TCPware® for RSX
Software Reference Manual

Version 2.2-5

MAN/TCPIPRSXRM-84

First Printing, March 1985
Revised Printing, May 1989
Revised Printing, July 1990
Revised Printing, July 1992
Revised Printing, May 1993

Contents

1

Introduction

1.1 What Is TCP-RSX?

TCP-RSX is a software product that provides TCP/IP networking services for PDP-11 computers made by Digital Equipment Corporation (DEC). These services include:

- the lower level network protocols (TCP, UDP, and IP)
- utilities for managing and monitoring the network
- a QIO programming interface for TCP and UDP

1.2 TCPware® for RSX Family of Software Products

TCP-RSX is included with each product in the TCPware for RSX family. The following table describes the other products in this family.

Product	**Purpose**
FTP-RSX	Allows users to transfer files between remote and local hosts. Supports both the client (user) and server ends of the FTP protocol.
TELNET-RSX	Allows local users to log in to remote hosts, and remote users to log in to hosts running TELNET-RSX. Supports both the client (user) and server ends of the TELNET protocol.

1.3 Other TCPware Families

Process Software Corporation develops TCP/IP networking software products for many Digital Equipment Corporation (DEC) operating systems, in addition to RSX. These products are members of the following families:

- TCPware for VMS
- TCPware for RT-11
- TCPware for TSX-PLUS
- TCPware for IAS

These products solve many problems that are associated with networking dissimilar computer systems, as shown in Figure 6-1.

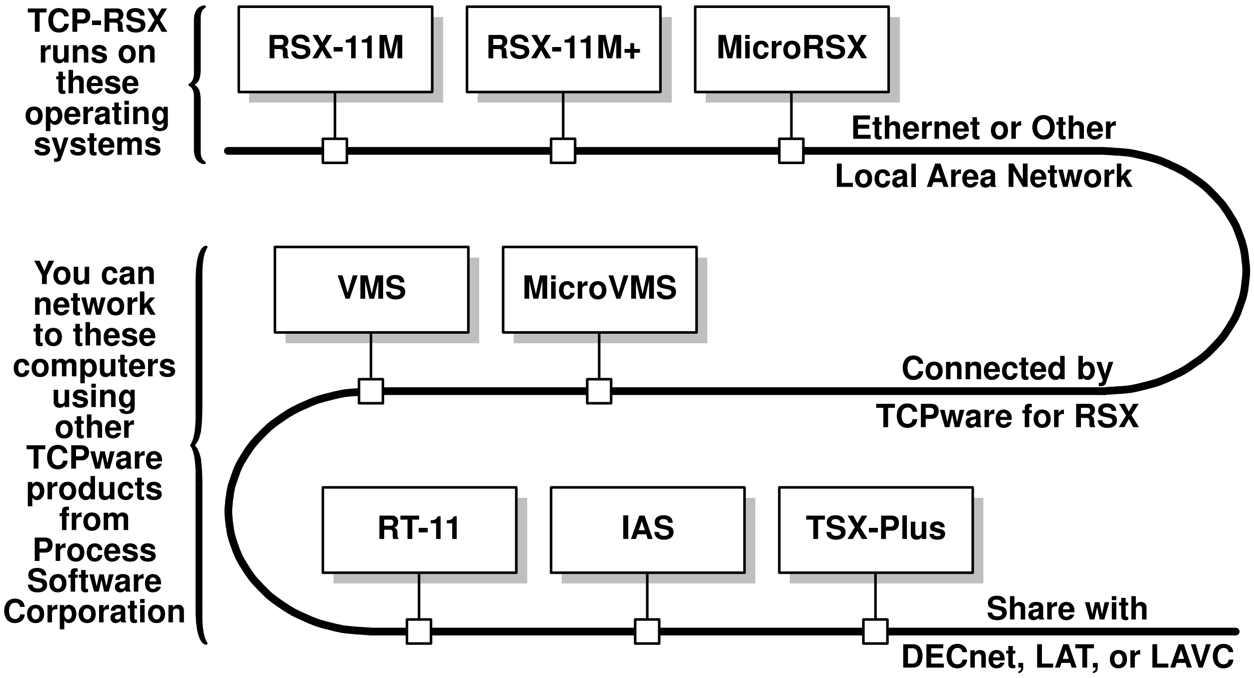

Figure 6-1: TCPware Connecting Different DEC Operating Systems

1.4 Intended Audience

TCPware for RSX Software Reference Manual is for:

- applications programmers who want to use TCP-RSX programming services
- system managers who install and maintain the FTP-RSX and TELNET-RSX software products

1.5 Prerequisite Knowledge

Before installing, maintaining, or using TCPware for RSX software products, you should be familiar with:

- the DEC RSX-11M, RSX-11M-PLUS, or MicroRSX operating systems
- the file systems for these operating systems
- computer networks in general

This manual does not document the RSX-11M, RSX-11M-PLUS, or MicroRSX operating systems.

1.6 Contents

This manual contains the following chapters and appendices:

- *Introduction* provides a brief overview of the TCP-RSX software product.
- *Overview of TCP/IP Networking*, for TCP/IP novices, introduces basic terms and concepts used in TCP/IP networking.
- *Installation*, for the system manager, explains how to install TCPware for RSX.
- *General Information* describes the hardware and software configuration. It provides an overview of the TCP-RSX components and lists reference material for the IP, TCP, UDP, and ARP protocols.
- *Network Control Utility*, for the system manager, explains what NETCU does and how to use it.
- *IN Device Driver Services*, for programmers, explains how to use the QIO programming interface (TCP and UDP).
- *Appendix A, Sample Programs*, is for programmers. It provides examples to show how to use TCP and UDP programming services.
- *Appendix B, Implementation Notes*, is for users who thoroughly understand the TCP, UDP, IP, and ARP protocols. It explains how TCPware implements these protocols.
- *Appendix C, Release Notes*, summarizes the enhancements made to each version of TCP-RSX.

1.7 The Documentation Set

The complete documentation set for the TCPware for RSX family of software products consists of:

- *TCPware for RSX Software Reference Manual*, which explains how to install and configure TCPware, use the support utilities, and use the QIO programming interface.
- *FTP-RSX Software User's Guide*, which explains how to use the FTP-RSX software product to transfer files between systems.
- *TELNET-RSX Software User's Guide*, which explains how to use the TELNET-RSX software product to access remote systems.

1.8 Conventions

This manual uses the conventions listed below.

`monospaced type`	Monospaced type is used in examples of system output or user input.
UPPERCASE	Uppercase letters are used in commands and examples to indicate that you should enter the characters shown. The system accepts uppercase or lowercase characters.
italics	Italics are used in commands and examples to indicate that either the system supplies a value or you should supply a value.
[ ]	Square brackets indicate the enclosed item is optional. Do not type the brackets when you enter the command.
^	A caret denotes a control character (ASCII characters from 1 to 31). To type a control character, press the “Ctrl” key and type the character that follows the caret.
Host	“Host” refers to any computer system on the network. The local host is the computer you are currently using (your RSX system). A remote host is any other computer system on the network.

1.9 Software License

Each copy of FTP-RSX and TELNET-RSX includes a *Software License Agreement*. You should read and understand the *Agreement* before installing TCPware.

The *Agreement* entitles you to install FTP-RSX/TCP-RSX or TELNET-RSX/TCP-RSX on one computer system. If you want to use these products on more than one computer, you must purchase additional products or licenses. Contact Process Software Corporation or your distributor for more information.

1.10 Software Update Service

Process Software Corporation periodically updates and enhances TCPware for RSX products to incorporate new features, fix bugs, and improve the documentation. The Software Update Service ensures that you receive the latest versions as they are released. FTP-RSX and TELNET-RSX each include updates for TCP-RSX. Contact Process Software Corporation to order this service.

2
Overview of TCP/IP Networking

This chapter provides an overview of networking using TCP/IP. It introduces you to terms and concepts used to describe TCP/IP. It is for applications programmers or system managers who are not familiar with TCP/IP networking.

2.1 What is TCP/IP?

TCP/IP, an acronym for *Transmission Control Protocol/Internet Protocol,* refers to a suite of protocols that was created especially for connecting dissimilar computer systems. TCP/IP provides a common set of communications rules (or protocols) for all interconnected computers to follow. Computers that are connected with TCP/IP form a network called an *internet.*

TCP/IP began in research funded by the Defense Advanced Research Projects Agency (DARPA), an agency of the U. S. Department of Defense. This research evolved from the need to connect computer systems that use different hardware, operating systems, and network technologies.

TCP/IP was first implemented in a DARPA internet called ARPANET. It was so successful in the ARPANET that DARPA designated TCP/IP as a networking standard.

The term *internet* can be used generically to describe any computer systems that are connected by TCP/IP. It is also the name of a worldwide DARPA network that connects thousands of major research institutions, universities, corporations, and government laboratories. Some organizations that belong to the Internet include:

- National Science Foundation
- Department of Energy
- National Aeronautics and Space Administration

Members of the Internet share data and resources through the network.

TCP/IP is often used to create internets that are independent of the DARPA Internet. TCP/IP is widely used in such diverse industries as telecommunications, health care, and agriculture. It can benefit any organization that wants to:

- Connect dissimilar systems (for example, connect a RSX system to a UNIX system)
- Connect hosts at different facilities that span large geographical distances
- Connect local or wide area networks
- Connect a host or network to the internet

An internet is also referred to as a *virtual network* because it allows your local host to communicate with other connected hosts as if all hosts were connected to the same local network.

2.2 Elements of an Internet

All internets contain three elements:

- Hosts
- Networks
- Gateways

TCP/IP must be implemented in a host or network before that host or network can be connected to an internet. The following sections explain each element in detail.

2.2.1 Hosts

A host is a computer or device that is part of an internet. These are some examples of hosts:

- Multiuser system such as a minicomputer
- Single user system such as a microcomputer
- Workstation
- Terminal server
- Printer server

Hosts in an internet can use different operating systems. The TCP/IP protocols give each operating system a set of shared rules to follow for internet communications. Figure 2-1 shows an internet created from different hosts that share the same network hardware interface.

2.2.2 Networks

A network consists of two or more hosts that are connected by one communications medium such as Ethernet. These are some examples of networks:

- A local area network (LAN) consisting of several hosts. A LAN typically runs at high speeds, and spans a small geographic area such as a single room or building.
- A wide area network (WAN) in which hosts are connected through leased lines (such as the U.S. telephone system). Although a WAN typically runs at much slower speeds than a LAN, it can transfer data across a large geographical distance such as a state or continent.

One goal of an internet is to connect various types of networks independent of the underlying hardware technologies. Gateways help to resolve hardware differences.

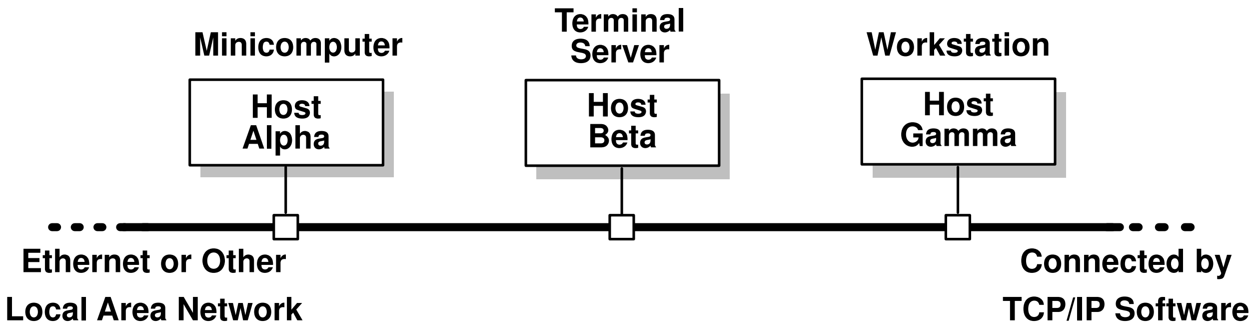

Figure 2-1: Internet Created from Different Hosts

2.2.3 Gateways

A gateway is a device that connects two or more networks to form an internet. A gateway might be a dedicated computer that resolves hardware differences between networks and routes packets between the networks. A host can serve as a gateway while also providing other services for its users. Figure 2-2 shows an internet consisting of two networks connected by a gateway.

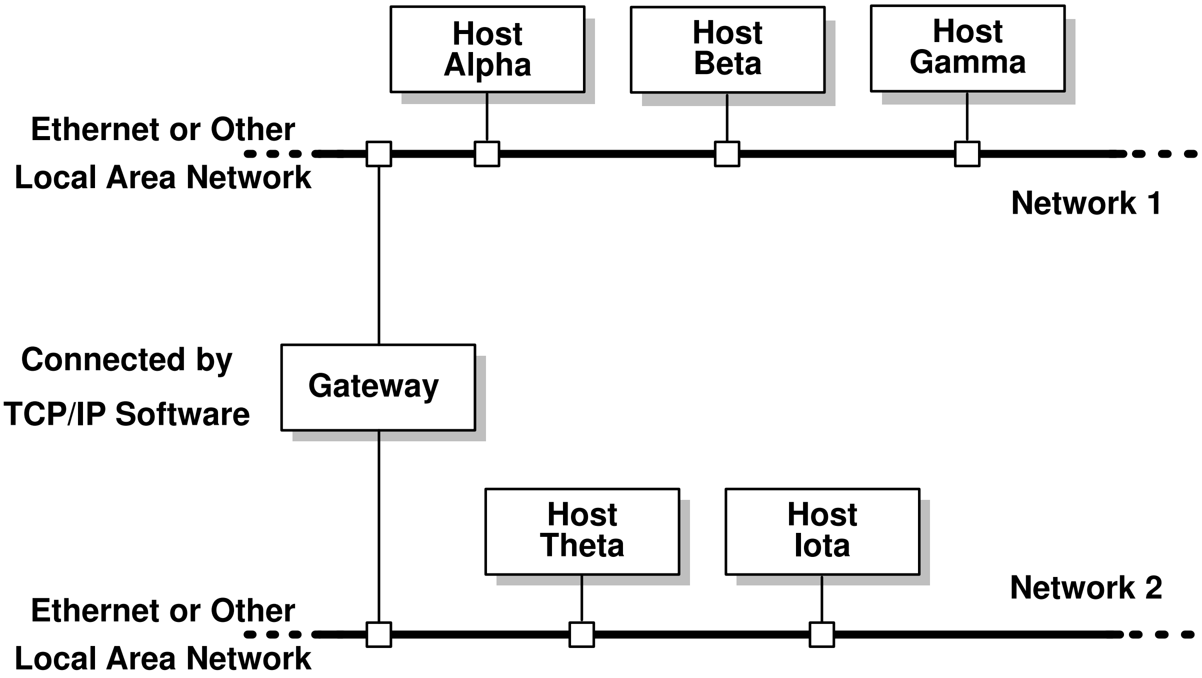

Figure 2-2: Two Networks Connected By A Gateway Form An Internet

In Figure 2-2, the gateway allows the hosts on network 1 to communicate with hosts on network 2. The two networks do not have to use the same hardware interfaces. The gateway resolves all differences in connectivity.

2.3 Internet Addresses

Internets use a consistent, global addressing scheme. Each host on an internet has a unique *internet address*. Internet addresses have several advantages over existing hardware addresses, because they:

- Cannot be duplicated or otherwise conflict with other networks on the internet.
- Permit an unlimited number of hosts or networks to connect to an internet.
- Allow networks that use different hardware addressing schemes to be included in the internet.

2.3.1 Internet Address Rules

Internet addresses are binary values expressed in decimal form. Each field of the address consists of one byte. A decimal point separates each byte. Examples of internet addresses are:

1.2.0.1 142.59.6.2 192.42.92.7

Internet addresses conform to the following rules:

1. Each internet address consists of 32 bits divided into four fields of eight bits (one byte) each.
2. An internet address has two parts: a network number and a host number (which is also referred to as a machine number).
3. All hosts on the same network must have the same network number.
4. No two hosts on the same network can have the same host number.
5. No two networks can have the same network number if they are connected in any way.

2.3.2 Internet Address Classes

Internet addresses fall into three primary categories or *address classes*. The address class is usually determined by the number of networks in proportion to the number of hosts at an internet site. However, a large internet like the Internet can use all three internet address classes. The address classes are described below.

- **Class A -** A class A address uses the first byte for a network number and the remaining three bytes for the host number. The first byte ranges in decimal value from 1 to 127. A class A address fits an internet situation that has up to 128 (2^7) networks and up to 16,777,216 (2^{24}) hosts per network.
- **Class B -** A class B address uses the first two bytes for the network number and the second two bytes for the host number. The first byte ranges in decimal value from 128 to 191. A class B address fits an intermediate situation with up to 16,384 (2^{14})networks and up to 65,636 (2^{16}) hosts per network.
- **Class C -** A class C address uses the first three bytes for the network number and the fourth byte for the host number. The first byte ranges in decimal value from 192 to 223. A class C address fits a situation with up to 2,097,152 (2^{21}) networks, and less than 256 (2^8) hosts per network.

Assigning the address class determines the *network mask* of the address. The network mask is used by hosts and gateways to determine the routing of internet packets. It is a 32 bit internet address in which the network number is set to all bits one and the host number is set to all bits zero. Hosts and gateways use the network mask to extract the network number of an internet address. They then compare that network number with their routing information to determine if the packet is bound for a local address.

The decimal value of each address class is listed below with its corresponding network mask. The first byte of the address determines the address class.

Address Class	First Byte	Network Mask
Class A	1. - 127.	255.0.0.0
Class B	128. - 191.	255.255.0.0
Class C	192. - 223.	255.255.255.0

The decimal notation of internet addresses is illustrated in Figure 2-3.

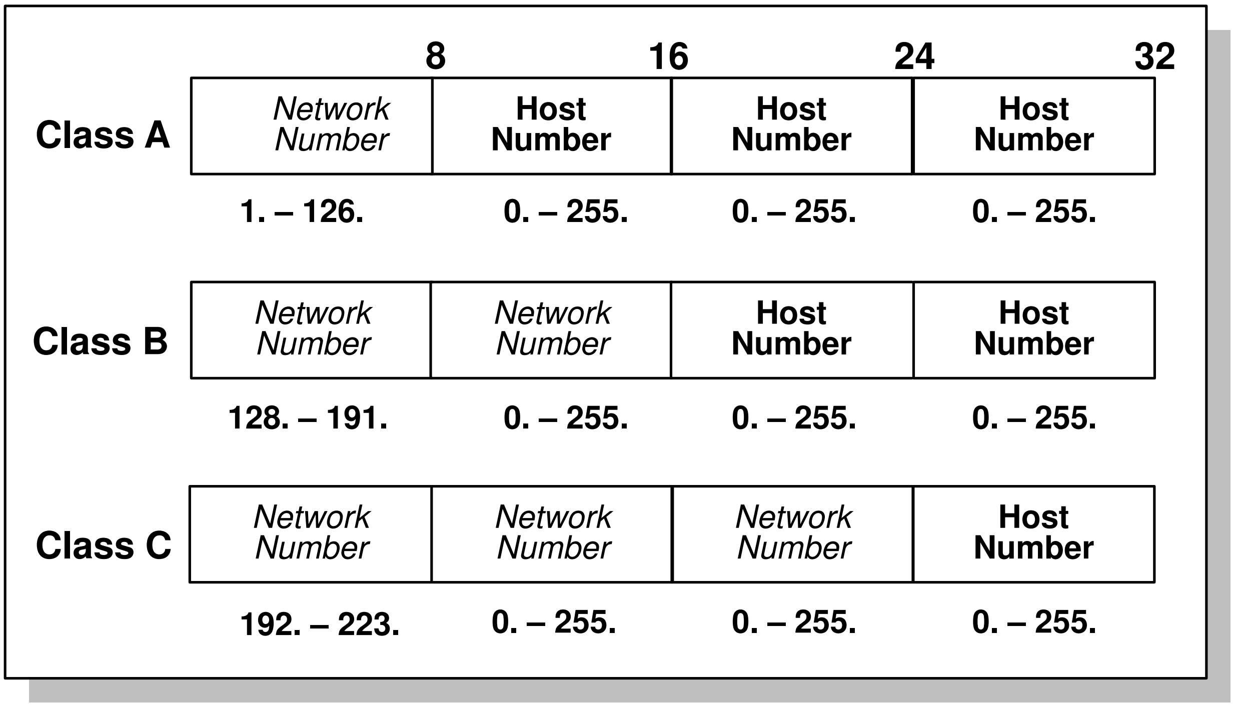

Figure 2-3: Internet Address Classes and Notation

2.3.3 Reserved Network and Host Numbers

Some network and host numbers are reserved for special purposes (such as broadcasting) and should not be assigned to a network or host. Reserved network numbers are 0 and 127. Both are class A addresses. Host reserved addresses are all bits zero, and all bits one. The reserved host numbers refer to all three classes.

2.3.4 Subnet Addresses

An internet network can be divided into several smaller networks called *subnetworks* (or *subnets*). Using subnets allows the logical grouping of hosts within a larger network. This may be practical in an organization that has many hosts connected to a single network, or one that has many networks and plans to connect to the Internet.

A subnet address is assigned by taking bits from the host number of the internet address and using them to extend the network number. Thus a host that is part of a subnet has an internet address that still consists of 32 bits, but is divided into three fields: network number, subnet number, and host number. The subnet number appears to hosts and gateways outside of the subnet as part of the host number. Only hosts or gateways that are connected to the subnet need to know the *subnet mask,* or the bits used to identify the subnet number.

The subnet mask is created by extending the network mask with the bits used to indicate the subnet number. Local hosts and gateways of a subnet then make routing decisions based on the subnet mask rather than the network mask.

2.3.5 Gateway Addresses

Gateways must also have internet addresses. The rules for assigning internet addresses to gateways are similar to those used to assign an internet address to a host with these additions:

1. A gateway must have an internet address for each physically connected network or subnet. For example, a gateway that connects network 1 and network 2 (as shown in Figure 2-2) has a separate internet address for each network.
2. All networks that use the same gateway must have different network numbers. A gateway cannot support two networks that both have the same network number. This statement also applies to any subnets connected to a gateway.
3. A gateway must have a unique host number for each network or subnet number. This host number cannot duplicate an existing host number on that network or subnet.

The reserved network and host numbers also apply to gateway addresses.

2.3.6 Assigning Internet Addresses

To illustrate how internet addresses might be assigned in particular situations, consider the following examples. Company A has three different hosts to interconnect to form an internet. This internet consists of one network made up of three hosts. Since Company A has a single network, a Class A address is chosen. All the hosts on the single network have the same network number: 1.

Company A assigns the hosts numbers as follows:

Host Alpha	1
Host Beta	2
Host Gamma	3

The internet addresses assigned to Company A's hosts are:

Host Alpha	1.0.0.1
Host Beta	1.0.0.2
Host Gamma	1.0.0.3

Let's further suppose that Company A adds a second network that connects to the first network by way of a gateway. The new internet situation would be similar to the one illustrated in Figure 2-2. Network 1 addresses remain as shown above.

Since networks 1 and 2 are connected by a gateway (gateway A), network 2 must use a different network number. The internet addresses assigned to the network 2 hosts are as follows:

Host Theta	2.0.0.1
Host Iota	2.0.0.2

More information on the FTP protocol can be found in the Request for Comments (RFC) documents published by the Network Information Center (NIC) at Government Systems, Inc., 14200 Park Meadow Drive, Suite 200, Chantilly, VA 22021 (800-365-3642, 703-802-4535 (7:00 AM to 7:00 PM EST)) The RFCs contain the specifications for all internet protocols. Gateway A is assigned these internet addresses:

Gateway A	1.0.0.4
	2.0.0.5

In summary, Company A's two networks must have different network numbers because they are joined by a gateway. Gateway A must have two internet addresses because it connects two networks.

Before assigning internet addresses to your network, check with your network administrator. You can also obtain a unique internet address from the Network Information Center (NIC) at Government Systems, Inc., 14200 Park Meadow Drive, Suite 200, Chantilly, VA 22021 (800-365-3642, 703-802-4535 (7:00 AM to 7:00 PM EST)). You can also send electronic mail directed to the "HOSTMASTER" and "REGISTRAR" mailboxes at network address NIC.DDN.MIL.

2.3.7 Routing

Both gateways and hosts store routing information. Gateway routing information consists of internet addresses for all networks that are directly connected to the gateway. A gateway also knows the internet addresses of gateways to other networks. When a gateway receives a packet of information from the internet, it consults its routing information and makes decisions as to the next destination of the packet. If the packet's internet address is local, the gateway delivers the packet directly. If not, the gateway makes decisions as to the packet's next destination.

In a host, routing information usually consists of the internet address of the local network, and the internet address of a gateway. When a host sends a packet, it firsts consults its routing information to see if the packet can be delivered directly. If not, the host sends the packet to the gateway.

2.4 TCP/IP Protocols

An internet is a virtual network that provides the user with the ability to communicate to all connected hosts as if they were part of the local network. All the network details are hidden from the user. The basis on which this virtual network exists is the **TCP/IP protocol suite**. These protocols are designed to deliver data from one host to another across all types of communications media.

The TCP/IP protocol suite is most often conceptualized in layers. TCP/IP consists of three layers of services that rest on a layer of hardware. Figure 2-4 illustrates the TCP/IP layer concept.

- The network layer provides the packet delivery service across the network hardware interface for the transport layer.
- The transport layer provides a datagram service and a reliable data transport service to the application layer.
- The application layer provides services to the user, including remote terminal login and file transfer.

When a **packet** is transmitted over an internet, it passes through all the TCP/IP layers. Each layer adds its own **header** to the packet that includes control information related to the service provided by that layer. When a packet reaches its destination, it is passed up through the layers. Each layer removes its header information before passing the packet to the next layer. Figure 2-5 illustrates the transmission path of a packet on an internet.

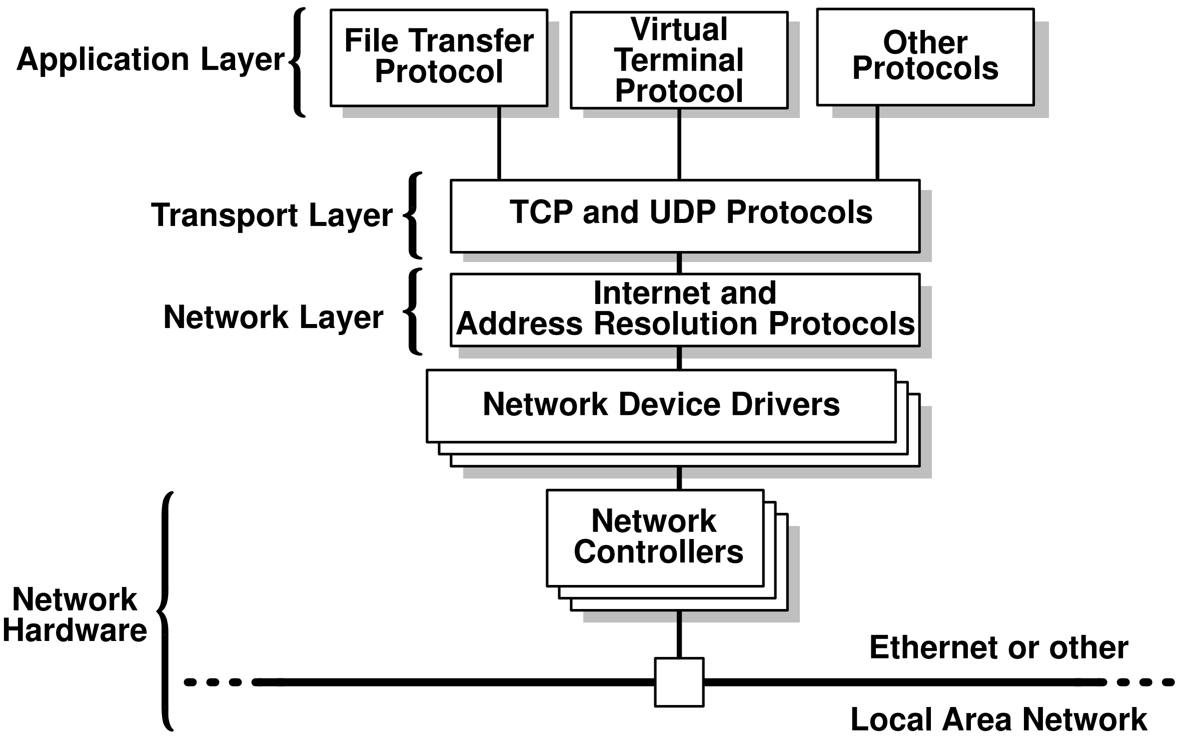

Figure 2-4: Layering of TCP/IP Protocols

2.4.1 Network Layer Protocols

The network layer protocols provide a connectionless packet delivery service. A **packet** is the unit of transfer for a physical network. **Connectionless** describes packet treatment: all packets are treated as separate entities. Network layer protocols provide efficient packet delivery by managing the addressing and routing of packets. The Internet Protocol (IP) provides the network layer protocol for TCP/IP.

Internet Protocol (IP)

The Internet Protocol (IP) is the foundation of TCP/IP. Its function is to send and receive packets of information over the physical network. IP calls the packet a **datagram**. Each datagram includes its source and destination addresses, control information, and any actual data passed from or to the host layer. The IP datagram is the unit of transfer of an internet.

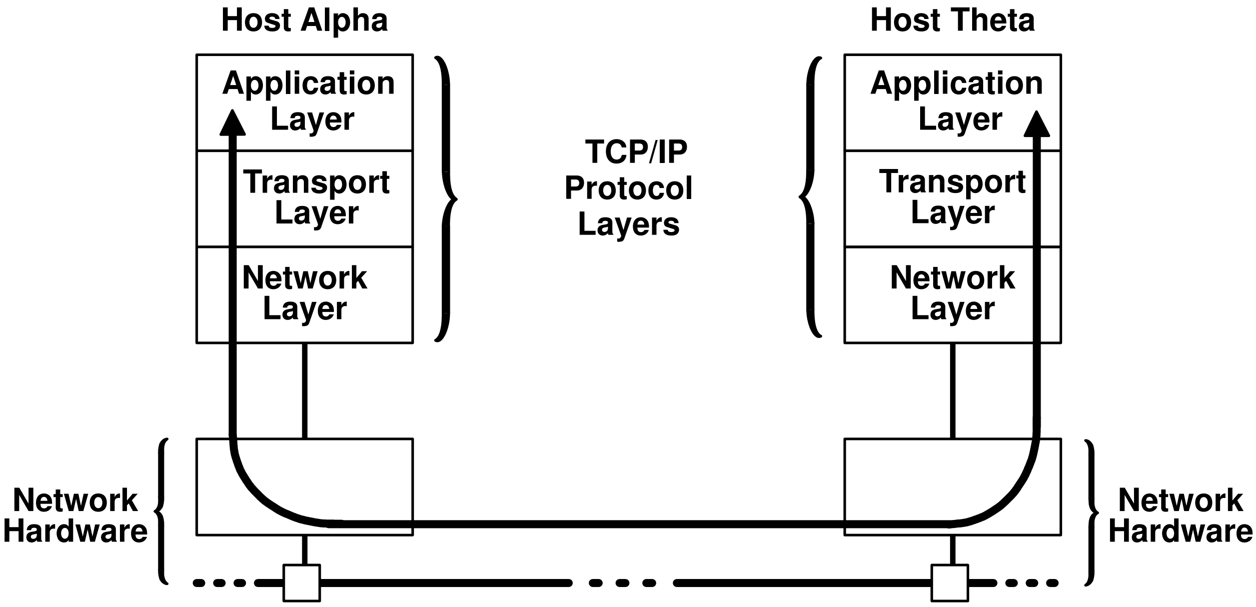

Figure 2-5: Transmission of a Packet Through an Internet

IP provides the mechanism by which hosts and gateways are able to route internet datagrams. IP routing is based on the destination address of each datagram. When IP receives a datagram, it checks the destination network number, then refers to a routing table. This table contains the addresses of known destination networks, gateways, and hosts. IP delivers datagrams with local internet addresses directly. It forwards datagrams with nonlocal addresses to their next destination based on the routing table information.

IP requires that data reliability be handled by the higher level (host or application layer) protocols.

Address Resolution Protocol (ARP)

The Address Resolution Protocol (ARP) provides an address conversion function. A host may have a hardware address that is unrelated to its internet address. ARP provides a method of mapping internet addresses to hardware addresses. ARP is used most often in hosts connected to Ethernet networks.

2.4.2 Transport Layer Protocols

The transport layer protocols provide the data transport services to the application layers. The Transmission Control Protocol (TCP) and the User Datagram Protocol (UDP) form the bridge between the user or application programs (such as an electronic mail facility) and the lower level protocols such as IP.

A primary function of the transport layer protocols is **multiplexing**. This term refers to the directing of message traffic to abstract delivery points called **ports**. When a host is running several application programs at once, the transport layer protocols direct incoming datagrams to the appropriate port so they reach the correct application program. Multiplexing allows many applications to access an internet at a time.

User Datagram Protocol (UDP)

The User Datagram Protocol (UDP) is a transaction-oriented protocol that transports data to or from the application layer. UDP requires very little overhead because it provides no protection against datagram loss or duplication. Applications that require orderly and reliable data transport should use the Transmission Control Protocol (TCP).

Transmission Control Protocol (TCP)

The Transmission Control Protocol (TCP) provides for reliable, process-to-process data transport by supplying the error checking facilities not provided by IP.

TCP is a stream oriented protocol. The service TCP provides guarantees that large amounts of data are transferred in order: all data is received by the destination host in the same order as it is sent by the source host. TCP also guarantees that data is not duplicated between the source and destination hosts.

2.4.3 Application Layer Protocols

Application layer protocols represent the highest level in the protocol layering model. These protocols convey the user interface to the host layer. The internet services, such as remote terminal login and file transfer, are provided by the application layer protocols. The protocols described here depend on TCP for reliable data transport.

The application protocols are often described in terms of the *client-server* model. This model views the local host as a *client* that initiates a networking service. The remote host, or *server*, then provides the service to the client. When referring to software that implements the TCP/IP application layer protocols, a *client utility* initiates a service, and a *server process* provides the service to the client utility.

The user on a local host typically invokes a client utility that initiates the networking service, such as TELNET or FTP. The client utility contacts the server process on the remote host. The server process usually performs the network service without user input.

For example, a user on host Alpha invokes the client FTP and requests that it open a connection with host Theta. The server FTP on host Theta answers the request, and thereby establishes a connection. Server FTP on Theta is now ready to service file transfer requests. The user on host Alpha enters commands “directly” to Theta to list directories, rename or delete files, or transfer files.

File Transfer Protocol (FTP)

The File Transfer Protocol (FTP) allows a user to log into a remote host and perform file functions. FTP file functions include: listing remote directories, copying files to or from the remote host, and renaming or deleting remote files.

When a user invokes FTP on a local host, a client FTP establishes a connection with the remote host server FTP. This first connection is used by the server and client to pass information. When the client has requested a file transfer, however, the server and client open a second connection over which to pass the data.

When a client FTP establishes a connection with a server, FTP demands a username and password before allowing the user access to the server host. This protects files by denying access to unauthorized users. FTP supports all popular file formats including ASCII and binary.

Virtual Terminal Protocol (TELNET)

TELNET (also known as Remote Login Protocol) allows a user at a client host to log into a server host as an interactive user. This allows a user at a local system to also be a user on a remote system. The user interacts with the remote system as if the user were directly logged in.

TELNET defines a device called a *network virtual terminal* (NVT). The NVT provides a standard network-wide representation of a terminal. This means that both the client and server have an agreed upon convention for how a terminal appears over the network.

TELNET also allows the client and server to negotiate options that can enhance their communications. By agreeing on certain options, TELNET clients and servers can, for example, change the format of the data they exchange from ASCII to binary.

2.5 Definitions

The terms defined in this section are commonly used to describe TCP/IP.

- **Address Resolution Protocol (ARP) -** Protocol used by IP to map internet addresses to Ethernet hardware addresses.
- **Client-Server Model -** Concept used to describe the application layer protocols where the process that initiates a service is called the client (or user) and the process that provides the service is the server. A client and a server can be on different hosts or on the same host.
- **Datagram -** Single message unit used by IP over an internet and consisting of protocol headers and data.
- **File Transfer Protocol (FTP) -** An application level protocol that allows a user on a client host to access a server host and perform file functions
- **Gateway -** A device used to connect two or more networks to form an internet. A gateway also has an internet address for each connected network, and performs routing functions.
- **Host -** A unique, addressable entity that is part of an internet. A multiuser minicomputer and a terminal server are examples of hosts.
- **Internet -** A network formed by connecting dissimilar hosts and networks with TCP/IP protocols. When capitalized, this term refers to the DARPA internet that formed the backbone of internet research.
- **Internet Address -** A unique 32 bit value assigned to each host in an internet. All internet communications with a particular host use its internet address.
- **Internet Protocol (IP) -** The basis of TCP/IP, IP provides the network interface and message routing services for the higher level protocols.
- **IP Routing -** Mechanism provided by IP to deliver datagrams from the source to the destination.
- **Local Area Network (LAN) -** Consists of two or more hosts connected by the same communications medium.

- **Network -** Element of an internet in which two or more hosts are connected with the same communications medium. A LAN is an example of an internet network element.
- **Network Mask -** A 32 bit internet address that represents an address class and is used by hosts and gateways to determine routing of datagrams.
- **Packet -** A single message as it appears to the physical network.
- **Port -** Abstract point through which a datagram passes from the host layer to the application layer protocols.
- **Remote Login Protocol -** *see* Virtual Terminal Protocol (TELNET).
- **Stream Service -** The transfer of data in an unbroken flow, without the use of markers to show the beginning or end of messages.
- **Subnet -** A subdivision of a network; used to provide a means to logically group hosts within a large network.
- **Subnet Mask -** A 32-bit internet address created by taking bits from the host number and using them to extend the network mask. Hosts and gateways local to a subnet use the subnet mask for local routing.
- **Transmission Control Protocol (TCP) -** A transport layer protocol that provides a reliable data transport service to the application layer protocols. TCP ensures data is delivered in order and without duplication.
- **User Datagram Protocol (UDP) -** A transport layer protocol that is *transaction-oriented* and transports data to or from the application layer. While there is very little overhead, UDP does not guarantee the reliable data transport exhibited by TCP.
- **Virtual Network -** A network in which all connected hosts are able to communicate to each other as if they were all on the same local network. An internet appears to a user as a virtual network.
- **Virtual Terminal Protocol (TELNET) -** An application layer protocol that allows a user at a client host to log into a server host. The user's terminal at the client host appears to the server as a directly connected terminal.
- **Wide Area Network (WAN) -** A network element of an internet in which hosts are connected over large geographic distances.

2.6 References

The *General Information* chapter lists material you can refer to for more information on TCP/IP networking.

3
Installation

This chapter is for system managers. It explains, step-by-step, how to install TCPware for RSX software products. It also provides a sample installation session.

3.1 Before You Begin

Before you begin installing TCPware, follow these guidelines:

1. Make sure your network hardware is properly installed. Refer to the appropriate hardware documentation for hardware installation instructions.
2. Ensure that you have approximately 1500 disk space blocks for installation.
3. If you are unfamiliar with TCP/IP networking, read the *Overview of TCP/IP Networking* chapter.
4. Read this *Installation* chapter.

NOTE

For a typical configuration, TCPware for RSX requires from 700 to 1200 words of system pool. Refer to the MCR SET /POOL command for details on determining the available pool space.

3.2 Summary of Steps

You perform one procedure to install TCP-RSX, FTP-RSX, and TELNET-RSX. You will obtain a software password from Process Software Corporation that determines which products you can actually use.

Follow these steps to install TCP-RSX, FTP-RSX, and TELNET-RSX:

1. **Copy** the distribution kit to a directory on the system.
2. **Build** the components for the product.
3. **Configure** the software.

4. **Obtain** a software password for your system.
5. **Start** TCPware.
6. **Test** TCPware.
7. **Clean up** after the installation.

The sections that follow describe each step in detail.

3.3 Step 1: Copy the Distribution Kit

One kit, labeled *TCPware for RSX,* contains all product files. Follow these steps to copy the files from the distribution kit:

1. **Log** in to the system as a privileged user.
2. **Set** your default directory to a directory that does not already contain files. If you do not have a directory free of files on the system, create one using the MCR UFD or DCL CREATE/DIRECTORY command.
3. **Copy** the files from the distribution kit.

   For the magnetic tape (16MT9) distribution kit, use the following MCR command to copy the files from the DOS-11 formatted tape:

   ```
   FLX SY:/RS=MS0:[*,*]*.*/DO
   ```

   For the RX01/RX02 floppy diskette distribution kit, use the following MCR command to copy the files from the RT-11 formatted diskette:

   ```
   FLX SY:/RS=DY0:*.*/RT
   ```

   For the RX50 floppy diskette distribution kit, use the following MCR commands to copy the files from the Files-11 formatted diskettes:

   ```
   MOUNT DUn: PSCTCP
   PIP SY:=DUn:[*,*]*.*
   DMO DUn:
   ```

   For the TK50 cartridge tape distribution kit, use the following MCR commands to copy the files from the tapes:

   ```
   MOUNT MU0: PSCTCP
   PIP SY:=MU0:*.*
   DMO MU0:
   ```

4. **Proceed** to Step 2: Build the Components.

NOTE

If you are using a previous version of FTP-RSX or TELNET-RSX, reboot the system so the previous version does not start (i.e., edit the system startup file) before proceeding to install the new version.

If you plan to use DECnet's Direct Line Access (DLX) support, make sure DECnet is up and running before proceeding to Step 2: Build the Components.

3.4 Step 2: Build the Components

To build the components, execute the CNFNET indirect command file from the directory containing the files copied from the distribution kit by **typing**:

```
@CNFNET
```

The command file explains the installation options and prompts you for the required information. The information you must provide is listed below.

- The number of the network device option to be used by the software and specifics for that device (such as the DLX line or the CSR and vector addresses of the controller).

  Choose option 1, for DECnet's Direct Line Access (DLX) support, if you are running DECnet and you want to use the same Ethernet controller for TCPware. You are prompted to enter the DECnet line specification for the controller. This line specification is usually QNA-0 or UNA-0.

  Choose option 2, for DEC XE device driver support, if your system uses a UNIBUS Ethernet controller (DELUA or DEUNA) and the XE device driver has been SYSGENed into your system. The XE driver is included in the RSX distribution kit. You are prompted to enter the DECnet line specification for the controller (usually UNA-0).

  Choose option 3, for XQ device driver support, if your system uses a Q-bus Ethernet controller (DELQA or DEQNA). The XQ driver is included with TCPware. You are prompted to enter the DEQNA/DELQA CSR and vector address for the controller. The default CSR is 174440 and the default vector is 120.

  Choose option 4, for XU device driver support, if your system uses a unibus Ethernet controller (DELUA or DEUNA). The XU driver is included with TCPware. You are prompted to enter the DEUNA/DELUA CSR and vector address for the controller. The default CSR is 174510 and the default vector is 120.

- The maximum number of Host Name Table entries required.

  To determine this value, you might want to obtain a list of all available hosts on the network. This value can be between 32 and 200, inclusive. Each table entry uses 38 bytes of INPOOL space. (Step 3: Configure the Software explains how to allocate space in INPOOL.) The default is 64 entries.

- Whether you want to build the FTP-RSX components.

  Enter Y if your company purchased FTP-RSX and is licensed to run it. Otherwise, enter N.

- The name of an optional FTPD event logging subroutine that you have written.

  You are prompted for this name only if you purchased FTP-RSX.

  Unless you need this feature, you would not normally use an event logging subroutine. The *Server-FTP* chapter in the *FTP-RSX Software User's Guide* provides more information on the event logging subroutine.

  If you have not written a routine, press return in response to this prompt.

- Whether you want to build the TELNET-RSX components.

  Enter Y if your company purchased TELNET-RSX and is licensed to run it. Otherwise, enter N.

After building all of the components, **proceed** to Step 3: Configure the Software.

3.5 Step 3: Configure the Software

When you configure the software you define your local environment. You are prompted for the following information:

- The internet address of the local host.

  Your network manager must assign an internet address to your local host.

  This address is used as the *local-address* parameter in the NETCU START command during network startup.

  The *Overview of TCP/IP Networking* chapter and the *General Information* chapter, *Definitions* section provide more information on internet addresses.

- The internet address of the default gateway.

  Your network may not have a gateway (it may be a local network, which is not connected to any other network). In this case, do not specify an internet address. Press return in response to this prompt.

  If your network does have a gateway and you want to access hosts not on the local network, you must specify the local network address of this gateway.

  If your network has more than one gateway, select the most reliable gateway or the gateway "closest" to the hosts you want to access on the other networks. The internet address of the gateway you specify must be on the local network.

  If one or more gateways on your network will respond to ARP (Ethernet Address Resolution Protocol) requests for non-local addresses, you may decide *not* to specify a gateway address because ARP can then be used to accomplish the routing much more efficiently.

  All datagrams addressed to hosts that are not on the local network will be sent to the specified gateway for forwarding to the ultimate destination.

  This address is used as the *gateway-address* parameter in the NETCU START command during network startup.

- The internet address network mask for your network.

  This prompt appears only if you provided an internet address for the default gateway.

  You must enter a value only if your network is a subnetwork.

  If your network is not a subnetwork, press return in response to this prompt. NETCU determines the default mask value based on the network class of the local internet address.

  This mask is used to determine whether or not an internet address is local. This is done by clearing all of the bits in the target internet address and local internet address that are *not* set in the network mask. If the resulting numbers match, the address is considered local.

  The default network masks are 255.0.0.0 for Class A networks, 255.255.0.0 for Class B, and 255.255.255.0 for Class C.

  This mask is used as the *network-mask* parameter of the NETCU START command.

- The size of the INPOOL region.

  INPOOL is used for:

  packet receive buffers

  packet transmit buffers

  TCP/IP data buffers

  the Host Name Table

  The size of INPOOL determines how many simultaneous connections TCPware can support. The value you specify is used as the *size* parameter in the NETCU CREATE command during network startup.

  When you start TCPware, it allocates about 16K bytes to the packet receive buffers and about 8K bytes to the Host Name Table.

  TCPware allocates the remaining space dynamically to the packet transmit buffers and TCP/IP data buffers. They need from 128 bytes to 4K bytes each.

  On the average, each connection requires about 4K bytes.

  Use this formula to calculate the size of INPOOL in K bytes:

  $4 * n + 16 + 8$

  where *n* is the maximum number of connections you want TCPware to support. Keep in mind that FTP users require two connections. Most other applications, such as TELNET, require just one.

  The following table shows some examples.

If You Want This Many Connections...	**Enter...**
2	32K bytes
4	40K bytes
6	48K bytes
8	56K bytes
16	88K bytes
32	152K bytes

  You can enter between 28K bytes and 1024K bytes, inclusive. The default is 48K bytes.

  For more information on the *size* parameter of the NETCU CREATE command, refer to the *Network Control Utility* chapter.

- The number of active tasks you want Server-FTP (FTPD) to support.

  This question is asked only if your company purchased FTP-RSX.

  Each server can support one remote user at a time. If you do not want to allow access from remote systems, you can specify 0. This value can be between 0 and 9, inclusive. The default is 2.

- The number of remote connections you want Server-TELNET to support simultaneously.

  This question is asked only if your company purchased TELNET-RSX.

  When determining this number, consider the following issues:

  > Each connection requires 200 bytes of system pool memory and adds 1280 bytes to the INTELD region. (These numbers are approximate.)
  >
  > How many remote users do you think will connect to Server-TELNET at any given time? Try to estimate the maximum number.

  A value of 0 removes TELNET server support from your RSX system.

  If you want to change the number of supported connections after TCPware is installed, you must reboot the RSX system.

- The host names and internet addresses for the hosts on the network. The host definition file is built from this information.

  Enter the host name and internet address for each host. To terminate the list, press return in response to the host name prompt.

  You can also edit the host definition command file, LB:[1,2]HOSTS.CMD, using any text editor at a later time.

  For more information on host names and internet addresses, refer to the *General Information* chapter, *Definitions* section.

After completing the configuration steps, **proceed** to Step 4: Obtain the Software Password.

NOTE

> If you need to change any configuration parameters, you can repeat the configuration step by typing @CNFNET CONFIGURE at the DCL/MCR prompt.

3.6 Step 4: Obtain the Software Password

You must obtain a software password from Process Software Corporation.

NOTE

> CNFNET loads some of the software components (the device driver(s) and INSACP) in order to determine the software password information. If any error messages display during these operations, your network device hardware controller is probably not installed correctly. Correct the problem and repeat this section by typing @CNFNET PASSWORD at the DCL/MCR prompt.

When you are prompted for the software password, **contact** your distributor or Process Software Corporation to get the software password. If you purchased TCPware from a distributor, contact that distributor.

NOTE

> If you are unable to obtain a software password at this time or if you have other systems to configure and you want to get all of your software passwords at once, press return in response to the software password prompt. Repeat this step later by typing @CNFNET PASSWORD.
>
> You cannot start up TCPware until you obtain a software password.

You may use any of the following methods to contact Process Software Corporation (ask for the Password Desk):

- Send a FAX to the phone number 508-879-0042. Use the Software Password Request Form.
- Send an electronic mail message to passwords@process.com.
- Call Process Software Corporation at 800-722-7770, or 508-879-6994. The Password Desk is available Monday through Friday between 8:15 A.M. and 5:00 P.M. Eastern Time.
- Mail the required information (see below) to Process Software Corporation, 959 Concord Street, Framingham, MA 01701.

You must **provide** the following information to get a software password:

- Your Software License Agreement number, if available (this number may be printed on the Software License Agreement form supplied with the distribution kit),
- Your company name
- Your name
- Your hardware identification number displayed during the above configuration steps
- Your system identification number displayed during the above configuration steps
- The names and versions of the product you are installing (FTP-RSX V2.2 and/or TELNET-RSX V2.2).

NOTE

> If you already have a software password for this system for one or more products, and you are adding additional products, you *must* obtain a new password. Provide the Software License Agreement number(s) and the product name(s) for the existing software password.

When you get your software password, **enter** it in response to the prompt.

The software password consists of five groups of four alphanumeric characters (not including the letters I, L, O, and Z) separated by hyphens. The following is an example of a software password, which is not valid on your system:

```
81BD-D640-CTUE-C86C-3F62
```

The networking software is now ready to be started. **Proceed** to Step 5: Start TCPware.

3.7 Step 5: Start TCPware

Start TCPware by typing the following command at the DCL/MCR prompt:

```
@LB:[1,2]STARTNET
```

This command file does the following:

- Loads the network device driver(s). The following message displays after loading:

  ```
  LOA -- Warning - Loadable driver larger than 4K
  ```

  Ignore this message. It does not indicate an error.
- Creates the INPOOL region (using the NETCU CREATE command).
- Starts the software (using the NETCU START command).
- Installs the FTP-RSX tasks and starts the Server-FTP (FTPD) tasks.
- Installs the TELNET-RSX tasks and sets up the Server-TELNET.

The startup should complete error-free. If any other error or warning message displays, you probably made a mistake during a previous installation step. Review the steps and repeat them if necessary.

NOTE

> If the IE.ONP (Hardware/Software option not present) error message display, the software password is invalid for the current configuration. Check that the software password was entered correctly.

3.8 Step 6: Test TCPware

If you purchased FTP-RSX, test the User-FTP utility by transferring files to a remote system. Test Server-FTP by transferring files from a remote system to a local system. Test both simultaneously by specifying your local host's name or internet address when opening a connection. Refer to the *Transferring Files* chapter in the *FTP-RSX Software User's Guide*.

If you purchased TELNET-RSX, test User-TELNET by logging in to a remote system. Test Server-TELNET by logging in to the TELNET-RSX system from a remote system. Refer to the *Using TELNET* chapter in the *TELNET-RSX Software User's Guide*.

3.9 Step 7: Clean Up

After testing the software, you will probably want to add a line to your system startup command file to start the networking software.

Add the following line to your system startup command file:

```
@LB:[1,2]STARTNET
```

NOTE

> If you configured the software to use the DECnet Direct Line Access (DLX) support, you MUST start up DECnet before invoking the STARTNET command file.

You can delete all of the files from the directory used during installation. However, it is recommended that you save the CNFNET.CMD and *.MAP files (you can copy these to some other directory).

The installation procedure adds all of the required files and moves them to their target directories, as follows:

- Directory LB:[1,1] contains the FTPLIB.OLB and/or TELLIB.OLB library file(s).
- Directory LB:[1,2] contains the SHUTNET.CMD, STARTNET.CMD, STARTNET.DAT, HOSTS.CMD, TCPWARE.HLP, and PSWRDNET.DAT files.
- The system directory, usually LB:[1,54], contains the INDRV.TSK, INDRV.STB, INSACP.TSK, NETCU.TSK, XQDRV.TSK/XQDRV.STB or XUDRV.TSK/XUDRV.STB files (depending on whether you selected the XQ or XU device driver), and TELCU.TSK, INTELD.TSK, TELNET.TSK and/or UFTP.TSK and FTPD.TSK files.

3.10 Stopping TCPware

If you want to stop TCPware, type the following command at the DCL or MCR prompt:

```
@LB:[1,2]SHUTNET
```

This command file does the following:

- Stops the software (using the NETCU STOP command),
- Removes the INPOOL region (using the NETCU CREATE command), and
- Removes the installed FTP-RSX and/or TELNET-RSX tasks from the system.

3.11 Sample Installation

This section contains a sample installation procedure. Information you enter during your installation might be different from this sample. In the sample, text the system manager enters is underlined.

```
>
> @CNFNET
> ;
> ;      TCPware(R) for RSX Version 2.2
> ;
> ;      Installation and Configuration procedure for:
> ;        o  FTP-RSX V2.2
> ;        o  TELNET-RSX V2.2
> ;        o  TCP-RSX V2.2 (provided with FTP-RSX and TELNET-RSX)
> ;
> ;      Copyright (c) 1990, 1991, 1992 by
> ;      Process Software Corporation
> ;      Framingham, MA   01701
> ;
> ;
> ;      Specify the network device by choosing one of the following
```

```
> ;     options, depending on your system's configuration:
> ;        1. For DECnet Direct Line Access (DLX) support of a DEC
> ;           DEQNA, DELQA, DEUNA, or DELUA Ethernet controller.
> ;        2. For DEC's XE (DEUNA or DELUA) device driver support.
> ;        3. For PSC's XQ (DEQNA or DELQA) device driver support.
> ;        4. For PSC's XU (DEUNA or DELUA) device driver support.
> ;
> * Network Device Option [D R:1.-4. D:1.]: 1
> ;
> ;     Enter the Direct Line Access Line Specification.
> ;
> ;     Examples: QNA-0, UNA-0.
> ;
> * DLX Line Specification [S R:5.-5. D:"QNA-0"]: QNA-0
> ;
> ;     Enter the number of Host Name Table entries.
> ;
> ;     This is the maximum number of entries permitted in the Host Name
> ;     Table.
> ;
> * Number of Host Name Table entries [D R:32.-200. D:64.]: 200
> ;
> MCR TKB @INDRV.TKB
> ;
> MCR TKB @INSACP.TKB
> ;
> MCR TKB @NETCU.TKB
> ;
> ;     Indicate whether you want to build the FTP-RSX components.
> ;     Answer YES if you purchased FTP-RSX and are licensed to run
> ;     FTP-RSX on this system.
> ;
> * Do you want to build the FTP-RSX components? [Y/N D:N]: Y
> ;
> MCR TKB @UFTP.TKB
> ;
> ;     Enter the name of the Event Logging subroutine module for the
> ;     Server-FTP (FTPD).
> ;
> ;     If you do not require Event Logging or have not written a
> ;     module, use the default module (NULLOG).
> ;
> ;     Refer to the FTP-RSX User's Guide for more details on the Event
> ;     Logging subroutine.
> ;
> * Event Logging module name [S R:1.-32. D:"NULLOG"]: NULLOG
> ;
> MCR TKB @FTPD.TKB
> ;
> ;     Indicate whether you want to build the TELNET-RSX components.
> ;     Answer YES if you purchased TELNET-RSX and are licensed to run
> ;     TELNET-RSX on this system.
> ;
> * Do you want to build the TELNET-RSX components? [Y/N D:N]: Y
> ;
> MCR TKB @TELNET.TKB
> ;
> MCR MAC INTSUB=LB:[1,1]EXEMC/ML,[1,54]RSXMC/PA:1,DU0:[1,14]INTSUB
```

```
> MCR TKB @INTELD.TKB
> ;
> MCR TKB @TELCU.TKB
> ;
> MCR SET /UIC=[1,3]
> MCR PIP LB:[1,2]/NV=DU0:[1,14]STARTNET.CMD,SHUTNET.CMD
> MCR PIP LB:[1,54]/NV=DU0:[1,14]NETCU.TSK,INDRV.TSK,INDRV.STB,INSACP.TSK
> MCR PIP LB:[1,54]/NV=DU0:[1,14]UFTP.TSK
> MCR PIP LB:[1,1]/NV=DU0:[1,14]FTPLIB.OLB
> MCR PIP LB:[1,54]/NV=DU0:[1,14]FTPD.TSK
> MCR PIP LB:[1,54]/NV=DU0:[1,14]TELNET.TSK
> MCR PIP LB:[1,1]/NV=DU0:[1,14]TELLIB.OLB
> MCR PIP LB:[1,54]/NV=DU0:[1,14]TELCU.TSK
> MCR PIP LB:[1,54]/NV=DU0:[1,14]INTELD.TSK
> MCR PIP LB:[1,2]/NV=DU0:[1,14]TCPWARE.HLP
> MCR SET /UIC=[1,14]
> ;
> ;  ****************************************************************
> ;
> ;     Proceed to the next step, "Configuring the Software."
> ;     You can repeat this configuration by typing:
> ;             @DU0:[1,14]CNFNET CONFIGURE
> ;
> ;
> ;     NOTE: Internet addresses are NOT validated at this time. If you
> ;     enter improper or incorrect values, the network startup
> ;     (@LB:[1,2]STARTNET) will fail.
> ;
> ;
> ;     Enter the local host's internet address. Examples of valid
> ;     internet address are:
> ;             1.0.2.3         132.4.5.34      192.127.50.1
> ;
> ;     NOTE: Internet and Ethernet addresses are not related. The
> ;     Address Resolution Protocol (ARP) maps internet addresses to
> ;     Ethernet addresses.
> ;
> * Local host's internet address [S R:7.-15.]: 1.0.0.3
> ;
> ;     Enter the default gateway's internet address. A gateway address
> ;     is required only if your network has a gateway (to other
> ;     networks) and you want to communicate with machines on other
> ;     networks. The default gateway's internet address MUST be on the
> ;     local network.
> ;
> ;     Press <return> if no default gateway exists.
> ;
> ;     Examples of valid internet address are:
> ;             1.0.2.3         132.4.5.34      192.127.50.1
> ;
> * Default gateway's internet address [S R:0.-15. D:""]:
> ;
> ;     Enter the size of the INPOOL common. TCPware uses this common
> ;     for communications buffer space.
> ;
> ;     Allow about 20K bytes for receive buffers and the Host Name.
> ;     Table. Allow an additional 4K bytes for each active connection.
> ;
```

```
> ;      You need about 54K bytes to support 8 active connections.
> ;
> ;      NOTE: These numbers are approximate. Their values depend on
> ;      which network device option you selected and the size of the
> ;      Host Name Table.
> ;
> * Size of INPOOL (in K bytes) [D R:28.-1024. D:48.]: 48
> ;
> ;      Enter the number of simultaneous, remotely initiated connections
> ;      you want FTP-RSX to support. This is the number of Server-FTP
> ;      tasks you want to run on this system.
> ;
> ;      These server tasks are checkpointable when they are not
> ;      servicing a remote connection. They also do not use up
> ;      active connections while waiting for a remotely initiated
> ;      connection.
> :
> ;      If you are not allowing any remotely initiated connections,
> ;      specify 0 for this parameter.
> ;
> * Number of Server-FTP (FTPD) tasks [D R:0.-9. D:2.]: 2
> ;
> ;      Enter the number of simultaneous, remotely initiated sessions
> ;      you want TELNET-RSX to support. This is the number of remote
> ;      TELNET users you want to support simultaneously.
> ;
> ;      If you are not allowing any remotely initiated connections,
> ;      specify 0 for this parameter.
> ;
> * Number of Server-TELNET sessions [D R:0.-16. D:0.]: 2
> ;
> MCR SET /UIC=[1,3]
> MCR PIP LB:[1,2]STARTNET.DAT/PU/NM
> MCR SET /UIC=[1,14]
> MCR SET /UIC=[1,3]
> ;
> ;      Enter the host names and the corresponding internet address for
> ;      hosts on the network. The host names are used locally and need
> ;      not match the names used on other hosts. However, the internet
> ;      addresses MUST correspond to the host.
> ;
> ;      When you are done, simply press return in response to the
> ;      Host’s Name prompt.
> ;
> ;      If you need to correct a mistake or want to edit the file
> ;      manually, you can do so later by using any text editor.
> ;
> * Host’s Name (<return> to end) [S R:0.-32. D:””]: ALPHA
> * Internet Address for alpha [S R:0.-15. D:””]: 1.0.0.1
> ;%CNFNET-I-ADDED, added alpha (1.0.0.1)
> ;
> * Host’s Name (<return> to end) [S R:0.-32. D:””]: BETA
> * Internet Address for BETA [S R:0.-15. D:””]: 1.0.0.2
> ;%CNFNET-I-ADDED, added BETA (1.0.0.2)
> ;
> * Host’s Name (<return> to end) [S R:0.-32. D:””]: GAMMA
> * Internet Address for GAMMA [S R:0.-15. D:””]: 1.0.0.3
> ;%CNFNET-I-ADDED, added GAMMA (1.0.0.3)
```

```
> ;
> * Host's Name (<return> to end) [S R:0.-32. D:""]:
> MCR SET /UIC=[1,14]
> ;
> ;  *************************************************************
> ;
> ;     Proceed to the next step, "Obtaining the Software Password."
> ;
> ;
> ;     Loading components required to display password information.
> ;
> MCR LOAD IN:/PAR=GEN/HIGH
LOA -- Warning - loadable driver larger than 4K
> MCR CON ONLINE ALL
> MCR INS $INSACP
> MCR INS $NETCU
> ;
> MCR NCU SHOW PASSWORD

Contact your distributor or Process Software Corporation with the
following information to obtain a software password for this system:

        o  The hardware ID: "2B-01-02-03".

        o  The system ID:   "82-00".

        o  The product name and version number for the product(s)
           you are installing on this system.

Refer to your product documentation for more information.
> ;
> ;     If you are unable to obtain the software password at this time,
> ;     press <return>.
> ;
> * Password [S R:0.-132.]: 81BD-D640-CTUE-C86C-3F62
> ;
> MCR SET /UIC=[1,3]
> MCR NCU SET PASSWORD "81BD-D640-CTUE-C86C-3F62"
> MCR SET /UIC=[1,14]
> ;
> ;     TCPware has been built and configured. Now you need to perform
> ;     the following operations:
> ;        o Start up TCPware by typing:
> ;             @LB:[1,2]STARTNET
> ;        o Test TCPware.
> ;        o Edit the system startup procedure to invoke the TCPware
> ;          startup procedure. Add the following line to the file:
> ;             @LB:[1,2]STARTNET
> ;          NOTE: You must add this line AFTER starting DECnet.
> ;
> ;     See the TCPware for RSX Software Reference Manual for additional
> ;     information.
> ;
> @ <EOF>
>
>
> @LB:[1,2]STARTNET
> MCR REM ...NCU
```

```
> MCR INS $NETCU
> MCR NCU CREATE 48.K GEN
> MCR SET /UIC=[1,3]
> MCR NCU START 1.0.0.3
> MCR SET /UIC=[1,14]
> MCR NCU @LB:[1,2]HOSTS.CMD
> MCR INS $UFTP
> MCR INS $FTPD/TASK=FTPD.1
> MCR RUN FTPD.1 2S
> MCR INS $FTPD/TASK=FTPD.2
> MCR RUN FTPD.2 2S
> MCR INS $TELNET
> MCR INS $TELCU
> MCR TCU CREATE 2 GEN
> MCR CON ONLINE ALL
> MCR TCU START
> MCR REM ...TCU
> @ <EOF>
```

4
General Information

This chapter provides background information on TCP-RSX. It includes an overview of the major software components and how they interact.

4.1 Network Hardware Configuration

To operate the software, each computer should be configured with one of the supported network controllers (DEC's UNIBUS DELUA or DEUNA, DEC's Q-BUS DELQA or DEQNA) and associated interconnecting hardware and cabling. The hardware components are essentially identical on each computer system.

During most communication operations, the communications network is configured logically as a two-node, point-to-point link between the two computers involved in the communication. The network may be physically part of a larger multi-node network with no impact on the other nodes in the network.

4.2 Communications Protocol

The TCPware for RSX software products use the Transmission Control Protocol (TCP), User Datagram Protocol (UDP), and Internet Protocol (IP). These protocols provides a reliable and efficient means for moving information between computer systems.

The Department of Defense (DoD) has adopted the IP and TCP protocols as DoD-wide standards for all DoD packet networks. All new DoD packet networks will be using these protocols exclusively.

4.3 Component Overview

The TCPware for RSX contains several components. Figure 4-1 shows how they interrelate.

The purpose and function of each component are described below.

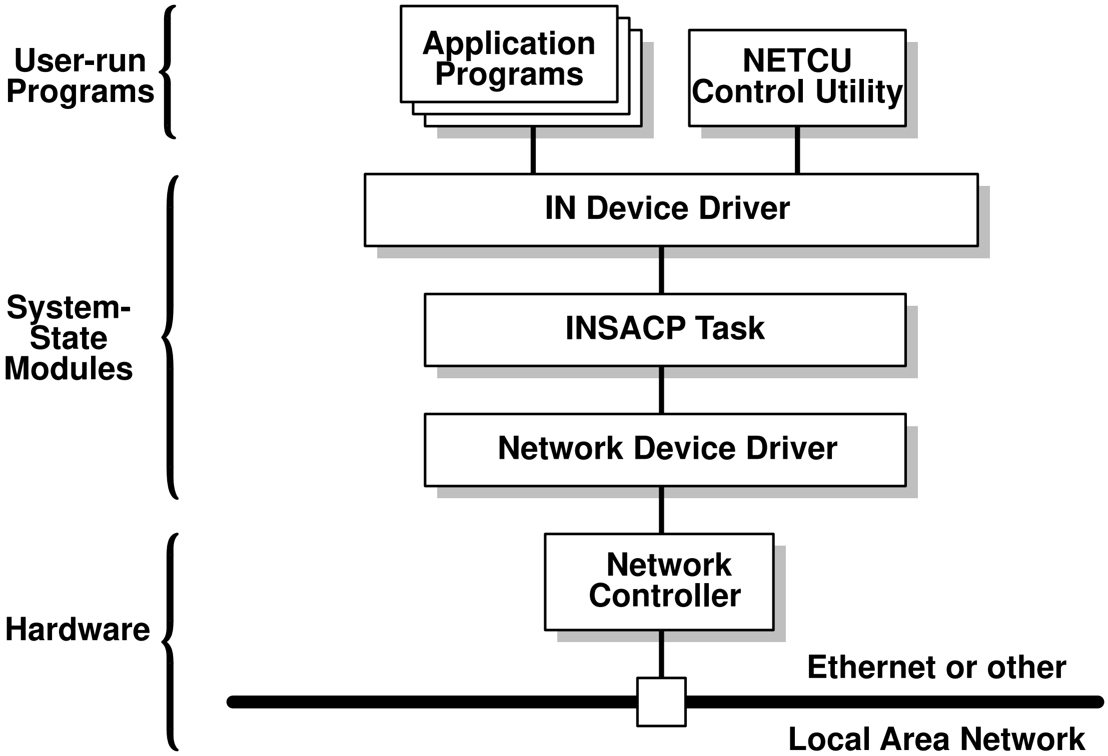

Figure 4-1: TCPware for RSX Software Components

4.3.1 Application Programs

The application programs use TCPware for RSX programming services. The *TCP Services* chapter provides more information on the programming services.

The User-FTP utility and Server-FTP task provided with FTP-RSX and the User-TELNET utility provided with TELNET-RSX are examples of application programs.

4.3.2 IN Device Driver, INSACP, and NETCU

The IN device driver, INSACP, and NETCU all work together to provide the lower protocol layers of the TCP/IP protocol stack (TCP, UDP, IP, ARP).

The IN device driver implements most of the lower layer protocols. INSACP provides an interface for the IN device driver to the network device and also provides some support functions for IN. The NETCU utility is used to start, stop, and control the networking software.

The IN device driver provides the TCP and UDP programming services and is used by the application programs to communicate with other systems on the network using the TCP/IP protocols.

The TCP and UDP programming services are described in the *IN Device Driver Services* chapter. The NETCU utility commands are described in the *Network Control Utility* chapter.

4.3.3 Network Device Driver

The network device driver operates the network controller hardware.

TCPware supports the following combinations of network controller hardware and network device drivers:

- DEC Q-BUS Ethernet controller (DEQNA or DELQA) using Process Software Corporation's XQ device driver (provided with TCP-RSX).
- DEC UNIBUS Ethernet controller (DEUNA or DELUA) using Process Software Corporation's XU device driver (provided with TCP-RSX).
- DEC Q-BUS or UNIBUS Ethernet controller (DEQNA, DELQA, DEUNA, or DELUA) using the DECnet/RSX's DLX (direct line access) support.
- DEC UNIBUS Ethernet controller (DEUNA or DELUA) using the XE device driver provided with the newer releases of RSX-11M and RSX-11M-PLUS.

TCPware provides two device drivers: XQ and XU. These are user-installable and designed for exclusive use by TCPware components.

The Ethernet device drivers (XQ or XU) support one DELQA, DEQNA, DELUA, or DEUNA network controller.

TCPware can also use DECnet's direct line access (DLX) interface.

4.4 Definitions

This section defines some of the terms used in this document.

- **Active Open -** An active open is used to actively open a connection. When an active open is requested, segments are sent to establish the connection with the destination host. A passive open must usually be pending at the destination host for the connection to be established.
- **Host Name -** A host name is a mnemonic given to a host in order to identify it on the local host. The host names are only used locally and need not match the names used on other hosts.

  Host names are from 1 to 32 characters long. Upper and lowercase characters are equivalent.
- **Host Name Table -** The host name table is maintained by the networking software. Each entry in the table consists of a host name and the internet address for that host. The host name table functions are used to maintain and query the table.
- **Internet Address -** Each host on a system is assigned a unique internet address. The lower layer protocols use the internet address to identify each host on an internet. An internet address is a 32-bit value. For more information on internet addresses, see the *Overview of TCP/IP Networking* chapter.

  Internet addresses are specified in reverse PDP-11 byte order (the most significant byte of the internet address is the least significant byte of the 2-word value). The internet address 1.2.3.4 is stored as .BYTE 1,2,3,4

- **Passive Open -** A passive open is used to “listen” and wait for a request from a remote host to establish a connection. Passive opens can be fully specified or partially specified. Partially specified opens are used when the requesting host is not known (for example, the server FTP specifies only the local port number). A passive open does not send segments to establish a connection until a segment is received from another host (that has issued an active open).
- **Port Number -** Port numbers are used by the Transmission Control Protocol and User Datagram Protocol to name the ends of logical connections. Port numbers are 16-bit values.

  The server port number for the FTP protocol is 21. Generally, servers use the port numbers from 0 to 255.

  User port numbers start at 1024.

  Port numbers are specified in normal PDP-11 byte order.

4.5 References

For more information on TCPware for RSX software products, refer to the following documentation:

- *FTP-RSX Software User’s Guide*
- *TELNET-RSX Software User’s Guide*

For information on the RSX-11M, RSX-11M-PLUS, and MicroRSX operating systems and utilities, refer to the appropriate DEC documentation.

For more information on the Internet Protocol (IP), Transmission Control Protocol (TCP), User Datagram Protocol (UDP) and Address Resolution Protocol (ARP), refer to the Request for Comments documents published by the Network Information Center (NIC) at Government Systems, Inc., 14200 Park Meadow Drive, Suite 200, Chantilly, VA 22021 (800-365-3642, 703-802-4535 (7:00 AM to 7:00 PM EST)). You can also send electronic mail directed to the “HOSTMASTER” and “REGISTRAR” mailboxes at network address NIC.DDN.MIL The RFCs contain the specifications for all internet protocols.

The following RFCs contain the protocol specifications and related information for the protocols implemented within TCP-RSX:

- *Internet Protocol*, RFC-791
- *User Datagram Protocol,* RFC-768
- *Transmission Control Protocol*, RFC-793
- *An Ethernet Address Resolution Protocol*, RFC-826
- *Trailer Encapsulations*, RFC-893
- *A Standard for the Transmission of IP Datagrams over Ethernet Networks*, RFC-894
- *Broadcasting Internet Datagrams*, RFC-919
- *Broadcasting Internet Datagrams in the Presence of Subnets*, RFC-922
- *Internet Standard Subnetting Procedure*, RFC-950
- *Assigned Numbers*, RFC-1060

The following documents contain more information on the Ethernet hardware and specification:

- *Ethernet Data Link Layer and Physical Layer Specifications*, DEC order no. AA-K759B-TK.
- The hardware documentation on your Ethernet controller, available from DEC.

5
Network Control Utility (NETCU)

This chapter describes how to use the Network Control Utility (NETCU). This utility is primarily for system managers.

NETCU provides commands for starting up and shutting down TCPware. It also provides commands to manipulate the Host Name Table and to display network status information. You can run NETCU directly from your terminal or from a command file.

To run NETCU, enter one of the following commands at the DCL/MCR prompt:

```
NCU
```

or

```
RUN $NETCU
```

The “NETCU>” prompt displays. This means NETCU is ready to accept your commands.

NETCU supports the following commands:

- Network Management:
  - **CREATE**
  - **FLUSH/NETWORK**
  - **START**
  - **STOP**
  - **SHOW CONNECTIONS**
  - **SHOW NETWORK**
- Software Password Management:
  - **SET PASSWORD**
  - **SHOW PASSWORD**

- Host Name Table Management:
  - **ADD**
  - **FIND**
  - **IFIND**
  - **LIST**
  - **REMOVE**
- **HELP**
- **EXIT**

5.1 CREATE Command

TCPware uses the CREATE command to create the INPOOL region when the networking software is started.

Format

CREATE *size* [*partition*]

Parameters

size

The size of the INPOOL region.

Refer to the *Installation* chapter for details on this parameter.

If you need to change the size of INPOOL once it is created, you need to first remove the region by entering CREATE 0. You can enter size 0 only if the region is not in use.

partition

The name of the partition in which the INPOOL region is created. This parameter is optional. If you omit it, GEN is used.

NOTE

> If your RSX system does not support the PLAS directives, you cannot use the CREATE command. You can still use TCPware by creating the INPOOL region using VMR or the supplied LDINPL.CMD command file.

5.2 FLUSH Command

Use the FLUSH command to erase the contents of the network caching tables. This command is provided primarily to erase the contents of the Ethernet Address Resolution Protocol (ARP) cache table.

Format

FLUSH

You must be privileged to use the FLUSH command.

Qualifier

/NETWORK

Required. This qualifier is used to request that the Ethernet ARP tables be flushed. All Ethernet ARP table entries are removed from the cache.

5.3 START Command

TCPware uses the START command to start the network when you execute the @LB:[1,2]STARTNET command file.

NOTE

> Use the @LB:[1,2]STARTNET command file to start up the network. It performs several operations described in the *Installation* chapter.

Format

START *local-address* [*gateway-address* [*network-mask*]]

This command defines the local internet address, default gateway internet address (if any), and network mask for the internet address (only specified if your network uses subnets).

The START command first attempts to load the INPOOL region into memory, if not already resident. Next, a special request is issued to IN: that will start the IN device driver and INSACP.

You must be privileged to use the START command.

Parameters

local-address

The internet address of the local host.

gateway-address

The internet address of the default gateway.

network-mask

The internet address network mask for your network.

Refer to the *Installation* chapter for details on these parameters.

5.4 STOP Command

TCPware uses the STOP command to shut down the network when you execute the @LB:[1,2]SHUTNET command file.

NOTE

> Use the @LB:[1,2]SHUTNET command file to shut down the network. It performs several cleanup operations described in the *Installation* chapter.

Format

STOP

When you enter this command, the IN device driver and INSACP are signaled to immediately stop all network activity. All connections are aborted immediately.

You must be privileged to use the STOP command.

5.5 SHOW Command

Use the SHOW command to display information about the network.

Format

SHOW CONNECTIONS

SHOW NETWORK

SHOW PASSWORD

5.5.1 SHOW CONNECTIONS Command

Use the SHOW CONNECTIONS command to display a list of the connections currently active (including listening servers).

Format

SHOW CONNECTIONS

For each connection the following are displayed: task name, receive queue byte count, send queue byte count, local internet address and port number, foreign internet address and port number, and connection state.

5.5.2 SHOW NETWORK Command

Use the SHOW NETWORK command to display the following information about the network:

- the amount of space that is allocated to and available in INPOOL, in octal bytes
- the number of network packets, IP datagrams, and TCP segments transmitted and received

Format

SHOW NETWORK

5.5.3 SHOW PASSWORD Command

Use the SHOW PASSWORD command to display the software password information.

Format

SHOW PASSWORD

This command displays the software password information that you must provide to your distributor or Process Software Corporation in order to get a software password. See the *Installation* chapter for additional information on getting a software password.

You must be privileged to use the SHOW PASSWORD command.

5.6 SET PASSWORD Command

Use the SET PASSWORD command to set the software password.

Format

SET PASSWORD "*software-password*"

Parameter

software-password

The software password. The *Installation* chapter explains how to obtain a software password.

You must be privileged to use the SET PASSWORD command.

5.7 ADD Command

Use the ADD command to add a host name/internet address pair to the Host Name Table.

Format

ADD *host-name internet-address*

When a host name and internet address are defined in the Host Name Table, users can connect to a host without knowing its internet address.

Use this command to define the host names and internet addresses for all hosts in the network. You can specify alias host names because more than one host name can have the same internet address.

A host can still be accessed if it is not defined in the Host Name Table. You can always specify a host's internet address when connecting to that host.

You must be privileged to use the ADD command.

Parameters

host-name

The name of the remote host you are adding to the Host Name Table.

Host names can contain alphanumeric characters, dots (.), hyphens (-), and underscores (_). They can also contain any printable non-control characters, except for spaces and tabs.

internet-address

The corresponding internet address of the remote host.

5.8 FIND Command

Use the FIND command to display the internet address for the specified host name.

Format

FIND *host-name*

Parameter

host-name

The name of the remote host you want to find. The name and internet address for that host will be displayed if found in the Host Name Table.

5.9 IFIND Command

Use the IFIND command to display the host name for the specified internet address.

Format

IFIND *internet-address*

Parameter

internet-address

The internet address of the remote host you want to find. The name and internet address for that host will be displayed if found in the Host Name Table.

If more than one host name exists for the internet address, only the alphabetically first name will be displayed.

5.10 LIST Command

Use the LIST command to list all entries in the Host Name Table.

Format

LIST

The name and internet address for each entry in the Host Name Table is displayed.

5.11 REMOVE Command

Use the REMOVE command to remove a host name/internet address pair from the Host Name Table.

Format

REMOVE *host-name*

You must be privileged to use the REMOVE command.

Parameter

host-name

The name of the remote host you want to remove from the Host Name Table.

5.12 HELP Command

Use the HELP command to obtain a brief list of the available commands and the syntax of each.

Format

HELP

5.13 EXIT Command

Use the EXIT command to exit from the NETCU utility. You can also use ^Z to exit from NETCU.

Format

EXIT

NOTE

The NETCU utility exits with status (using the EXST$S directive). It returns the EX$SUC status code only if no errors occurred. If an error did occur, NETCU returns the EX$ERR status code. This status information is useful primarily to indirect command files.

6
IN Device Driver Services

This chapter describes the Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) programming services.

TCP is a connection-oriented, full-duplex, end-to-end reliable protocol. It provides for reliable intertask communication between pairs of tasks in computers that are interconnected.

TCP services are available through the RSX QIO$ and QIOW$ directives. You can use these directives to:

- Open and close connections
- Send and receive data over connections
- Perform status checks on connections

UDP allows an application on one machine to send a datagram to an application on another machine. While these application programs can send messages to other programs with minimum effort, UDP is transaction-oriented, and therefore does not guarantee delivery and non-duplication of data. If your application requires ordered and reliable delivery of data, you should use TCP.

UDP services are also available through the RSX QIO$ and QIOW$ directives. You can use these directives to:

- Open and close a port
- Send and receive data over the port
- Perform status checks on the port

This chapter describes only the user interface of this implementation of TCP and UDP. The TCP specifications are in RFC-793 (see the *References* section in the *General Information* chapter). The UDP specifications are in RFC-768 (see the *References* section in the *General Information* chapter).

The *Sample Program* appendix contains examples of a program that uses TCP services and a program that uses UDP services.

6.1 Sequence of Operations

The following sequence of operations is typically used to open a connection:

1. **Assign** a logical unit number to the IN0: device using the ALUN$ directive or the FORTRAN ASNLUN subroutine.
2. **Open** the port or connection using either a passive or active open function (TCP).
3. After the connection or port has opened, **perform** receive, send, and control functions as desired.
4. After all data has been received and/or sent, **close** the connection or port using the close function.

Use the QIO$ and QIOW$ directives (or the FORTRAN QIO and WTQIO subroutines) to open and close connections, to send and receive data over connections, and to perform control functions on connections.

The synchronous and asynchronous forms of the directives can be used, depending on your application's requirements.

Refer to the appropriate RSX-11M, RSX-11M-PLUS, or MicroRSX documentation for information on the directives (ALUN$, QIO$, QIOW$) and FORTRAN subroutines (ASNLUN, QIO, WTQIO) and on related directives, subroutines, and concepts (such as those used with asynchronous system traps (ASTs) and event flags).

6.2 Open Connection Function (TCP)

Use the open function to open an active or a passive connection. An active open initiates a connection. A passive open waits for an active open to complete the connection.

Before issuing this request, assign a logical unit number to the IN0: device. You will need a different logical unit number for each connection you want to open simultaneously.

You need to specify the port numbers and internet addresses to be used in a buffer. You can use the "lookup internet address of host name" function described in the *Host Name Functions* section to translate a host name to its internet address (this function also converts an ASCII internet address string to binary form).

The open function is a buffered-I/O function (if supported by your RSX system). This means the task may be checkpointed while waiting for the connection to open.

Function Code

IO.CTL!1 for an active open or IO.CTL!201 (octal) for a passive open

NOTE

> The standard RSX I/O function codes and I/O status codes are defined by the QIOSY$ macro. Refer to the DEC *I/O Drivers Reference Manual* for additional details on I/O function codes and I/O status values.

Parameters

P1 contains the address of the buffer containing the internet address and port number information. The address must be even. This buffer contains:

- Local internet address (2 words). This value is currently ignored but should be 0.
- Local port number (1 word). For a passive open, this is the port number on which to listen for incoming requests. If this value is 0, a unique port number is assigned for you.
- Foreign internet address (2 words). For a passive open, you can enter this value to restrict incoming requests to a particular system. For an active open, this value is required. If this value is 0, IN driver assumes it is not specified.
- Foreign port number (1 word). For a passive open, you can enter this value to restrict incoming requests to a particular port. For an active open, this value is required. If this value is 0, IN driver assumes it is not specified.

For an active open, you must specify both the foreign internet address and foreign port number as non-zero values. For a passive open, you must specify the foreign internet address and the foreign port number or neither the address nor the port number. You cannot specify only one.

P2 contains the byte length of the buffer specified in **P1**. This value must be 12.

P3 contains the time-out time in seconds. This unsigned 16-bit number is used as the default time-out time. It If you specify 0, five minutes (300 seconds) is used as the default. The minimum value is 20 seconds. This parameter is ignored for passive opens. If you specify this parameter for a passive open, it is used only after a connection has been opened.

P4 contains the window size. This value can be 0 (for the default size of 2048), or from 256 to 4096. The window size controls how many bytes of unsolicited data the TCP layer allows the peer to send before it prevents the peer from sending more. For some applications, such as a control connection in which commands and replies are exchanged, it makes sense to use small values and conserve INPOOL resources. For other applications, such as a data connection in which files are transferred, larger sizes can improve performance.

Status Codes

This request can return the following function-specific status codes:

- **IE.ABO** - if the open request was aborted (using the abort or IO.KIL function).
- **IE.ALN** - if a connection or port has already been opened for the logical unit number. Use a different logical unit number or close the previous connection or port.
- **IE.BAD** - if an invalid combination of internet addresses and port numbers was specified, or the combination is not unique (if fully specified). Check the values specified.
- **IE.BYT** - if the buffer address or buffer length is odd.
- **IE.NDR** - if there is insufficient dynamic memory available for the connection. Close some other connections or ports or increase the amount of memory allocated to INPOOL during network startup.
- **IE.OFL** - if the networking software is not active. Ask your system manager to start the networking software.
- **IE.ONP** - if the software password is invalid or expired. See your system manager.

- **IE.RES** - if the networking software was shut down while the open request was pending.
- **IE.SPC** - if the buffer specified is invalid. Check the length and address specified in the request.
- **IE.TMO** - if the active open timed-out, or the connection was rejected by the remote system.

NOTE

If an open function fails, you must issue the abort function. This ensures that everything within the IN device driver is cleaned up.

6.3 Open Port Function (UDP)

Use the open port function to open a new UDP port.

When opening a port, you have a choice of either fully or partially specifying the port characteristics.

To open a fully specified port, you must specify both the foreign address and foreign port number. UDP then delivers only datagrams received from that address and port and destined for the local port number.

To open a partially specified port, do not specify the foreign address and foreign port number. UDP then delivers all datagrams destined for the local port number, except those to be delivered to fully specified ports for the same local port number.

Before issuing this request, assign a logical unit number to the IN0: device. You will need a separate logical unit number for each port you want to open simultaneously.

The open function is a buffered-I/O function (if supported by your RSX system). This means the task may be checkpointed while waiting for the port to open.

Function Code

IO.CTL!101 for an open port.

NOTE

The standard RSX I/O function codes and I/O status codes are defined by the QIOSY$ macro. Refer to the DEC *I/O Drivers Reference Manua*l for additional details on I/O function codes and I/O status values.

Parameters

P1 contains the address of the buffer containing the internet address and port number information. The address must be even. This buffer contains:

- Local internet address (2 words). This value is currently ignored but should be 0.
- Local port number (1 word). If this value is 0, a unique port number is assigned for you.
- Foreign internet address (2 words). If this value is 0, IN driver assumes it is not specified.
- Foreign port number (1 word). If this value is 0, IN driver assumes it is not specified.

If you specify a foreign address, you must specify a foreign port number. If you specify a foreign port number, you must specify a foreign address. You cannot specify only one.

P2 contains the byte length of the buffer specified in **P1**. This value must be 12.

Status Codes

This request can return the following function-specific status codes:

- **IE.ALN** - if a connection or port has already been opened for the logical unit number. Use a different logical unit number or close the previous connection or port.
- **IE.BAD** - if an invalid combination of internet addresses and port numbers was specified, or the combination is not unique (if fully specified). Check the values specified.
- **IE.BYT** - if the buffer address or buffer length is odd.
- **IE.NDR** - if there is insufficient dynamic memory available for the port. Close some other connections or ports or increase the amount of memory allocated to INPOOL during network startup.
- **IE.OFL** - if the networking software is not active. Ask your system manager to start the networking software.
- **IE.ONP** - if the software password is invalid or expired. See your system manager.
- **IE.RES** - if the networking software was shut down while the open request was pending.
- **IE.SPC** - if the buffer specified is invalid. Check the length and address specified in the request.

NOTE

If an open function fails, you must issue the abort function. This ensures that everything within the IN device driver is cleaned up.

6.4 Receive Function

Use the receive function to receive data over a connection or port. The received data is returned in the buffer specified.

For TCP, the amount of data returned to the buffer may be less than that requested if a push was sent by the remote system. If a push caused the receive to be completed, the high byte of the first word in the I/O status block is 1 (otherwise, it is 0). The second word of the I/O status block always contains the number of bytes returned to the buffer.

For UDP, the amount of data returned depends on the size of the datagram sent to you. This size may be less than the size requested by P2, below.

Function Code

IO.RVB or IO.RLB

Parameters

P1 contains the address of the buffer to receive the data. The buffer address may be odd.

P2 contains the byte length of the buffer specified in **P1**. The byte count may be odd.

P4 contains the address of the 6-byte buffer to receive the foreign internet address and port number. Specify this parameter for UDP only. When the receive function completes, the buffer contains:

- The foreign internet address (2 words)
- The foreign port number (1 word)

Status Codes

This request can return the following function specific status codes:

- **IE.ABO** - if the receive request was aborted (using the abort or IO.KIL function).
- **IE.DAO** - if the received datagram was larger than the buffer specified by P2, the truncated datagram is returned. (UDP only)
- **IE.NLN** - if no connection or port has been opened on the logical unit number.
- **IE.EOF** - to indicate that the remote host closed its end of the connection (TCP only). If both ends have now issued a close, the connection is fully closed.
- **IE.OFL** - if the networking software is not active. Ask your system manager to start the networking software.
- **IE.RES** - if the networking software was shut down while the receive request was pending.
- **IE.TMO** - if the connection timed-out (TCP only). Usually means that the remote system has crashed or been shut down.
- **IE.SPC** - if the buffer specified is invalid. Check the length and address specified in the request.

NOTE

The TCP protocol is stream oriented, not message oriented. This means that data sent might be received in more than one buffer, or data from several sends might be received in one buffer.

6.5 Send Function

Use the send function to send data over an open connection or port. The request is completed as soon as the data has been queued for transmission (therefore, a successful status does not mean that the other system has received the data).

For a connection, you should *always* use the push subfunction code unless you can issue another send request without waiting for the previous one to complete (you cannot reuse a buffer until the request that used that buffer completes).

Function Code

IO.WVB or IO.WLB

IO.WVB!200 or IO.WLB!200 (octal) to request a push of the data

Parameters

P1 contains the address of the data buffer to be sent. The buffer address may be odd.

P2 contains the byte length of the buffer specified in **P1**. The byte count may be odd.

P3 contains the time-out time in seconds (optional). If the time-out time is not specified, the current value is not changed.

P4 contains the address of an optional 6-byte buffer that contains the foreign internet address and foreign port number. This parameter is optional if you opened a fully specified port. Specify this parameter for UDP only. See the *Receive Function* section for the buffer's format.

Status Codes

This request can return the following function-specific status codes:

- **IE.ABO** - if the send request was aborted (using the abort or IO.KIL function).
- **IE.NLN** - if no connection or port has been opened on the logical unit number.
- **IE.OFL** - if the networking software is not active. Ask your system manager to start the networking software.
- **IE.RES** - if the networking software was shut down while the send request was pending.
- **IE.TMO** - if the connection timed-out (TCP only). Usually means that the remote system has crashed or been shut down.
- **IE.SPC** - if the buffer specified is invalid. Check the length and address specified in the request.

NOTE

The TCP protocol is stream oriented, not message oriented. This means that data sent might be received in more than one buffer, or data from several sends might be received in one buffer.

6.6 Close Function

Use the close function to close your end of the connection or port. All outstanding data is transmitted, and no additional send requests are accepted.

This request does not complete until both ends have closed the connection. If the remote host has not closed the connection, you should continue to issue receive requests until it has done so.

Function Code

IO.CTL!2

Parameters

None.

Status Codes

This request can return the following function-specific status codes:

- **IE.ABO** - if the close request was aborted (using the abort or IO.KIL function).
- **IE.NLN** - if no connection or port has been opened on the logical unit number.
- **IE.OFL** - if the networking software is not active. Ask your system manager to start the networking software.
- **IE.RES** - if the networking software was shut down while the close request was pending.
- **IE.TMO** - if the connection timed-out (TCP only). Usually means that the remote system has crashed or been shut down.

6.7 Abort Function

Issue the abort function to stop using a connection or port immediately. All pending requests are aborted.

Function Code

IO.CTL!3

Parameters

None.

Status Codes

This request can return the following function-specific status codes:

- **IE.NLN** - if no connection or port has been opened on the logical unit number.
- **IE.OFL** - if the networking software is not active. Ask your system manager to start the networking software.

6.8 Status Function

Issue the status function to get information on a connection or port. This request may be issued after a port or connection is opened (either passive or active, even before the open completes).

This request returns the following status information for the connection:

- Local internet address (2 words)
- Local port number (1 word)

  If a local port number was not specified, the value of the assigned port number is returned.
- Foreign internet address (2 words)

  This value may be 0 if a passive open is pending on the connection.
- Foreign port number (1 word)

  This value may be 0 if a passive open is pending on the connection.

- Connection state (1 word)

State	**Value (octal)**
Listen	0001
SYN-Sent	0002
SYN-Received	0004
Established	0010
Fin-Wait-1	0020
Fin-Wait-2	0040
Close-Wait	0100
Closing	0200
Last-ACK	0400
Time-Wait	1000
Closed	2000

  This value is 0 for a UDP port.

  For more information on the meaning of each state, refer to RFC-793 (see the *References* section in the *General Information* chapter).

- Connection time-out time in seconds (1 word)

Function Code

IO.CTL!4

Parameters

P1 contains the address of the status information buffer to be filled in. It must be 16 bytes long. The address must be even.

P2 is the byte length of the buffer specified in **P1**. This must be 16.

Status Codes

The following function-specific status codes can be returned by this request:

- **IE.BYT** - if the buffer address or buffer length is odd.
- **IE.NLN** - if no connection or port has been opened on the logical unit number.
- **IE.OFL** - if the networking software is not active. Ask your system manager to start the networking software.
- **IE.SPC** - if the buffer specified is invalid. Check the length and address specified in the request.
- **IE.TMO** - if the connection timed-out (TCP only). Usually means that the remote system has crashed or been shut down.

6.9 Show Connections Function

You can use this function to get information on all of the active connections. Information regarding each connection, up to the number active or the available buffer space, is returned in the buffer specified.

For each active or listening connection, the following information is returned:

- Address of the connection's transmission control block in pool (1 word)
- Task name in RAD50 of the owner of this connection (2 words)
- Number of bytes in this connection's receive queue (1 word)
- Number of bytes in this connection's send queue (1 word)
- Local internet address of this connection (2 words)
- Local port number of this connection (1 word)
- Foreign internet address of this connection (2 words)
- Foreign port number of this connection (1 word)
- Connection's state bit mask (1 word) or 0 for UDP (see the *Status Function* section for the state definitions)

The second word of the I/O status block contains the number of bytes of data returned in the buffer.

Function Code

IO.CTL!6

Parameters

P1 contains the address of the buffer into which to return the information. The address must be even.

P2 contains the byte length of the buffer specified in **P1**. Each connection's or port's information requires 24 bytes.

Status Codes

The following function-specific status codes can be returned by this request:

- **IE.BYT** - if the buffer address or buffer length is odd.
- **IE.DAO** - if the buffer was too small to return all of the available information.
- **IE.OFL** - if the networking software is not active. Ask your system manager to start the networking software.
- **IE.SPC** - if the buffer specified is invalid. Check the length and address specified in the request.

6.10 Host Name Functions

Use the host name functions to manipulate the Host Name Table and to translate host names into internet addresses. You can issue host name functions whether or not a connection is open.

The following operations are supported:

- Add host name/internet address pair to Host Name Table.

  Both the internet address and host name must be specified in your buffer. A new entry is added if the host name is not already in the Host Name Table. Any number of host names may be assigned the same internet address (you can use alias host names).

  This operation requires that the task or terminal be privileged.

- Remove host name from Host Name Table.

  Only the host name must be specified in your buffer. The entry for that host name, if found, is removed from the Host Name Table.

  This operation requires that the task or terminal be privileged.

- Lookup internet address of host name.

  Only the host name must be specified in your buffer. The internet address for that host name is returned in the internet address field of your buffer if found in the Host Name Table.

  If the host name is an ASCII internet address string, it is converted to the binary internet address and returned in the internet address field of your buffer.

- Lookup host name of internet address.

  Only the internet address must be specified in your buffer. The alphabetically first host name for that internet address is returned in the host name field of your buffer if found in the Host Name Table.

- Read entire Host Name Table.

  The Host Name Table is returned to the buffer specified. Each entry requires 36 bytes (4 bytes for the internet address followed by 32 bytes for the host name).

  The maximum number of Host Name Table entries is selected during installation. However, this maximum value is always less than or equal to 200.

Function Code

IO.WAT

Parameters

P1 contains the address of the host name buffer. This address must be even. The host name buffer may be read or written, or both, depending on the operation requested.

The buffer usually consists of one host name entry (except for the read entire Host Name Table operation). A host name entry is 36 bytes long: 4 bytes for the internet address followed by 32 bytes for the (zero filled) host name. Host names may be specified in upper and lower case; they are always stored and returned in upper case.

P2 contains the byte length of the buffer specified in **P1**. It must be 36 bytes long (except for the read entire Host Name Table operation, in which case you should specify a buffer large enough to hold all of the entries).

P3 contains the Host Name Table operation code (1 for lookup by internet address, 2 for lookup by host name, 3 for add host name/internet address, 4 for remove host name, and 5 for read all entries).

Status Codes

The following function-specific status codes can be returned by this request:

- **IE.BNM** - if the host name is invalid (for example, if the host name is an ASCII internet address string for an add operation).
- **IE.BYT** - if the buffer address or buffer length is odd.
- **IE.DAO** - if the buffer was too small to return all of the available information (only returned for the read entire Host Name Table operation).
- **IE.FOP** - if an entry for the host name already exists in the Host Name Table (only returned for the add operation).
- **IE.IFC** - if the operation code is invalid.
- **IE.NDR** - if the Host Name Table is full (only returned for the add operation).
- **IE.OFL** - if the networking software is not active. Ask your system manager to start the networking software.
- **IE.PRI** - if your task or terminal is not privileged (only returned for the add and remove operations).
- **IE.SPC** - if the buffer specified is invalid. Check the length and address specified in the request.
- **IE.UKN** - if the host name was not found in the Host Name Table, or if there is no entry in the Host Name Table for the internet address.

A
Sample Programs

This appendix contains sample programs that use the IN Device Driver programming services.

A.1 TCP Programming Example (DSCRDD)

This program is a server for the Discard Protocol (see RFC-863, *Discard Protocol*). The TCP-based discard service specifies that a server listens for TCP connections on TCP port 9. When a connection is established, the server throws away any data it receives. It does not send a response. This continues until the calling user terminates the connection.

This sample program is provided on the distribution kit in the DSCRDD.MAC file.

The sample program does the following:

1. Listens for a connection on TCP port 9.
2. Receives data from the host.
3. Discards the data.
4. Closes the connection.
5. Displays the number of bytes received, the internet address of the remote host, the port number, and the host name (if available).

The program uses the following functions: passive open, status, receive, and close connection. It also issues a host name function to look up the host name for the internet address of the remote host.

```
        .TITLE  DSCRDD  -- SERVER-DISCARD
        .SBTTL  Main Program
;
; **    Server-DISCARD
```

```
;
;       This program will listen for a connection on the discard
;       port number (9). When that connection opens, it will
;       receive (and discard) all data until the sender closes
;       the connection.
;
;       To build:
;               MACRO DSCRDD
;               LINK DSCRDD
;
        .MCALL  ALUN$S  EXIT$S  QIOW$S

DSCRDD: ALUN$S  #1,#"IN,#0                ; Assign LUN
        CALL    CHKDSW                    ; Check directive status
;
;       Wait for a passive connection to open:
;
100$    CLR     DATCNT+0                   ; Zero received data counters
        CLR     DATCNT+2
        QIOW$S   #IO.CTL!201,#1,#1,,#IOSB,,<#SOCKET,#12.>
        CALL     CHKISB                    ; Check I/O status
        BPL      200$                      ; Branch on success
        CMPB     R0,#IE.TMO                ; Timed-out?
        BEQ      100$                      ; Branch if so (ok to try again)
        EXIT$S                             ; Otherwise, exit
;
;       Read the socket information (to find foreign IA + port)
;       and then lookup the foreign IA's host name:
;
200$:   QIOW$S  #IO.CTL!4,#1,#1,,#IOSB,,<#DATBUF,#16.>
        CALL    CHKDSW                    ; Check directive status
        MOV     #DATBUF+6,R0              ; Build $EDMSG data buffer and
        MOV     #CPEMDB+2,R1              ;  host name buffer for lookup
        MOV     #HNBUFF,R2                ;  by IA
        .REPT   4
        MOVB    (R0),(R1)+
        CLRB    (R1)+
        MOVB    (R0)+,(R2)+
        .ENDR
        MOV     (R0)+,(R1)+
        CLRB    (R2)+
        QIOW$S  #IO.WAT,#1,#1,,#IOSB,,<#HNBUFF,#36.,#1>
        CALL    CHKDSW                    ; Check directive status
;
;       Wait for data to be received:
;
210$:   QIOW$S  #IO.RVB,#1,#1,,#IOSB,,<#DATBUF,#2048.>
        CALL    CHKISB                    ; Check I/O status
        BMI     220$                      ; Branch on error
        ADD     IOSB+2,DATCNT+2           ; Update received byte count
        ADC     DATCNT
        BR      210$                      ; And receive more

220$:   CMPB    R0,#IE.EOF                ; Foreign end closed connection?
        BNE     230$                      ; Branch if not
        QIOW$S  #IO.CTL!2,#1,#1           ; Close the connection
        MOV     #DATBUF,R0
        MOV     #CPEMCS,R1
```

```
        MOV     #CPEMDB,R2
        CALL    $EDMSG                  ; Format message
        SUB     #DATBUF,R0              ;  and output it
        QIOW$S  #IO.WVB,#5,#1,,#IOSB,,<#DATBUF,R0,#40>

230$:   JMP     100$                    ; And continue

CHKISB: MOVB    IOSB,R0                 ; Load I/O status
CHKDSW: BCC     10$                     ; Branch if directive OK
        EXIT$S                          ; Exit

10$:    RETURN                          ; Return

        .PAGE
        .SBTTL  Working Storage

SOCKET: .WORD   0,0,9.                  ; Passive open socket (listen
        .WORD   0,0,0                   ;  on DISCARD port)

CPEMCS: .ASCIZ  "%T. bytes received from %M.%M.%M.%M.%M (%I)"
        .EVEN

CPEMDB: .WORD   DATCNT                  ; $EDMSG data buffer to display
        .BLKW   5                       ;  CPEMCS message
        .WORD   HNBUFF+4

DATCNT: .BLKW   2                       ; Count of bytes received

HNBUFF: .BLKB   4.+32.                  ; Host Name Buffer
        .BYTE   0                       ;  (guard byte for %I in case
        .EVEN                           ;   host name is 32 bytes long)

IOSB:   .BLKW   2                       ; I/O status block

DATBUF: .BLKB   2048.                   ; Data buffer

        .END    DSCRDD
```

A.2 UDP Programming Example (UECHOD)

This program is a server for the Echo Protocol (see RFC-862, Echo Protocol). The UDP-based echo service specifies that a server listens for a UDP datagram on port 7. When a datagram is received on this port, the server sends the datagram back to the sender.

The sample program performs the following steps:

1. Opens UDP port 7.
2. Receives a datagram from any host.
3. Sends the datagram back to that host.
4. Repeats steps 2 and 3 forever.

You can find this program on the distribution kit in the UECHOD.MAC file.

```
        .TITLE  UECHOD  -- UDP Echo Server
        .SBTTL  UDP Echo Server
;
;       This program is a UDP echo server. It:
;       1) Opens a UDP port using local port number 7 (port 7 is
;          reserved for the Echo protocol).
;       2) Waits to receive a datagram from any host.
;       3) When a datagram is received, it is sent back to the
;          sender.
;
;       To build:
;               MACRO UECHOD
;               LINK  UECHOD
;
        .MCALL  ALUN$S  EXIT$S  QIOW$S

UECHOD: ALUN$S  #1,#"IN,#0              ; Assign LUN
        CALL    CHKDSW                  ; Check directive status
;
;       Open the UDP port:
;
        QIOW$S  #IO.CTL!101,#1,#1,,#IOSB,,<#SOCKET,#12.>
        CALL    CHKISB                  ; Check I/O status
        BPL     RDLOOP                  ; Branch on success
        EXIT$S                          ; Otherwise, exit
;
;       Wait for a datagram to be received:
;
RDLOOP: QIOW$S  #IO.RVB,#1,#1,,#IOSB,,<#DATBUF,#DATSIZ,#0,#SRCBUF>
        CALL    CHKISB                  ; Check I/O status
        BMI     200$                    ; Branch on error
        MOV     IOSB+2,EREMDB+0         ; Set up to format a message
        MOVB    SRCBUF+0,EREMDB+2
        MOVB    SRCBUF+1,EREMDB+4
        MOVB    SRCBUF+2,EREMDB+6
        MOVB    SRCBUF+3,EREMDB+10
        MOV     SRCBUF+4,EREMDB+12
        MOV     #MSGBUF,R0
        MOV     #EREMCS,R1
        MOV     #EREMDB,R2
```

```
        CALL    $EDMSG                  ; Format the message
        SUB     #MSGBUF,R0              ;  and output it
        QIOW$S  #IO.WVB,#5,#1,,,,<#MSGBUF,R0,#40>
;
;       Send the datagram back to the sender ...
;
        QIOW$S  #IO.WVB,#1,#1,,#IOSB,,<#DATBUF,IOSB+2,#0,#DSTBUF>
        CALL    CHKISB                  ; Check I/O status
        BPL     RDLOOP                  ; Branch on success
        MOV     #MSGBUF,R0              ; Set up to format a message
        MOV     #EWEMCS,R1
        MOV     #IOSB,R2
        CALL    $EDMSG                  ; Format the message
        SUB     #MSGBUF,R0              ;  and output it
        QIOW$S  #IO.WVB,#5,#1,,,,<#MSGBUF,R0,#40>
        JMP     RDLOOP

200$:   EXIT$S                          ; Exit

CHKISB: MOVB    IOSB,R0                 ; Load I/O status
CHKDSW: BCC     10$                     ; Branch if directive OK
        EXIT$S                          ; Exit

10$:    RETURN                          ; Return

        .PAGE
        .SBTTL  Working Storage

SOCKET: .WORD   0,0,7.                  ; Open ECHO port
        .WORD   0,0,0

IOSB:   .BLKW   2                       ; I/O status block

DSTBUF:
SRCBUF: .BLKW   2                       ; Foreign internet address
        .BLKW   1                       ; Foreign port number

DATBUF: .BLKB   2048.                   ; Data buffer
DATSIZ  =       .-DATBUF                ; Size of data buffer

EWEMCS: .ASCIZ  "UECHOD -- Failed to echo datagram - IOSB is %M %M."
EREMCS: .ASCIZ  "UECHOD -- Received %M. byte datagram from %M.%M.%M.%M,%M"
        .EVEN

EREMDB: .BLKW   1                       ; $EDMSG data buffer to display
        .BLKW   4                       ;  CPEMCS message
        .BLKW   1

MSGBUF: .BLKB   132.                    ; Message buffer

        .END    UECHOD
```

B
Implementation Notes

This appendix describes the implementation of the Transmission Control Protocol (TCP), Internet Protocol (IP), and Address Resolution Protocol (ARP).

Most of the material in this appendix requires a thorough understanding of the protocols used and is not intended for most readers.

This appendix does not describe the protocols themselves. Please refer to the documents listed in the *References* section of the *General Information* chapter for complete protocol descriptions.

B.1 Transmission Control Protocol (TCP)

The Transmission Control Protocol (TCP) implemented within this product does not support the urgent pointer and urgent option. These fields are ignored.

B.2 User Datagram Protocol (UDP)

The following restrictions apply:

- The largest amount of user data that you can transmit or receive in a single datagram is 1116 bytes.
- For unsolicited receives, UDP only saves one datagram per open port.

B.3 Internet Protocol (IP)

Version 4 of the Internet Protocol (IP) is implemented within this product, and the following restrictions apply:

- Datagram fragmentation and reassembly are not supported.
- The type of service field is ignored.
- All options are ignored (options may be present in incoming packets).

- All received Internet Control Message Protocol (ICMP) packets are ignored. No ICMP packets are transmitted.
- A single default gateway address is supported for access to non-local networks.
- Received packets may be a maximum of 1144 bytes long (IP + TCP/UDP + data). Transmitted packets are a maximum of 1144 bytes long (IP + TCP + data).

B.4 Address Resolution Protocol (ARP)

The Address Resolution Protocol (ARP) is used to dynamically map between internet and Ethernet addresses. This implementation of the ARP protocol does not watch for or probe for impostors.

C
Release Notes

This appendix summarizes the enhancements and changes for each version of TCP-RSX. If you have been using an earlier version, read this appendix to determine whether any changes affect you.

C.1 Version 2.2-5

This section describes enhancements and changes for TCP-RSX V2.2-5.

- UDP support was added. See the *IN Device Driver Services* chapter and the *Implementation Notes* appendix for more information.

C.2 Version 2.2-4

This section describes enhancements and changes for TCP-RSX V2.2-4.

- The Ethernet XU device driver (XUDRV) now handles UNIBUS Mapping Registers (UMRs) correctly. This resolves a problem previously noticeable on systems with heavy UMR usage.
- The XUDRV reset timeout time was increased to allow sufficient time for a DELUA controller to reset if the controller failed the self-test during a power-up.

C.3 Version 2.2

This section describes enhancements and changes for TCP-RSX V2.2.

- The TCP/IP-RSX product has a new name: TCP-RSX. TCP-RSX is part of the TCPware for RSX family of software products.
- INPOOL memory is allocated dynamically, instead of statically as in previous versions. This enhancement allows TCPware to support a dynamic number of connections. The maximum number of connections depends on the resources available when a connection is opened. The *Installation* chapter of this manual explains how you can determine how large to make INPOOL for your environment.

- You can specify the size of the receive window for TCP connections when you open a connection. Refer to the *TCP Services* chapter, *Open Function* section for more information.
- TCPware processes ARP packets that are longer than 64 bytes. In the past, these packets were discarded.
- The NETCU SHOW NETWORK command displays the amount of space available in INPOOL, in octal bytes.
- NETCU supports host names containing dots (.), hyphens (-), underscores (_), and any printable non-control characters, except for spaces and tabs.
- INSACP no longer uses the context switching disable flag ($CXDBL). This change was implemented because some versions of the RSX operating system contain a bug that causes this flag to be ignored.
- TCP-RSX V2.2 contains additional minor improvements and corrections.

C.4 Version 2.1

TCP/IP RSX Version 2.1 includes the following enhancements and changes:

- Several minor problems in the CNFNET indirect command file have been corrected. In particular, CNFNET now works properly if the system files are in a directory other than LB:[1,54].
- A problem in INDRV that could sometimes hang the system has been corrected.
- A problem in XUDRV that could sometimes hang the system has been corrected.
- Other minor internal changes have been made.

Prior to this release, most of the material in this document was published in the *FTP-RSX Software User's Guide*.

Index

U

V

W