **tcp_servers** on *both* nodes.

Task 3: Install the Software

Install the TCP/IP software as described in the Release Notes of your TCP/IP product. (A hardcopy of the Release Notes is shipped with the product; an online version is available in the */doc* directory.) The installation procedure in this document tells you which software revision level of the operating system is required. If you are installing the operating system software at this time, you must install the software in the following order:

1. DOMAIN

2. DOMAIN/IX

NOTE: The DOMAIN/IX installation procedure instructs you to give */etc/run_rc* root ownership. If */etc/run_rc* does not have root ownership, processes required for TCP/IP will not run properly. If this was not done as part of the installation procedure; use the **chown** command to do it now.

3. TCP/IP

   When you install TCP/IP according to the Release Notes, the install procedure asks if you are installing software for an administrative, gateway, and host node. Answer YES accordingly. (The installation procedure installs certain files depending on which node type you select. For a list of these files, see the Release Notes. After the install, use the DOMAIN/IX **ls** command to make sure that you have the required files.)

Task 4: Edit /etc/hosts

Add an entry to the */etc/hosts* file to list all the TCP/IP hosts you want to access. If you are configuring an administrative node for the first time, add an entry for *each* BSD4.2 node on the network. Each line has the following format:

Internet-address host-name

For example:

```
127.0.0.1          localhost
197.9.8.1          timeix
```

You must edit */etc/hosts* if you have DOMAIN/IX BSD4.2 TCP/IP on a DOMAIN network and you want to run the following: **lpr**(1), **rcmd**(3X), **rcp**(1), **rlogin**(1), **rsh**(1), **rexec**(3X), **ftp** or **telnet**.

Task 5: Edit /etc/networks

Make sure there is an entry in the */etc/networks* file for the network you're on. Since this is a single DOMAIN network or internet, the file should contain only one entry in the following format:

network-name network-number

For example:

```
domain-ring  129.9.0.0
```

Task 6: Edit /sys/tcp/gateways file on routing servers nodes

If you are configuring a routing server node that serves as a gateway between two DOMAIN networks in an internet, edit the */sys/tcp/gateways* file. This file contains routing information that **tcp_server** uses when it starts. Include an entry for each gateway in the DOMAIN internet.

Each entry in the file is a single line in the following format:

addr1, addr2 : name : protocols

For example:

```
198.8.8.253 : 198.8.6.253 : nyc : IP/GW, GW/PRIME,
198.8.6.241 : 198.8.4.241 : london : IP/GW, GW/DUMB
```

Note that both DOMAIN TCP/IP and DOMAIN/IX BSD4.2 TCP/IP use */sys/tcp/gateways*. However, DOMAIN TCP/IP users don't have to edit this file because the **makehost.sh** Shell script does it automatically. For details on the format of this file, see the section on */sys/tcp/hostmap/local.txt* in Chapter 3.

NOTE: Do not confuse */sys/tcp/gateways* with */etc/gateways*. We suggest that you do *not* use */etc/gateways* in a DOMAIN-only environment, and that you replace it with an empty file.

Task 7: Update /etc/hosts.equiv

The */etc/hosts.equiv* file contains the names of hosts that are equivalent to this node for login purposes. That is, any host listed in this file does not have to provide a password when executing certain programs that require one; for example, **lpr**(1), **lprm**(1), **rcmd**(3X), **rcp**(1), **rlogin**(1), **rsh**(1).

You must list all nodes that use **lpr**, including the **lpd** printer daemon, in the **lpd** node's */etc/hosts.equiv* file.

PROCEDURE 5-4

The */etc/hosts.equiv* file contains the name of each equivalent host on a separate line. For example:

```
paris
brussels
berlin
nyc
seattle
```

As a general rule, the */etc* directory resides on the DOMAIN/IX administrative node and all other nodes access it through links. In most cases, only the system administrator has access rights to edit this file. For details, see Section 3.4.1. "/etc/hosts.equiv."

Task 8: Edit the Node Start-up Files

Use the following steps to update your node start-up files. See Chapter 4,"Starting TCP/IP Servers and Daemons" for a description of the processes that you can start using each of these files.

1. Edit the */sys/node_data*[.node_id]*/startup*[.type] file to include the following commands. You must start **tcp_server** first; otherwise, processes that */etc/run_rc* initializes will not run.

   ```
   env SYSTYPE  'bsd4.2'
   cps /sys/tcp/tcp_server -n tcp_server
   cps /etc/run_rc
   ```

2. Edit the */etc/rc* file (which is a link to *'node_data/etc.rc*) by removing the comment character (#) from the lines that contain processes that you want to run *on this host*. You can start the processes: **ined**(8), **routed**(8), **rwhod**(8), **sendmail**(8), **tftpd**(8). Note that you should run the **routed** daemon on each routing server that serves as a TCP/IP gateway in an internet. For example, to specify **inetd** uncomment the following lines in */etc/rc*:

   ```
   if [-f /etc/inetd ]; then
      /etc/inetd &
   fi
   ```

   NOTE: To edit this file, you must give *etc.rc* root ownership with the **chown** command. If you do not give this file root ownership, TCP/IP won't run properly. To give root ownership, log in as root, and type the following command:

   ```
   % chown root etc.rc
   % chmod 4755 etc.rc
   ```

3. If you specified **inetd** in Step 2, edit the */etc/inetd.conf* file (which is a link to *'node_data/etc.inetd.conf*) by removing the comment character (#) from the lines that contain processes that you want to run *on this host*. You can start the processes **ftpd**(8), **rexecd**(8), **rlogind**(8), **rsh**(8), **telnetd**(8).

For details, see Section 4.3., "Running DOMAIN/IX BSD4.2 Daemons."

Task 9: Edit /sys/node_data/thishost

The */sys/node_data*[.node_id]*/thishost* file lists the Internet name of the local host. You must have this file on each node that serves as a TCP/IP host or gateway. The file consists of the host's Internet name on a single line. Supply the node's name *without* the slashes. For example, the *thishost* file for //PARIS is simply `paris`.

Note that the TCP/IP installation procedure creates template versions of */sys/node_data/thishost*. Also, this file should be a link to *'node_data/thishost*.

Disked Nodes: If your node has a disk, add the host name to the */sys/node_data/thishost* file.

Diskless Nodes: If your node is diskless, copy the partner node's */sys/node_data/thishost* file to */sys/node_data*.node_id/*thishost*. Then edit this file to replace your partner's host name with your host's name.

Task 10: Edit /sys/node_data/networks

The */sys/node_data*[.node_id]*/networks* file defines the Internet addresses and the physical interface names (which identify the physical medium) of the local host. Also, if this node is part of a DOMAIN internet that's subdivided into subnets, you must supply the subnet mask in this file.

Specify this information in the networks file in the following format:

internet_address on physical_interface_symbol ; [subnet mask W.X.Y.Z ; comment]

where **W.X.Y.Z** fields can contain either a one (255) to denote the network or subnet field, or a zero (0) to denote the host field.

For example, the following is a two-byte subnet mask for a Type A address:

```
10.9.9.7  on eth0 ;  mask 255.255.255.0
```

For gateways, list *each* physical medium to which the node is connected. Note that you don't have to specify the loopback interface in this file. TCP/IP automatically assigns this interface (**lo0**) the Internet address 127.0.0.1.

Disked Nodes: If your node has a disk, edit the */sys/node_data/networks* file to specify the host's associations between Internet addresses and physical interfaces.

Diskless Nodes: If your node is diskless, copy the partner node's */sys/node_data/networks* file to */sys/node_data*.node_id/*networks*. Then edit this file to specify the host's associations between Internet addresses and physical interfaces.

The TCP/IP installation procedure creates template versions of */sys/node_data/networks*. This file should be a link to *'node_data/networks*. For details on this file, see Chapter 3, "Editing TCP/IP Information Files."

Task 11: Initialize TCP/IP

Initialize TCP/IP by starting the node's **tcp_server** process and any other server processes required for TCP/IP communications. You can do this in either of two ways:

- Restart the Display Manager (DM) to start all the servers (such as **routed** and **inetd**) automatically. Since you've included the appropriate commands in the node start-up file, the servers initialize when you reboot your node.

- Start each server manually with the **cps** command. Do this if you don't want to shut down your node.

When the **tcp_server** initializes, it automatically runs the following programs:

/sys/tcp/tcpinit
/sys/tcp/makegate

NOTE: The **tcp_server** might not be able to initialize these programs. If not, you can try executing these programs yourself. Appendix B describes these programs.

PROCEDURE 5-4

To restart the DM, do the following:

1. Type the DM **ex** command to exit the DM:

   ```
   Command: ex
   ```

   All current processes stop executing, the Display Manager exits, and the node enters the bootshell, which prompts you with a parenthesis.

2. Enter the **go** command to restart the DM:

   ```
   ) go
   ```

   The DM restarts and returns you to the **login** message. You can now log in and use TCP/IP.

Start the server and daemon processes manually use the **cps** command. Start the **tcp_server** first, then you can start any other TCP/IP servers the node might want to run. To start **tcp_server**:

```
Command: cps /sys/tcp/tcp_server -n tcp_server
```

To start any daemons, such as **inetd**(8), **lpd**(8), **routed**(8), **rwhod**(8), **sendmail**(8), or **tftpd**(8), by running the **/etc/run_rc** program. To do so, enter the following DM command:

```
Command: cps /etc/run_rc
```

END OF PROCEDURE 5-4.

5.4. Configuring Non–DOMAIN Hosts

In the same way that you must add the names and Internet addresses of foreign networks, gateways, and hosts to the */sys/tcp/hostmap/local.txt* file on the DOMAIN network, you must add the names and Internet addresses of the DOMAIN network, gateway, and hosts to some equivalent file or files on the *other* side of the connection. Each foreign host and gateway node that will communicate with nodes on the DOMAIN network must have access to this information.

We can't provide procedures for all possible situations; however, we *can* provide some general rules that may be helpful if you are using TCP/IP for the first time. The following two sections describe what to do if the other network is a standard DARPA Internet TCP/IP implementation, or a BSD4.2 UNIX TCP/IP implementation.

5.4.1. Configuring DARPA Internet TCP/IP Hosts

If you are configuring a host that uses the standard DARPA Internet TCP/IP, locate its *hosts.txt* (or equivalent) file. This is the file that contains a list of hosts that aren't listed by the NIC. You must include all the entries (for your DOMAIN network, gateways, and hosts) that you listed in Procedure 5–1, Task 4.

5.4.2. Configuring BSD4.2 UNIX Hosts

If you are configuring a host that uses BSD4.2 UNIX, you must follow the steps described in Procedure 5–5 *at the non–DOMAIN host*. We assume several things in this procedure, including:

- You have the permissions required to edit such files as */etc/hosts* on the non–DOMAIN host.
- You understand the */etc/rc* file on the UNIX system well enough to set up the servers and daemons that are appropriate for the UNIX host.

PROCEDURE 5-5. Configuring a Non-DOMAIN BSD4.2 Host to Communicate with a Host on a DOMAIN Network

Task 1: Edit /etc/networks

Make sure that there is an entry in the */etc/networks* file for the DOMAIN network that you want to access. Unless you are adding the first DOMAIN host, this file should already have a DOMAIN network entry.

Task 2: Edit /etc/hosts

Make sure that there is an entry in the */etc/hosts* file for the DOMAIN host (or hosts) that will communicate with this host.

NOTE: The method you use to manage the */etc/hosts* and */etc/networks* files depends on your system administration procedures. For example, you might edit these files directly. Or, you might edit a file similar to *local.txt* and use the **/etc/htable**(8) program to put the entries in the files. See your host's BSD4.2 documentation for more information on the **/etc/htable**(8) program.

Task 3: Edit /etc/hosts.equiv

The */etc/hosts.equiv* file contains the names of hosts that are equivalent to this node for login purposes. That is, any host listed in this file does not have to provide a password when executing certain programs that require one; for example, **lpr**(1), **lprm**(1), **rcmd**(3X), **rcp**(1), **rlogin**(1), **rsh**(1). Edit the file according to the details described in Section 3.4.1.,"/etc/hosts.equiv."

Task 4: Edit /etc/rc

Edit the non-DOMAIN host's */etc/rc* file, if necessary. This file controls the services that this host provides to other hosts; it is a UNIX Shell script that runs automatically when the UNIX system is rebooted. Some installations use a */etc/rc.local* file for commands that are pertinent to a single site. For more details on **rc**(8) see the *BSD4.2 UNIX Programmer's Manual.*

The */etc/rc* file must specify the **routed**(8) routing daemon and any other daemons, such as **telnetd**(8) and **ftpd**(8), that you require to enable TCP/IP communications with the DOMAIN hosts. Section 4.3, "Running DOMAIN/IX BSD4.2 Daemons" discusses these processes.

Task 5: Ensure that the Daemons are Running

Make sure the routing daemon, **routed**, and any other daemons that you require for TCP/IP communications with DOMAIN hosts run on this host. You can check whether this process is running by using the UNIX **ps**(1) command. If necessary, reboot the system or start the processes manually.

END OF PROCEDURE 5-5.

5.5. Verifying TCP/IP on Your Configured Network

After configuring TCP/IP on *each* node in the network, use the following procedure to verify that it's running correctly.

PROCEDURE 5-6. Verifying TCP/IP on Your Configured Network

Task 1: Run tcp_server in a Window

The **tcp_server** produces messages when it initializes and encounters errors. To monitor **tcp_server**, you can run it in a window as follows:

1. If it's currently running, stop **tcp_server** with the DOMAIN **sigp** or DOMAIN/IX **kill**(1) command.
2. Restart **tcp_server** in a window in which a Shell process is running by typing:

   $ **/sys/tcp/tcp_server**

3. When **tcp_server** initializes, it should identify the network interfaces you supplied when configuring the node.

Task 2: Run TCP/IP Network Status Command

To check network status, run the network status command with various options. If you're running DOMAIN, use **tcpstat**. If you're running DOMAIN/IX, use **netstat**.

DOMAIN Status Option:	DOMAIN/IX Option:	Displays Information About:
tcpstat -i	**netstat -i**	Physical interfaces between the host and network.
tcpstat -g	**netstat -r**	Displays information about gateways.
tcpstat	**netstat**	Displays information about each open connection.

For more information on **tcpstat**, see Chapter 7, "Troubleshooting TCP/IP."

Task 3: Start a telnet Session

You can tell when a node is configured for TCP/IP if it can run an application. To check this, try to use **telnet** from the node as follows. If **telnet** is successful, you'll be able to login in to the specified node.

1. Start **telnet**, specifying your own node as the destination.
2. Start **telnet**, specifying a node on your local network.
3. Start **telnet**, specifying a node on the other side of a gateway.

END OF PROCEDURE 5-6.

Chapter 6

Managing TCP/IP

This chapter describes procedures for managing TCP/IP once you have configured all the nodes on your network. It includes procedures for managing DOMAIN TCP/IP and other procedures for managing DOMAIN/IX BSD4.2 TCP/IP on a DOMAIN-only network or internet.

This chapter includes procedures for the following:

- Updating TCP/IP software
- Starting and stopping server processes
- Maintaining configuration files for DOMAIN TCP/IP
- Maintaining configuration files for DOMAIN/IX BSD4.2 TCP/IP
- Maintaining internal tables on hosts and gateways

6.1. Updating TCP/IP Software

To update your TCP/IP software, you simply install the latest revision of TCP/IP according to the installation procedures described in the Release Notes of the TCP/IP software. (A hardcopy of the Release Notes is shipped with the product; an online version is available in the */doc* directory.) Note that the install procedure indicates the version of the operating system that the node must be running before you can install the latest version of TCP/IP on that node.

After you successfully complete the installation according to the Release Notes, shut down and reboot your node to invoke the new TCP/IP software. After you reboot, make sure the **tcp_server** process started by invoking the **pst** command (in DOMAIN) or the **ps** command(in DOMAIN/IX). If the **tcp_server** isn't running, start it with the following DM command:

```
Command:  cps /sys/tcp/tcp_server -n tcp_server
```

6.2. Starting and Stopping Servers and Daemons

Generally, you use start-up files located in the */sys/node_data* directory to start the various server processes whenever your node reboots (as described in Chapter 4). However, in some cases, you may want to stop and start these processes manually. For example, you might want to stop **tcp_server** and then restart it in a window so that you can monitor its activity.

> NOTE: You must stop the **tcp_server** before chaning any TCP/IP files and restart the server after completing the changes.

The following sections describe how to start and stop DOMAIN servers and DOMAIN/IX daemons.

6.2.1. Starting and Stopping DOMAIN Servers

You use the standard Display Manager **cps** command to start, and the DOMAIN Shell **sigp** command to stop the following DOMAIN servers:

- **tcp_server**
- **rip_server**
- **ftp_server**
- **telnet_server**

If the node that runs, or will run, a server process also runs the Server Process Manager (SPM) you can start or stop the processes from any node on the network. Enter the following commands to stop the process:

```
$ crp -on node_id -me
Connected to node nnnn  "//node_name"
$ sigp process_name
```

To create a process on a remote node, use the **crp** command line with the **-cps** option. For example, to start the **ftp_server** on the node //EGIL, type:

```
$ crp -on //egil -cps /sys/tcp/ftp_server -n ftp_server
```

6.2.2. Starting and Stopping DOMAIN/IX BSD4.2 Daemons

Use the standard DOMAIN/IX commands to start and stop the DOMAIN/IX BSD4.2 daemons. To start a daemon from a Shell, specify the daemon pathname followed by an ampersand (&). (The ampersand signifies to start the daemon in background mode.)

To stop a daemon, use the following procedure:

1. Use the **ps**(1) command with the **-ax** option to get the process number of the process to be killed.
2. Use the **kill**(1) command to kill the process.

For example, to kill the **inetd** daemon:

```
% ps ax
  PID STAT    TIME COMMAND
    2 R    2052:54 null
    3 S       2:41 wired-DXM
    4 S       7:00 purifier
    5 S       8:55 unwired-DXM
    6 S       0:01 netreceive
    7 S       1:20 netpaging
    8 S       3:37 netrequest
    1 S      15:47 /sys/dm/dm
   81 R       0:10 ps ax
   86 S       0:01 /etc/inetd
% kill 86
```

6.3. Maintaining Configuration Files for DOMAIN TCP/IP

As you add and remove nodes from the network, you must be sure to update the appropriate TCP/IP information files. Which configuration files you change depends on whether you're running DOMAIN TCP/IP on a network connected to foreign networks, or DOMAIN/IX BSD4.2 TCP/IP on a DOMAIN network or internet. This section specifies which files you must update, and how to make these changes for DOMAIN TCP/IP.

Follow the procedures in this section if you are using DOMAIN TCP/IP or if you are using both DOMAIN TCP/IP *and* DOMAIN/IX BSD4.2 TCP/IP. Follow the procedures in the Section 6.4., "Maintaining Configuration Files for DOMAIN/IX BSD4.2 TCP/IP," if you are using *only* DOMAIN/IX BSD4.2 TCP/IP.

The following sections cover the following procedures:

- Adding hosts to a network
- Removing hosts or gateways from the network
- Changing the name of a host or gateway
- Changing the Internet address of a host or gateway
- Changing a DOMAIN network's Internet network number
- Subdividing an internet into subnets
- Getting an official *hosts.txt* file from the Network Information Center (NIC)

6.3.1. Adding Hosts or Gateways

To add hosts or gateways to an existing TCP/IP network, follow the procedures to configure a host or gateway as described in Chapter 5, "Configuring TCP/IP."

6.3.2. Removing a TCP/IP Host or Gateway

You should remove a TCP/IP host or gateway from the TCP/IP host address mapping files when you either physically remove the node from its network, or when you stop using TCP/IP on the node.

To remove a host or gateway entry, delete its **HOST** entry and its **GATEWAY** entry from the */sys/tcp/hostmap/local.txt* file on the DOMAIN TCP/IP administrative node or nodes. Then run **/sys/tcp/maphost.sh** on the TCP/IP administrative nodes. Also, be sure to delete the same information from all foreign host and gateway mapping files.

In some cases, you may be using the same Internet address again. For example, after removing a node, you can associate the same Internet address with a new node name. In this case, you must clear the old association between the Internet address and the local address from the hosts' internal address mapping tables. To do so, you must run the **maphost** command on *each* TCP/IP host and gateway.

```
$ /sys/tcp/maphost -c
```

Note that you must stop the **tcp_server** before changing any TCP/IP information files and restart the server after completing the changes.

6.3.3. Changing a Host or Gateway Name

If you change the name of any DOMAIN TCP/IP host or gateway, you must update the TCP/IP information files by following Procedure 6-1. This procedure applies whether you are changing the name of a DOMAIN node or a host on the foreign host.

Task 1: Stop tcp_server

Stop **tcp_server** by entering the following DOMAIN or DOMAIN/IX commands. If you're running DOMAIN, type the following:

```
$ sigp tcp_server
```

If you're running DOMAIN/IX, type the following to list all the processes:

```
% ps ax
```

Locate the number of the process corresponding to the **tcp_server** process and type:

```
% kill process_number
```

Task 2: Edit the thishost file

If you are changing the name of a DOMAIN node, replace the old name in the node's */sys/node_data[.nodeid]/thishost* file with the new name.

Task 3: Edit the local.txt file

Edit the */sys/tcp/hostmap/local.txt file* on the TCP/IP administrative node as follows. (Note that the host's */sys/tcp/hostmap* directory is a link to the TCP/IP administrative node.)

1. Change the name field of **HOST** entry in the */sys/tcp/hostmap/local.txt* file to the new host name.

2. If the host is also a gateway, change the name field of **GATEWAY** entry in the */sys/tcp/hostmap/local.txt* file to the new host name.

3. If you have more than one TCP/IP administrative node, make sure you change all copies of the *local.txt* file.

Task 4: Run makehost.sh

Run the **/sys/tcp/hostmap/makehost.sh** Shell script. This script converts the *local.txt* file into a format that TCP/IP software can use. If you have more than one TCP/IP administrative node, you must run the Shell script on each administrative node. To run the script, enter the following command in a DOMAIN or DOMAIN/IX Shell:

```
$ /sys/tcp/hostmap/makehost.sh
```

Task 5: Update the Remote Mapping Tables

Follow the procedures required to update the host name in the mapping information tables maintained by your foreign hosts.

Task 6: Restart tcp_server

Enter the following command to restart **tcp_server**:

```
Command:  cps /sys/tcp/tcp_server -n tcp_server
```

To create **tcp_server** on a remote node, type:

```
$ crp -on //node_name  -cps /sys/tcp/tcp_server -n tcp_server
```

END OF PROCEDURE 6-1.

6.3.4. Changing Internet Addresses or Network Numbers

If you change the Internet address of a DOMAIN TCP/IP host or gateway, you must update the appropriate TCP/IP information files by following Procedure 6–2. This procedure applies whether you are changing the address of a DOMAIN node or a host on the foreign host.

You can also use this procedure to change your network number. You will want to replace your network number with the official NIC–supplied network number.

Task 1: Stop tcp_server

Stop **tcp_server** by entering the following DOMAIN or DOMAIN/IX commands. If you're running DOMAIN, type the following:

```
$ sigp tcp_server
```

If you're running DOMAIN/IX, type the following to list all the processes:

```
% ps ax
```

Locate the number of the process corresponding to the **tcp_server** process and type:

```
% kill process_number
```

Task 2: Edit the networks file

If you are changing the address of a single DOMAIN node, replace the old address in the node's */sys/node_data[.nodeid]/networks* file with the new address.

If you are changing the Internet network number of a DOMAIN network, you must edit the */sys/node_data[.nodeid]/networks* file on *each host and gateway* to add the node's new address.

Task 3: Edit the local.txt file

Edit the */sys/tcp/hostmap/local.txt* file on the TCP/IP administrative node as follows. (Since the host's /sys/tcp/hostmap directory is usually a link to the TCP/IP administrative node, you can simply edit the local.txt file on the host node.) When updating this file make sure you change *both* host and gateway entries, since a single node will have entries in both places if it's a host as well as a gateway.

1. Change the entry in the address field of the host or gateway's **HOST** entry in the */sys/tcp/hostmap/local.txt* file to the new host address.

2. If you are changing a gateway address, replace the old address in the */sys/tcp/hostmap/local.txt* file's **GATEWAY** entry with the new address.

3. If you have more than one administrative node, make sure you change all copies of the */sys/tcp/hostmap/local.txt* file.

Task 4: Run makehost.sh

Run the **/sys/tcp/hostmap/makehost.sh** Shell script on the TCP/IP administrative node. This script converts the local.txt file into a format that TCP/IP software can use. If you have more than one TCP/IP administrative node, you must run the Shell script on each node. To run the script, enter the following command in a DOMAIN or DOMAIN/IX shell:

```
$ /sys/tcp/hostmap/makehost.sh
```

Task 5: Run /sys/tcp/maphost

You must clear the old address from the hosts' internal address mapping tables. To do so, you must run the following command on *each* TCP/IP host and gateway:

```
$ /sys/tcp/maphost -c
```

Task 6: Update the Remote Mapping Tables

Follow the procedures required to change the host or gateway's address in the mapping information tables maintained by your foreign hosts and gateways.

Task 7: Restart tcp_server

Enter the following command to restart **tcp_server**:

Command: **cps /sys/tcp/tcp_server -n tcp_server**

To create **tcp_server** on a remote node, type:

$ crp -on //node_name -cps /sys/tcp/tcp_server -n tcp_server

END OF PROCEDURE 6-2.

6.3.5. Subdividing an Internet into Subnets

You can change your existing TCP/IP configuration to support subnet numbers (this is useful if you have an internet containing many networks). Rather than maintaining a separate TCP/IP network number for each network within your internet, you can assign a network number to your entire internet and subnet numbers to each network within the internet. For details on creating subnets, see Chapter 2, "Selecting Internet Addresses."

Note, though, to incorporate subnets within an existing TCP/IP network, you have to change the Internet addresses for *each* host within the internet. Procedure 6-3 describes the steps you must take.

Task 1: Stop tcp_server

Stop **tcp_server** by entering the following DOMAIN or DOMAIN/IX commands. If you're running DOMAIN, type the following:

```
$ sigp tcp_server
```

If you're running DOMAIN/IX, type the following to list all the processes:

```
% ps ax
```

Locate the number of the process corresponding to the **tcp_server** process and type:

```
% kill process_number
```

Task 2: Decide on Type A, B, or C Internet address format

To implement subnets, you might want to change your current Internet address format. The Type B address format allows you to specify 255 subnets and 254 hosts. However, the Type C address format limits you to 15 subnets and 14 hosts.

Task 3: Decide on the network number for your network

If you currently have several network numbers in your internet, decide on a single network number to represent the entire internet.

Task 4: Select subnet numbers and host numbers

When using subnet numbers, you must subdivide the host portion of your Internet address into a subnet and host number. Select a unique subnet number for each *network* within your internet. Then select a unqiue host number for each *host* within each network. Note that subnets (or networks) within the internet share the same network number, while hosts within each network share the same subnet number.

Task 5: Edit your /sys/tcp/hostmap/local.txt file

Edit the */sys/tcp/hostmap/local.txt* file to incorporate the new network, subnet, and host numbers.

Task 6: Edit your /sys/node_data[.node_id]/networks file

Edit the */sys/node_data[.node_id]/networks* file for *each* host to supply the changed Internet address and a subnet mask. The subnet mask has the following format:

```
internet_address on physical_interface_symbol ; subnet_mask W.X.Y.Z [ ; comment ]
```

where **W, X, Y, Z** can contain either a one (255) to denote the network or subnet field, or a zero (0) to denote the host field.

For example, the following is a one-byte subnet mask for a Type B address where the first two bytes represent the network number, and the last byte represents the host number.

```
129.9.9.7; mask 255.255.255.0
```

Task 7: Run makehost.sh

Run the **/sys/tcp/hostmap/makehost.sh** Shell script on each TCP/IP administrative node. This script converts the */sys/tcp/hostmap/local.txt* file into a format that TCP/IP can use. To run the script, enter the following command in a DOMAIN or DOMAIN/IX shell:

```
$ /sys/tcp/hostmap/makehost.sh
```

Task 8: Run /sys/tcp/maphost

You must clear the old address from the hosts' internal address mapping tables. To do so, you must run the following command on *each* TCP/IP host and gateway:

```
$ /sys/tcp/maphost -c
```

Task 9: Update the Remote Mapping Tables

Follow the procedures required to change the host or gateway's address in the mapping information tables maintained by your foreign hosts and gateways.

Task 10: Restart tcp_server

Enter the following command to restart **tcp_server**:

```
Command:  cps /sys/tcp/tcp_server -n tcp_server
```

To create **tcp_server** on a remote node, type:

```
$ crp -on //node_name  -cps /sys/tcp/tcp_server -n tcp_server
```

END OF PROCEDURE 6-3.

6.3.6. Getting the Official hosts.txt File from the NIC

The Network Information Center (NIC)maintains a master *hosts.txt* file. This file contains the names of all networks, gateways, and addresses on the ARPANET as well as on several other networks that conform to the DARPA Internet standard. If your network is connected to any of these networks, you should occasionally update your local */sys/tcp/hostmap/hosts.txt* file from the NIC master by following either Procedure 6-4 or Procedure 6-5. Use Procedure 6-4 if the *hosts.txt* file resides on an administrative node running the AEGIS operating system. Use Procedure 6-5 if the files resides on an administrative node running DOMAIN/IX.

PROCEDURE 6-4. Updating /sys/tcp/hostmap/hosts.txt on a DOMAIN Node

Task 1: Copy the File from NIC

Use the FTP utility to copy the hosts.txt file from the Network Information Center (NIC) to the */sys/tcp/hostmap/hosts.txt* file on the TCP/IP administrative node. Use the following steps to copy the file:

1. Run the **ftp** command with the host name **SRI-NIC**, for example:

```
$ ftp SRI-NIC
TCP trying 10.0.0.73
connections established
```

2. Login as user **ANONYMOUS** with password **GUEST**, for example:

```
> log ANONYMOUS GUEST
331 Password required for anonymous
230 User anonymous logged in
```

3. Use the **get** or **retrieve** command to copy the file from *[SRI-NIC]<NETINFO>RFC810.TXT* on the remote host to /sys/tcp/hostmap/hosts.txt on the TCP/IP administrative node. For example, enter:

```
> get [SRI-NIC]<NETINFO>RFC810.TXT  /sys/tcp/hostmap/hosts.txt
```

4. When the transfer is complete, use the **quit** or **bye** command to log off the SRI-NIC host and exit FTP. For example:

```
> quit
221 Goodbye.
```

Task 2: Run makehost.sh

Run the **/sys/tcp/hostmap/makehost.sh** Shell script. This script converts the *local.txt* file into a format that TCP/IP software can use. To run the script, enter the following command in a DOMAIN Shell:

```
$ /sys/tcp/hostmap/makehost.sh
```

If you have more than one TCP/IP administrative node, copy */sys/hostmap/local.txt* from the node you have updated to each additional TCP/IP administrative node and run **makehost.sh** on the other administrative nodes. You can run **makehost.sh** on the other nodes using the DOMAIN **/com/crp** command as follows:

```
$ crp -on TCP/IP-administrative-node-specifier -me
Connected to node nnnn "//node-name"
$ //node_name/sys/tcp/hostmap/makehost.sh
```

END OF PROCEDURE 6-4.

PROCEDURE 6-5. Updating /sys/tcp/hostmap/hosts.txt on a DOMAIN/IX Node

Task 1: Copy the File from NIC

Use the **gettable**(8) program to copy the hosts.txt file from the Network Information Center (NIC) to the */sys/tcp/hostmap/hosts.txt* file on the TCP/IP administrative node. (See the *DOMAIN/IX Programmer's Reference for BSD4.2* manual for a reference description of the **gettable** utility.) Use the following steps to copy the file:

1. Set your working directory to */sys/tcp/hostmap*:

   % cd /sys/tcp/hostmap

2. Run **/etc/gettable** with a host name **SRI-NIC**:

   % /etc/gettable SRI-NIC

Task 2: Run makehost.sh

Run the **/sys/tcp/hostmap/makehost.sh** Shell script. This script converts the hosts.txt file into a format that TCP/IP software can use. To run the script, enter the following command in a DOMAIN/IX shell:

% /sys/tcp/hostmap/makehost.sh

If you have more than one TCP/IP administrative node, copy */sys/tcp/hostmap/hosts.txt* from the node you have updated to each additional TCP/IP administrative node and run **makehost.sh** on each node. You can run **makehost.sh** on each remote node from the node you're at by using the DOMAIN **/com/crp** command as follows:

```
$ /com/crp -on TCP/IP-administrative-node-specifier -me
Connected to node nnnn "//node-name"
$ //node_name/sys/tcp/hostmap/makehost.sh
```

END OF PROCEDURE 6-5.

6.4. Maintaining Configuration Files for DOMAIN/IX BSD4.2 TCP/IP

Managing TCP/IP configuration files on a DOMAIN/IX BSD4.2 network or internet is somewhat simpler than managing files for DOMAIN TCP/IP because you don't have to maintain DARPA Internet address mapping files. The following sections describe procedures for maintaining TCP/IP files on the BSD4.2 network. They cover the following procedures:

- Adding hosts to a network
- Removing hosts or gateways from the network
- Changing the name of a host or gateway node

6.4.1. Adding Hosts or Gateways to the Network

To add a host or gateway to an existing DOMAIN/IX BSD4.2 TCP/IP network, follow Procedure 5-4 in Chapter 5, "Configuring TCP/IP."

6.4.2. Removing a TCP/IP Host or Gateway

You must remove a TCP/IP host or gateway from the appropriate information files when you either physically remove the node from its network, or when you stop using TCP/IP on the node.

If you are removing a DOMAIN/IX BSD4.2 node from a DOMAIN network or internet, delete the node's entry in each administrative node's */etc/hosts* file.

Note that you must stop the **tcp_server** before changing any TCP/IP information files and restart the server after completing the changes.

6.4.3. Changing a Host or Gateway Name of a DOMAIN/IX BSD4.2 Node

Use Procedure 6–6 to change the host or gateway name of a node that is running DOMAIN/IX BSD4.2 on a DOMAIN network or internet.

PROCEDURE 6-6. Changing a DOMAIN/IX BSD4.2 TCP/IP Host's Name

Task 1: Stop tcp_server

Stop **tcp_server** by entering the following DOMAIN or DOMAIN/IX commands. If you're running DOMAIN, type the following:

```
$ sigp tcp_server
```

If you're running DOMAIN/IX, type the following to list all the processes:

```
% ps ax
```

Locate the number of the process corresponding to the **tcp_server** process and type:

```
% kill process_number
```

Task 2: Edit the thishost File

Change the name in the node's */sys/node_data[.nodeid]/thishost* file to the new name.

Task 3: Edit /etc/hosts

Change the host's name in the */etc/hosts* file. If you have more than one administrative node, you must correct the host name in each copy of the */etc/hosts* file on the network.

Task 4: Restart tcp_server

Enter the following command to restart **tcp_server**:

```
Command:  cps /sys/tcp/tcp_server -n tcp_server
```

To create **tcp_server** on a remote node, type:

```
$ crp -on //node_name  -cps /sys/tcp/tcp_server -n tcp_server
```

END OF PROCEDURE 6-6.

6.4.4. Changing DOMAIN/IX BSD4.2 Internet Addresses

Use Procedure 6-7 to change the Internet address of a node running DOMAIN/IX BSD4.2 on a DOMAIN network or internet.

PROCEDURE 6-7. Changing a DOMAIN/IX BSD4.2 TCP/IP Host Internet Address

Task 1: Stop tcp_server

Stop **tcp_server** by entering the following DOMAIN or DOMAIN/IX commands. If you're running DOMAIN, type the following:

```
$ sigp tcp_server
```

If you're running DOMAIN/IX, type the following to list all the processes:

```
% ps ax
```

Locate the number of the process corresponding to the **tcp_server** process and type:

```
% kill process_number
```

Task 2: Edit /sys/node_data/networks

Change the address in the */sys/node_data[.nodeid]/networks* file to the node's new address.

Task 3: Edit /etc/hosts

Change the host's address in the */etc/hosts* file. If you have more than one administrative node, you must correct the host address in each copy of the */etc/hosts* file on the network.

Task 4: Run /sys/tcp/maphost

You must clear the old address from the host's internal address mapping tables. To do so, you must run the following command on each TCP/IP host and gateway.

```
$ /sys/tcp/maphost -c
```

Task 5: Restart tcp_server

Enter the following command to restart **tcp_server**:

```
Command:  cps /sys/tcp/tcp_server -n tcp_server
```

To create **tcp_server** on a remote node, type:

```
$ crp -on //node_name  -cps /sys/tcp/tcp_server -n tcp_server
```

END OF PROCEDURE 6-7.

6.5. Maintaining Internal Tables

After you've configured your network and TCP/IP is running, you have to make sure TCP/IP uses the most current host name and addressing information. The procedures in this chapter describe how to edit the appropriate configuration files when updating TCP/IP. In addition, TCP/IP software continuously updates its *internal* routing, and address mapping, and interface files. The **tcp_server** updates routing, address mapping and network interface information for each host. Meanwhile, the **rip_server** and **routed** daemon updates routing information on gateways to foreign networks.

In most cases, the processes keep these tables current by running certain TCP/IP utilities when **tcp_server** initializes. Table 6-1 lists these utilities. Following Table 6-1, sections 6.7.1., "Internal Routing Table" ; 6.7.2., "Address Mapping Files"; and 6.7.3. "The Physical Interface Table" describe the internal tables in greater detail.

You can run the utilities yourself when you change the tables manually. However, we recommend that you stop the **tcp_server** before changing the tables so that, when you restart the server, **tcp_server** automatically runs the utilities to update the tables.

Table 6-1. TCP/IP Utilities for Maintaining Internal Tables

Utility	Server or Daemon	Description
/sys/tcp/makegate	**tcp_server**	Initializes the gateways table after changing the order of gateways to redirect messages. The default input file is */sys/tcp/gateways*. The gateways table lists available gateways and the networks to which this host or gateway connects. Use the **-c** option to clear old associations.
/sys/tcp/maphost	**tcp_server**	Initializes the address mapping table to load addresses of remote hosts that don't follow Address Resolution Protocol (ARP) protocol. The default input file is */sys/tcp/host_addr*. Use the **-c** option to clear old associations.
/sys/tcp/tcpinit	**tcp_server**	Initializes the physical interface table after updating it. The default input file is */sys/node_data[.nodeid]/networks*. This table relates the host's physical interfaces to their corresponding physical networks.
/sys/tcp/setroute	**rip_server**	Changes information in network routing table on gateways that run the DOMAIN operating system. Any changes that you make using **setroute** remain in effect until the **tcp_server** stops running on the node or until you use **setroute** again to change them.
route(8C)	**routed** (daemon)	Changes DOMAIN/IX network routing tables, which provide information about routes to remote destinations. Any changes that you make using **route** remain in effect until the **tcp_server** stops running on the node or until you use **route** again to change them.

6.5.1. Maintaining the Internal Routing Table

The TCP/IP routing (or gateways) table keeps a list of accessible destination addresses and which gateways to use from your network to reach each destination network. The table also indicates whether the gateway is a prime gateway or a "dumb" gateway. A **prime** (dynamic) gateway exchanges routing information with other prime gateways through a routing protocol such as the Routing Information Protocol (RIP). A **dumb** gateway has static routing tables. DOMAIN gateways are prime gateways.

The utilities, **makegate**, **setroute**, and **route** help you maintain the internal routing table. **makegate** maintains the routing table for each host, while **setroute** and **route** maintain the routing tables on each gateway.

The Routing Table on Gateways

The internal routing table on gateways lists *all* accessible destination networks. It indicates the *next* gateway in the route to each destination.

TCP/IP continuously updates the gateway's routing table with information received over the networks to which the gateway is connected. The gateway server (either **rip_server** or **routed**) updates the table with information, and, in turn, broadcasts the updated tables over the networks. In addition, the servers purge old information if it is not marked as static. This technique ensures that all gateways have the most current routing tables.

To maintain the TCP/IP routing tables, the gateway servers follow the Routing Information Protocol (RIP). Both DOMAIN and DOMAIN/IX BSD4.2 TCP/IP conform to the BSD4.2 RIP specifications. RIP enables gateways to exchange Internet routing information. It defines the information that the gateways broadcast, when to broadcast, and the packet format. The protocol also defines the procedures and timeouts used to update the routing tables and delete out-of-date entries. While TCP/IP hosts can listen to RIP messages, they are not required to run a RIP process.

Because some gateways on a TCP/IP network may not support the RIP protocol, we provide a method of putting information permanently in a gateway's routing table. On DOMAIN gateways, you can use the **/sys/tcp/setroute** program. On BSD4.2 gateways, you can edit the */etc/gateways* table to include the information when you configure TCP/IP, or you can run the **/etc/route**(8) program. We describe the */etc/gateways* format in Chapter 3, "Editing TCP/IP Files." Appendix B, "TCP/IP Reference" describes **setroute**, and **route** is documented in the *DOMAIN/IX Programmer's Reference for BSD4.2* manual.

The Routing Table on Hosts

The internal routing table for hosts lists only the networks that are listed in */sys/tcp/gateways*. Usually, this includes only networks that are directly connected to gateway nodes on the DOMAIN network to which the host is attached. The host's routing table does not usually include any information about further destinations.

Like the gateway's routing table which gets updated dynamically, the host's routing table gets updated dynamically. However, the host's table is updated only when the host doesn't have a gateway entry that it should have. That is, a smart gateway keeps track of routing, and if it must redirect a packet to another smart gateway across the *local* network, it enters that smart gateway in the host's routing table.

You can also update the host routing table to add entries to dumb gateways that cannot perform dynamic routing. To do so, edit the */sys/tcp/gateways* file and run the **/sys/tcp/makegate** program. Normally, you'd do this by shutting down the **tcp_server**, the server automatically runs **makegate** when it starts executing.

As with the gateway routing tables, you can provide information about gateways on a TCP/IP network that may not support the RIP protocol. On DOMAIN gateways, you can use the **/sys/tcp/setroute** program. On BSD4.2 gateways, you can edit the */etc/gateways* table to include the information when you configure TCP/IP, or you can run the **/etc/route**(8) program. We describe the */etc/gateways* format in Chapter 4, "Editing TCP/IP Files." Appendix B describes **setroute**, and **route** is documented in the *DOMAIN/IX Programmer's Reference for BSD4.2* manual.

6.5.2. Address Mapping Files

TCP/IP uses an internal address mapping table to convert Internet addresses and local network addresses. This table is maintained dynamically on all nodes, and is updated by using the Address Resolution Protocol (ARP).

DOMAIN TCP/IP uses ARP to update and maintain the internal address mapping tables that relate Internet and local network addresses. Whenever a host cannot find the local address that corresponds to the Internet address of a host or gateway that is on its local network, it uses ARP to get the address and add it to the table. Similarly, it uses ARP to update its address mapping tables with local addresses on both networks that the gateway connects.

If remote hosts don't follow the ARP protocol, the **maphost** utility loads the hosts in the address mapping table from the */sys/tcp/host_addr* file. Information that you enter into the table this way remains until the **tcp_server** process stops executing. Chapter 3, "Editing TCP/IP Files," describes the */sys/tcp/host_addr* file and its format.

The address mapping table maps addresses that are on the host or gateway's local networks. Therefore, hosts use the table to map between Internet addresses and local DOMAIN addresses. Gateways map addresses between Internet addresses and local addresses for both networks that they connect.

A gateway between two DOMAIN networks maps addresses for both the local DOMAIN network and the secondary network. However, routing nodes that contain an ETHERNET controller can only map addresses of other DOMAIN routing nodes. This gateway between two DOMAIN networks cannot map addresses of any foreign gateways or hosts.

For more information on the ARP protocol , see the Network Information Center (NIC) publication RFC 826 *An ETHERNET Address Resolution Protocol.*

6.5.3. The Physical Interface Table

In addition to the routing and address mapping tables, the TCP/IP software maintains a physical level interface table that is used in routing each packet. This table associates the host or gateway's networks with the physical level interface for each network. Essentially, it tells TCP/IP which physical network to use for a given network number. If a physical interface is not working, it is marked as being in error in the table, so that TCP/IP can either try another interface or not send the message. To initialize the physical interface table, the **/sys/tcp/tcpinit** program uses the */sys/node_data*[.nodeid]*/networks* file.

Chapter 7

Troubleshooting TCP/IP

This chapter describes how to troubleshoot problems that may occur with DOMAIN and DOMAIN/IX BSD4.2 TCP/IP. Your major concern in troubleshooting problems with a TCP/IP system is to determine which node is causing the problem.

In general, when determining the cause of your problem, you will rely on the error messages that you receive while running the TCP/IP software. Appendix C, "Error Messages" lists error messages that TCP/IP generates and indicates possible causes.

We also provide you with other tools to aid you in troubleshooting. These tools include the **tcpstat**, **tcpreset**, and **maphost** programs. The **tcpstat** program reports network status, the **tcpreset** program removes broken network connections, and the **maphost** program clears or updates the **tcp_server** internal address mapping table.

If you cannot fix the problem, and you determine that the cause of the problem is in DOMAIN TCP/IP or in the network controller hardware, you should call your service representative and report the problem.

This chapter contains the following sections:

- Section 7.1., "Locating the Cause of the Problem" describes general guidelines that you should follow to locate the component causing the problem.
- Section 7.2., "Using tcp_server" provides instructions for running the **tcp_server** command to debug your TCP/IP software.
- Section 7.3., "Using DOMAIN TCP/IP Utilities" provides instructions for running the DOMAIN utilities **tcpstat**, **tcpreset**, and **maphost**.

7.1. Locating the Component Causing the Problem

Your first step in troubleshooting problems with TCP/IP is to locate the component of your TCP/IP system that is causing the problem. The problem could either be hardware- or software-related.

7.1.1. Checking the Hardware Controller

If you believe that the hardware controller may be the cause of your problem, use the appropriate diagnostic. For example, use the **ether_diag** diagnostic to check the COM-ETH controller. For more information, see the installation manual for your controller.

7.1.2. Checking the TCP/IP Software

When you are troubleshooting, we suggest that you begin by examining the TCP/IP software on host and the gateway nodes. Try the following:

1. Use the DOMAIN **pst** command or the DOMAIN/IX **ps**(1) command (with the **-ax** option) to check that the **tcp_server** is running.

2. Make sure your administrative node is running. To find out which administrative node you're linked to, use the DOMAIN **ld -ll -lt** command or the DOMAIN/IX **ls -l** command on your */sys/tcp* directory. Your administrative node is the node which */sys/tcp/hosts.hst* references.

3. To monitor TCP/IP activity, use **tcpstat** as described in Section 7.3.1., "Using tcpstat." Or, if you are running DOMAIN/IX BSD4.2, use the **netstat**(1) command as described in *DOMAIN/IX Command Reference for BSD4.2*.

4. Run the **tcp_server** in a window, or use **tcp_server** with an option. See Section 7.2., "Using tcp_server" for more information.

5. If you find errors that indicate the remote computer is not responding, check that the remote host is functioning and that all required servers are running. Table 7-1 gives examples of such errors, and possible reasons; Appendix C, "Error Messages" lists additional error messages.

6. Run TCP/IP using the **tcp_server** internal software loopback, described in Section 7.2.5.,"Using the Software Loopback." This technique will help you determine if the problem is located within your host's software.

7. Try to establish a connection to another host node on the *same* physical network. Use the DOMAIN **crp** command to make sure the host is running the appropriate TCP/IP servers or daemons. Once you establish a connection, use **tcpstat** or **netstat** to monitor TCP/IP activity.

8. Try to establish a connection to a host across a gateway to another physical network. Use **tcpstat -g** or **netstat -r** to get the name of a gateway. Once you establish a connection, use **tcpstat** or **netstat** to monitor TCP/IP activity.

Table 7-1. Common Error Messages from Remote Hosts

Message	Possible Reasons
Destination not responding	Remote host is probably down or not running TCP/IP, or its routing table does not include the DOMAIN gateway.
Destination unreachable	Gateway is probably down or not running TCP/IP. Local gateway table is not installed (to check, use **tcpstat -g** or **netstat -r**).
Destination refused	Remote host does not provide the service you're attempting to use (such as **ftp** or **telnet**).

7.2. Using tcp_server

You can use the the **tcp_server** in several ways when you debug TCP/IP. You can:

- Run it in a window.
- Run it with the debug option.
- Start it without initializing the internal tables.
- Determine the TCP/IP software version you are using.

The following sections describe these uses.

7.2.1. Running tcp_server in a Window

The **tcp_server** produces messages when it initializes and when it encounters certain errors. You can observe these messages by running the process in a window as follows:

1. Stop the **tcp_server** and any other TCP/IP servers or daemons if they are running, by using the DOMAIN Shell **sigp** or the DOMAIN/IX **kill**(1) command. You might also want to stop other servers that are running on the node such as **telnet_server**, **ftp_server**, or the **inetd** daemon.

2. Restart **tcp_server** in a window in which a Shell process is running by entering the following pathname at the Shell prompt:

   /sys/tcp/tcp_server

   Running **tcp_server** in a window allows you to monitor the actions of the node.

3. After running **tcp_server** for a while, check the transcript pad for error messages. Any such messages should help you locate the cause of the problem.

7.2.2. Running tcp_server with the Debug Option

The **tcp_server** can also display debugging information such as flow of control and indications of all messages received, including message size and the TCP header. To display this information, run the **tcp_server** in a window with the debug option by entering the following line at the Shell prompt:

/sys/tcp/tcp_server -debug [bitmask]

You can get additional debug information by supplying optional values with the **-debug** option. The values are hexadecimal values that correspond to a 16-bit mask. Table 7-2 lists the debug information you get with each bit.

Table 7–2. Getting Additional Debug Information

Bit Value	Debug Information
0001 (default)	General information
0002	IP level information
0004	ARP information
0008	TCP information
0010	Data in TCP packets
0020	UDP information
0200	Broadcasts
1000	TCP finite state machine information
2000	Device level information
4000	Additional detail at any level

To specify additional information, specify the bit values corresponding to the information you need. For example, to specify TCP (0008) and IP (0002) information, you add the bits 0002 and 0008 to get 000a. Specify the following hexadecimal value on the command line as follows:

```
/sys/tcp/tcp_server -debug 000a
```

To specify TCP, IP and device level (2000) information, add the bits 0002, 0008, and 2000 to get 200a. So, you specify the following hexadecimal value on the command line:

```
/sys/tcp/tcp_server -debug 200a
```

At times, you might want to get all the available debug information *except* for one certain type — such as broadcast information. (You might want to suppress broadcast information when **rip_server** or **routed** are running on the local network because they generate many broadcasts.) To get all the available debug information except broadcast information, supply the following hexadecimal value on the command line:

```
/sys/tcp/tcp_server -debug f0ff
```

7.2.3. Running tcp_server without Initializing Internal Tables

The **tcp_server** usually initializes the TCP/IP internal routing and physical interface tables from the default files */sys/tcp/gateways* and */sys/node_data*[.nodeid]*/networks* when it starts running. However, for debugging purposes, you might not want the server to start these tables automatically. You can start **tcp_server** without initializing the tables by entering the following command:

```
/sys/tcp/tcp_server -n
```

You can then use any of the following commands to initialize the tables.

To initialize the physical interface table, type:

```
/sys/tcp/tcpinit
```

To initialize the internal routing table, type:

```
/sys/tcp/makegate
```

Appendix B, "TCP/IP Reference" describes these programs in detail.

7.2.4. Determining the tcp_server Software Version

You can determine the date the TCP/IP software you are running was built without stopping an existing **tcp_server** process from running. You do this by specifying the **tcp_server** with the **-version** option in a window. The **tcp_server** prints the version information and exits.

For example:

```
$ /sys/tcp/tcp_server -version
Apollo TCP/IP server Version 3.0, Fri Oct 17 12:15:53 EDT 1986
```

7.2.5. Using the Software Loopback

The **tcp_server** provides a software loopback interface, which the server configures automatically when it initializes. You access the interface by sending a message to Internet address 127.0.0.1.

By convention, address 127.0.0.1 is assigned the host name **localhost.** If you include this host in your **local.txt** file when you configure TCP/IP, you can then use the name **localhost,** rather than the address, to access the software loopback. For example, you can issue a **telnet connect** command to **localhost.**

Messages that you send to the software loopback interface are redirected back to the host within the host's **tcp_server.** The messages never go below the IP (network) protocol layer within the host. Therefore, the software loopback limits and isolates the processes that handle the message. (This does not necessarily mean any increase in speed; in fact, the opposite may be true.)

Sending to the software loopback is equivalent to sending to your own address. As with any other destination, you must have a process to receive the connection. For example, you can make a **telnet** connection to **localhost** only if you run **telnet_server** or **telnetd** on your host.

7.3. Using DOMAIN TCP/IP Utilities

You use the **tcpstat, tcpreset,** and **maphost** programs to troubleshoot and correct problems with your DOMAIN TCP/IP system. The **tcpstat** program reports network status and the **tcpreset** program removes inactive network connections. The **maphost** program allows you to clear or update the internal TCP/IP address mapping tables. You run these programs in a window in which a Shell process is running. The remaining sections describe these utilities.

7.3.1. Using tcpstat

The **tcpstat** command reports network status. This section shows the syntax of the command and each of its options, and gives information about typical uses of **tcpstat** during monitoring or troubleshooting of the TCP/IP configuration. This section refers to tables, such as the gateway or host tables, which are discussed in Chapter 3, "Editing TCP/IP Information Files." The syntax of the **tcpstat** command follows:

```
$ tcpstat [option]
```

Table 7–3 lists and summarizes the options.

Table 7–3. tcpstat Options

Option	Description
-a	Report all information including miscellaneous network connection and buffer management statistics. This is the same as using the **-c**, **-m**, and -s options together.
-c (default)	Report status information for each open connection.
-g	Report information about the internal gateway table.
-h	Report mapping information from the internal host map table.
-i	Report information about the physical network interface(s).
-m	Report buffer management statistics.
-n	When used with one of the other options, show Internet addresses, instead of host or gateway names.
-s	Show miscellaneous network statistics.
-t	Report detailed TCP–specific information for each TCP connection.

The following subsections show the use of these options and describe the values that they report. Appendix B, "TCP/IP Reference" includes a standard reference–format description of the **tcpstat** command.

Using tcpstat -a

tcpstat -a is the equivalent of the **-c**, **-m**, and -s options used together. See the description of each option for details on their reports.

Using tcpstat -c

tcpstat -c (the default) produces a line of information about each open connection. Use this command during ordinary operation to get connection information. The following example shows the default **tcpstat** display.

```
$ tcpstat

MODE  TCP STATE  HOST/ROUTE    FPRT LPRT   TCB      SB   RB    SA    RA STAT IF
T     LISTEN     anyhost       0000 0015 00BE7F1C   0    0  8192  8192 0000
T     ESTAB      apollo-vax    0017 03ff 00BE7E98   0    0  8192  8192 0000 dr0
```

Table 7–4 explains each of the fields.

Table 7–4. Fields of tcpstat –c Option

Field		Description
MODE		Lists the Internet Protocol. T indicates the TCP protocol.
TCP STATE		Shows the state of the connection as follows:
	???	Unknown or invalid state.
	LISTEN	Waiting for a connection request from any remote TCP and port.
	SYN–SENT	Sent a connection request; waiting for a matching connection request.
	SYN–RCVD	Sent and received a connection request; waiting for a confirming connection request.
	L–SYN–RCVD	Received and sent a connection request; waiting for a confirming connection request.
	ESTAB	Connection established; data can be transferred.
	FIN–WAIT1	Waiting for a connection termination request from the remote TCP or for an acknowledgement of the termination request that was sent.
	FIN–WAIT2	Waiting for a connection termination request from the remote TCP.
	CLOSE–WAIT	Waiting for a connection termination request from the local user.
	CLOSING1	Waiting for a connection termination request acknowledgement from the remote TCP.
	CLOSING2	Waiting for an acknowledgement of the connection termination request that was previously sent to the remote TCP.
	TIME–WAIT	Waiting for enough time to pass to be sure the remote TCP received the acknowledgement of its connection termination request.
	CLOSED	Connection has been terminated.
HOST/ROUTE		Lists the name of the remote host to which you have established a connection (HOST) and the name of the gateway (ROUTE). **tcpstat** gets this information from the host tables. If a "connection" is a passive listener (indicated by the LISTEN state), there is no remote host and the identifier **anyhost** appears. **tcpstat** truncates names that don't fit in the display. If you can reach the remote host directly from the local host, only the remote host name appears. If you connect to a host in your local network, there's no connection to a gateway, so you see only the remote host name.
FPRT		Shows the number of the port that is open on the remote host (a *foreign* port).
LPRT		Shows the number of the port that is open on the local host (a *local* port).

Table 7–4. Fields of tcpstat -c Option (Continued)

Field	Description
TCB	Shows a number you can use with the **tcpreset** command to forcibly reset a TCP connection.
SB, RB	Shows the number of bytes of send and receive buffering currently in use.
SA, RA	Shows the maximum number of bytes of send and receive buffering that are available.
STAT	Shows whether the remote host has forcibly reset the connection. If it has not, the number is 0001.
IF	Shows the physical network interface used by the local host.

Using tcpstat -g

tcpstat -g displays the TCP/IP internal gateway table and the local networks it has connected. An example of output follows:

```
$ tcpstat -g

LOCAL_NET       ->    FOREIGN_NET via LOCAL_ADDR      TYPE
apollo-ring           apollo-ether      val           routing
apollo-ring           apol-zeus-ether   shrew         routing
apollo-ring           faceoff           throgs_neck   routing
```

Table 7–5 explains each of the fields.

Table 7–5. Fields of tcpstat -g Option

Field	Description
LOCAL_NET	Shows the name of the source network.
FOREIGN_NET	Shows the name of the destination network.
LOCAL_ADDR	Shows the the name of the gateway node.
TYPE	Shows the gateway type. If this a routing gateway, it displays ROUTING; otherwise, it's blank.

Using tcpstat -h

tcpstat -h displays the TCP/IP internal host address mapping table. This address mapping table associates Internet addresses with local addresses. Each association is an entry in the table. Before TCP/IP can make a connection, the local host's address mapping tables must contain an entry for the destination host. During **tcp_server** operation, the Adress Resolution Protocol (ARP) fills the mapping tables with correspondences as they are required.

If you use **tcpstat -h** after attempting to establish a connection, and no remote host name (or Internet address) appears, the correspondence was not in the address mapping table and the ARP has not been able to obtain the correspondence.

Inability to map will prevent any further progress towards a connection, so you might want to investigate why mapping has not taken place. Perhaps there are network problems on the DOMAIN network, since ARP must broadcast a packet over the network and receive a reply in order to get the correspondence. Also, the remote host must be running **tcp_server** or, for non-DOMAIN hosts, an equivalent TCP listening proccess.

If **tcpstat -h** shows that a mapping exists for the proper remote host, the problem is at a less fundamental level. Use other options to the **tcpstat** command to investigate further.

tcpstat -h can also show if one name is used for two different hosts. If the same name appears twice with different addresses, the name has been incorrectly assigned to two hosts.

The example that follows shows output from **tcpstat -h**.

```
$ tcpstat -h
HOST          STAT  REF RFNM  PCT  MAPADDR
bliss         GM      3    0    0   0.0.6f.fe.0.0
hobbit         M      0    0    0   0.0.84.4b.0.0
```

Table 7-6 explains each of the fields.

Table 7-6. Fields of tcpstat -h Option

Field	Description
HOST	The name of the host for which mapping exists.
STAT	The type of entry in the host table, where **G** is a gateway entry, **M** is a mapping entry, and **T** is a temporary entry.
REF	The reference count, the number of connections using one mapping entry. The ARP address resolution protocol (and the **maphost** utility) create mapping entries when they map Internet to local addresses. Typically, if there are several connections from a host, TCP/IP will attempt to use the same mapping entry. (In our example, there are three connections using the same mapping entry). If the REF count is 0, there are no connections currently using mapping entries.
RFNM	The RFNM count for ARPANET host entries. This statistic is currently not useful for DOMAIN TCP/IP implementations.
PCT	The gateway probe count for gateway entries in the host table. **tcp_server** sends out periodic packets to the gateway to ensure that the gateway will respond. Non-gateway entries won't receive them, so they will always show a zero. If a gateway does not respond to four consecutive probe packets, **tcp_server** will look for another gateway to use.
MAPADDR	The local network address (physical address) of the remote host. In our example, the remote host's physical address is a DOMAIN node ID.

Using tcpstat -i

tcpstat -i shows information about the physical interfaces between the host and the network or networks. Most hosts display two reports — one for the network and one for the software loopback interface. Gateways show a line for each network to which they are attached.

The following example shows a **tcpstat -i** display.

```
$ tcpstat -i

UNIT ADDRESS        STAT    IPKTS   OPKTS RSTS FLSH OERR IERR COLL MASK
dr0  192.9.10.112    AEI   618778      45    1    0    1    0    0  255.255.255.0
lo0  127.0.0.1        AI        0       0    0    0    0    0    0  255.0.0.0
```

NOTE: If there is a problem with a connection and the **tcpstat -i** display shows large numbers of packet transmissions or errors (in the IPKTS, OPKTS, OERR or IERR fields), the problem may actually be in the physical network.

Table 7-7 explains each of the fields.

Table 7-7. Fields of tcpstat -i Option

Field	Description
UNIT	The name of the physical interface on which the host runs. For your TCP/IP implementation, allowable names are DOMAIN (**drn**), ETHERNET (**ethn**), where n is the interface number, and **lo0**, the software loopback interface. The information for this field comes from the */sys/node_data*[.nodeid]*/networks* file that you edit for each host during TCP/IP configuration.
ADDRESS	The Internet address for the host on this interface.
STAT	The status flags for the physical interface. A STAT of **AI** indicates a healthy physical interface. A STAT of **E** indicates that the interface is working, although some errors have occurred. The STAT flags are: **A** for available; **C**, initialization completing; **D**, disabled; **E**, error; **F**, flushing; **G**, global; **I**, initialized; **M**, subnets in use; **W**, waiting for initialization.
IPKTS	The number of input packets received by the interface.
OPKTS	The number of output packets sent by the interface.
RSTS	The number of times the interface has been reset.
FLSH	The number of packets flushed by the interface for lack of buffers. This number should always be 0.
OERR	The number of output errors for the interface.
IERR	The number of input errors for the interface.
COLL	For ETHERNET interfaces, the number of collisions. (If you are not familiar with this concept, refer to your ETHERNET documentation. This is usually not an error condition; rather, it indicates the relative amount of activity on the ETHERNET.)
MASK	The subnet mask in use. Ones (225) denote network and subnet fields; zeros (0's) denote host fields of an Internet address.

Using tcpstat -m

tcpstat -m shows information about buffer pools used for TCP/IP. An example follows:

```
$ tcpstat -m
pool size=     80 bufs=  112
pool size=   1520 bufs=  104
pool size=   9216 bufs=  112
```

Table 7–8 explains each of the values.

Table 7–8. Values of tcpstat -m Option

Value	Description
Pool Size	The size of the buffers in the particular pool, in bytes.
Bufs	The number of buffers currently available (not in use) in the pool.

Using tcpstat -n

tcpstat -n modifies the information reported by other **tcpstat** options. With this option, hosts and gateways are identified by their Internet addresses instead of their ASCII Internet names. You can use this option in the same command with the **-a**, **-c**, **-g**, **-h**, or **-t** option. If you only specify the **-n** option you get the default (**-c**) report.

Using tcpstat -s

tcpstat -s reports numerous statistics about network activity, much like the **netstat** Shell command when used with the **-l** option. You might want to use **tcpstat -s** if you suspect a network problem. Even if there are no network problems, you may see some non-zero numbers in these statistics.

The following example shows output from **tcpstat -s**.

```
$ tcpstat -s
mem drops= 0 net drops=0 ip drops= 7 ip badsums= 0
tcp badsums= 0 tcp rejects= 0 tcp unacked= 0
icmp drops= 21 icmp badsums= 0 src quenches= 0 redirects= 0
icmp echoes= 0 time exceeded= 0 udp badsums= 0 udp drops= 407555
ip reroutes= 39 ip bounces= 39
icmp type 0 sent= 0 recved= 19
icmp type 8 sent= 19 recved= 0
```

Table 7–9 explains each of the values.

Table 7–9. Values of tcpstat -s Option

Value	Description
mem drops	The number of messages dropped due to insufficient buffers.
net drops	The number of local network messages that have been dropped because no connection exists to accept them.
ip drops	The number of IP messages dropped because no connection exists to accept them.
ip badsums	The number of messages dropped due to bad checksums (corrupted packets) in the Internet header.
tcp badsums	The number of messages dropped due to bad checksums (corrupted packets) in the TCP header or data.
tcp rejects	The number of messages that TCP rejected because they do not correspond to a listening TCP connection.
tcp unacked	The number of messages that TCP has not acknowledged.
icmp drops	The number of Internet Control Message Protocol (ICMP) packets dropped because no connection exists to accept them. The IP protocol uses ICMP to send information about problems in reaching destinations. The NIC publication RFC 792 describes ICMP.
icmp badsums	The number of ICMP packets dropped due to bad checksums (corrupted packets) in the Internet header.
icmp echoes	The number of ICMP "echo" messages sent (see RFC 792).
src quenches	The number of ICMP "source quench" messages received (see RFC 792).
redirects	The number of ICMP redirects received from gateways. These tell a host to use another gateway.
time exceeded	The number of ICMP "time-exceeded" messages received (see RFC 792).
udp badsums	The number of User Datagram Protocol (UDP) packets dropped due to bad checksums (corrupted packets) in the UDP header or data. The UDP protocol is used by **rip_server, rwhod, tftp,** and **routed**. The NIC publication RFC 768 describes UDP.
udp drops	The number of UDP packets dropped because no listener exists to accept them.
ip reroutes	The number of IP messages that IP rerouted. Always zero for non-gateways.
ip bounces	The number of ICMP redirects that IP sent. Always zero for non-gateways.
icmp type	The number of ICMP packets of type n sent and received (see RFC 792).

Using tcpstat -t

tcpstat -t produces detailed, TCP-related information about the state of each TCP connection. A sample of output follows. (The output for a single entry would normally appear on one line. We've put it on two lines for printing purposes.)

```
$ tcpstat -t
TCB       HOST      FPRT  LPRT  TCP STATE   RCV_NEXT
1E9A48    apollo    0017  0573  ESTAB       2711B2BD

SND_NEXT   SND_UNA   FLAGS  WIND   XT   RT
2711B1EF   2711B1EF  110C8  8192   2    0
```

Table 7-10 explains each of the fields.

Table 7-10. Fields of tcpstat -t Option

Value	Description
TCB	The address of the transmission control block data structure for this connection. You supply this number to **tcpreset** to reset the connection.
HOST	The host name of the remote (foreign) host for the connection.
FPRT	The number of the remote (foreign) port.
LPRT	The number of the local port.
TCP STATE	The TCP state of the connection. See the explanation of **tcpstat -c**.
RCV_NEXT	Sequence number of the next piece of data expected.
SND_NEXT	Sequence number of the next piece of data to be sent.
SND_UNA	Sequence number (+1) of the last piece of data that the remote host acknowledged.
FLAGS	Flags for the TCP connection.
WIND	Size of the window advertised by the foreign host.
XT	Average round trip time (in seconds) for a packet.
RT	Number of retransmissions for the last segment sent on the connection.

7.3.2. Using tcpreset

The **tcpreset** program forcibly resets a TCP connection. It removes inactive connections that have not been properly closed. You can use this command if the connection is hung in an unexpected state, as indicated by the **tcpstat -c** command. To run the **tcpreset** utility, enter the following command at the Shell prompt.

```
$ /sys/tcp/tcpreset [tcb_address]
```

The tcb_address argument specifies the connection that will be reset. To find the TCB address for a connection, use the **tcpstat** command with the **-t** option.

7.3.3. Using maphost

The **maphost** program can either add entries to the **tcp_server** address mapping table or forcibly clear the table. This ability to clear the table helps in troubleshooting because it removes any incorrect Internet address. That is, it clears any local address associations that may prevent messages from being delivered correctly.

To use the **maphost** utility to clear the address mapping table, enter the following command at the Shell prompt.

```
$ /sys/tcp/maphost -c
```

See the command description in Appendix B, "TCP/IP Reference" for details on other **maphost** options.

Appendix A

How TCP/IP Sends Packets

This appendix describes the way DOMAIN TCP/IP software routes messages between application programs. It does not describe the TCP/IP protocol or the DOMAIN implementation in detail; but it does provide a conceptual overview.

A.1. Sending Packets

TCP/IP data is sent in packets that are routed through the network. Different packets in a single message could theoretically go by different routes to the end destination. Therefore, the routing process is performed for each individual packet.

Figure A–1 shows how a packet is sent between the local host and a remote host. It provides a simplified overview of the steps required to transmit the packet between hosts.

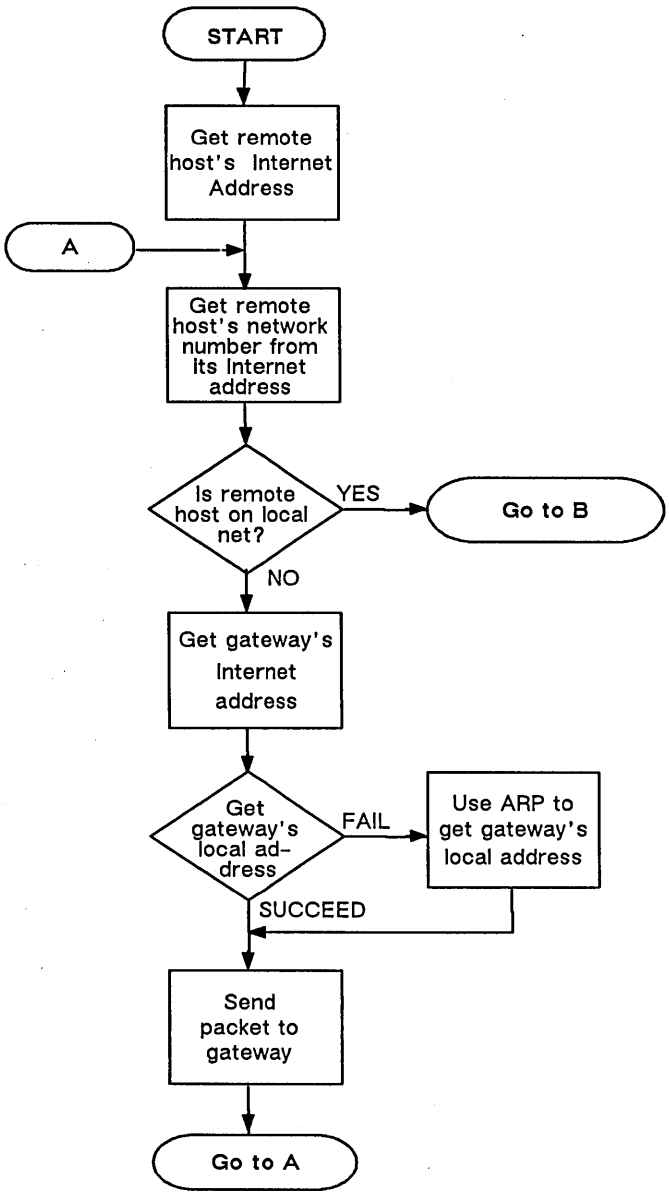

Figure A–1. Sending a Packet

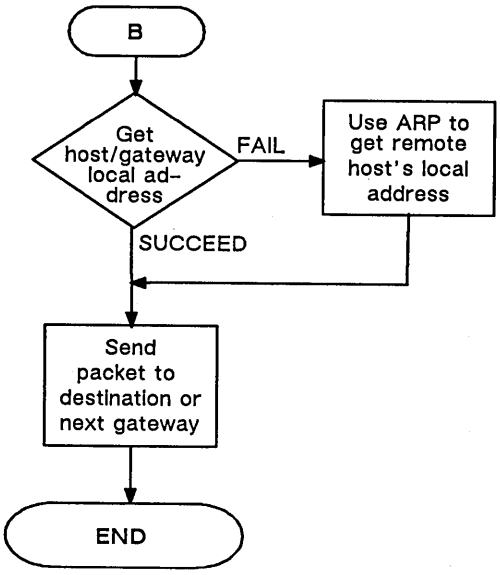

Figure A–1. Sending a Packet (Continued)

Tables A–1 and A–2 describe the process of sending a packet in more detail. They show the name and address mappings that must be performed, and indicate the files or tables that are used. Table A–1 shows how a host transmits a packet. Note that the host performs Stages 1 and 2 only when it opens the connection; it retains this information for the remaining packets. Table A–2 shows how the host delivers the packet to another host on the same network, or how a gateway delivers the packet to another network.

Table A–1. How a Local Host Sends a Packet

Stage	File or Table Used	Host's Action
1	*/sys/tcp/hosts.txt* */etc/hosts* (BSD4.2)	Gets remote host's Internet address from */sys/tcp/hosts.txt* or */etc/hosts* file.
2	Remote host's Internet address	Parses the host's Internet address to get the remote host's network number.
3	Physical Interface table	Checks the physical interface table. If the remote host is on the same network, it goes to Table A–2, Step 4.
4	Internal routing table	Checks the internal routing table for the Internet address of the gateway that's connected to the remote host's network. If there's no gateway listed for the network, it selects the first prime gateway in the routing table.
5	Address mapping table	Gets the gateway's local address from the address mapping table. If it's not listed, it goes to Step 6. Otherwise, it goes to Step 7.

Table A–1. How a Local Host Sends a Packet (Continued)

Stage	File or Table Used	Host's Action
6	Address Resolution Protocol (ARP)	Gets gateway's local address from ARP and updates the address mapping table.
7		Sends the packet to the gateway.

Table A–2. How a Gateway or Host Delivers the Packet

Stage	File or Table Used	Host's Action
1	Physical interface table	Checks the physical interface table. If the remote host is on either of the gateway's networks, it goes to Step 4. Otherwise, it goes to Step 2.
2	Internal routing table	Checks the routing table. If this gateway is on the most efficient path to the remote host, it goes to Step 3. Otherwise, it chooses the best gateway, sends the packet to that gateway, and sends a redirection message with the gateway's address to the sending host. It then goes to Table A–1, Step 5.
3	Internal routing table	Gets the Internet address of the next gateway in route to the remote host.
4	Address mapping table	Checks the address mapping table for the host or next gateway's Internet address. If it finds the address, it goes to Step 5. If not, it goes to Step 6.
5	Address Resolution Protocol (ARP)	Gets the destination's local address from ARP and updates the address mapping table.
6		Sends the packet to the remote host or the next gateway using the destination's local network address. The next gateway repeats this process until it reaches the remote host.

A.2. A Simple Example

Figure A–2 shows an example of two interconnected networks and a gateway that we will use to illustrate sending a packet.

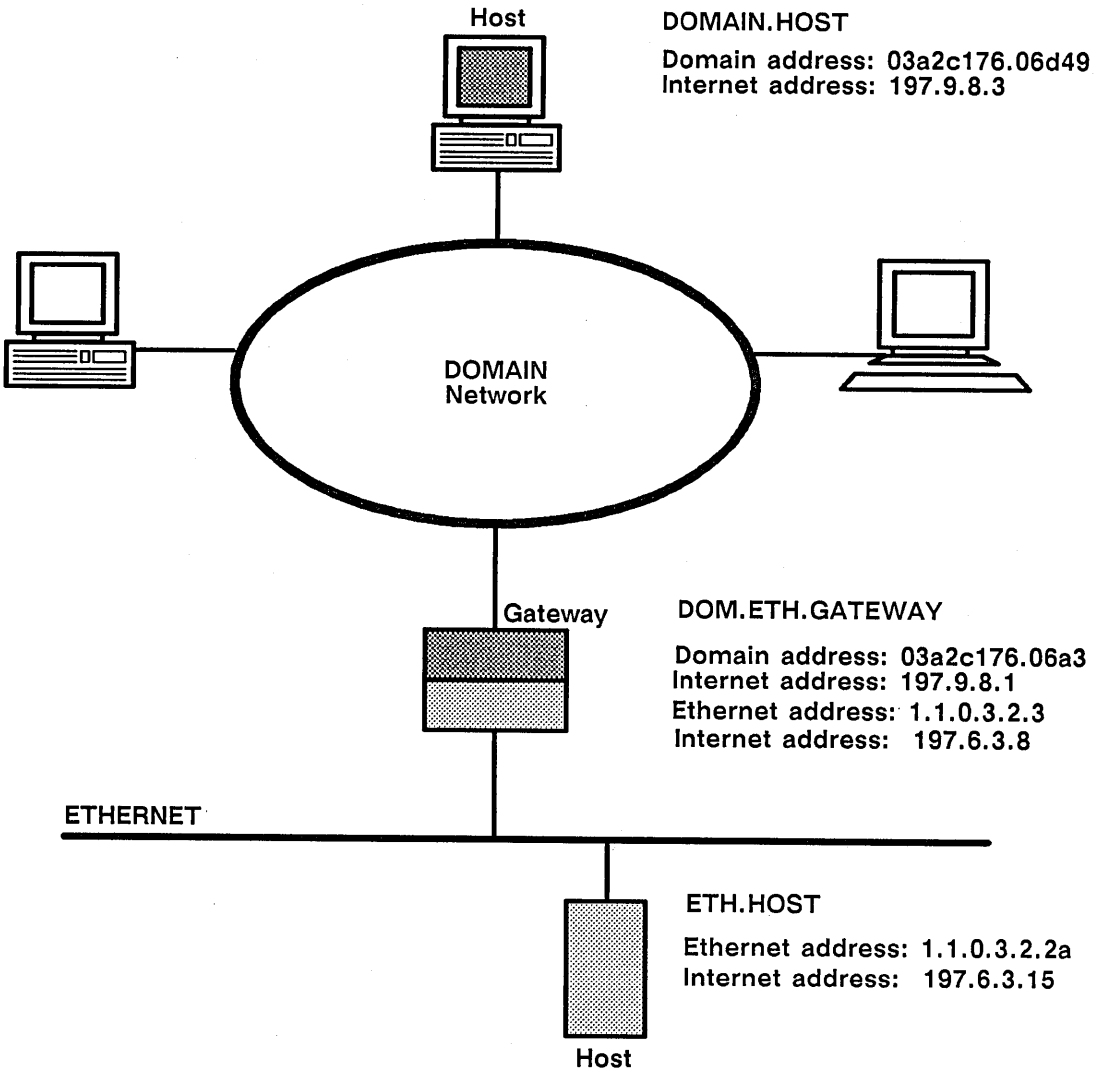

Figure A–2. Network Configuration that Illustrates Sending Packets

In this example, suppose you are working at a DOMAIN node called //DOMAIN.HOST. You make a request for a connection by typing **ftp ETH.HOST**. ETH.HOST is a remote computer that runs in an ETHERNET LAN linked to a DOMAIN network. TCP/IP uses the following procedure:

1. **ftp** uses the */sys/tcp/hosts.hst* (or, if you are using DOMAIN/IX BSD4.2 **ftp** */etc/hosts)* table to associate the name ETH.HOST with its Internet adress 197.6.3.15.

2. TCP/IP parses ETH.HOST's Internet address and strips off the local address to get the remote host's network number, 197.6.3.

3. TCP/IP checks to see whether ETH.HOST's network number matches DOMAIN.HOST's network number, 197.9.8. TCP/IP gets the local network number from the local host's physical interface table.

   Since the network numbers for ETH.HOST and DOMAIN.HOST don't match, TCP/IP knows they are in different networks, so TCP/IP must check the gateway's table. (If they did match, we would skip to Step 7.)

4. TCP/IP searches the internal routing table on DOMAIN.HOST for the Internet address of a gateway to network 197.6.3. In our case, only one gateway, 197.9.8.1, appears in this table, since this DOMAIN network has only one link to another network. (For convenience, we'll use the gateway's name, DOM.ETH.GATEWAY. TCP/IP uses only addresses, not names, during the routing process.)

5. TCP/IP searches its internal address mapping table for the local network address, 03a2c176.06a3 that corresponds to the DOM.ETH.GATEWAY Internet address.

   If DOM.ETH.GATEWAY's local network address is not in the address mapping table, TCP/IP uses ARP to broadcast a message to the TCP/IP nodes on the DOMAIN network. DOM.ETH.GATWAY would respond with its local network address. Once it finds the local network address, TCP/IP would add it to the address mapping table for faster access next time around.

6. TCP/IP sends the packet to DOM.ETH.GATEWAY, using its local network address. DOM.ETH.GATEWAY then transmits the packet to its ETHERNET side. Now the packet is in a local network that recognizes ETH.HOST's local network address, but the gateway's TCP/IP still does not know this address.

7. The gateway's TCP/IP uses its address mapping table to associate ETH.HOST Internet address 197.6.3.15, with its local network address, 1.1.0.3.2.2a. TCP/IP uses the ARP if the required local address is not in the host address mapping tables.

8. Using ETH.HOST's local network address, the packet finally reaches its destination.

Appendix B

TCP/IP Reference

This appendix contains reference descriptions, in alphabetical order, of the following TCP/IP management commands, utilities, and Shell script:

Commands	Utilities	Shell Script
host	**/sys/tcp/makegate**	**/sys/tcp/hostmap/makehost.sh**
net	**/sys/tcp/maphost**	
tcpstat	**/sys/tcp/setroute**	
	/sys/tcp/tcpinit	
	/sys/tcp/tcpreset	

DOMAIN/IX BSD4.2 also includes the following TCP/IP management programs.

gettable(8)

htable(8)

netstat(1)

route(8)

See the *DOMAIN/IX Programmer's Reference for BSD4.2* for detailed descriptions of these programs. Also see the BSD4.2 reference manual for descriptions of all BSD4.2 TCP/IP–related daemons and for the IPC system calls that use TCP/IP.

host — List host address mapping information.

FORMAT

host [hostname|address[,netname|number] ...]

The **host** command displays information from the TCP/IP host address mapping table file */sys/tcp/hosts.hst*. This command displays the following information for each host that you specify:

- Name
- Any aliases
- Internet address
- The host communication protocol type; 1 = TCP
- Supported protocols

If the host has more than one Internet address (that is, if it is also a gateway), the **host** command lists the addresses and protocols for all networks.

ARGUMENTS

hostname (optional)
: The TCP/IP name or alias of the host for which you want the information.

address (optional)
: The Internet address of the host for which you want the information.

  Default if both are omitted: Display information for the host on which you are logged in. That is, for the hostname listed in the node's */sys/node_data*[.nodeid]*/thishost* file.

netname (optional)
: The Internet network name of the network of the host for which you want the information.

netnumber (optional)
: The Internet network number for the network of the host for which you want the information.

  Default if both are omitted: display information for the host on which you are logged in. That is, for the hostname in the node's */sys/node_data*[.nodeid]*/thishost* file.

EXAMPLE

Display the host map information for the host ARAN.

```
$ host aran
aran rodian:194.6.4.2 hostcap=1 tcp/ftp tcp/telnet
```

makegate — Create an internal gateway table.

FORMAT

/sys/tcp/makegate [**-i** infile] [-c] [-w]

The **makegate** command creates the **tcp_server** internal routing, or gateway, table and completely replaces any existing internal gateway table. **makegate** runs automatically whenever the **tcp_server** starts executing. If you are explicitly running **/sys/tcp/tcpinit** and **makegate**, you must run **tcpinit** first.

makegate uses a file containing ASCII network/gateway correspondence entries to initialize the gateway server's internal gateway table. (By default, this is the */sys/tcp/gateways* file.)

makegate verifies each entry in the file for proper syntax and recognizable network/gateway names. Errors in individual entries do not cause the entire command to abort. **makegate** then adds each valid entry to the internal gateway table.

NOTE: **makegate** is located in the */sys/tcp* directory, not in the */com* directory. Therefore, in most cases, you must enter the utility's pathname.

OPTIONS

-i infile

Use the specified infile for the gateway definition information. Each entry in the ASCII input file consists of a line of the following format:

lnet_add, fnet_add : gway [: protocols] [; comment]

where lnet_add and fnet_add are Internet addresses of the gateway on the local and foreign networks, gway is the host name (or address) that is the gateway between the two networks, and protocols is an optional string of transport/service protocols, separated by commas (usually **IP/GW, GW/PRIME** for DOMAIN gateways).

A line can contain a semicolon, after which all following text on the line is ignored.

Default if omitted: */sys/tcp/gateways* is used as the input file.

-c

Specifies that the new gateway input files are checked for proper syntax without updating the gateway table.

-w

Suppresses warning messages generated by **makegate**. **makegate** reports entry syntax errors and unknown gateway addresses and network names.

EXAMPLE

Use the file */sys/tcp/mygates* to supply gateway information.

```
$ /sys/tcp/makegate -i /sys/tcp/mygates
```

In this case */sys/tcp/mygates* looks as follows:

```
137.5.3.1, 156.9.8.3 :aella :ip/gw, gw/dumb
137.5.3.7, 149.3.11.6 :busla :ip/gw, gw/dumb
137,5,3,33, 3.0.0.62 :grettir :ip/gw, gw/prime
```

FORMAT

/sys/tcp/maphost/makehost.sh

The **makehost.sh** Shell script creates several files used by TCP software.

In all cases, this command creates the host name file */sys/tcp/hosts.hst* and gateway file */sys/tcp/gateways*. It completely replaces any existing */sys/tcp/hosts.hst* or */sys/tcp/gateways* files.

If DOMAIN/IX BSD4.2 is installed on your node and the **/etc/htable** program exists, this Shell script also creates the BSD4.2 host file */etc/hosts*, gateways file */etc/gateways*, and networks file */etc/networks*. It completely replaces these tables if they exist. Note that */etc/gateways* is created empty since it's required only in special cases. See the *DOMAIN/IX Programmer's Reference for BSD4.2* for details.

makehost.sh uses two input files: */sys/tcp/hostmap/hosts.txt* and */sys/tcp/hostmap/local.txt*. These files contain Internet name and address information. Their format and contents are described in detail in Chapter 3, "Editing TCP/IP Files."

/sys/tcp/hostmap/hosts.txt is a standard file that the Network Information Center (NIC) maintains. It provides information for all hosts, gateways, and networks on the DARPA Internet. We provide a copy of this file. However, if you will use the DARPA Internet from your DOMAIN nodes, you should use the procedures described in Chapter 6, "Managing TCP/IP," to update this file from the NIC.

If you do *not* want to use a DARPA Internet from your DOMAIN nodes, replace the */sys/tcp/hostmap/hosts.txt* file with an empty file; so the files that **makehost.sh** creates will be considerably smaller. You must have an empty *hosts.txt* file because **makehost.sh** requires that a */sys/tcp/hostmap/hosts.txt* exist.

/sys/tcp/hostmap/local.txt defines your local networks. It should contain information about all networks, gateways, and hosts that you define yourself. Chapter 3, "Editing TCP/IP Files" describes this file and its format.

NOTE: **makehost.sh** is located in the */sys/tcp/hostmap* directory. You must enter the Shell script's pathname.

EXAMPLE

Create the host, gateway, and networks files on a DOMAIN network with nodes that use DOMAIN and DOMAIN/IX. The node on which you run this command, a TCP/IP administrative node, runs DOMAIN/IX BSD4.2. However, the Shell script must be invoked in an AEGIS shell.

```
$ /sys/tcp/hostmap/makehost.sh
Formatting tables
Sorting Tables
Formatting tables (pass 2)
Hashing
input: hosts.tmp1, 1466 lines
output: hosts.hst, 215519 bytes
hash: 1542 bytes, 257 entries, 0 holes
keys: 97942 bytes, 5349 entries (largest 662, longest 37)
data: 116019 bytes, 1466 entries (largest 740)
(file) "hosts.hst" moved.
(file) "gateways" moved.
Updating 4.2bsd host tables
(file) "/etc/hosts" moved.
(file) "/etc/gateways" moved.
(file) "/etc/networks" moved.
(file) "hosts.tmp" deleted.
(file) "hosts.tmp1" deleted.
(file) "nets.tmp" deleted.
(file) "hosts.txt1" deleted.
(file) "nets.txt1" deleted.
```

maphost — Update address mapping tables.

FORMAT

/sys/tcp/maphost [**-i** infile] [**-c**]

The **maphost** command manages the TCP/IP internal address translation tables that convert between a local network and Internet addresses. This command allows you to add entries to the table or clear all entries from the table. You *must* run **maphost** to add entries for non-DOMAIN hosts and gateways that do not use the Address Resolution Protocol (ARP) as specified in RFC 826.

You run the **maphost** command after you add entries to the host address file.

NOTE: **maphost** is located in the */sys/tcp* directory, not in the */com* directory. Therefore, in most cases, you must enter the utility's pathname.

OPTIONS

-i infile

Specifies the file that contains the host address mapping information. Each entry in this ASCII input file consists of a line of the following format:

internet_addr, local_addr

where internet_addr is the host's internet address and local_addr is its local (ETHERNET) network address.

Default if omitted: update the table by adding the entries in */sys/tcp/host_addr*.

-c

Clear the table of all entries. This command clears any invalid entries that might cause addressing errors. The **tcp_server** then uses ARP to update the table when it must make connections.

Default if omitted: do not clear the table.

EXAMPLE

Use the file */sys/tcp/no_arp_hosts* to supply gateway information.

```
$ /sys/tcp/maphost -i /sys/tcp/no_arp_hosts
```

In this case, */sys/tcp/no_arp_hosts* looks as follows:

```
156.9.8.3, 2.7.1.0.3.a4
156.9.8.4, 2.7.1.0.3.a8
156.9.8.9, 2.7.1.0.e.dc
```

net — List network mapping information.

FORMAT

net [netname | netnumber] ...

The **net** command displays network information from the */sys/tcp/hosts.hst* file. This command displays the following information for each network that you specify:

- Name
- Internet network number

ARGUMENTS

netname (optional)	The Internet network name of the network for which you want the information.
netnumber (optional)	The Internet network number of the network for which you want the information.
	Default if both are omitted: Display information for the host on which you are logged in. The network used is the first entry in the file */sys/node_data*[.nodeid]*/networks*.

EXAMPLE

Display the network information for the networks NORWAY and 198.2.4.

```
$ net norway 198.2.4
norway:198.2.3
sweden:198.2.4
```

setroute — Update routing tables.

FORMAT

/sys/tcp/setroute {cmnd dest gate [metric] [**-f** infile] | **-f** infile}

The **setroute** utility updates the TCP/IP internal routing table that is created by the **makegate** utility.

The internal routing table retains any routes that you add with this command until you use **setroute** to delete the route or the **tcp_server** stops executing. Use this command to define routes through gateways that do not use a routing process that conforms to the Routing Information Protocol (RIP).

NOTE: **setroute** is located in the */sys/tcp* directory, not in the */com* directory. Therefore, in most cases, you must enter the utility's pathname.

ARGUMENTS

cmnd (required)
: The update operation; it must be either of the following:

  add Add an entry to the table.

  delete Remove an entry from the table.

dest (required)
: The Internet address of the destination network. This argument is a network number followed by an all-zero host address, for example: 197.2.3.0.

gate (required)
: The Internet address of the next gateway in the route to the destination. That is, the gateway on the network that is connected to the gateway you are updating. This gateway is the first hop in the route.

metric (optional)
: The number of hops. This is the number of changes from network to network, and therefore the number of gateways, between this gateway and the destination.

  Default if omitted: 0. In this case, the destination must be on a network that is connected to this gateway; therefore you should not use the default.

NOTE:
: You can omit all arguments if you specify the **-f** to flush the routing table.

OPTIONS

-f infile
: Flush all entries from the routing table. If you use this with the **add** command, **setroute** flushes the tables before adding the entry. You can also enter this option without specifying any command. When **setroute** executes this option it lists each route as it is deleted.

EXAMPLE

Add a route to the network 203.2.4. There are two gateways between this gateway and the destination network. The next gateway in the route is 201.1.5.6.

```
$  /sys/tcp/setroute add 203.2.4.0 201.1.5.6  2
```

tcpinit — Associcate Internet addresses with physical interface identifiers.

FORMAT

/sys/tcp/tcpinit [**-i** infile] [**-d** [interface]]

The **tcpinit** utility reads a file whose entries associate network addresses with network physical level interface identifiers. For each entry, **tcpinit** assigns the network address to the specified interface and initializes the **tcp_server**'s internal interface tables. **tcpinit** runs automatically when **tcp_server** starts running.

NOTE: **tcpinit** is located in the */sys/tcp* directory, not in the */com* directory. Therefore, in most cases you must enter the utility's pathname.

OPTIONS

-i infile

Use the specified infile for the information on correspondences between decimal Internet addresses and network interfaces. This file must contain lines of the following format:

address **on** interface

where address is the node's Internet address on the specified interface, **on** is a keyword, and interface identifies the physical interface. The interface argument must have one of the following values:

drn identifies the nth interface between the node and a DOMAIN network; n must be 0 for the network and 1 for the gateway node between two DOMAIN networks.

ethn identifies the nth interface between the node and an ETHERNET network; n can be 0, 1, 2 or 3.

lo0 identifies the **tcp_server**'s internal software loopback interface.

Default if omitted: use */sys/node_data*[.nodeid]*/networks* as the input file.

-d [interface]

Disables the specified network interface. This option ends I/O operations and clears queues, resets connections associated with the disabled interfaces, and marks the interface as down.

tcpinit reports entry syntax errors. Fatal errors terminate the process with a non-zero exit status code. Fatal errors only occur when **tcpinit** can initialize *none* of the specified interfaces in a file. As long as **tcpinit** can initialize one interface specified in the file, it is not a fatal error.

See Chapter 3, "Editing TCP/IP Files" for more information on networks files. See Chapter 7, "Troubleshooting TCP/IP" for more information on the **tcpinit** command.

EXAMPLE

Use the file */sys/tcp/mynets* to supply Internet address and physical interface device associations for a TCP/IP gateway between two DOMAIN networks.

```
$ /sys/tcp/tcpinit -i /sys/tcp/mynets
$
```

In this case, the */sys/tcp/mynets* file looks as follows:

```
197.8.3.2 on dr0
197.8.5.45 on dr1
```

tcpreset — Reset a TCP connection.

FORMAT

/sys/tcp/tcpreset tcb_address

The **tcpreset** utility forcibly resets, that is, closes without the normal handshaking, an existing TCP connection. This command enables you to remove inactive network connections that have not been closed and to recover from situations where the connection hangs in an abnormal state.

Your host sends a TCP reset (RST) message to the remote host in response to this command. When the connected host receives this message, it should enter the CLOSED state.

NOTE: tcpreset is located in the */sys/tcp* directory, not in the */com* directory. Therefore, in most cases, you must enter the utility's pathname.

ARGUMENTS

tcb_address (required)	The address of the transmission control block for this connection. Use the default TCPSTAT command to get this address.

EXAMPLE

Reset the TCP connection with a TCB address of 2A710C:

```
$ /sys/tcp/tcpreset 2a710c
```

tcpstat — Display TCP status.

FORMAT

tcpstat [options]

The **tcpstat** command reports network status. See Chapter 7, "Troubleshooting TCP/IP" for details on the reports produced with each option.

OPTIONS

The following options denfine the **tcpstat** command's operation.

Default options are indicated by "(D)."

-a		Report all information including miscellaneous network connection and buffer management statistics. Equivalent to using the **-c**, **-m**, and **-s** options together.
-c	(D)	Report status information for each open connection.
-g		Report information about the internal gateway table.
-h		Report mapping information from the internal host map table.
-i		Report information about the physical network interface(s).
-m		Report buffer management statistics.
-n		Show Internet addresses, instead of host or gateway names.
-s		Show miscellaneous network connection statistics.
-t		Report detailed TCP-specific information for each TCP connection.

EXAMPLE

Display the default TCP status information. The report for each connection takes a single line. We have split the report for printing purposes.

```
$ tcpstat
MODE  TCP STATE  HOST/ROUTE       FPRT  LPRT
T     LISTEN     anyhost          0000  0015
T     ESTAB      apl-vax/apoll    0201  03ff

TCB       SB   RB   SA   RA   STAT   IF
2A710C     0    0    2    6   0000
2A57C4     0    0    2    6   0001   dr0
```

Appendix C

Error Messages

This appendix is a guide to the error messages you may see when attempting to use TCP/IP or any of the BSD4.2 socket IPC facilities. We've tried to include, where appropriate, information about what may be causing the error, and how to fix that condition. We've also tried to indicate whether a program error, an error in configuring TCP/IP, or an anomalous network condition is most likely to be responsible for generating a particular message.

Address already in use

This message is the result of a programming error or an error in configuring TCP/IP. The programming error occurs when an application attempts to reopen a connection on an address before a previous occurrence of that address closed or was timed out. If you've attempted to start the same server twice (on one host), you will also get this message. For example, this error message would occur if you tried to start **ftpd** on a host that already had the **ftp_server** or **ftpd** running.

All network ports in use

No **ptys** (pseudo-tty devices that provide socket connection facilities) are available or all ptys already have **telnet** connections active. First, check */dev* to be sure at least one pty pair exists:

```
% ls /dev/*typ?
/dev/ptyp0 /dev/ptyp1 /dev/ttyp0 /dev/ttyp1
```

You must have at least one pair of ptys (e.g., **ptyp0** and **ttyp0**) in */dev*. If you don't, run **/etc/crpty** to create the ptys. The example below creates two pairs of ptys.

```
% /etc/crpty 2
```

Can't assign requested address

This could be caused by a program error if the program tries to perform an operation, usually a *bind*, on an Internet address that is different from the host's Internet address. It might also be an error in the */sys/node_data*[.nodeid]*/networks* file. Check the */sys/node_data*[.nodeid]*/networks* file to make sure that the network address for this host agrees with the Internet address for this host in the */sys/tcp/hostmap/local.txt* file.

Can't send after socket shutdown

This is the result of a programming error; the program attempted to send data to a socket that the program had already shut down with the **shutdown** call.

Connection refused

You tried to connect to a remote host for a service that didn't have a server running. For example, you may have tried to **telnet** to a host that didn't have **telnet_server** or **telnetd** running. In this instance, the remote host *actively* refused the connection.

Connection reset by peer

This error message generally results from a problem with the network. The connection on the network was broken for some period longer than a timeout set by the remote system. Try to reconnect.

Connection timed out

This error might result from a network problem or a configuration error. The message signifies that you attempted to connect to a machine and received no response. The network connection may be broken, the remote host may not be running TCP/IP, or there may be no gateway entry on the foreign host for the DOMAIN gateway.

Check that either **routed** or **rip_server** is running on the gateway. Also check the foreign host to see if it recognizes your DOMAIN network. If the foreign host runs UNIX, you can use the **netstat -r** command at that host to check the networks that it recognizes. If the DOMAIN network doesn't appear on the listing, use the following command to add your network:

```
% /etc/route add apollo-net gateway-name hop-count
```

where apollo-net is the name or network address of the DOMAIN network, gateway-name is the name or address of the (next) gateway or network between the foreign host and the DOMAIN network, and hop-count is the number of gateways and networks between the foreign host and the DOMAIN gateway.

Destination address required

This error message results from a programming error. Some of the DOMAIN/IX BSD4.2 socket calls require an address, and when this error returns, the address was not supplied in the program.

I/O error

This error message occurs in many cases; it is a catchall for a number of miscellaneous error conditions. The most likely cause of it, however, is that the **tcp_server** is not running, or is running in some undetectably altered state. Stop any TCP/IP proceses that are running on this host, and restart them.

Message too long

This error is caused by a programming error. The program attempted to send a datagram larger than legal size over a UDP socket.

Network dropped connection on reset

The remote host system to which you were connected crashed and rebooted. Reconnect to the remote system.

Network is unreachable

Your **tcp_server** does not have a gateway entry for the network you're trying to reach. Check the physical path to the remote host; then run the program **/sys/tcp/makegate**. If neither of these actions solves the problem, make sure there is a gateway entry for the foreign network in */sys/tcp/hostmap/local.txt*; if not, add the entry and run **/sys/tcp/hostmap/makehost.sh**.

Operation not supported on socket

This message is usually caused by a programming error. You've attempted to perform an operation on a type of socket that doesn't support that operation.

Protocol family not supported; Protocol not available; Protocol not supported.

These three messages are caused by programming errors. Your program has tried to create a socket type other than an Internet socket.

Protocol wrong type for socket

You specified a protocol which does not support the semantics of the socket type you requested in the program. For example, you specified the UDP protocol with type SOCK_STREAM.

Socket is already connected

Your program attempted to connect to an already–connected socket, or a BSD4.2 **sendto** or **sendmsg** request specified a destination other than the connected socket.

Socket is not connected

Your program attempted to send or receive data to or from a socket that was not connected.

Socket operation on non–socket

This message results from a programming error. The program attempted to perform an operation on a file descriptor that was not associated with a socket; the attempted operation can only be performed on a socket.

Socket type not supported

You specified a socket type in your program that isn't supported on this machine.

Unkown host name for your address

This is a configuration problem. There is no entry in the foreign host's host table for your DOMAIN host. On a DOMAIN/IX BSD4.2 node, add the DOMAIN host's name to */etc/hosts*, and be certain that the DOMAIN network address is entered in */etc/networks*. On other hosts, you may have to edit the *local.txt* or equivalent file.

Unknown host specifier

This is a configuration problem. There is no entry in your */sys/tcp/hostmap/local.txt* file for this host. On a DOMAIN node, add the DOMAIN host's name to */sys/tcp/hostmap/local.txt*. On a DOMAIN/IX BSD4.2 node, add the DOMAIN host's name to */etc/hosts*, and be certain that the DOMAIN network address is entered in */etc/networks*.

Unknown service

This message could result if the destination network is down, or if the DOMAIN/IX BSD4.2 **/etc/services** file on either machine has been deleted or corrupted. See the *DOMAIN/IX Programmer's Reference for BSD4.2* for information about **/etc/services**.

Glossary

Address
A series of numbers that uniquely identifies a node on a network. Each network and protocol family has its own address format. See **DOMAIN Address**.

Address Mapping Table
A table that TCP/IP uses to convert **Internet addresses** to and from **local addresses.**

Address Resolution Protocol
(ARP) The protocol used by TCP/IP hosts and gateways to maintain their **address mapping tables.** ARP updates the table whenever a host cannot find the local address that corresponds to the Internet address of a host or gateway on its local network.

Administrative Node
See **TCP/IP Administrative Node** and **DOMAIN/IX Administrative Node.**

Alias
A secondary name for a TCP/IP host. You can use a TCP/IP host alias as if it were the host name.

ARP
See Address Resolution Protocol.

ARPANET
The Department of Defense Advanced Research Projects Agency Network. The ARPANET is one of the largest nationwide networks that uses TCP/IP protocols.

Bridge
A device that physically links two or more networks. For example, a bridge can connect a token ring network to an **ETHERNET** network. In DOMAIN internets, network controllers and their associated software perform bridge functions. See also, **Router** and **Routing Node.**

Configuration
The arrangement of a computer system or network as defined by the nature, number, and chief characteristics of its functional units. More specifically, the term configuration can refer to a hardware configuration or a software configuration.

daemon
A DOMAIN/IX BSD4.2–specific **server** process.

DARPA
The Defense Advanced Research Projects Agency of the U.S. Department of Defense.

DARPA Internet
1. A TCP/IP Internet consisting of all networks listed in the **hosts.txt** file.
2. See Internet, definition 1.

Destination Address
The field in a packet that identifies the intended recipient of the packet.

DOMAIN Address
A DOMAIN node ID is a 20–bit node address, expressed in hexadecimal numbers, for a single DOMAIN network. In some contexts, a 32–bit DOMAIN network number precedes the node ID. DOMAIN network numbers are assigned by Apollo Computer Inc.

DOMAIN/IX Administrative Node
The DOMAIN/IX BSD4.2 node that contains the */etc* directory. Other BSD4.2 nodes gain access to the contents of this directory through soft links.

DOMAIN Network
A network consisting of nodes that use DOMAIN protocols. At present, DOMAIN nodes can use the **DOMAIN token ring network** or the ETHERNET 802.3 media.

DOMAIN Token Ring Network
A **local area network** that uses coaxial cable as its transmission medium and operates at 12 megabits per second.

Dumb Gateway
A gateway that does not perform **routing**. See also **Prime Gateway**.

ETHERNET
A **local area network** that uses coaxial cable as its communications medium and operates at 10 megabits per second. ETHERNET is a registered trademark of the Xerox corporation.

ETHERNET Address
A 48-bit number that identifies a device on an ETHERNET network. ETHERNET addresses are expressed as six hexadecimal octets. ETHERNET addresses are assigned by the Xerox Corporation.

File Transfer Protocol
(FTP) A protocol for transmitting files between host computers. FTP is defined by the **Defense Advanced Research Projects Agency**. FTP uses TCP and IP as underlying protocols.

FTP
See File Transfer Protocol.

ftpd
The **daemon** process that accepts incoming FTP requests on DOMAIN/IX nodes.

ftp_server
The **server** process that accepts incoming FTP requests on AEGIS nodes.

Gateway
A device or set of devices that connects usually unlike networks, such as DOMAIN and ETHERNET, by providing protocol translation. (In contrast, a **bridge** connects two like networks and so requires no protocol information). See also **TCP/IP Gateway**.

Hop
A packet's passage through a **routing node** on its way to its final destination. The number of hops in a route is the same as the number of gateways a packet passes through.

Host
A computer or workstation that communicates over a network. A host can both initiate communications and respond to communications that are addressed to it.

Host Number
The portion of a TCP/IP **Internet address** that uniquely identifies the host on its local network.

hosts.txt File
A file containing information on the networks, gateways, and hosts that make up the **ARPANET** and other DARPA Internets that are supported by the **Network Information Center** (NIC).

inetd
The **daemon** process that listens for incoming requests for programs listed in the */etc/inetd.conf* file. When a request for a certain program listed in this file arrives, **inetd** forks the desired program.

Internet

1. Two or more connected networks that may or may not use the same communication protocol. The device that connects the networks may perform routing and/or gateway functions.
2. A DARPA Internet conforms to a set of protocols defined by the Defense Advanced Research Projects Agency that include the Internet Protocol and Transmission Control Protocol.

Internet Address

1. An address that conforms to the **DARPA-defined Internet protoocol.** A unique, four-byte number that identifies a host or gateway on the Internet, consisting of a **network number** followed by a **host number.** The host number can be further divided into a **subnet number.** Internet addresses are expressed as four decimal numbers, ranging between 0-255 and separated by periods.
2. An address that uniquely identifies a destination on an **internet.**

Internet Gateway

See Gateway.

Internet Protocol

(IP) A protocol defined by the Defense Advanced Research Projects Agency for connecting networks through **gateways.**

IP

See Internet Protocol.

LAN

See Local Area Network.

Local Address

An address that uniquely identifies a destination within a single network.

Local Area Network

(LAN) A communications network linking a number of devices that are located within a relatively short distance, typically less than a mile.

Local Network

The network to which a node is directly attached.

Local Node

The node executing the commands. For example, the processes created by the DOMAIN **crp** command execute on the node specified in the **-on** option. For contrast, see **remote node.**

Network

Transmission media and software that links nodes and peripherals.

Network Address

An Internet address created by appending zeros to the **network number.** For example, 205.3.1.0 is a **Type C** network address.

Network Information Center

(NIC) A centralized information resource managed by SRI International (Menlo Park, CA). The Network Information Center assigns DARPA Internet network numbers, and maintains the master *hosts.txt* table and copies of the DARPA Internet specifications.

Network Number

The component of an **Internet address** on an internet that uniquely identifies the network. See **Internet Address, Network Address.**

NIC

See Network Information Center.

Node ID

The unique 20-bit identifier for a DOMAIN node.

Node Specification

An operating system identifier for a node. A node specification can consist of a **node ID,** the **DOMAIN address, Internet address** or the node name.

Packet

A sequence of binary digits that is transmitted as a unit in a computer network. A packet usually contains control information plus data. Packets are transferred as a single unit over a **packet-switched network.**

Packet-Switched Network

A network that transmits data in the form of packets. Each **packet** is transmitted separately over the network; they are dynamically routed from source to destination. The DARPA Internet is a packet-switched network.

Physical Layer

The lowest communications layer. It provides the mechanical, electrical, functional, and procedural means to provide communications over a physical medium.

Physical Layer Interface

The interface between **TCP/IP** and the **physical layer** software that sends messages over a particular transmission medium. Each physical layer interface identifier indicates a particular network to which the host is attached. For example, **dr0** specifies the **DOMAIN token ring.**

Prime Gateway

A gateway that maintains up-to-date information about routes to destinations on the TCP/IP Internet. A prime gateway uses a protocol such as **RIP** to dynamically maintain its routing tables.

Protocol

A set of rules that governs the procedures used in exchanging information between two communication processes.

Remote Network

A network not directly connected to a node. A node must send packets through a **router** or **gateway** to communicate on a network.

Remote Node

A node other than the node executing commands.

Request for Comment

(RFC) A specification or proposed specification that applies to the **DARPA Internet.** You can obtain copies of any RFC from the **Network Information Center.**

RFC

See Request for Comment.

RIP

See Routing Information Protocol.

rip_server

The **server** process that maintains the **routing table** on DOMAIN gateways.

Route

1. To determine the path by which a packet will reach its destination, when the packet is being transmitted through an internet.
2. The path a packet takes from its source to its destination.

routed

The **daemon** process that maintains the **routing table** on DOMAIN/IX BSD4.2 gateways.

Router

1. The software process that controls the transmission of packets between networks. A router manages data transfer across a bridge.
2. A node that runs routing software. See **Routing Node.**

Routing Information Protocol

(RIP) A **protocol** used by the **rip_server** and **routed** processes to dynamically maintain the routing tables on gateways and, in some (non-DOMAIN) cases, hosts.

Routing Node

A node that runs the routing process and transmits packets between different networks, especially through a bridge. A node that transmits packets between dissimilar networks is called a **gateway.**

Routing Server

Same as **router.**

Routing Table

A table maintained by hosts and gateways that indicates the next gateway in the route to a destination.

Server

A process that is dedicated to managing a certain function. A variety of servers support TCP/IP communications. See also **daemon**.

Software Loopback Interface

A **physical layer interface** simulator within TCP/IP software that directs messages to be received within the node without transmitting them to the physical layer software.

Subnet Number

The portion of the **Internet Address** that identifies networks within an internet. A **network number** identifies a single **internet** while subnet numbers identify networks within that internet.

TCP

See Transmission Control Protocol.

TCP/IP

Transmission Control Protocol/Internet Protocol. An acronym used to refer to the TCP and IP protocols and related Internet protocols, such as FTP and Telnet, defined by the Defense Advanced Projects Agency.

TCP/IP Administrative node

A node on which the DOMAIN TCP/IP */sys/tcp/hostmap* directory and the */sys/tcp/hosts.hst* and /sys/tcp/gateways files are located. The TCP/IP administrative node for DOMAIN/IX BSD4.2 TCP/IP contains */etc/hosts, /etc/gateways, /etc/neworks, /etc/hosts.equiv*.

TCP/IP Gateway

A gateway that routes information on a TCP/IP internet. A TCP/IP gateway usually translates protocols for unlike networks. However, TCP/IP gateways are required on DOMAIN routing nodes to maintain TCP/IP services across the physical link between two DOMAIN networks.

tcp_server

The **server** process that manages TCP/IP communications on all DOMAIN nodes.

Telnet

A remote terminal emulation protocol defined by the Defense Advanced Research Projects Agency for internetwork communications. Telnet uses TCP and IP as underlying protocols.

telnetd

The **daemon** process that accepts incoming Telnet requests on DOMAIN/IX nodes.

telnet_server

The **server** process that accepts incoming Telnet requests on AEGIS nodes.

Topology

The arrangement of networks and systems on those networks.

Transmission Control Protocol

(TCP) A protocol for sending datagrams from one network to another. It was defined by the DefenseAdvanced Research Projects Agency for internetwork communications.

Type A Address

An **Internet address** where the network number is represented by a single byte with a leftmost 0 bit, and the local address consists of three bytes.

Type B Address

An **Internet address** where the network number is represented by two bytes with the leftmost two bits having the value 10, and the local address consisting of two bytes.

Type C Address

An **Internet address** where the network number is represented by three bytes with the leftmost three bits having the value 110, and the local address consisting of one byte.

Index

The letter *f* means "and the following page"; the letters *ff* mean "and the following pages". Symbols are listed at the beginning of the index.

E

F

G

H

I

J

K

L

M

S

T

U

Reader's Response

Please take a few minutes to send us the information we need to revise and improve our manuals from your point of view.

Document Title: *Configuring and Managing TCP/IP*
Order No.: 008453 Revision: 01 Date of Publication: January, 1987

What type of user are you?

_____ System programmer; language _____
_____ Applications programmer; language _____
_____ System maintenance person
_____ System Administrator
_____ Student Programmer
_____ Other

_____ Manager/Professional
_____ Technical Professional
_____ Novice

How often do you use the DOMAIN system?_____

What parts of the manual are especially useful for the job you are doing?

_____

What additional information would you like the manual to include?

_____

Please list any errors, omissions, or problem areas in the manual. (Identify errors by page, section, figure, or table number wherever possible. Specify additional index entries.)

_____

_____
Your Name Date

_____
Organization

_____
Street Address

_____
City State Zip

No postage necessary if mailed in the U.S.

cut or fold along dotted line

FOLD

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 78 CHELMSFORD, MA 01824

POSTAGE WILL BE PAID BY ADDRESSEE

APOLLO COMPUTER INC.
Technical Publications
P.O. Box 451
Chelmsford, MA 01824

FOLD

Reader's Response

Please take a few minutes to send us the information we need to revise and improve our manuals from your point of view.

Document Title: *Configuring and Managing TCP/IP*
Order No.: 008453 Revision: 01 Date of Publication: January, 1987

What type of user are you?

_____ System programmer; language _____
_____ Applications programmer; language _____
_____ System maintenance person
_____ System Administrator
_____ Student Programmer
_____ Other

_____ Manager/Professional
_____ Technical Professional
_____ Novice

How often do you use the DOMAIN system?_____

What parts of the manual are especially useful for the job you are doing?

_____

What additional information would you like the manual to include?

_____

Please list any errors, omissions, or problem areas in the manual. (Identify errors by page, section, figure, or table number wherever possible. Specify additional index entries.)

_____

_____
Your Name Date

_____
Organization

_____
Street Address

_____
City State Zip

No postage necessary if mailed in the U.S.

cut or fold along dotted line

FOLD

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 78 CHELMSFORD, MA 01824

POSTAGE WILL BE PAID BY ADDRESSEE

APOLLO COMPUTER INC.
Technical Publications
P.O. Box 451
Chelmsford, MA 01824

FOLD