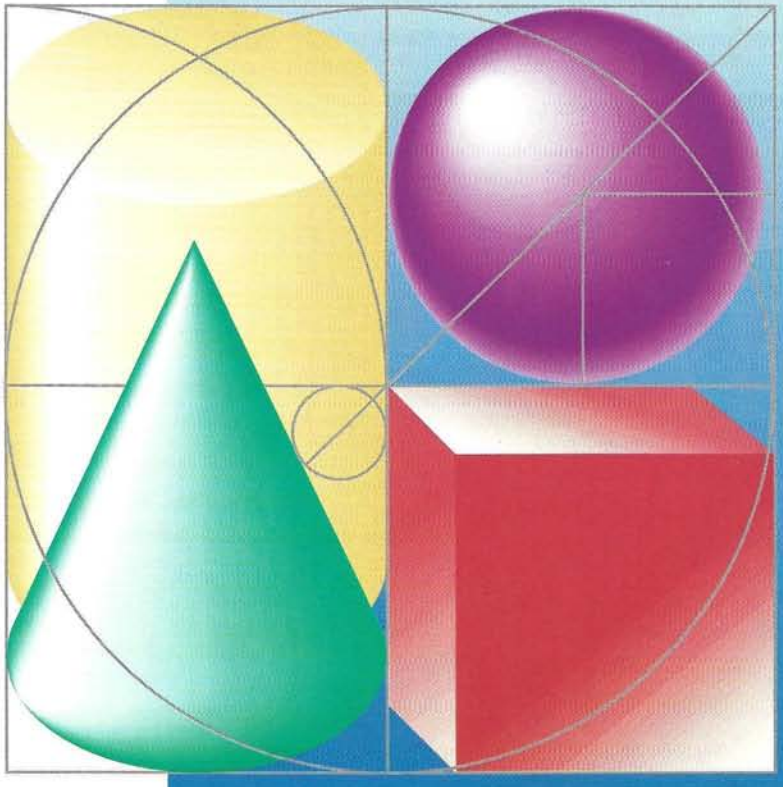
MacTCP™ v.1.0, Evaluation Kit

with Interim Manual

APDA # M0230LL/A



MacTCP™ v.1.0, Evaluation Kit

with Interim Manual

APDA# M0230LL/A

Apple Programmers and
Developers Association

Dear MacTCP™ Customer:

Thank you for your interest in MacTCP. Enclosed is your purchase.

Please note that your product may have a different part number printed on it than the number you saw on a data sheet or read in the APDAlog. We needed to change the numbering system late in the production cycle for inventory purposes, but chose to ship the product anyway rather than cause delays in delivery. Please be assured that you are in fact receiving the correct product you ordered.

Again, thank you for your support and for your patience. I hope this did not cause you any undue confusion. If you have any questions, please call us toll-free at 1-800-282-2732 or write us at:

APDA Customer Service
20525 Mariani Avenue, M/S 33G
Cupertino, California 95014.

Sincerely,

Jennifer Gugerty

Jennifer Gugerty
APDA Customer Service Supervisor

MacTCP™ v.1.0, Evaluation Kit

with Interim Manual

This package contains:

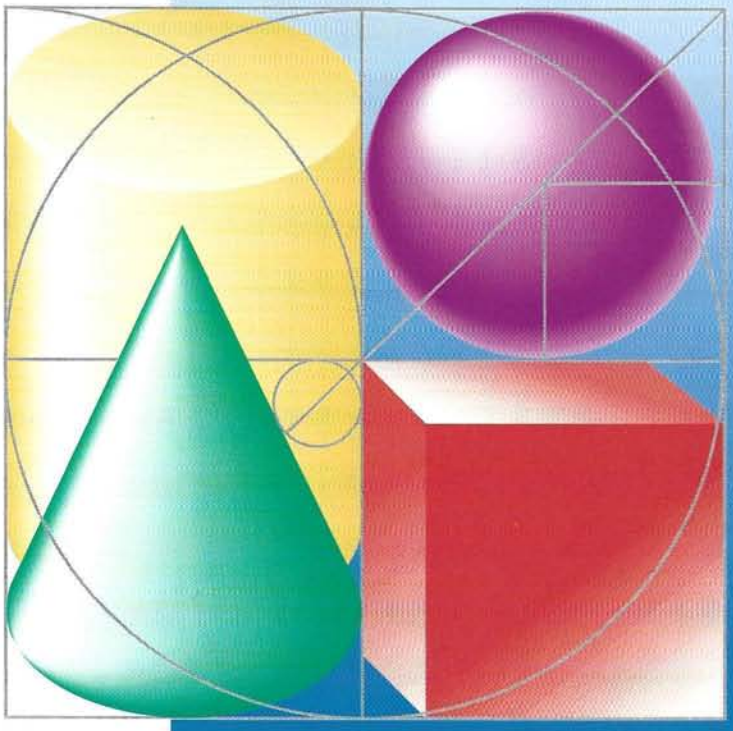
2	Manuals:	*MacTCP Administrator's Guide* *MacTCP Programmer's Guide, Interim Release*	*M0260LL/A*
	Set of Release Notes	*None*	
2	Disks	*MacTCP Developer Disk, Disk 1* *MacTCP Sample Distribution Disk, Disk 2*	
2	1-inch Binders		
2	1-inch Binder Covers		
2	1-inch Spine Identifications		

If you have any questions, please call:

1-800-282-2732 (U.S.)
1-408-562-3910 (International)
1-800-637-0029 (Canada)

MacTCP™ v.1.0, Administrator's Guide

APDA # M0260LL/A



Limited Warranty on Media and Replacement

If you discover physical defects in the manuals distributed with an Apple product or in the media on which a software product is distributed, Apple will replace the media or manuals at no charge to you, provided you return the item to be replaced with proof of purchase to Apple or an authorized Apple dealer during the 90-day period after you purchased the software. In addition, Apple will replace damaged software media and manuals for as long as the software product is included in Apple's Media Exchange Program. While not an upgrade or update method, this program offers additional protection for up to two years or more from the date of your original purchase. See your authorized Apple dealer for program coverage and details. In some countries the replacement period may be different; check with your authorized Apple dealer.

ALL IMPLIED WARRANTIES ON THE MEDIA AND MANUALS, INCLUDING IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, ARE LIMITED IN DURATION TO NINETY (90) DAYS FROM THE DATE OF THE ORIGINAL RETAIL PURCHASE OF THIS PRODUCT.

Even though Apple has tested the software and reviewed the documentation, **APPLE MAKES NO WARRANTY OR REPRESENTATION, EITHER EXPRESS OR IMPLIED, WITH RESPECT TO SOFTWARE, ITS QUALITY, PERFORMANCE, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. AS A RESULT, THIS SOFTWARE IS SOLD "AS IS," AND YOU, THE PURCHASER, ARE ASSUMING THE ENTIRE RISK AS TO ITS QUALITY AND PERFORMANCE.**

IN NO EVENT WILL APPLE BE LIABLE FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES RESULTING FROM ANY DEFECT IN THE SOFTWARE OR ITS DOCUMENTATION, even if advised of the possibility of such damages. In particular, Apple shall have no liability for any programs or data stored in or used with Apple products, including the costs of recovering such programs or data.

THE WARRANTY AND REMEDIES SET FORTH ABOVE ARE EXCLUSIVE AND IN LIEU OF ALL OTHERS, ORAL OR WRITTEN, EXPRESS OR IMPLIED. No Apple dealer, agent, or employee is authorized to make any modification, extension, or addition to this warranty.

Some states do not allow the exclusion or limitation of implied warranties or liability for incidental or consequential damages, so the above limitation or exclusion may not apply to you. This warranty gives you specific legal rights, and you may also have other rights which vary from state to state.

Apple Computer, Inc.
20525 Mariani Avenue
Cupertino, California 95014
(408) 996-1010
TLX 171-576

To reorder products, please call:
Apple Programmers and Developers Association
1-800-282-APDA



Apple® MacTCP™ Administrator's Guide

APPLE COMPUTER, INC.

This manual and the software described in it are copyrighted, with all rights reserved. Under the copyright laws, this manual or the software may not be copied, in whole or part, without written consent of Apple, except in the normal use of the software or to make a backup copy of the software. The same proprietary and copyright notices must be affixed to any permitted copies as were affixed to the original. This exception does not allow copies to be made for others, whether or not sold, but all of the material purchased (with all backup copies) may be sold, given, or loaned to another person. Under the law, copying includes translating into another language or format.

You may use the software on any computer owned by you, but extra copies cannot be made for this purpose.

The Apple logo is a registered trademark of Apple Computer, Inc. Use of the "keyboard" Apple logo (Option-Shift-K) for commercial purposes without the prior written consent of Apple may constitute trademark infringement and unfair competition in violation of federal and state laws.

© Apple Computer, Inc., 1989
20525 Mariani Avenue
Cupertino, CA 95014
(408) 996-1010

Apple, the Apple logo, AppleShare, AppleTalk, LaserWriter, and Macintosh are registered trademarks of Apple Computer, Inc.

EtherTalk, LocalTalk, and MacTCP are trademarks of Apple Computer, Inc.

Apollo is a registered trademark of Apollo Computer, Inc.

DEC is a trademark of Digital Equipment Corporation.

IBM is a registered trademark of International Business Machine Corporation.

ITC Zapf Dingbats is a registered trademark of International Typeface Corporation.

Microsoft and MS-DOS are registered trademarks of Microsoft Corporation.

POSTSCRIPT is a registered trademark, and Illustrator is a trademark, of Adobe Systems Incorporated.

Sun is a trademark of Sun Microsystems, Inc.

UNIX is a registered trademark of AT & T Information Systems.

Varityper is a registered trademark, and VT600 is a trademark, of AM International, Inc.

Simultaneously published in the United States and Canada.

Contents

About This Guide xi
Who should read this guide xii
How to use this guide xii
Conventions xiii
What you need xiii
Associated documents xiv

Part One Concepts

1 About MacTCP 3
Overview 4
Network environment 5
DDP-IP gateway 6

2 About TCP/IP 7
The family of protocols 9
Internet Protocol (IP) 10
Transmission Control Protocol (TCP) 11
User Datagram Protocol (UDP) 11
Telnet 12

File Transfer Protocol (FTP) 12
Simple Mail Transfer Protocol (SMTP) 12

3 TCP/IP Addressing Conventions 13

Internet addresses 14
Subnetwork addressing 15
Subnetwork masks 17
The domain name system 18
Address Resolution Protocol (ARP) 19
Reverse Address Resolution Protocol (RARP) 20
Bootstrap Protocol (BootP) 21

4 Routing 23

Gateways 24
Routing tables 25
Routing protocols 25
Routing Information Protocol (RIP) 25
Internet Control Message Protocol (ICMP) 25

Part Two Procedures

5 Installation 29

MacTCP software 30
Creating a master disk 30
User installation instructions 33
Installing MacTCP on a floppy disk-based system 34
Installing MacTCP on a hard disk 34

6 MacTCP Configuration 37

Configuration tools 38
The MacTCP Control Panel 38

The Administrator dialog box 40
Obtain Address box 41
IP Address box 41
Routing Information box 41
Domain Name Server Information box 41
Protected button 42
Configuration scenarios 42
Bringing up the MacTCP Control Panel 44
Entering the IP address in decimal notation 45
Bringing up the Administrator dialog box 46
Setting the IP address 47
Setting the address manually 48
Setting the address class 48
Setting the subnet mask 50
Setting the IP address (integer format) 50
Having the address set automatically 52
Setting the node number dynamically 53
Setting the gateway address 54
Setting domain name server information 55
Protecting the configuration 56
Giving the user full configuration power 56
Alert message 57
MacTCP is configured 57

7 User Configuration 63

Bringing up the MacTCP Control Panel 60
Selecting the MacTCP icon 60
Setting link level information 61
Network configuration 1 61
Network configuration 2 63
Network configuration 3 64
Network configuration 4 65
Setting the IP address 67
Entering the node number 68
Entering the IP address 68
Viewing the Administrator dialog box 69
Closing the Control Panel 70

Part Three Appendixes

Appendix A
Name-to-Address Mapping 73
Hosts file syntax 74
Editing the hosts file 75

Appendix B
MacTCP on the Macintosh 512K Enhanced Computer 77
Creating a master disk 78
The Macintosh 512K enhanced Control Panel 80

Appendix C
Getting More Information 83
Books 84
Request for Comments (RFC) 84
Military Standards 85

Glossary 87
Index 91

Figures and tables

CHAPTER 1 **About MacTCP 3**

Figure 1-1 MacTCP network configuration 5

CHAPTER 2 **About TCP/IP 7**

Figure 2-1 An internet 8
Figure 2-2 Comparison of OSI and TCP/IP 10

CHAPTER 3 **TCP/IP Addressing Conventions 13**

Figure 3-1 IP address structure 15
Figure 3-2 Subnetting at Pundit University 16
Figure 3-3 ARP request and reply 20
Figure 3-4 RARP broadcast and server reply 21

Table 3-1 Top-level domains 18

CHAPTER 4 **Routing 23**

Figure 4-1 Gateway routing table 24

CHAPTER 5 **Installation 29**

Figure 5-1 Disk folders 31
Figure 5-2 Icons in the MacTCP Folder 32

Figure 5-3 Disk folders 35
Figure 5-4 Icons in the System Folder 35

CHAPTER 6 **MacTCP Configuration 37**

Figure 6-1 View of the MacTCP Control Panel 39
Figure 6-2 View of the Administrator dialog box 40
Figure 6-3 Roadmap to configuration scenarios 43
Figure 6-4 The MacTCP Control Panel 44
Figure 6-5 IP Address box in the MacTCP Control Panel 45
Figure 6-6 IP Address box in the Administrator dialog box 45
Figure 6-7 The Administrator dialog box 46
Figure 6-8 Obtain Address box 47
Figure 6-9 Menu of address classes 48
Figure 6-10 Settings for classes A, B, and C 49
Figure 6-11 Values set in Net, Subnet, and Node boxes 51
Figure 6-12 Node Range boxes 53
Figure 6-13 Routing Information box 54
Figure 6-14 Domain name server information 55
Figure 6-15 The Protected button 56
Figure 6-16 The alert box 57

CHAPTER 7 **User Configuration 59**

Figure 7-1 The MacTCP Control Panel 60
Figure 7-2 Network configuration 1 61
Figure 7-3 Icons displayed for network configuration 1 62
Figure 7-4 Zone pop-up menu 62
Figure 7-5 Network configuration 2 63
Figure 7-6 Icons displayed for network configuration 2 63
Figure 7-7 Network configuration 3 64
Figure 7-8 Icons displayed for network configuration 3 65
Figure 7-9 Network configuration 4 66
Figure 7-10 Icons displayed for network configuration 4 67
Figure 7-11 Entering the node number 68
Figure 7-12 Entering the IP address 68
Figure 7-13 The Administrator dialog box 69
Figure 7-14 The alert box 70

APPENDIX B **MacTCP on the Macintosh 512K Enhanced Computer** 77

Figure B-1 Icon in the Installer Script Folder 79
Figure B-2 Installer window 79
Figure B-3 Control Panel on the Macintosh 512K enhanced computer 81

P R E F A C E

About This Guide

THIS GUIDE CONTAINS INSTRUCTIONS FOR INSTALLING AND CONFIGURING MacTCP™, Apple Computer's implementation of the protocol suite known as Transmission Control Protocol/Internet Protocol (TCP/IP).

Who should read this guide

This guide is intended for the network administrator who is responsible for network maintenance. The major part of administering a MacTCP network involves using the Control Panel to configure MacTCP for network users. Before you start, you must determine how much you want network users involved in the configuration process; for instance, you can set up the Control Panel so the user must enter all the configuration information or none at all. See the section "Configuration Scenarios" in Chapter 6 for more information.

You should be familiar with Macintosh® computer operation and general networking concepts. However, it is not necessary to be a TCP/IP expert; Part I of this guide provides conceptual information if you need to learn more about it. Appendix C points you to other documentation that might be useful.

How to use this guide

This guide begins with basic concepts in Part I and proceeds to explicit instructions in Part II. If you are an experienced network administrator, you may want to spend less time on introductory chapters and move quickly to operating instructions. However, it is important to understand TCP/IP concepts, such as IP addressing, in order to configure the system correctly.

After you install and configure MacTCP, you must distribute the software to network users who in turn install and configure MacTCP for their individual machines. Instructions have been provided that you can copy and distribute to users.

To get MacTCP up and running you need to follow the instructions in Part II of this guide, as described here:

- Create a master disk as described in Chapter 5.
- Configure MacTCP on the master disk using the instructions in Chapter 6.

- Give each network user a copy of the master disk along with a copy of the user installation instructions in Chapter 5 and the configuration instructions in Chapter 7.
- If you give users full configuration power you must provide them with a copy of Chapter 6.

Conventions

Terms that appear in **boldface** type also appear in the glossary. The glossary also defines other terms that are related to your use of MacTCP.

❖ *Note:* Text set off in this manner presents sidelights or interesting points of information.

△ **Important** Text set off in this manner—with the word *Important*—presents important information or instructions. △

What you need

To use MacTCP for network communication, you need the following hardware and software:

- An Ethernet interface card in a Macintosh SE, Macintosh SE/30, Macintosh II, or Macintosh IIx computer connected to an Ethernet network, or a Macintosh 512K enhanced, Macintosh Plus, Macintosh SE, Macintosh SE/30, Macintosh II, or Macintosh IIx computer connected to a LocalTalk™ network.

  ❖ *Important:* If your Macintosh computer is using an Ethernet interface card, make sure the card's software is installed. If you are using an Apple EtherTalk™ Interface Card, make sure the software is installed according to the instructions in the *EtherTalk User's Guide*. After the software is installed, refer back to this manual to configure MacTCP.

- On a LocalTalk network, a Datagram Delivery Protocol-Internet Protocol (DDP-IP) gateway.

- Macintosh System file version 6.0.3 or later for the Macintosh Plus, Macintosh SE, Macintosh SE/30, Macintosh II, or Macintosh IIx computer. Macintosh System file version 3.4 for the Macintosh 512K enhanced computer.
- MacTCP software.

Associated documents

For additional information about MacTCP networks, see the *Apple MacTCP Programmer's Guide*. This guide is for third-party developers creating application programs for the MacTCP driver.

You might also find it useful to refer to *Inside AppleTalk* for a complete description of the AppleTalk® network system.

PART ONE

Concepts

CHAPTER 1

About MacTCP

MacTCP™, Apple Computer's TCP/IP product, increases the Macintosh® computer's ability to operate in a heterogenous computer environment. MacTCP allows the Macintosh computer to talk to such diverse systems as IBM, DEC™, Sun, and Apollo computers. This chapter describes the features and operation of MacTCP.

Overview

MacTCP is a software driver for the Macintosh Operating System that implements the following TCP/IP protocols:

- Internet Protocol (IP)
- Internet Control Message Protocol (ICMP)
- User Datagram Protocol (UDP)
- Address Resolution Protocol (ARP)
- Reverse Address Resolution Protocol (RARP)
- Routing Information Protocol (RIP)
- Bootstrap Protocol (BootP)
- Transmission Control Protocol (TCP)

These protocols provide core transmission services that are used by third-party products such as electronic mail, remote login, file transfer, remote printing, and database access. Chapters 2, 3, and 4 describe these protocols and the services they provide.

MacTCP conforms to Internet Request for Comments (RFC) and Military Standards (MIL-STD), ensuring interoperability with systems on the Defense Advanced Research Projects Agency (DARPA) Internet.

The Control Panel is used to configure MacTCP, simplifying installation and setup procedures. Chapters 6 and 7 of this guide describe how to use the Control Panel to configure MacTCP.

The MacTCP driver includes these features:

- C and assembly language interfaces that provide programmers with a familiar development environment.
- A domain name resolver that maps domain names to internet addresses. The domain name resolver is compatible with domain name server implementations that comply with RFC 1034 and 1035.

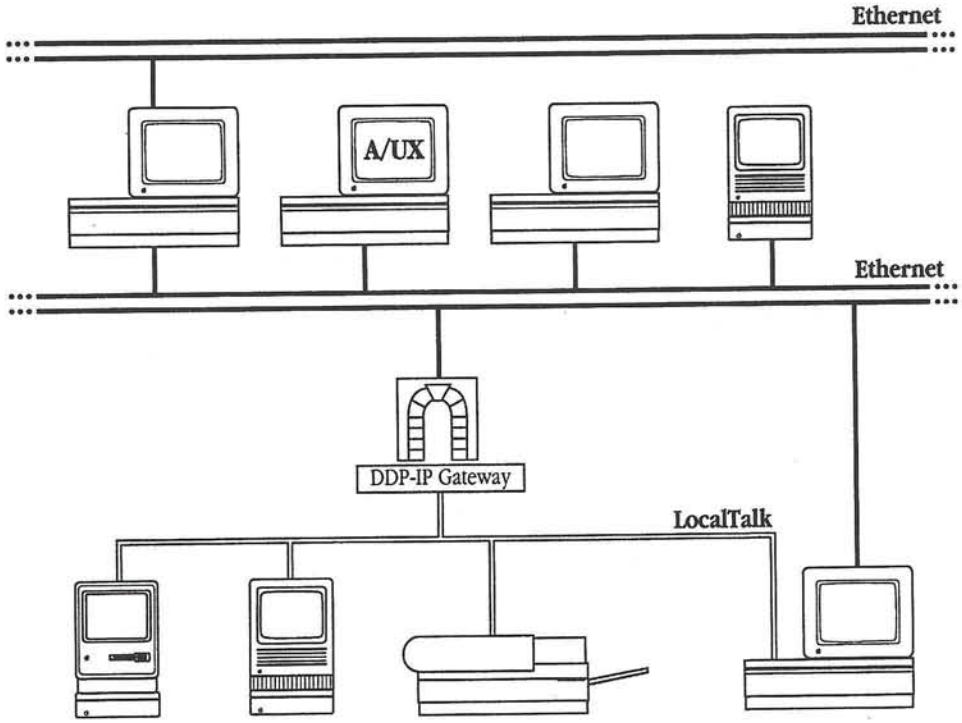
Network environment

MacTCP runs over both Ethernet and LocalTalk™-compatible cabling systems. A Macintosh 512K enhanced, Macintosh Plus, Macintosh SE, Macintosh SE/30, Macintosh II, or Macintosh IIx computer can run MacTCP on LocalTalk-compatible cable. A Macintosh SE, Macintosh SE/30, Macintosh II, or Macintosh IIx computer with an Ethernet interface card can run MacTCP on Ethernet.

MacTCP is coresident with AppleTalk® protocols so there can be concurrent TCP/IP and AppleTalk operation. For example, MacTCP can be run while a print job goes out to an Apple® LaserWriter® printer over LocalTalk cabling. AppleTalk and MacTCP can run over the same media, or one protocol can run over one media while the other protocol runs over a different media.

Figure 1-1 shows a typical MacTCP network configuration.

■ **Figure 1-1** MacTCP network configuration



DDP-IP gateway

To run MacTCP on LocalTalk cable (or any other AppleTalk-compatible media), you need a Datagram Delivery Protocol-Internet Protocol (DDP-IP) gateway. This device takes a TCP/IP packet that is encapsulated in DDP, an AppleTalk protocol, and converts it to Ethernet format. DDP-IP gateways can also assign addresses to MacTCP nodes and handle routing to other networks.

If MacTCP is running on LocalTalk, a DDP-IP gateway must be provided on the network as shown in Figure 1-1. By default, the DDP-IP gateway should be located in the same AppleTalk zone as the Macintosh computer running MacTCP. Alternatively, a single DDP-IP gateway can support Macintosh computers in multiple AppleTalk zones; in this case, the zone where the DDP-IP gateway resides must be selected by the user using the MacTCP Control Panel.

CHAPTER 2

About TCP/IP

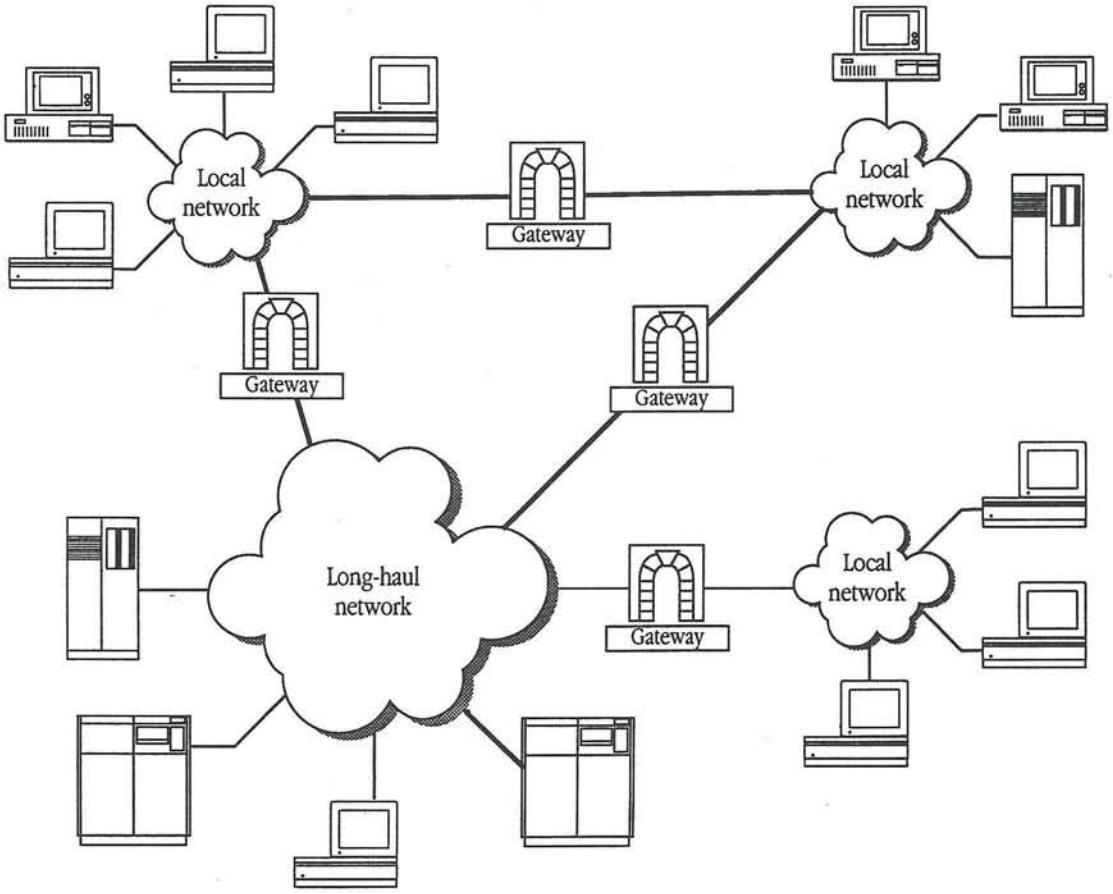
THE TRANSMISSION CONTROL PROTOCOL/INTERNET PROTOCOL (TCP/IP) IS A widely used industry standard for connecting multivendor computers. The TCP/IP protocol layers are fully compatible across all implementations on different hardware platforms, allowing different vendors' computers that run TCP/IP to interoperate and share data and services.

TCP/IP development began when the Defense Advanced Research Projects Agency (DARPA) wanted more reliable communication protocols on the ARPANET, its packet-switched wide area network. DARPA initiated a research project to define and implement a suite of protocols, and the researchers developed TCP/IP. Eventually TCP/IP became the standard protocol suite used on the DARPA Internet, a collection of networks that includes the ARPANET, Military Network (MILNET), National Science Foundation Network (NSFnet), and networks at universities, research insitutions, and military installations. Since then, hundreds of vendors have developed products that support TCP/IP and all kinds of networks use it.

TCP/IP supports an architecture of multiple networks interconnected by gateways. This interconnected set of networks is called an internetwork or **internet** (the term *internet* refers to the DARPA Internet). Figure 2-1 shows an example of an internet.

For example, TCP/IP protocols can be used to connect networks on a college campus as well as geographically distant sites on a wide area network.

■ **Figure 2-1** An internet



The family of protocols

TCP/IP is a family of protocols named after the best known protocols in the suite, the Transmission Control Protocol (TCP) and the Internet Protocol (IP). The protocols described in this guide are the ones most commonly supported by computers attached to TCP/IP networks.

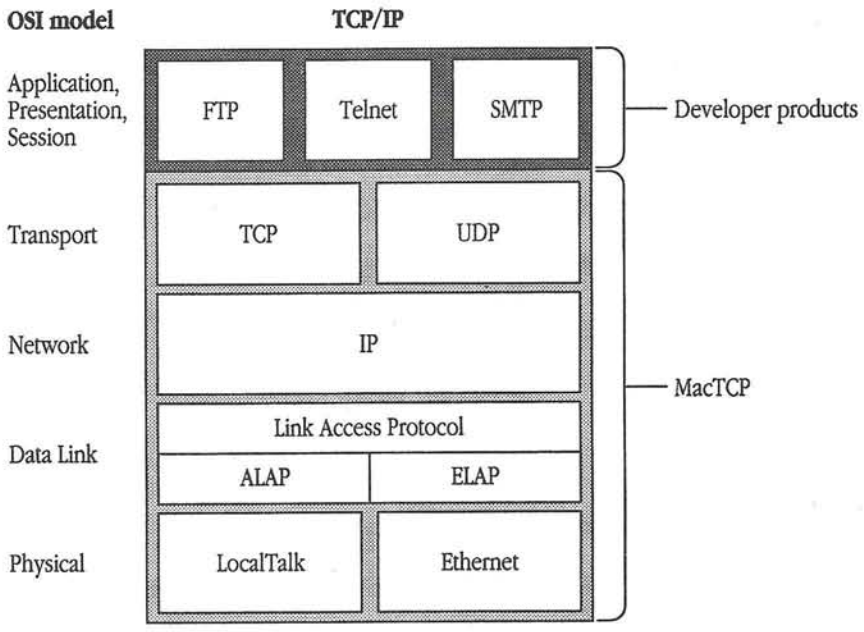
TCP, IP, and the User Datagram Protocol (UDP) provide basic transmission facilities that are augmented by application services in higher-level protocols such as Telnet, the File Transfer Protocol (FTP), and the Simple Mail Transfer Protocol (SMTP).

Other protocols in the TCP/IP family are described in later chapters of this manual. The Address Resolution Protocol (ARP), the Reverse Address Resolution Protocol (RARP), and the Bootstrap Protocol (BootP) are described in Chapter 3. The Routing Information Protocol (RIP) and the Internet Control Message Protocol (ICMP) are described in Chapter 4.

Although TCP/IP was designed and tested before the International Standards Organization (ISO) defined the Open Systems Interconnection (OSI) Reference Model, it partially conforms to the OSI layers of networking functionality. Figure 2-2 shows a comparison of the OSI and the TCP/IP communications architecture.

The TCP/IP protocols shown in Figure 2-2 are described in the following sections.

■ **Figure 2-2** Comparison of OSI and TCP/IP



Internet Protocol (IP)

The **Internet Protocol** (IP) is responsible for sending data across multiple networks. IP accepts segments of data from TCP or UDP, places the data in packets called datagrams, and determines the correct path for the datagram to take. The datagrams are sent across the internet, through as many gateways as needed, until they reach the destination host.

IP provides an addressing mechanism that allows routing between networks. The header of an IP datagram contains source and destination internet addresses so that any host in a network can route a packet to a destination, either directly or through a gateway.

IP has the ability to fragment a datagram as it is transmitted across a network. Since IP can be used with many different physical network implementations that specify different sizes for physical data frames, datagrams can be fragmented to fit into a small data frame. Fragments are reassembled as they arrive at the destination.

IP is often referred to as an unreliable delivery system because it makes a best-effort attempt to deliver all datagrams, but delivery is not guaranteed (TCP guarantees delivery). It is also called a connectionless delivery system because it routes each datagram separately. When IP receives a sequence of datagrams from TCP or UDP, it routes each datagram in the sequence individually and each datagram may travel over a different path to the destination.

Transmission Control Protocol (TCP)

The **Transmission Control Protocol** (TCP) provides reliable transmission of data between processes. (Processes are application programs that communicate; for instance, a file transfer process on one host talks to a file transfer process on another host.) It ensures that data is delivered error-free, without loss or duplication, and in sequence.

Upper-layer protocols such as Telnet pass data to TCP for delivery to peer processes. TCP encapsulates the data into segments and passes the segments to IP, which puts the segments into datagrams and passes them across the internet. TCP at the receiving end checks for errors, acknowledges error-free segments, and reassembles the segments for delivery to upper-layer protocols. If a segment is lost or damaged, it will not be acknowledged, and the sending process will retransmit.

TCP has a flow control mechanism so that computers of different speeds and sizes can communicate. When TCP at the receiving end sends an acknowledgment, it also advertises how much data it is prepared to accept on the next transmission.

User Datagram Protocol (UDP)

The **User Datagram Protocol** (UDP) provides unreliable transmission of data between processes. UDP's transport of data is unreliable because, unlike TCP, it does not provide error checking, it does not acknowledge that data has been successfully received, and it does not order incoming messages. UDP messages can be lost, duplicated, or arrive out of order. Like TCP, UDP messages are encapsulated in IP datagrams for delivery.

The advantage of UDP is that the overhead associated with establishing and maintaining an error-free TCP session is avoided. Upper-layer protocols that don't require reliability use UDP to transmit data. For instance, the domain name system uses UDP because reliability is not critical; if there is no response to a domain name query, the resolver simply retransmits. (The domain name system is described in Chapter 3.)

Telnet

Telnet is a remote access protocol that allows a terminal on one host to appear as if it were directly connected to a remote host on an internet. Telnet also makes a personal computer act like a terminal. It is usually implemented as user software that initiates sessions to a remote location and server software that listens for connections from remote users.

File Transfer Protocol (FTP)

The **File Transfer Protocol** (FTP) is used to transfer files across an internet. A host can connect to a remote host on an internet and send or receive files, list directories, and execute simple commands. Like Telnet, FTP is usually implemented as user and server software. The user software interacts with the user at a terminal and the server software receives requests from remote users to store or retrieve files.

Simple Mail Transfer Protocol (SMTP)

The **Simple Mail Transfer Protocol** (SMTP) transfers electronic mail messages from one host to another, across an internet. SMTP specifies the commands necessary to send mail and is used with a standard that specifies the general structure of a mail message.

CHAPTER 3

TCP/IP Addressing Conventions

IN ORDER FOR COMPUTERS ON THE INTERNET TO BE ABLE TO REFER TO EACH OTHER, each machine must be assigned a universal address. This chapter describes the addressing scheme that was developed to allow each TCP/IP host on the internet to be uniquely identified.

Internet addresses

Each host on a TCP/IP internet is assigned a unique 32-bit **internet address**. The address is divided into two fields, called the network field and the host field. The network field identifies a network on the internet to which the host attaches (so that all hosts on the same network share the same network field) and the host field identifies a particular host attached to that network.

The network field is assigned by a central authority, the Network Information Center (NIC) located at SRI International. Local administrators assign the host field of the address. (Note that only networks that might attach to the DARPA Internet need to obtain the network address from the NIC.)

Three IP address classes—A, B, and C—provide for the following network configurations:

- Class A addresses are used for a few networks with many hosts; for instance, the ARPANET.
- Class B addresses are used for medium-sized networks; for instance, a university network.
- Class C addresses are used for a large number of small networks; for instance, an Ethernet local area network (LAN).

The first bits of the address identify the address class, and the number of bits assigned to the network and host field of the address differs for each class. For instance, Class A addresses have 1 class identifier bit, 7 network identifier bits, and 24 host identifier bits. This allows for 128 Class A networks, where each network can support up to 16 million hosts. Figure 3-1 shows the class identifier bits and the number of bits allocated to the network and host fields for each address class.

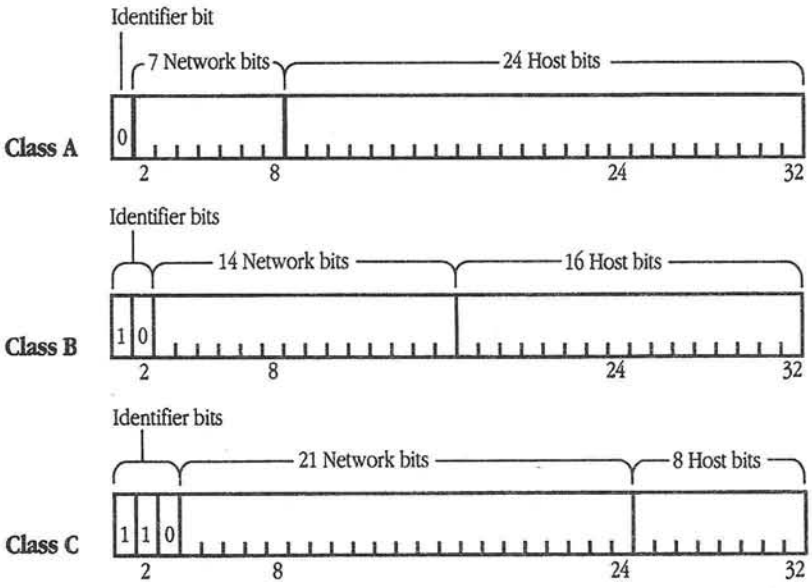
To make the addresses easier to work with, the addresses are written in dotted decimal notation. Each octet of the 32-bit address is assigned its decimal equivalent, and decimal points separate the integers. For example, the 32-bit address

10000100 00001101 00000010 00011110

is written in dotted decimal notation as

132.13.2.30

■ **Figure 3-1** IP address structure



Subnetwork addressing

The internet addressing scheme, designed for a few hundred networks, did not anticipate the proliferation of LAN technology. Organizations created large networks consisting of many LANs connected by gateways and there was an explosive growth in the number of networks on the internet. Assigning every LAN on the internet its own address created two problems:

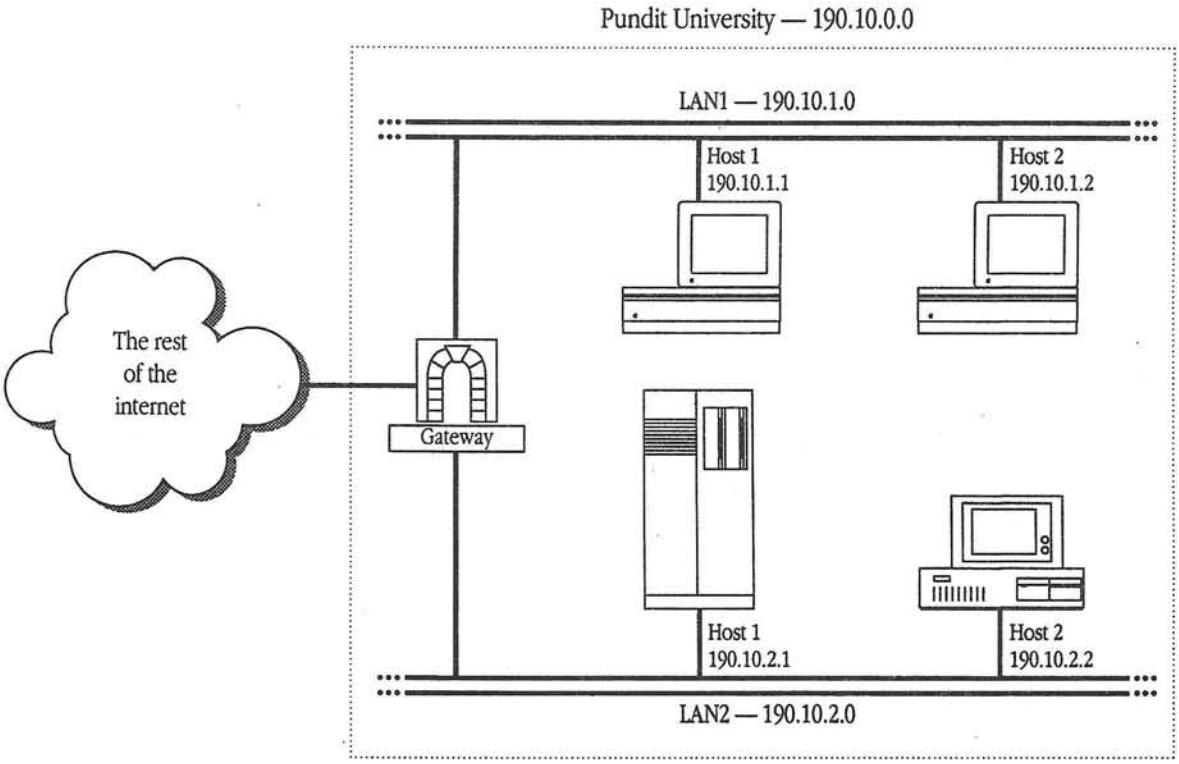
- Immense administrative overhead is required to manage the addresses.
- Internet routing tables cannot accommodate many network addresses.

A technique called **subnetwork addressing** (or subnetting) was devised to deal with this problem. Subnetting allows multiple physical networks (called subnetworks) to share the same internet network number. For example, a university with two LANs can use subnet addressing so that both campus subnetworks share a single network number. The subnet structure is not visible to the rest of the internet; the route to the network is the same no matter what subnet the host is on.

The last section stated that the internet address is divided into network and host fields. With subnetting, the address is conceptually divided into a network and local field, and the local field is divided into subnet and host fields. Various schemes (defined by subnet addressing standards) are used to divide the local field of the address into subnet numbers; the internet site determines which scheme is implemented.

Consider how Pundit University has subdivided its assigned Class B internet address, 190.10.0.0, for the two LANs on campus. They divided the local field of the address into an 8-bit subnet identifier and 8-bit host identifier. LAN1 has the address 190.10.1.0 and LAN2 has the address 190.10.2.0. Figure 3-2 illustrates the addressing scheme at Pundit University.

■ **Figure 3-2** Subnetting at Pundit University



In Figure 3-2, the rest of the internet perceives that it is sending data to one network. The gateway routes packets to the appropriate LAN by examining the third field in the address.

Subnetwork masks

An internet site that has implemented subnet addressing must choose a **subnetwork mask** (or subnet mask), which is used by network software to identify the host number field from the subnet number and network number fields. The portion of the address to be allocated to the subnet is defined (for example, Pundit University used the third octet of the address to identify the subnet). Then bits are set in the 32-bit subnet mask to correspond to the IP address. A subnet mask bit is set to 1 if the corresponding bit in the internet address is part of the network number and subnet number fields, and to 0 if the corresponding bit in the address is part of the host number field. The subnet mask for a Class A address might be

11111111 11111111 00000000 00000000

(or 255.255.0.0 decimal). Since this is a Class A address, octet 1 identifies the network field and octets 2, 3, and 4 identify the local field. The mask demarcates octets 3 and 4 as the host number field; therefore, the system knows that octet 2 of the address identifies the subnet number.

In theory, each LAN at the internet site could have a different number of bits allocated to its subnet field, and therefore have different subnet masks. However, it is generally recommended that a given network have a single subnet mask that is the same for all of its component LANs.

The domain name system

Because users would rather refer to machines using meaningful, symbolic names rather than long strings of numbers, the **domain name system** was created to map internet addresses to names.

The hierarchical naming scheme accommodates a large set of names and allows local autonomy in assigning names. A domain name is divided into subnames (called *labels*), separated by periods, for instance

```
english.pundit.edu
```

In this example, `edu` is the top-level domain and each label further specifies a subdomain. The Network Information Center (NIC) administers the top-level domains and is responsible for assigning subdomains. Table 3-1 shows the top-level domains specified by the NIC.

■ **Table 3-1** Top-level domains

Domain name	Meaning
COM	Commercial organization
EDU	Educational institution
GOV	Government institution
MIL	Military group
NET	Major network support centers
ORG	Organizations other than above
ARPA	Temporary ARPANET domain
ISO country code	Countries other than USA

After an organization obtains authority for a domain from the NIC, it can assign subordinate domain names. For instance, Pundit University obtained authority for the domain `pundit.edu` and the English Department at Pundit University obtained authority for the domain `english.pundit.edu`.

The organization with authority over a domain must maintain a domain name server that maps domain names to internet addresses. If the English Department at Pundit University obtained a new computer, Hamlet, the name

```
hamlet.english.pundit.edu
```

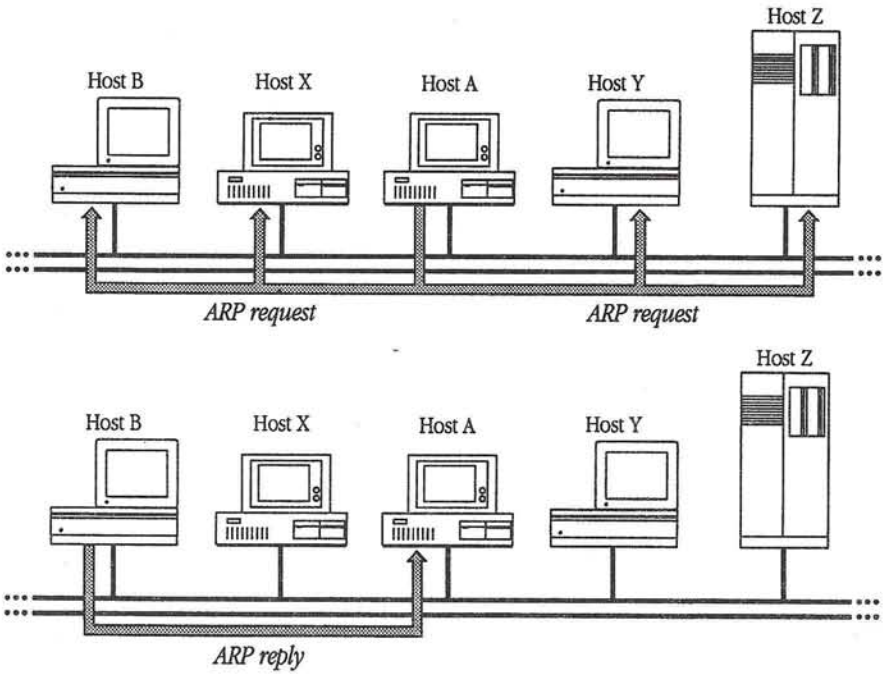
would be added to a domain name server's database, along with the computer's internet address.

Hosts participating in the domain name system must have **domain name resolvers** that request domain name information from **domain name servers**. The name resolver contacts a local name server to obtain the internet address associated with the domain name. The local name server may need to contact other domain name servers to obtain the internet address.

Address Resolution Protocol (ARP)

The **Address Resolution Protocol** (ARP) is a protocol in the TCP/IP protocol suite that maps internet addresses to physical network addresses. If Host A wants to communicate with Host B and it only knows B's internet address, it can use ARP to obtain B's physical address. Host A broadcasts an ARP request that contains Host B's internet address. All hosts on the network receive the request, but only Host B recognizes the internet address and replies with its physical address. Host A receives the reply, learns B's physical address, and delivers its packet directly to B. Figure 3-3 shows an ARP request and reply.

■ **Figure 3-3** ARP request and reply

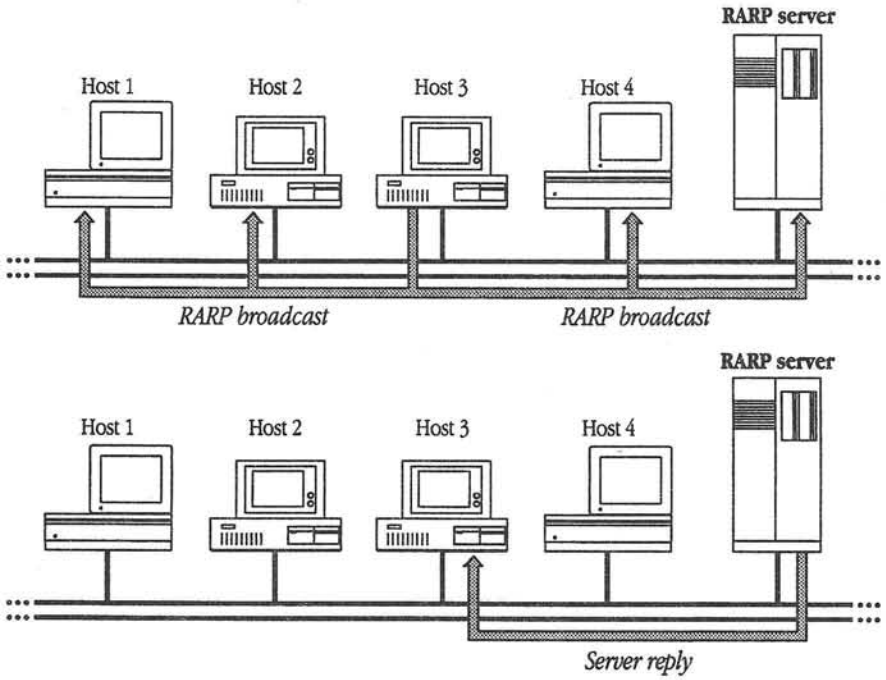


Reverse Address Resolution Protocol (RARP)

The **Reverse Address Resolution Protocol** (RARP) maps physical network addresses to internet addresses (the opposite function of ARP).

RARP provides a way for a host to obtain its internet address. The host broadcasts a RARP request that contains its physical network address. All hosts on the local network receive the request; however, only RARP servers can process the request and provide the internet address (RARP will work only if there's a RARP server on the network). The RARP server consults its database of internet addresses, then sends the information back to the host that made the request. Figure 3-4 shows a RARP broadcast and a RARP server reply.

■ **Figure 3-4** RARP broadcast and server reply



Bootstrap Protocol (BootP)

The **Bootstrap Protocol** (BootP), like RARP, provides a way for a host to find its internet address. A host running BootP broadcasts a BootP request. A BootP server returns a response that contains the host's internet address, the address of a bootserver, the address of an intervening gateway (if present), and other useful configuration information such as the subnet mask and addresses of domain name servers.

CHAPTER 4

Routing

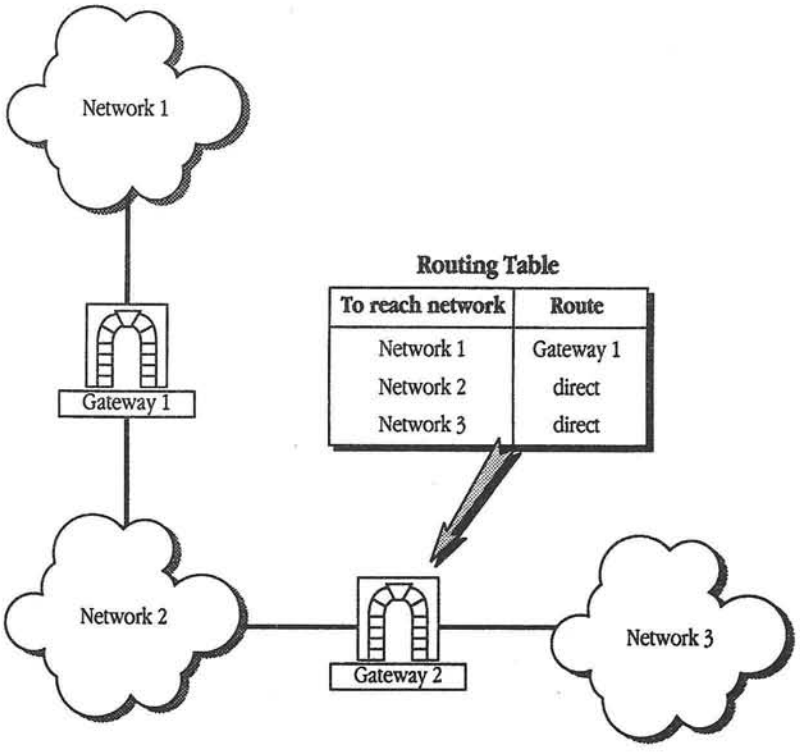
A PACKET TRANSMITTED BY A HOST ON THE INTERNET MAY ONLY NEED TO take a short jaunt across the local network or it may have to cross many gateways and networks to reach its destination. The process of finding a path over which the packet can travel to reach its destination is called **routing.** This chapter describes how TCP/IP handles routing.

Gateways

If a host is transmitting to a destination on its own network, routing a packet involves finding the destination host's physical address (using the Address Resolution Protocol) and sending the packet over the physical network to its destination. When the destination is on a different network, the packet is routed to the nearest gateway. A **gateway** is a machine that interconnects two networks and passes packets from one network to another. In Figure 4-1, three networks are interconnected by two gateways.

If the local gateway is not connected to the destination, the packet is forwarded to other gateways until it reaches a gateway that is directly connected to the destination network.

■ **Figure 4-1** Gateway routing table



Routing tables

All gateways in the internet have **routing tables** that contain pairs of network addresses and gateway addresses. Each network address is linked with the gateway address of the gateway to be used to get to that network. Routing decisions are based on the network number of the destination address rather than the host number to keep routing tables small. Figure 4-1 shows an example gateway routing table.

Routing protocols

Routing protocols provide a way for gateways to find each other, keep up-to-date routing information, and report communication problems.

Routing Information Protocol (RIP)

Some gateways use the **Routing Information Protocol** (RIP) to exchange network routing information. Gateways broadcast their routing tables to neighboring gateways. If a RIP message contains new information, gateways update their routing tables.

RIP is intended for low-delay local area networks (although it is used in wide area networks). The most widely used version of RIP is the **routed software** that is released with the 4.3BSD UNIX system.

RIP can also be used to locate gateways on a LAN. Any computer broadcasting RIP packets is likely to be a gateway.

While other routing protocols are used by gateway manufacturers, RIP is by far the most common protocol found in gateways.

Internet Control Message Protocol (ICMP)

Gateways and hosts use the **Internet Control Message Protocol** (ICMP) to report communication problems. For instance, if a packet cannot be delivered because the destination host is disconnected from the network, an ICMP message may be returned to the sending host stating that the destination is unreachable. ICMP is primarily used by gateways to notify the source host of delivery or routing problems.

ICMP is an integral part of the Internet Protocol (IP). Successful IP routing requires ICMP services to report error conditions. ICMP messages are encapsulated in IP datagrams, transmitted to the destination gateway or host, and processed by IP software.

PART TWO

Procedures

CHAPTER 5

Installation

THIS CHAPTER DESCRIBES HOW TO INSTALL THE MACTCP SOFTWARE ON A master disk. You start up from this disk before you perform the configuration tasks described in Chapter 6. Since users must install the software on their machines after you configure MacTCP, this chapter also includes a user installation section that you can copy and distribute to network users.

MacTCP software

The distribution disk contains the following folders:

- the Application Folder
- the MacTCP Folder
- the 512Ke Installer Script Folder
- the Hosts File Folder

The Application Folder is an empty folder into which you can put any applications, such as FTP or Telnet, for distribution along with the MacTCP software.

The MacTCP Folder contains the following files:

- the MacTCP file
- the AdminTCP file

The MacTCP file contains the MacTCP driver as well as other resources. The AdminTCP file is used to configure MacTCP. The configuration information is recorded in resource records that are stored in the MacTCP file.

The 512Ke Installer Script Folder contains the 512Ke Installer Script that is used to install MacTCP software on a master disk for a Macintosh 512K enhanced computer.

The Hosts File Folder contains the TeachText application and a sample hosts file. Appendix A describes how to edit the hosts file.

Creating a master disk

To install the MacTCP software on a master disk, perform the following steps:

1. **Lock the distribution disk.**

   To lock the disk, slide the colored tab toward the edge of the disk (revealing a small hole).

2. **Make a duplicate of the distribution disk.**

   To copy the files in the distribution disk to another disk (which will be the master disk), insert both the distribution disk and a blank disk into your Macintosh disk drives. (The blank disk may have to be initialized.) If you have a one-drive system, insert the blank disk, eject it by choosing Eject from the File menu, and then insert the distribution disk. Drag the distribution disk icon to the blank disk icon. Eject the distribution disk and put it in a safe place.

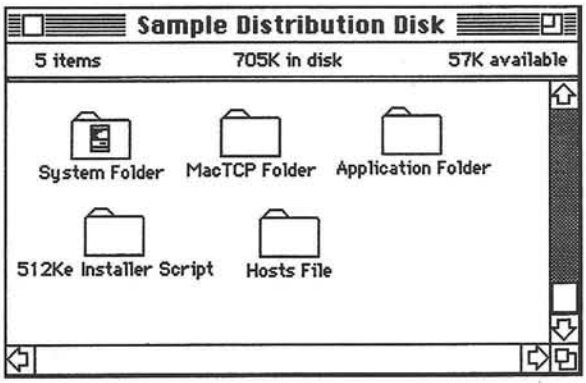
3. **If you have a one-drive system, insert the duplicate, master disk into your disk drive.**

4. **Install System file version 6.0.3 on the duplicate distribution disk.**

5. **Open the disk icon.**

   Select the disk icon and choose Open from the File menu, or double-click the icon. Five folders are displayed in the disk window as shown in Figure 5-1.

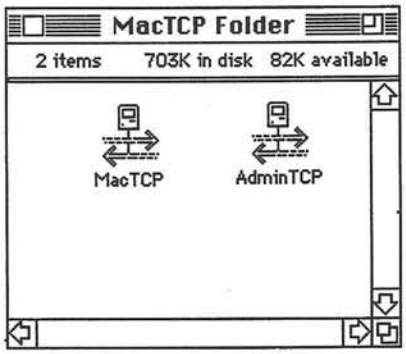
■ **Figure 5-1** Disk folders



6. **Open the MacTCP Folder.**

   Select the MacTCP Folder and choose Open from the File menu, or double-click the folder. The folder contains the icons displayed in Figure 5-2.

■ **Figure 5-2** Icons in the MacTCP Folder



7. **Drag the MacTCP and AdminTCP icons into the System Folder.**

   Select both the MacTCP and AdminTCP icons (click one icon and then hold down the Shift key and click the other icon). Drag them into the System Folder (on the master disk).

8. **Throw away the MacTCP Folder.**

   Select the MacTCP folder and drag it into the Trash.

9. **If you want to use the Hosts file for name-to-address mapping, open the Hosts File Folder and drag the Hosts file into the System Folder.**

   Select the Hosts File Folder and choose Open from the File menu, or double-click the folder. Then select the file called Hosts and drag it into the System Folder (on the master disk). To customize the contents of the Hosts file, refer to Appendix A.

10. **Start up from the master disk.**

    Choose Restart from the Special menu. The disk will be ejected.

    If your system has no hard disk, a disk icon with a blinking question mark will appear. Reinsert the disk.

    If your Macintosh computer has a hard disk, reinsert the disk quickly before your system starts up from the hard disk. The floppy disk icon will appear in the upper-right corner of the screen, above the hard disk icon, indicating that the floppy disk is the startup disk.

11. **Configure the MacTCP software as described in Chapter 6.**

12. **Throw away the AdminTCP icon.**

    Double-click the System Folder on the master disk. Select the AdminTCP icon and drag it into the Trash.

13. **Put a copy of your network application in the Application Folder.**

14. **Distribute the disk to a network user along with the appropriate installation instructions.**

User installation instructions

This section describes installation tasks that must be performed by network users. Copy the appropriate pages and distribute the copies to users along with a master disk containing MacTCP software that you've configured for each user.

Installing MacTCP on a floppy disk-based system

To install MacTCP on a floppy disk-based system, perform the following steps:

1. **Get the master disk from your network administrator.**

2. **Create a duplicate master disk.**

   To copy the files in the master disk to another disk, insert both the master disk and a blank disk into your disk drives. (The blank disk may have to be initialized.) If you have a one-drive system, insert the blank disk, eject it by choosing Eject from the File menu, and then insert the master disk. Drag the master disk icon to the blank disk icon. Eject the master disk and put it in a safe place.

3. **Start up from the duplicate master disk.**

   Choose Restart from the Special menu. The disk is ejected; then a disk icon with a blinking question mark appears. Reinsert the disk.

4. **Your network administrator will provide you with instructions on how to configure MacTCP on your Macintosh computer.**

Installing MacTCP on a hard disk

To install MacTCP on your Macintosh computer's hard disk, perform the following steps:

1. **Get the master disk from your network administrator.**

2. **Create a duplicate master disk.**

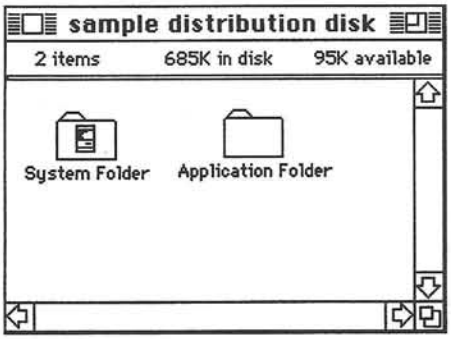
   To copy the files in the master disk to another disk, insert both the master disk and a blank disk into your disk drives. (The blank disk may have to be initialized.) If you have a one-drive system, insert the blank disk, eject it by choosing Eject from the File menu, and then insert the master disk. Drag the master disk icon to the blank disk icon. Eject the master disk and put it in a safe place.

3. **If you have a one-drive system, insert the duplicate master disk into your disk drive.**

4. **Open the disk icon.**

Select the disk icon and choose Open from the File menu, or double-click the icon. Two folders are displayed in the disk window as shown in Figure 5-3.

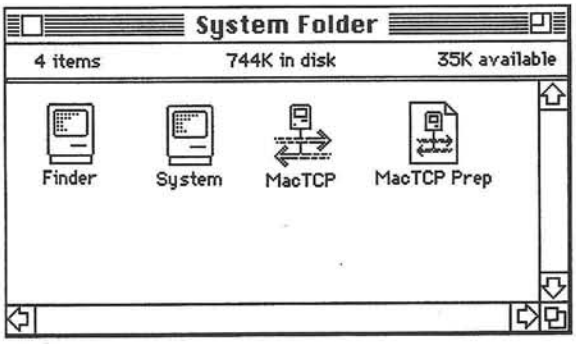
■ **Figure 5-3** Disk folders



5. **Open the System Folder.**

Select the System Folder on the disk and choose Open from the File menu, or double-click the folder. The folder contains the icons displayed in Figure 5-4.

■ **Figure 5-4** Icons in the System Folder



6. **Put the MacTCP icon into your hard disk's System Folder.**

   Select the MacTCP icon, shown in Figure 5-4, and drag it into the System Folder on your hard disk.

7. **Put the Application Folder on your hard disk.**

   Select the Application Folder on the master disk and drag it onto your hard disk icon.

8. **Choose Restart from the Special menu.**

9. **Your network administrator will provide you with instructions that describe how to configure MacTCP on your Macintosh computer.**

CHAPTER 6

MacTCP Configuration

THIS CHAPTER DESCRIBES HOW TO CONFIGURE MACTCP FOR YOUR NETWORK users. Read the section "Configuration Tools" before you start because this section provides an overview of the MacTCP Control Panel and the Administrator dialog box, the two tools that are used to configure MacTCP.

You can set up MacTCP in several ways, depending on how much of the configuration you want to leave up to your network users. Read the section "Configuration Scenarios" to determine your strategy.

Recall from Chapter 5 that the master disk's System Folder contains two files, MacTCP and AdminTCP. When you configure MacTCP, the AdminTCP file modifies resources in the MacTCP file and the MacTCP Prep file. After the configuration session, you throw away the AdminTCP file. You can then distribute the configured master disk to network users.

This chapter assumes that you are bringing up the MacTCP Control Panel screens for the first time. If you have configured the driver once, the values that appear will be somewhat different from those depicted here.

Make sure that you have installed the MacTCP software according to the instructions in Chapter 5 before you proceed with the tasks described in this chapter.

Configuration tools

There are two MacTCP configuration tools: the MacTCP Control Panel and the Administrator dialog box. The way you use these tools depends on the configuration scenario you choose to implement. (These scenarios are described in the section "Configuration Scenarios" later in this chapter.)

As network administrator, you use the Administrator dialog box to configure MacTCP. In two cases, however, you must use the MacTCP Control Panel:

- If you are configuring MacTCP for a Macintosh 512K enhanced computer, you must set the link level information using this panel.
- If you decide to enter the IP address in decimal notation, you must do so using this panel.

The network user configures MacTCP using the MacTCP Control Panel. The user can view the Administrator dialog box but not change it. However, you can give the user full configuration powers and he or she can use the panels just as you would.

The MacTCP Control Panel

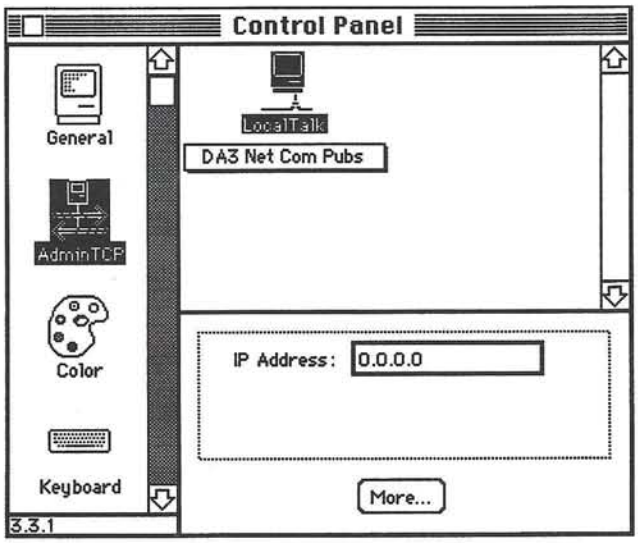
Figure 6-1 shows the MacTCP Control Panel.

The top half of the screen is used to set link level information. Different icons are displayed depending on the user's network configuration. The user selects the link level protocol on which MacTCP is to run.

❖ *Note:* If you are configuring MacTCP for a Macintosh 512K enhanced computer, you must set the link level information because the user will not have access to this Control Panel.

The other half of the screen is used to set IP address information.

■ **Figure 6-1** View of the MacTCP Control Panel



If you want to enter the IP address in dotted decimal notation, enter it in the IP Address box. If you enter the address here, portions of the Administrator dialog box are filled out.

The way the user fills in the IP address information depends on how you configured the Administrator dialog box. There are five possibilities:

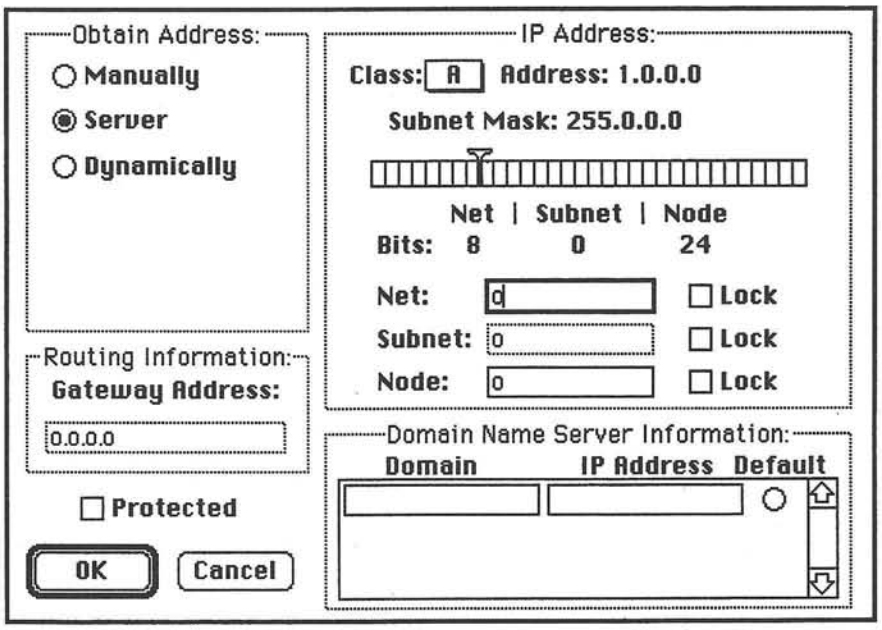
- You have selected the Manually button in the Administrator dialog box and preconfigured the entire IP address so the user doesn't have to enter it. The IP Address box appears on the MacTCP Control Panel with the address already set.
- You have not set the IP address in the Administrator dialog box so the user must enter the address in the IP Address box. Only the IP Address box appears, and the user must enter the address in this box (in decimal notation).
- You have entered the net and subnet portions of the address in the Administrator dialog box, and the user must enter the node number. Net, Subnet, and Node boxes appear on the MacTCP Control Panel, and the user must enter the node number in the Node box.

- You have selected the Server button in the Administrator dialog box so that the IP address is obtained automatically from a Server. The IP Address box appears on the MacTCP Control Panel with the address already set.
- You have selected the Dynamically button in the Administrator dialog box so that the node portion of the IP address is assigned dynamically from the indicated range. The IP Address box appears on the MacTCP Control Panel with the address already set.

The Administrator dialog box

Figure 6-2 shows the Administrator dialog box. You and the user click the More button on the MacTCP Control Panel to display this screen. For the user without configuration power, the screen is displayed with all the fields grayed out; the user can view the screen but not change it.

- **Figure 6-2** View of the Administrator dialog box



Obtain Address box

This is where you select the method in which you want to set the IP address. You have three options:

- If you select the Manually button, you must also fill in some or all of the fields in the IP Address box.
- If you select the Server button, the IP address is set for you (a network server is queried for the address). A Macintosh computer that has MacTCP configured in this way automatically obtains address information every time it boots up. This is a way to create a master disk that can be distributed to several network users.

  If you select this option, there must be a RARP or BootP server on your Ethernet or a KIP-compatible Datagram Delivery Protocol-Internet Protocol (DDP-IP) gateway on your LocalTalk network.
- If you select the Dynamically button, the node portion of the IP address is set dynamically If you use this option, you must also set some of the fields in the IP Address box. A Macintosh computer that has MacTCP configured in this way assigns itself a node number dynamically every time it boots up. This is another way to create a master disk that can be distributed to several network users.

IP Address box

This is where you set the address class, subnet mask, net, subnet, and node numbers. You may fill in some or all of these fields depending on the configuration scenario that you choose to implement.

Routing Information box

This is where you set the gateway address in dotted decimal notation if the Routing Information Protocol (RIP) is not implemented on your network.

Domain Name Server Information box

This is where you enter the IP addresses of domain name servers and the domains for which they have authority.

Protected button

If you select this button, the user will not be able to modify any of the fields on the Administrator dialog box.

Configuration scenarios

You can configure MacTCP in several ways; you must decide how much of the configuration you want to leave up to network users. The roadmap in Figure 6-3 shows the chapter sections you must refer to in order to accomplish each scenario.

Possible configuration scenarios are as follows:

Scenario 1 Configure MacTCP so that the user does not have to fill in any IP address information in the MacTCP Control Panel. If this is your strategy, you have two options. You can enter the IP address in the MacTCP Control Panel (in decimal notation) and then enter the subnet information in the Administrator dialog box (if appropriate). Or you can enter the address class, subnet mask, net, subnet, and node numbers in the Administrator dialog box. If you select this method, you must create and configure a separate master disk for each network user.

Scenario 2 Configure MacTCP so that the user must fill in the IP address in the MacTCP Control Panel (in decimal notation). If this is your strategy, enter the address class and subnet mask but do not fill in the Net, Subnet, and Node boxes. If you select this method, you can distribute the master disk to several users.

Scenario 3 Configure MacTCP so that the user must fill in the node portion of the IP address in the MacTCP Control Panel. If this is your strategy, enter the address class, subnet mask, net number, and subnet number but do not enter the node number in the Node box. If you select this method, you can distribute the master disk to several users.

Scenario 4 Configure MacTCP so that the address is automatically assigned. (The user does not have to fill in any IP address information in the MacTCP Control Panel.) If this is your strategy, select the Server button. If you select this method, you can distribute the master disk to several users.

Scenario 5 Configure MacTCP so that the node portion of the address is assigned dynamically. (The user does not have to fill in any IP address information in the MacTCP Control Panel.) If this is your strategy, select the Dynamically button and enter the address class, subnet mask, net, and subnet numbers.

Scenario 6 The user has been granted full configuration power and can use the Administrator dialog box to configure MacTCP.

■ **Figure 6-3** Roadmap to configuration scenarios

The dots indicate sections you should read to accomplish each scenario.

Refer to these sections:	Scenario 1	Scenario 2	Scenario 3	Scenario 4	Scenario 5	Scenario 6
Bringing up the MacTCP Control Panel	•	•	•	•	•	
Entering the IP Address in Decimal Notation	•					
Bringing up the Administrator Dialog Box	•	•	•	•	•	
Setting the IP Address	•	•	•	•	•	
Setting the Address Manually	•	•	•		•	
Setting the Address Class	•	•	•		•	
Setting the Subnet Mask	•	•	•		•	
Setting the IP Address (Integer Format)	•		•		•	
Entering the Net Number	•		•		•	
Entering the Subnet Number	•		•		•	
Entering the Node Number	•					
Having the Address Set Automatically				•		
Setting the Node Number Dynamically					•	
Setting the Gateway Address	•	•	•	•	•	
Setting Domain Name Server Information	•	•	•	•	•	
Protecting the Configuration	•	•	•	•	•	
Giving the User Full Configuration Power						•

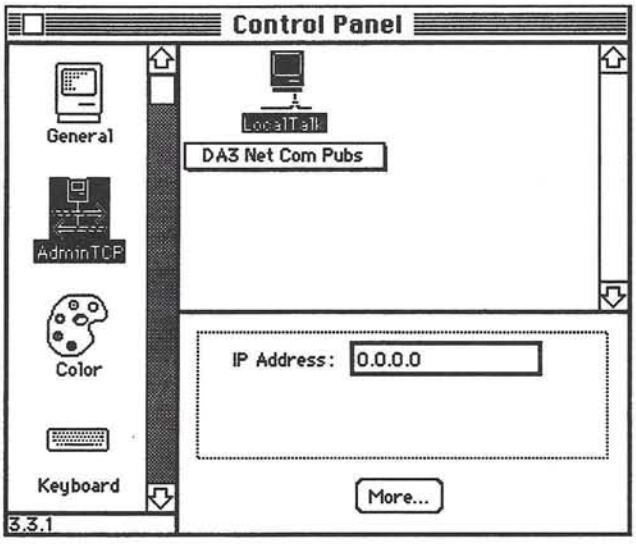
Bringing up the MacTCP Control Panel

Choose Control Panel from the Apple menu, and the panel appears with a scrollable list of icons in the left quarter of the window.

Click the AdminTCP icon (it may be necessary to use the scroll bar to bring the AdminTCP icon into view). The MacTCP Control Panel appears on the right three-quarters of the window as shown in Figure 6-4.

△ **Important** If you are configuring MacTCP for a Macintosh 512K enhanced computer, you must completely configure the driver, including setting the link level information on this panel as described in Chapter 7. The Macintosh 512K enhanced user cannot access the MacTCP Control Panel because System file version 3.4 does not support an extendible Control Panel. △

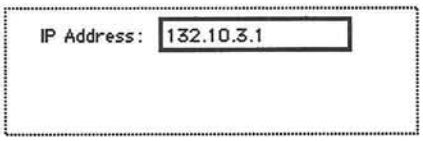
■ **Figure 6-4** The MacTCP Control Panel



Entering the IP address in decimal notation

If you have decided to enter the entire IP address for the user and you prefer to enter the address in dotted decimal notation (for example, 132.10.3.1), enter the address in the IP Address box as shown in Figure 6-5.

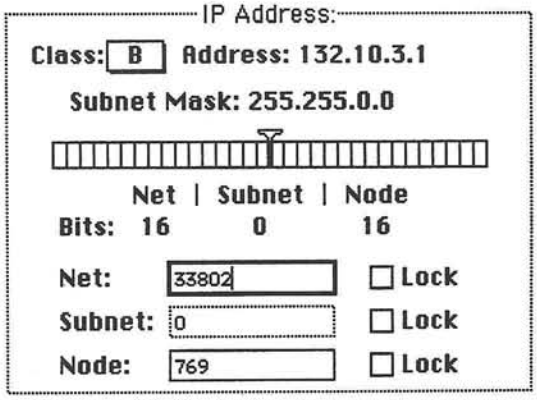
■ **Figure 6-5** IP Address box in the MacTCP Control Panel



When you click the More button and the Administrator dialog box is displayed, the class, address, and net and node numbers are set as shown in Figure 6-6.

❖ *Note:* Entering the address in the MacTCP Control Panel does not set the subnet mask and subnet number.

■ **Figure 6-6** IP Address box in the Administrator dialog box

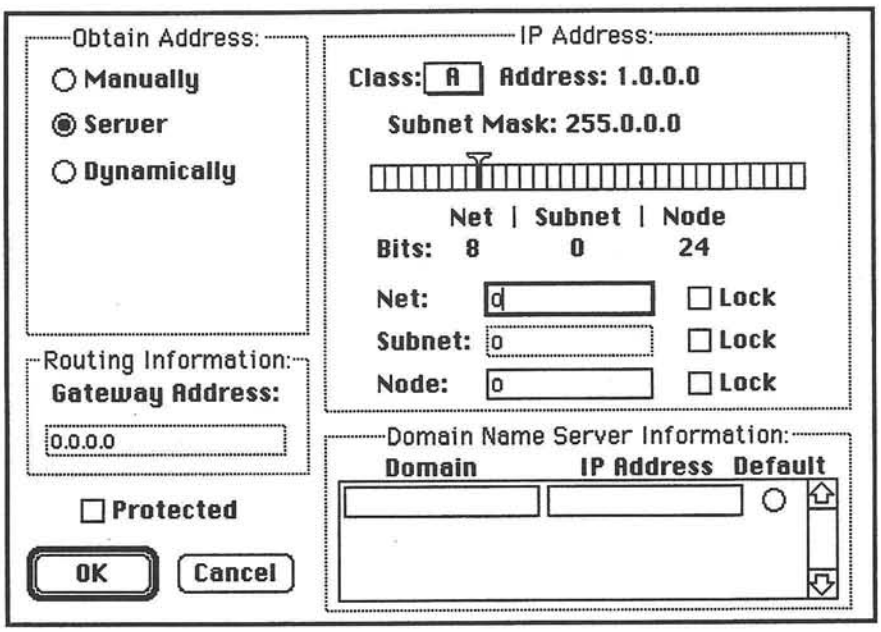


Bringing up the Administrator dialog box

Click the More button to get to the Administrator dialog box shown in Figure 6-7.

This panel is your main tool for configuring MacTCP.

■ **Figure 6-7** The Administrator dialog box

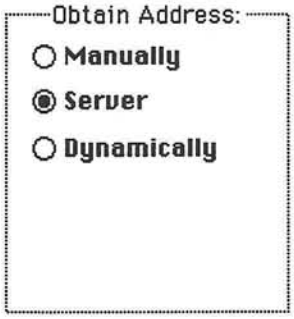


Setting the IP address

MacTCP provides you with three ways to set the IP address (if you did not enter the decimal address in the MacTCP Control Panel). The Obtain Address box in the upper-left corner of the panel (shown in Figure 6-8) allows you to select your preferred method for setting the address.

- If you select Manually, you must set the address manually using the fields in the IP Address box. See the next section "Setting the Address Manually."
- If you select Server, the address is set automatically. See the section "Having the Address Set Automatically" later in this chapter.
- If you select Dynamically, you must set the net and subnet portions of the address using the fields in the IP Address box, but the node portion of the address will be allocated dynamically (within the range of node numbers specified). See the section "Setting the Node Number Dynamically" later in this chapter.

■ **Figure 6-8** Obtain Address box



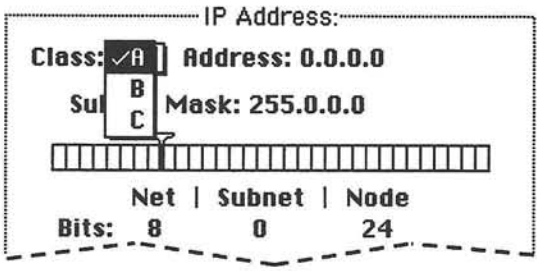
Setting the address manually

To set the address manually, click the Manually button located in the Obtain Address box. Then follow the steps described in the following sections.

Setting the address class

Move to the IP Address box and position the pointer on the Class box. Class A, the default setting, appears in the box. Hold down the mouse button and a menu of classes appears as shown in Figure 6-9. The ✓ indicates which class is the current setting. If you want to change the class, drag the pointer to B or C.

■ **Figure 6-9** Menu of address classes

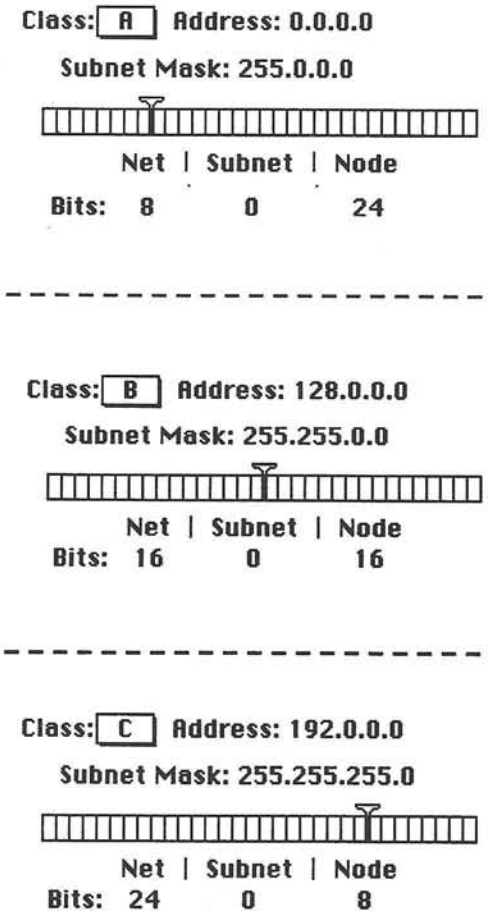


Net: 12582912

If you change the class, the new value that appears in the Net box is the minimum value allowable for the selected address class. For instance, if you select class C, the value in the net box changes to 12582912, which is the minimum value that can be used for class C addresses.

Note that changing the class causes the slider on the ruler to move. The address, subnet mask, and bits allocated to net, subnet, and node also change. Figure 6-10 shows the address, subnet mask, position of the slider, and the bit allocation for each class selection.

■ **Figure 6-10** Settings for classes A, B, and C



Setting the subnet mask

❖ *Note:* Perform this step only if you have implemented subnet addressing on your network.

Use the slider on the ruler to set the subnet mask. Each box in the ruler represents one bit of the 32-bit IP address. A dark line on the ruler indicates the number of bits allocated to the net portion of the address (as determined by the class selected). The slider can be moved anywhere along the ruler to the right of this darkened line. Place the pointer on the slider, hold down the mouse button, and drag the slider to the appropriate location on the ruler. When you move the slider, the subnet mask, the subnet bits, and the node bits change.

If you move the slider all the way to the left, the Subnet box is dimmed.

Setting the IP address (integer format)

△ **Important** If you want the user to enter the IP address in the IP Address box of the MacTCP Control Panel (scenario 2 in the earlier section "Configuration Scenarios"), skip this step. Continue with the configuration, referring to the section "Setting the Gateway Address" later in this chapter. The user will have to enter the IP address in the MacTCP Control Panel as described in Chapter 7. △

In integer format, the IP address is broken down into bits (four sets with eight bits per set). The bits are allocated to net, subnet, and node according to address class and subnet mask, then converted to a decimal number. For instance, the address 90.25.3.240, broken down into bits, looks like this:

01011010.00011001.00000011.11110000

If the address class is A (8 bits comprise the net portion of the address) and the subnet mask is 255.255.252.0 (or 14 bits of subnet address and 10 bits of node), then the integer form of the address is as follows:

- Net - 90
- Subnet - 1600
- Node - 1008

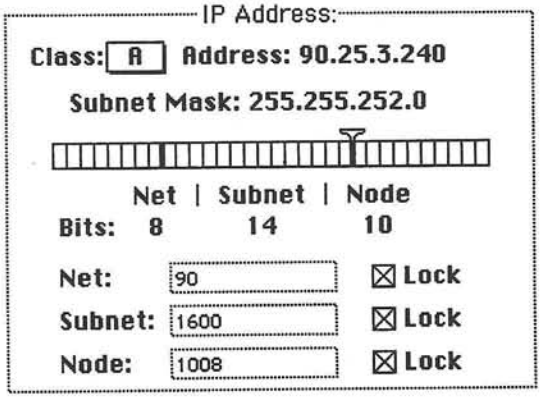
If you want to enter the IP address in integer format, enter this information in the Net, Subnet, and Node boxes shown in Figure 6-11.

△ **Important** If you are configuring MacTCP so that the user has to enter the node number in the MacTCP Control Panel (scenario 3 in the earlier section "Configuration Scenarios"), you must enter the net and subnet portion of the address in integer format in the Net and Subnet boxes. △

Entering the net number The Net box contains the minimum value that can be used, determined by the address class that was selected. Click twice in the Net box and enter the net portion of the IP address in integer form. If you try to enter a value over the maximum allowed for the selected address class, the last digit in the number you enter isn't accepted; you have to reenter a valid number. If you enter a number under the minimum allowed for the selected address class, a valid number is automatically assigned when you save your changes to the screen; therefore, be sure to enter a value over the minimum allowed for the selected address class.

If you click the Lock box to the right of the Net box, the net number is protected. The Net box is dimmed and you cannot change its value unless you click the Lock box to unlock it. If you lock the value in the Net box, the user will be unable to change this number on the MacTCP Control Panel.

■ **Figure 6-11** Values set in Net, Subnet, and Node boxes



Entering the subnet number Note that you can only enter the subnet number if you set the subnet mask. Click the Subnet box and enter the subnet portion of the IP address in integer form. If you click the Lock box to the right of the Subnet box, the subnet number is protected. The Subnet box is dimmed, and you cannot change the value in the box unless you click the Lock box to unlock it. If you lock the value in the Subnet box, the user will be unable to change this number on the MacTCP Control Panel.

Entering the node number You now have three choices: enter the node number, leave the Node box at the default and have the user enter the node number, or have the node number assigned dynamically.

If you decide to enter the node number, click the Node box and enter the node portion of the IP address in integer form. If you click the Lock box to the right of the Node box, the node number is protected. The Node box is dimmed and you cannot change the value in the box unless you click the Lock box to unlock it. If you lock the value in the Node box, the user will be unable to change this number on the MacTCP Control Panel.

If you decide to have the user enter the node number in the MacTCP Control Panel, leave the box at the default and do not select the Lock box. Then continue with the configuration, starting with the section "Setting the Gateway Address" later in this chapter.

If you want the node number to be assigned dynamically every time the user starts up, see the section "Setting the Node Number Dynamically" later in this chapter.

Figure 6-11 shows values set and locked in the Net, Subnet, and Node boxes.

Having the address set automatically

If you select the Server button in the Obtain Address box, a network server is queried for the address assigned to your machine. On an Ethernet network, the protocols BootP or RARP are used. On a LocalTalk network, a DDP-IP gateway sets the address.

All you have to do is click on the Server button (when you bring up the Administrator dialog box for the first time, Server is the default setting). There is no need to enter class, subnet mask, or net, subnet, or node numbers; the Server button does it for you. The Macintosh computer that uses this master disk will be assigned an address every time it starts up as long as you have properly configured your server.

After you restart and your configuration changes are saved (see the section "Alert Message" later in this chapter), the class, subnet mask, and net, subnet, and node numbers set by the Server button are reflected in the Administrator dialog box.

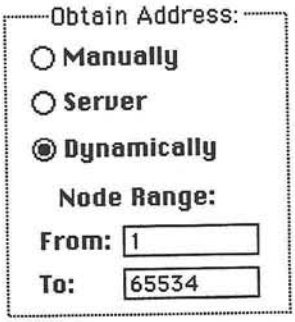
Setting the node number dynamically

You can select Dynamically in the Obtain Address box so that the node portion of the IP address is assigned dynamically when the user boots up.

An IP address has a range of valid node numbers that are determined by the address class and subnet mask. With dynamic addressing, MacTCP randomly selects a node address in that range and broadcasts to other nodes on the network, "Is there anyone out there using this address?" If there is no response, that node number will be used; if there is a response, the computer will rebroadcast until it finds a number that is not being used by other machines on the network.

To use dynamic addressing, enter the address class, subnet mask, and net and subnet numbers as described in the earlier section "Setting the Address Manually." Click on the Dynamically button (located in the Obtain Address box). Two boxes appear, illustrated in Figure 6-12, that show the range of valid node numbers (determined by class and subnet mask). After changes made to the Administrator dialog box are saved, a node number is assigned dynamically when the Macintosh computer is started up using this master disk.

■ **Figure 6-12** Node Range boxes



If you know there are static addresses on your network, you can change the values in the Node Range boxes so the addresses in the static range are not used. To change the minimum range value, double-click the From box and enter an appropriate number. To change the maximum range value, double-click the To box and enter an appropriate number.

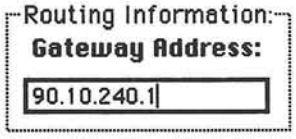
Setting the gateway address

MacTCP automatically monitors Routing Information Protocol (RIP) traffic to determine active gateways. If your network does not use RIP, you must manually set the address of a gateway.

- *Note:* The default gateway address can also be configured from a server using BootP.

To manually set the gateway address, go to the Routing Information section of the Administrator dialog box. Click the pointer in the Gateway Address box and enter your gateway's decimal IP address as shown in Figure 6-13.

■ **Figure 6-13** Routing Information box



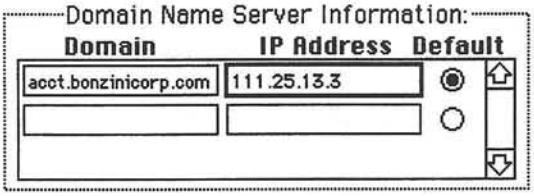
Setting domain name server information

This part of the Administrator dialog box allows you to enter the IP address of domain name servers and the domains over which they have authority. The box allows you to enter this information for your network's domain name server and other domain name servers on the internet.

Click the pointer in the Domain box and enter a domain. Press the Tab key to move the pointer to the IP Address box and enter the address of that domain. As you enter the IP address, boxes appear (shown in Figure 6-14) to allow you to enter more domain name server information.

Click the default button to indicate your default domain name server. Generally, it should be a domain name server that has authority over your domain.

■ **Figure 6-14** Domain name server information



Protecting the configuration

To protect your configuration, click the Protected button (shown in Figure 6-15). If you don't select the Protected button, the user who receives the configured master disk will be able to modify any of the fields on the Administrator dialog box.

■ **Figure 6-15** The Protected button



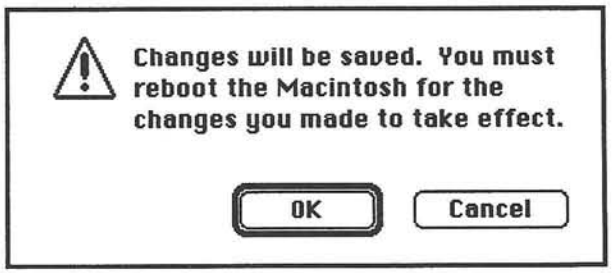
Giving the user full configuration power

Recall the sixth scenario in the section "Configuration Scenarios" earlier in this chapter: you can configure MacTCP so that the user can make changes through the Administrator dialog box, giving him or her full configuration power. To accomplish this, take the AdminTCP file out of the System Folder on the master disk (throw it into the Trash) and distribute the disk along with a copy of this chapter. Since the Protected button has not been selected (see the previous section, "Protecting the Configuration"), the user can access the Administrator dialog box and use it to configure MacTCP. In the other configuration scenarios, you select the Protected button so that the user is not able to make changes through the Administrator dialog box.

Alert message

After you have filled in all the appropriate information in the Administrator dialog box, click the OK button. The alert message shown in Figure 6-16 is displayed.

■ **Figure 6-16** The alert box



This message alerts you that if you click the OK button, you must restart your Macintosh computer from the master disk in order for the configuration changes to take effect. Click the Cancel button if you don't want the changes saved (for instance, if you realize you entered the wrong gateway address).

If you click the OK button, the MacTCP Control Panel is displayed; if you click the Cancel button the Administrator dialog box is redisplayed.

MacTCP is configured

You have now configured MacTCP for at least one of your network users. The configuration settings are stored in the MacTCP driver and also in a file called MacTCP Prep.

Click the Close box to close the Control Panel window. Drag the AdminTCP file into the Trash and distribute the software to users as described in Chapter 5.

CHAPTER 7

User Configuration

THIS SECTION HAS BEEN PROVIDED TO HELP YOU DESCRIBE THE STEPS users must follow in order to configure MacTCP. Copy and distribute it as appropriate.

If you selected the Protected button on the Administrator dialog box when you configured MacTCP, the user can only make changes to the MacTCP Control Panel. If you did not select the Protected button, the user has full configuration power and can use the Administrator dialog box to configure MacTCP; provide him or her with the configuration instructions in Chapter 6.

Bringing up the MacTCP Control Panel

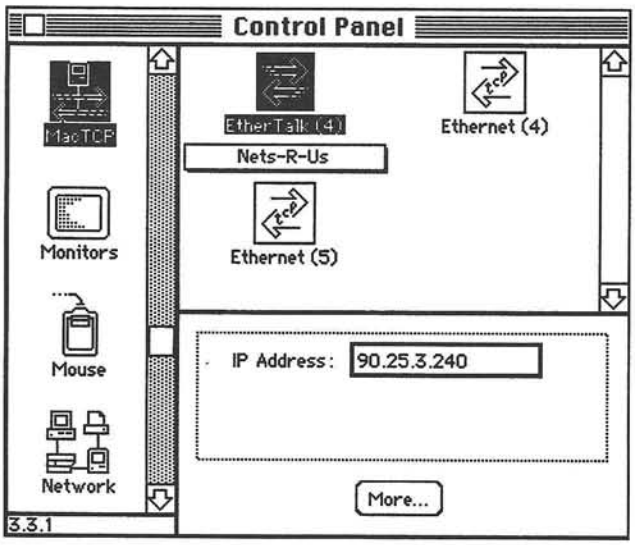
Choose Control Panel from the Apple menu and the panel appears with a scrollable list of icons in the left quarter of the window.

Selecting the MacTCP icon

Click the MacTCP icon (it may be necessary to use the scroll bar to bring the MacTCP icon into view). A new display appears on the right three-quarters of the window as shown in Figure 7-1.

Unless your network administrator has given you full configuration power, use this Control Panel to configure your system. If you do have full power, the network administrator should provide you with the appropriate instructions.

■ **Figure 7-1** The MacTCP Control Panel



Setting link level information

The top half of the screen displays the available link level protocols on which MacTCP can run. Different icons will be displayed depending on your network configuration.

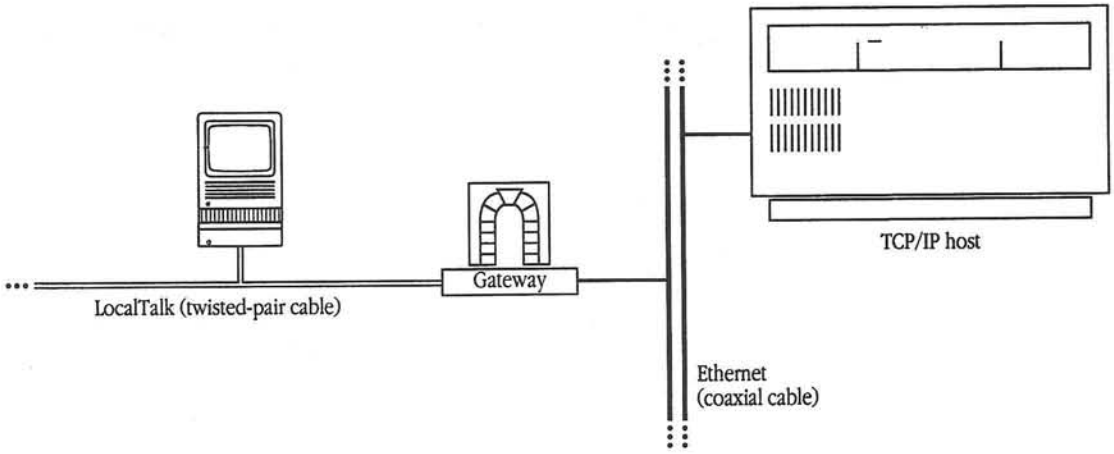
If your Macintosh computer is using an Ethernet interface card, make sure the card's software is installed.

❖ *Note:* You may need to consult your network administrator to determine your network configuration and the zone where your DDP-IP gateway is located.

Network configuration 1

Figure 7-2 illustrates one possible network configuration: a Macintosh computer on an AppleTalk network using LocalTalk cabling with a gateway to an internet.

■ **Figure 7-2** Network configuration 1



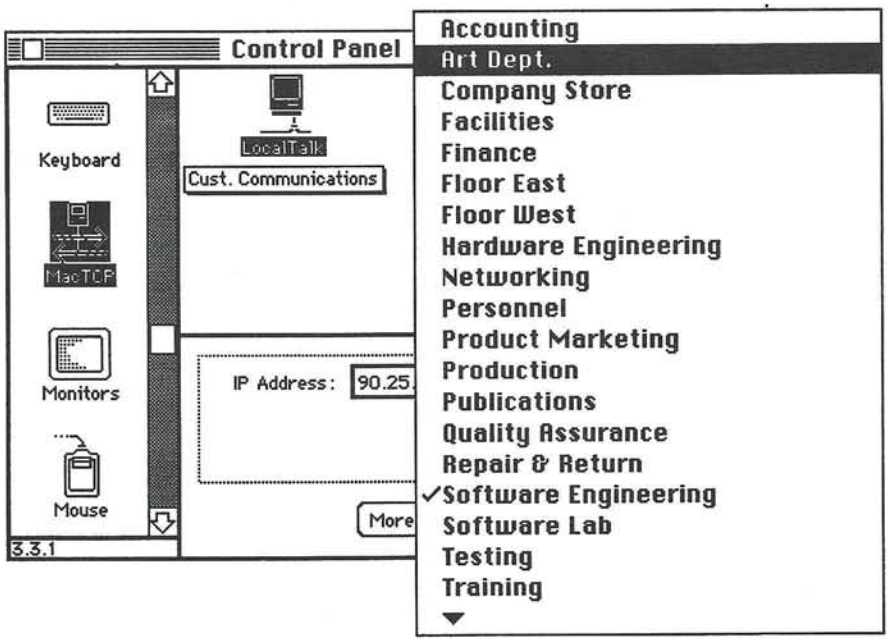
If this is your network configuration, the upper-right section of the MacTCP Control Panel appears as shown in Figure 7-3.

■ **Figure 7-3** Icons displayed for network configuration 1



The box beneath the icon displays your network zone. If your gateway is located in another zone, direct the pointer to the zone box and hold down the mouse button. A pop-up menu appears with a list of zone names as shown in Figure 7-4. Keeping the mouse button down, slide the pointer to the list and highlight the appropriate zone. When you let go of the mouse button, the pop-up menu disappears and the zone name you selected appears in the box beneath the icon.

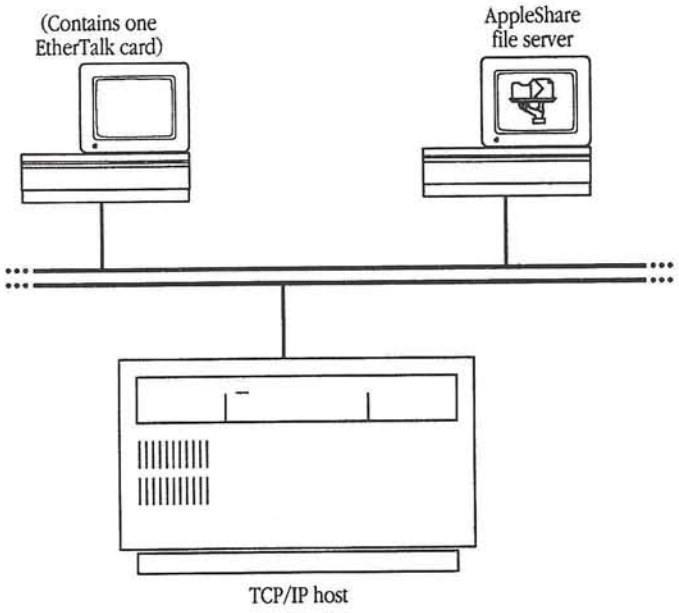
■ **Figure 7-4** Zone pop-up menu



Network configuration 2

Figure 7-5 illustrates a second possible network configuration: a Macintosh computer with one EtherTalk card on an AppleTalk network using Ethernet cabling.

■ **Figure 7-5** Network configuration 2



If this is your network configuration, the upper-right section of the MacTCP Control Panel appears as shown in Figure 7-6.

■ **Figure 7-6** Icons displayed for network configuration 2

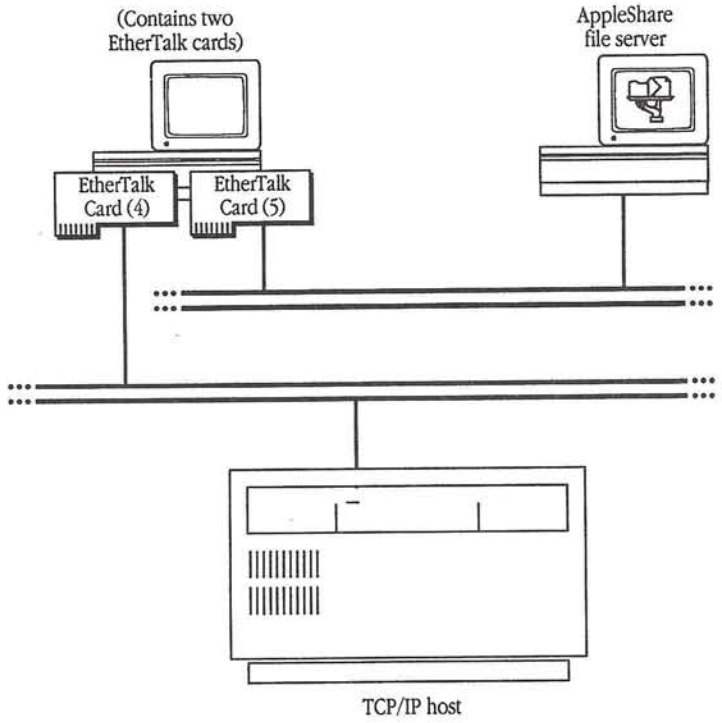


Select the Ethernet icon to enable your Macintosh computer to use TCP/IP to communicate with other TCP/IP hosts on the network. (AppleTalk is already being used to communicate with the AppleShare® File Server and LaserWriter.)

Network configuration 3

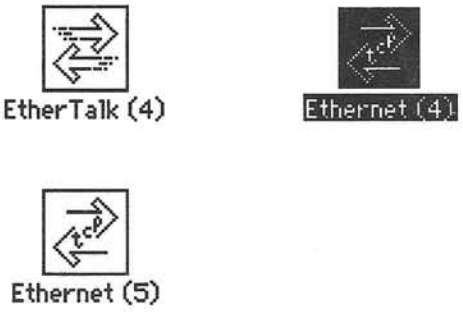
Figure 7-7 illustrates a third possible network configuration: a Macintosh II computer with two EtherTalk cards (located in slots 4 and 5) on an AppleTalk network using Ethernet cabling.

■ **Figure 7-7** Network configuration 3



If this is your network configuration, the upper-right section of the MacTCP Control Panel will appear as shown in Figure 7-8. (If the EtherTalk cards are located in slots other than 4 and 5, the numbers in the icon names will be different.)

■ **Figure 7-8** Icons displayed for network configuration 3

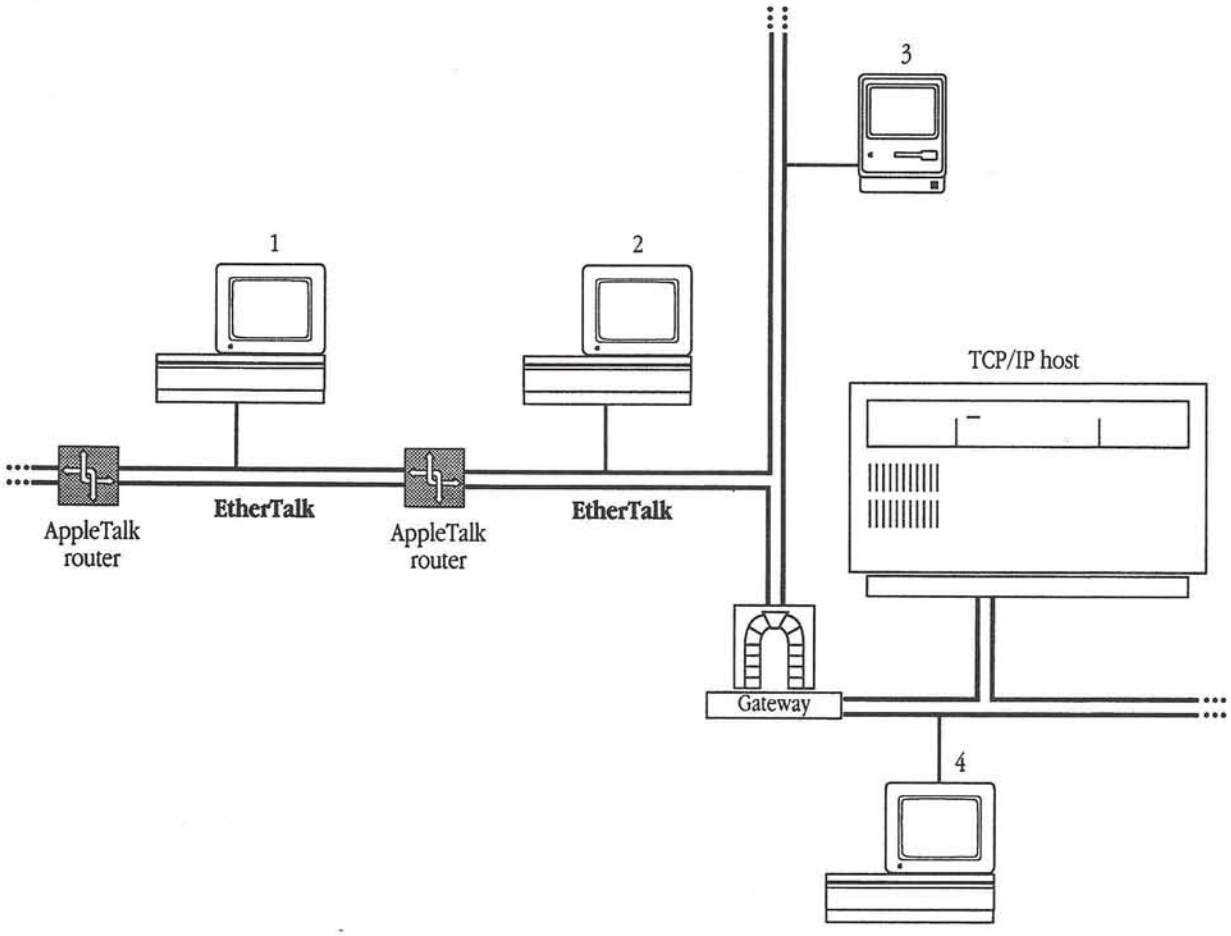


Select the Ethernet (4) icon to enable your Macintosh computer to use TCP/IP to communicate with other TCP/IP hosts on that network.

Network configuration 4

Figure 7-9 illustrates a fourth possible network configuration. Macintosh computers 1, 2, and 3 are on EtherTalk cable segments separate from the TCP/IP host. These computers use AppleTalk protocols to talk to the gateway and the gateway uses TCP/IP protocols to talk to the TCP/IP host. Macintosh computer 4, on the same Ethernet as the TCP/IP host, uses standard TCP/IP protocols to talk directly to the TCP/IP host.

■ **Figure 7-9** Network configuration 4



If this is your network configuration, the upper-right section of the MacTCP Control Panel will appear as shown in Figure 7-10.

■ **Figure 7-10** Icons displayed for network configuration 4



If you must go through a gateway to reach a TCP/IP host (as hosts 1, 2, and 3 do in Figure 7-9), make sure the EtherTalk icon is selected. If you are on the same Ethernet as the TCP/IP host (as host 4 is in Figure 7-9), select the Ethernet icon.

Setting the IP address

The way your network administrator has configured MacTCP determines the information you enter in the IP address portion of the MacTCP Control Panel. You have three possibilities:

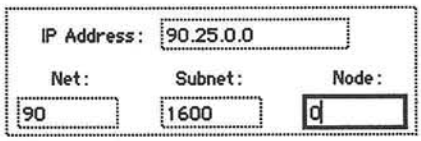
- The administrator has set the full IP address or has configured MacTCP so that the address information is automatically assigned. In this case, you don't have to enter the address and the following sections do not apply to you.
- The administrator has set the net and subnet portions of the address and you must enter the node number.
- You must enter the IP address in decimal notation.

Entering the node number

If your administrator has set up MacTCP so that all you have to do is enter the node portion of the address, refer to the part of your screen shown in Figure 7-11. The Net and Subnet boxes contain numbers but the Node box is set to 0. Double-click the Node box and enter the node number (your administrator should provide you with this information).

The address displayed in the IP Address box changes to include the node portion of the address if you select the More button and then return to the MacTCP Control Panel, or if you close the Control Panel and then bring it back up. You cannot change the address in the IP Address box.

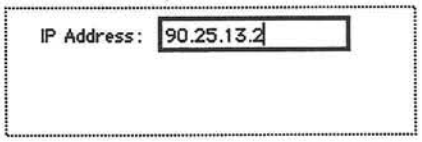
■ **Figure 7-11** Entering the node number



Entering the IP address

If your administrator has set up MacTCP so that you must enter the IP address in dotted decimal notation, an IP Address box is displayed as shown in Figure 7-12. Click in the box and enter the address (your administrator should provide you with this information).

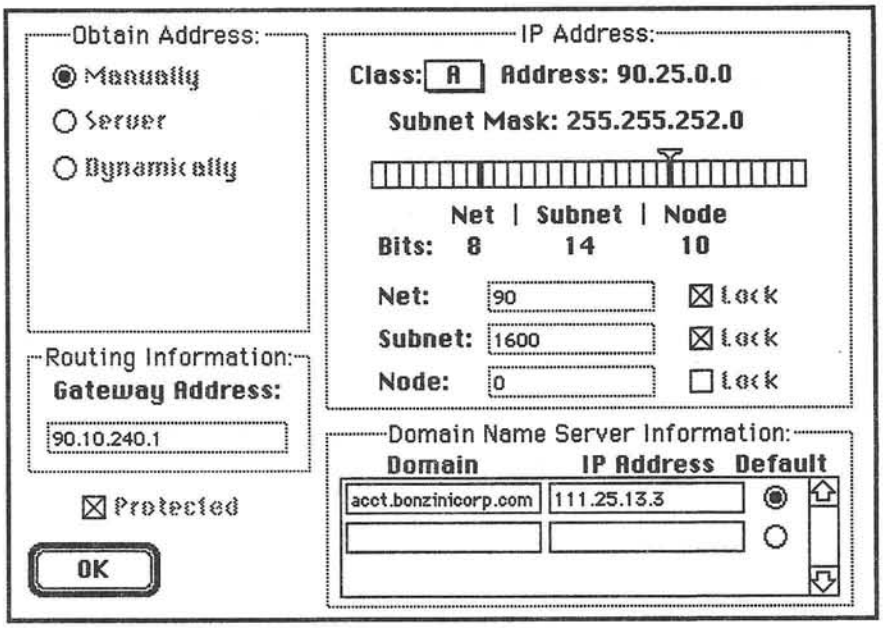
■ **Figure 7-12** Entering the IP address



Viewing the Administrator dialog box

Your system has been set up so that you cannot change the information set in the Administrator dialog box. However, if you want to view this panel, click the More button. The Administrator dialog box appears with all the fields grayed out as shown in Figure 7-13.

■ **Figure 7-13** The Administrator dialog box

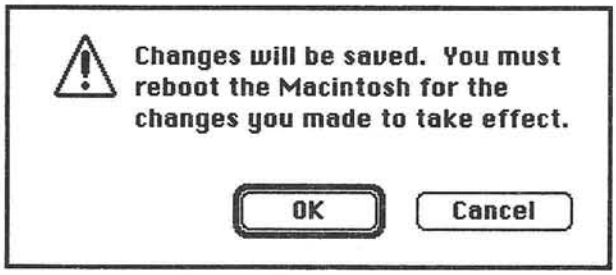


Click the OK button. If you made changes to the MacTCP Control Panel, the alert message shown in Figure 7-14 is displayed. This message alerts you that you must restart your Macintosh computer in order for the configuration changes to take effect. If you click the OK button on the alert box, the MacTCP Control Panel is displayed. If you click the Cancel button on the alert box, the Administrator dialog box is redisplayed.

Closing the Control Panel

After you enter the appropriate information in the MacTCP Control Panel (and if you didn't already save your changes after you viewed the Administrator dialog box in the previous section), click the Close box to close the Control Panel window. The alert message shown in Figure 7-14 is displayed.

■ **Figure 7-14** The alert box



This message alerts you that you must restart your Macintosh computer in order for the configuration changes to take effect. Click the Cancel button if you don't want the changes saved (for instance, if you realize you entered the wrong node number).

If you click the OK button, the Control Panel closes; if you click the Cancel button, the MacTCP Control Panel is redisplayed.

When you restart, MacTCP is configured on your Macintosh computer.

❖ *Note:* The configuration settings are stored in the MacTCP driver and also in a file called MacTCP Prep in your System Folder. If you install a later version of the MacTCP driver on your system, configuration settings will be read from MacTCP Prep, eliminating the need to use the MacTCP Control Panel to reconfigure the new driver.

PART THREE

Appendixes

APPENDIX A

Name-to-Address Mapping

MacTCP includes a Hosts file that maps machine names to Internet addresses, the same service provided by the domain name system. You can use the Hosts file if there is no domain name server on your network. It is also convenient to place frequently used name-to-address mappings in this file.

To use the Hosts file, you must edit the sample file that is included with the MacTCP software and add text that defines name-to-address mappings. Each host on your network that uses Hosts file services must have the file resident on the disk.

This appendix describes the syntax of the Hosts file and how to edit it.

Hosts file syntax

The Hosts file syntax conforms to the master file syntax specified in Request for Comment (RFC) 1035. Refer to this RFC for more information.

❖ *Note:* $INCLUDE has not been implemented.

The syntax of the Hosts file is as follows:

<name> <type> <data> [;<comment>]

where

name is the name assigned to a host or domain on the internet.

type is A (address), NS (name server), or CNAME (canonical name).

data is determined by the type specified:

- If type = A, the data field contains an internet address.
- If type = NS, the data field contains the name of the domain name server that has authority over the domain specified in the name field.
- If type = CNAME, the data field contains the canonical (or official) name for the name field.

comment allows you to comment the entry. A semicolon is used to start a comment.

Any combination of tabs and spaces can be used as a delimiter between each item in a line. Lines end with the characters return and line feed.

Example entries in a Hosts file are as follows:

```
acct.xco.com        A       128.8.1.1         ; address of host "acct"
xco.com             NS      server.xco.com    ; name server for domain xco.com
fred.xco.com        CNAME   bonzini.xco.com   ; canonical name for alia fred.xco.com
```

Editing the Hosts file

MacTCP software includes a sample Hosts file and the TeachText editor. To create Hosts files for network users, modify the sample file using TeachText, adding text that defines name-to-address mappings for your particular network configuration.

To edit the Hosts file, perform the following steps:

1. Select the sample Hosts file icon entitled Hosts, included with your MacTCP software, and open it by choosing Open from the File menu or by double-clicking the icon.

❖ *Note:* You can edit the Hosts file with any word processor as long as you save it as Text Only without formatting commands.

2. Add names and internet addresses in the syntax specified in the previous section. TeachText allows you to use all the usual editing functions.

△ **Important** TCP/IP protocols do not allow the use of International and Graphics characters. Use of these characters is allowed by TeachText; however, using them can cause unpredictable results. △

A P P E N D I X B

MacTCP on the Macintosh 512K Enhanced Computer

THIS APPENDIX DESCRIBES HOW TO CREATE A MASTER DISK FOR THE MACINTOSH 512K enhanced computer. The procedure is different from that described in Chapter 5 because of limited memory space and the fact that System file version 3.4 does not support an extendible Control Panel. You must preconfigure MacTCP on a Macintosh Plus, Macintosh SE, or Macintosh II computer and then install it on the Macintosh 512K enhanced computer.

This appendix also shows the Macintosh 512K enhanced Control Panel. The network user must bring it up and make sure that one of the buttons on the panel is selected. Distribute the section "The Macintosh 512K Enhanced Control Panel" in this appendix to any of your network users who are working on a Macintosh 512K enhanced computer.

Creating a master disk

To create a master disk for the Macintosh 512K enhanced computer, you must use the Installer with the MacTCP 512Ke Installer Script that is supplied with the MacTCP driver.

On a Macintosh Plus, Macintosh SE, or Macintosh II computer, do steps 1 through 10 in the section "Creating a Master Disk" in Chapter 5. Configure MacTCP so that the address is automatically assigned (see the section "Having the Address Set Automatically" in Chapter 6). Then perform the following steps:

1. **Install System file version 3.4 on a blank disk.**

   If your system has only one disk drive, eject this disk by choosing Eject from the File menu.

2. **Insert the duplicate master disk into your disk drive.**

   This is the master disk created in step 2 of the section "Creating a Master Disk" in Chapter 5.

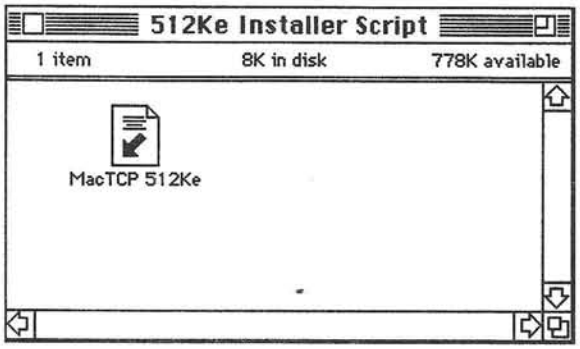
3. **Open the disk icon.**

   Select the disk icon and choose Open from the File menu, or double-click the icon.

4. **Open the Installer Script Folder.**

   Select the Installer Script Folder and choose Open from the File menu, or double-click the folder. The folder contains the icon displayed in Figure B-1.

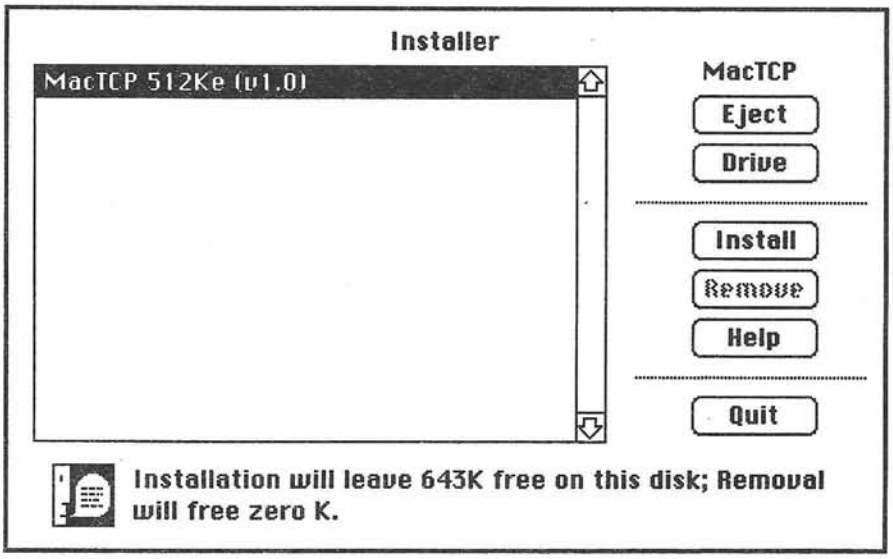
■ **Figure B-1** Icon in the Installer Script Folder



5. To run the Installer, double-click the MacTCP 512Ke icon.

The Installer window shown in Figure B-2 is displayed.

■ **Figure B-2** Installer window



6. **Select the disk on which MacTCP is to be installed.**

   The Installer installs MacTCP on the disk named in the upper-right corner of the dialog box. Click the Drive button until you see the name of the disk you created in step 1.

7. **Click MacTCP 512Ke to select it.**

8. **Click the Install button.**

   The Installer script will copy the driver from your system to the disk specified in step 6.

9. **When the installation is complete, click the Quit button.**

10. **Put a copy of your network application in the Application Folder.**

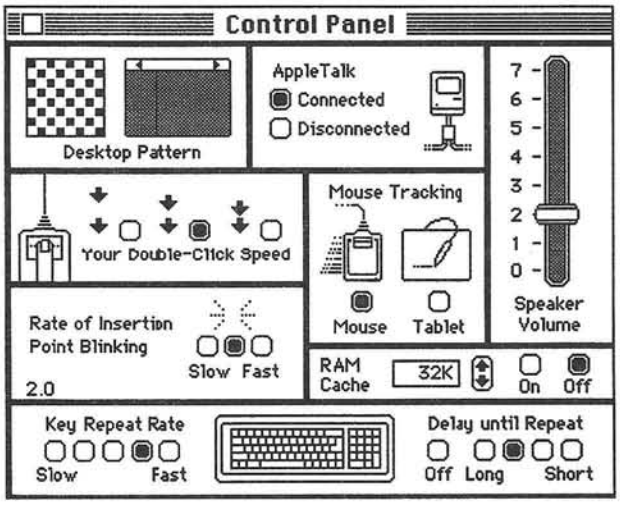
11. **Distribute the disk to the network user along with the appropriate installation instructions.**

The Macintosh 512K enhanced Control Panel

Install MacTCP using the instructions in the section "Installing MacTCP on a Floppy Disk-Based System" in Chapter 5. Then choose Control Panel from the Apple menu, and the Control Panel shown in Figure B-3 is displayed.

In the AppleTalk box, make sure the Connected button is selected as shown in Figure B-3.

■ **Figure B-3** Control Panel on the Macintosh 512K enhanced computer



A P P E N D I X C

Getting More Information

THIS APPENDIX PROVIDES AN OVERVIEW OF AVAILABLE TCP/IP documentation in case you need more information than has been provided by the scope of this guide.

Books

The following books provide information on TCP/IP:

- *An Introduction to TCP/IP*, by John Davidson, Springer-Verlag.
- *Handbook of Computer-Communication Standards, Volume 3: Department of Defense (DOD) Protocol Standards*, by William Stallings, Macmillan Publishing Company.
- *Internetworking With TCP/IP—Principles, Protocols, and Architecture*, by Douglas Comer, Prentice-Hall.

Request for Comments (RFC)

Request for Comments (RFC) is a series of technical notes used by the Internet community that contain reports of work, proposals, and protocol specifications. Important RFCs include 791 (IP), 768 (UDP), 793 (TCP), 826 (ARP), 903 (RARP), 792 (ICMP), and 951 (BootP). RFCs 999, 1000, and 1012 provide a guide to RFCs. RFC 1011 lists the official protocols.

RFCs are available from the DDN Network Information Center (NIC) at SRI International. That address is

DDN Network Information Center
SRI International, Room EJ 291
333 Ravenswood Avenue
Menlo Park, CA 94025

You can order hard copies of RFCs by calling 800-235-3155 or 415-859-3695. To obtain an RFC electronically over the Internet, use anonymous FTP from the <RFC> directory on host SRI-NIC.ARPA. If you are outside the Internet and don't have access to FTP, you can receive RFCs by sending electronic mail to SERVICE@SRI-NIC.ARPA. Put the RFC number in the "Subject:" field.

The DDN NIC publishes the *DDN Protocol Handbook*, which contains most important RFCs related to TCP/IP.

Military standards

The Internet Protocol (IP), Transmission Control Protocol (TCP), File Transfer Protocol (FTP), Telnet, and Mail are described in the five Military Standards (MIL-STD) about TCP/IP protocols. These standards contain errors; when there are discrepancies, RFCs always take precedence over MIL-STDs.

You can obtain MIL-STDs from

Naval Publications and Forms Center, Code 3015
5801 Tabor Avenue
Philadelphia, PA 19120

Glossary

Address Resolution Protocol (ARP): A protocol in the TCP/IP protocol suite that maps internet addresses to physical hardware addresses.

AppleTalk network system: The system of network software and hardware used in various implementations of Apple's communications network.

ARPANET: A wide area network that served as a basis for networking research and provided a central backbone during development of the Internet. See also **Defense Advanced Research Projects Agency Internet.**

Bootstrap Protocol: A protocol in the TCP/IP protocol suite that allows a host to find its internet address and other useful configuration information.

broadcast: A network transmission technique in which data is sent to all attached hosts.

datagram: A packet of data passed across an internet. IP attaches a header to data received from TCP to make an IP datagram. See also **segment.**

Datagram Delivery Protocol-Internet Protocol (DDP-IP) gateway: A gateway that connects AppleTalk networks to TCP/IP networks. This gateway takes a TCP/IP packet that is encapsulated in AppleTalk's Datagram Delivery Protocol (DDP) and converts it to Ethernet format.

Defense Advanced Research Projects Agency (DARPA) Internet: The collection of networks and gateways, including the ARPANET, MILNET, and NSFnet that use TCP/IP protocols to communicate. Also called the Internet. See also **ARPANET, MILNET, NSFnet.**

domain name resolver: Software used by the domain name system to request information from domain name servers.

domain name server: Software used by the domain name system to map domain names to internet addresses.

domain name system: An internet service for mapping a name to an internet address.

dotted decimal notation: The representation for a 32-bit internet address. Each address is written as four decimal integers separated by periods. See also **internet address.**

Ethernet: A high-speed local area network that consists of a cable technology and a series of communication protocols. The hardware (cable) provides the physical link to connect systems together.

EtherTalk: A high-speed AppleTalk network system that uses the cables of an Ethernet network.

File Transfer Protocol (FTP): A protocol in the TCP/IP protocol suite for transferring files across an internet.

fragmentation: The process of breaking an IP datagram into smaller pieces so that it can be transferred over a network that has a small maximum packet size.

frame: A group of bits forming a logical transmission unit that is sent between data-link layer entities.

gateway: A computer that connects two or more networks and routes packets from one to the other.

hardware address: An address used by a physical network. In an Ethernet network, each machine is assigned a 48-bit hardware address.

International Standards Organization (ISO): An international body that specifies network protocol standards. The ISO developed the Open Systems Interconnection (OSI) model. See also **Open Systems Interconnection model.**

internet: An interconnected group of networks. When written as Internet, refers to the DARPA Internet.

internet address: A 32-bit address assigned to every host that wants to use TCP/IP to communicate across an internet. The address consists of a network and host field.

Internet Control Message Protocol (ICMP): A protocol in the TCP/IP protocol suite that hosts and gateways on an internet use to inform each other of error conditions.

Internet Protocol (IP): One of the fundamental protocols in the TCP/IP protocol suite, IP is responsible for sending data across multiple networks.

local area network (LAN): A system that connects computers together within restricted geographical areas.

LocalTalk: A system of cables, cable extenders, and connector boxes that link computers and peripheral devices in an AppleTalk network system.

MILNET (MILitary NETwork): A wide area network on the DARPA Internet that provides network service to military installations. See also **Defense Advanced Research Projects Agency Internet.**

NSFnet (National Science Foundation NETwork): A wide area network on the DARPA Internet that provides network service to the scientific community. See also **Defense Advanced Research Projects Agency Internet.**

octet: 8 bits. IP addresses consist of 4 octets. The term *octet* is used instead of *byte* because not all hosts use 8-bit bytes.

Open Systems Interconnection (OSI) model: A 7-layer reference model developed by the ISO that is used to describe network systems architecture.

packet: A unit of data transmitted on a network.

process: Programs that communicate; for instance, a file transfer process on one host talks to a file transfer process on another host. A host may be running several processes.

protocol: A set of rules that computers must follow to exchange data over a network.

Request for Comments (RFC): A series of technical notes used by the Internet community that contain reports of work, proposals, and protocol specifications.

Reverse Address Resolution Protocol (RARP): A protocol in the TCP/IP protocol suite that maps physical hardware addresses to internet addresses.

route: The path that network traffic takes to get from source to destination.

routed software: The implementation of the RIP protocols for 4.3BSD UNIX.

Routing Information Protocol (RIP): A protocol in the TCP/IP protocol suite that allows gateways and hosts to exchange network routing information.

routing table: A table maintained in each internet gateway that contains, for each possible destination network, the next gateway to which data should be sent.

segment: A unit of data sent from TCP on one host to TCP on another. Each segment travels across the internet in an IP datagram. See also **datagram.**

server: A program running on a host that offers a service to other hosts on the network. For instance, a file server provides access to its files. Computers that run server programs are often referred to as servers. See also **domain name server.**

Simple Mail Transfer Protocol (SMTP): A protocol in the TCP/IP protocol suite that allows electronic mail to be transferred across an internet.

subnetwork address: A technique that allows multiple physical networks to share the same internet network address.

subnetwork mask: Software used to select bits from an internet address for subnet addressing.

Telnet: A protocol in the TCP/IP protocol suite that allows a terminal on one host to appear as if it were directly connected to a remote host on an internet.

Transmission Control Protocol (TCP): One of the fundamental protocols in the TCP/IP protocol suite, TCP provides for the reliable exchange of data between two processes.

User Datagram Protocol (UDP): A protocol in the TCP/IP protocol suite that provides unreliable data transmission between two processes.

wide area network: A system for connecting computers over a large geographical area.

Index

A

address. *See also* internet address; subnetwork addressing
  class, setting of 48-49
Address Resolution Protocol (ARP) 9, 19-20
Admin TCP file 30
Admin TCP icon 44
Administrator dialog box 38, 40-42
  bringing up the 46
  user viewing of 69
alert message 57
AppleTalk
  network configurations with 61-67
  running MacTCP while using 5
Application Folder 30
ARPANET 7, 14
assembly language interfaces 4

B

Bootstrap Protocol (BootP) 9, 21

C

Class A addresses 14-15
  settings for 49
Class B addresses 14-15
  settings for 49
Class C addresses 14-15
  settings for 49
Comer, Douglas *(Internetworking With TCP/IP—Principles, Protocols, and Architecture)* 84
comment, in Hosts file syntax 74
configurations
  full configuration power to user 56
  MacTCP configuration 37-57
  network configurations for users 61-67
  protection of 56
  roadmap to scenarios for 43
  scenarios 42-43
  user configuration 59-70
Control Panel. *See* MacTCP Control Panel

D

DARPA. *See* Defense Advanced Resarch Projects Agency
data, in Hosts file syntax 74
Datagram Delivery Protocol-Internet Protocol (DDP-IP) gateway xiii, 6, 52, 61, 87
datagrams, IP protocol and 10
Davidson, John *(Introduction to TCP/IP, An)* 84
DDN Network Information Center (NIC) 84
DDN Protocol Handbook 84
decimal notation, entering IP address in 45
default gateway address 54
Defense Advanced Research Projects Agency (DARPA)
  Internet 4
  TCP/IP and 7
domain name resolvers 4, 19
domain name servers 19
  information box 41
  setting information for 55
domain name system 18-19

E

error checking, UDP and 11
Ethernet 5
  configuration for 5
  interface card required, xiii, 5
  local area network (LAN) 14
  network configuration with 63-67
EtherTalk, network configuration with 63-67

F

File Transfer Protocol (FTP) 9, 12
  Military Standards and 85
512Ke Installer Script Folder 30
floppy disk-based system, installing MacTCP on 34
4.3BSD UNIX system 25
full configuration power to user 56

G

gateways 24
  setting gateway address 54
graphics characters 75

H

Handbook of Computer-Communication Standards, Volume 3: Department of Defense Protocol Standards (Stallings) 84
hard disk, installing MacTCP on 34-36
hardware requirements xiii
Hosts file
  editing of 75
  File Folder 30
  syntax 74
Hosts File Folder 30

I

installation of MacTCP 29-36
  on Macintosh 512K enhanced computer 78-80
  user installation instructions 33-36
integer format for setting IP address 50-52
international characters 75
International Standards Organization (ISO) 9
internet 8. *See also* internet address
  example of 8
internet address 14-15
  Administrator dialog box and 41
  automatic setting of 52-53
  Bootstrap Protocol and 21
  decimal notation, entering IP address in 45
  dynamically setting the 53-54
  integer format for setting the 50-52
  IP address information 38-40
  manual setting of 48-52
  setting the 47-54
  user setting 67-69
  name-to-address mappings 73-75
  RARP and 19
  subnetwork addressing 15-17
Internet Control Message Protocol (ICMP) 9, 25-26
Internet Protocol (IP) 9, 10-11. *See also* internet address
  ICMP and 26
  Military Standards and 85
Internet Request for Comments (RFC) 4, 84
internetwork 8
Internetworking With TCP/IP—Principles, Protocols, and Architecture (Comer) 84
Introduction to TCP/IP, An (Davidson) 84
IP address. *See* internet address
IP address box 41

L

labels 18
LaserWriter printers, running MacTCP while using 5
link level information setting 61-67
local area networks (LAN) 14
  subnetwork addressing and 15-17
  subnetwork masks and 17
LocalTalk xiii, 5
  configuration for 5
  network configuration with 61-62

M

Macintosh 512K enchanced computer xiii, 5
  bringing up MacTCP Control Panel with 44
  Control Panel for 38, 80-81
  MacTCP on 77-81
Macintosh II xiii, 5
  network configurations with 64-65
Macintosh IIx xiii, 5
Macintosh Plus xiii, 5
Macintosh SE xiii, 5
Macintosh SE/30 xiii, 5
Macintosh System file version 6.0.3 xiv
MacTCP Control Panel 4, 38-40. *See also* Transmission Control Protocol (TCP)
  bringing up the 44-45, 60
  closing the Control Panel 70
  for Macintosh 512K enhanced computer 80-81
MacTCP Folder 30
Mail
  Military Standards and 85
  Simple Mail Transfer Protocol for (SMTP) 12
manually setting IP address 48-52
master disk
  creating a 30-33
  on Macintosh 512K enhanced computer 78-80
Military Network (MILNET) 7
Military Standards (MIL-STD) 4, 85

N

name in Hosts file syntax 74
name-to-address mappings 73-75
National Science Foundation Network (NSFnet) 7
Naval Publications and Forms Center 85
Net box 51
net number, entering the 51
network configuration 5
  for users 61-67
network field 14
Network Information Center (NIC) 14
  domain name system and 18
  RFCs available from 84
network zone 62
node number
  dynamically setting the 53-54
  entering the 52
  user entering 68
Node Range boxes 54

O

Obtain Address box 47
Open Systems Interconnection (OSI) Reference Model 9
  comparison of TCP/IP and 10

P

protected button 42, 56

R

Request for Comments (RFC) 4, 84
Reverse Address Resolution Protocol (RARP) 9, 20-21
routed software 25
routing 23-26
  gateways 24
  information box 41
  protocols 25-26
Routing Information Protocol (RIP) 9, 25
  determining active gateways with 54
routing tables 25

S

Server button 52
Simple Mail Transfer Protocol (SMTP) 9, 12
software requirements xiii
SRI International 14, 30
Stallings, William (*Handbook of Computer-Communication Standards, Volume 3: Department of Defense Protocol Standards)* 84
subnames 18
subnetting 15-17
subnetwork addressing 15-17
  example of 16
subnetwork masks 17
  setting the 50
subnetwork number, entering the 52
subnetworks 15

T

TCP-IP. *See* Transmission Control Protocol/Internet Protocol (TCP/IP)
TeachText 75
Telnet 9, 12
  Military Standards and 85
  TCP and 11
Transmission Control Protocol (TCP) 9, 11
  Military Standards and 85
Transmission Control Protocol/Internet Protocol (TCP/IP) 7-12
  configuration using 65-67
  conventions for 13-21
  family of protocols 9-12
  list of protocols 4
type in Hosts file syntax 74

U

user configuration 59-70
User Datagram Protocol (UDP) 9, 11
user installation instructions 33-36

THE APPLE PUBLISHING SYSTEM

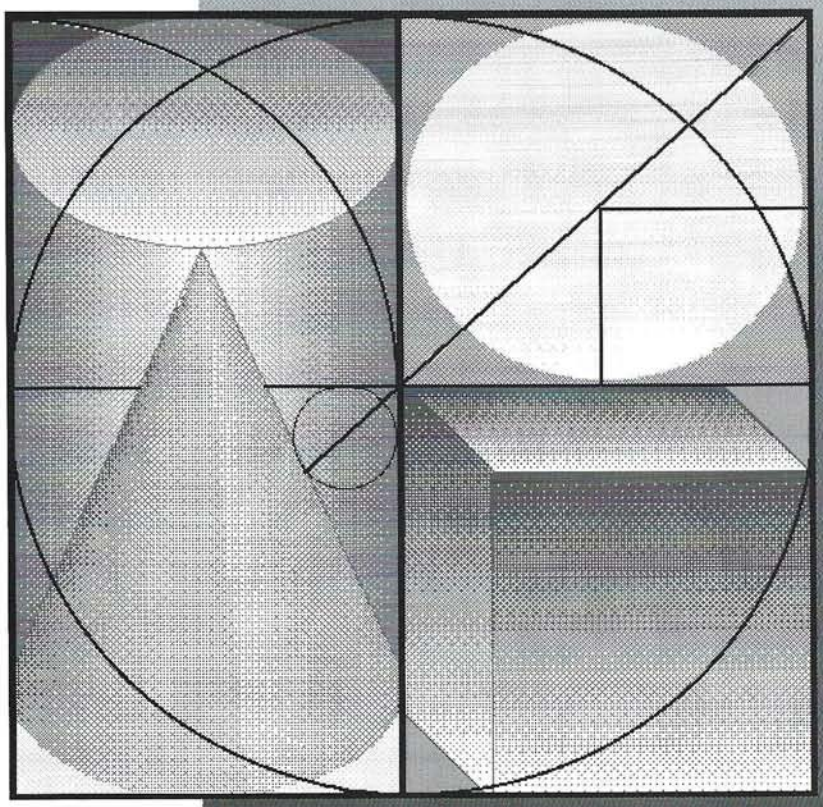
This Apple® manual was written, edited, and composed on a desktop publishing system using Apple Macintosh® computers and Microsoft® Word. Proof pages were created on the Apple LaserWriter® printers; final pages were printed on a Varityper® VT600™. Line art was created using Adobe Illustrator™. POSTSCRIPT®, the LaserWriter page-description language, was developed by Adobe Systems Incorporated.

Text type and display type are Apple's corporate font, a condensed version of Garamond. Bullets are ITC Zapf Dingbats®. Some elements, such as program listings, are set in Apple Courier, a fixed-width font.

MacTCP™ v.1.0, Programmer's Guide

with Interim Manual

APDA # M0260LL/A



Apple Computer, Inc.
20525 Mariani Avenue
Cupertino, California 95014
(408) 996-1010
TLX 171-576

To reorder products, please call:
Apple Programmers and Developers Association
1-800-282-APDA



Apple® MacTCP™ Programmer's Guide
Interim Release

APPLE COMPUTER, INC.

This manual and the software described in it are copyrighted, with all rights reserved. Under the copyright laws, this manual or the software may not be copied, in whole or in part, without written consent of Apple, except in the normal use of the software or to make a backup copy of the software. The same proprietary and copyright notices must be affixed to any permitted copies as were affixed to the original. This exception does not allow copies to be made for others, whether or not sold, but all of the material purchased (with all backup copies) may be sold, given, or loaned to another person. Under the law, copying includes translating into another language or format.

You may use the software on any computer owned by you, but extra copies cannot be made for this purpose.

© Apple Computer, Inc., 1989
20525 Mariani Avenue
Cupertino, CA 95014
(408) 996-1010
Apple, the Apple logo, AppleTalk, and Macintosh are registered trademarks of Apple Computer, Inc.

LocalTalk and MacTCP are trademarks of Apple Computer, Inc.

ITC Garamond is a registered trademark of International Typeface Corporation.

Microsoft is a registered trademark of Microsoft Corporation.

PostScript is a registered trademark of Adobe Systems Incorporated.

Simultaneously published in the United States and Canada.

Contents

Figures and tables / vii

Preface / ix

What's in this guide / ix
Who should read this guide / ix
Document conventions / ix
Related documents / x

1 Introduction / 1

Architectural overview / 2
- Application, presentation, and session layers / 2
- Transport layer / 3
  - User Datagram Protocol (UDP) / 3
  - Transmission Control Protocol (TCP) / 3
- Network layer / 3
- Link access layer / 4
- Physical layer / 4

2 The MacTCP Driver / 5

The PBOpen call / 6
The PBControl call / 6
The PBClose call / 7

3 User Datagram Protocol / 9

Data structures / 10
- Write Data Structures / 10
- Receive buffer area / 11

Using UDP / 11

UDP routines / 12
UDPCreate / 13
UDPRead / 14
UDPBfrReturn / 15
UDPWrite / 16
UDPRelease / 17
UDPMTU / 18
UDP asynchronous notification routine / 19
C parameter block definitions / 21

4 Transmission Control Protocol / 23

Data structures / 24
Read and Write Data Structures / 24
Receive buffer area / 25
Using TCP / 25
Streams and connections / 25
Asynchronous notification routine / 25
Connection opening / 26
Receiving data / 26
Sending data / 26
Time-outs / 26
Pushed data / 27
Urgent mode / 27
Connection closing / 28
Network management information / 28
Formatting MacTCP commands / 28
TCP routines / 29
TCPCreate / 30
TCPPassiveOpen / 31
TCPActiveOpen / 34
TCPSend / 35
TCPNoCopyRcv / 36
TCPBfrReturn / 38
TCPRcv / 39
TCPClose / 41
TCPAbort / 42
TCPStatus / 43

TCPRelease / 45
TCPGlobalInfo / 46
TCP asynchronous notification routine / 48
C parameter block definitions / 52

5 Name-to-Address Resolution / 59

The AddressXlation.h header file / 60
The OpenResolver call / 60
The StrToAddr call / 60
The AddrToStr call / 61
The EnumCache call / 61
The CloseResolver call / 62
The AddrToName call / 62
Binding the DNR to the application / 63

6 Miscellaneous Interfaces / 65

MacTCPCommontypes / 66
Result codes / 66
Miscellaneous types / 68
Internet Control Message Protocol report structures / 69
GetMyIPAddr / 70

Appendix Constants / 71

Command codes / 72
UDP asynchronous event codes / 72
TCP asynchronous event codes / 72
Reasons for TCP termination / 73

Figures and Tables

CHAPTER 1 **Introduction / 1**

Figure 1-1 MacTCP protocols and OSI network layers / 2

CHAPTER 3 **User Datagram Protocol / 9**

Figure 3-1 UDP Write Data Structure / 10
Table 3-1 UDP routines / 12

CHAPTER 4 **Transmission Control Protocol / 23**

Figure 4-1 TCP Read and Write Data Structures / 24
Table 4-1 TCP routines / 29

Preface

THIS GUIDE DESCRIBES how to create application programs for the MacTCP™ driver. MacTCP, Apple® Computer's implementation of the protocol suite known as Transmission Control Protocol/Internet Protocol (TCP/IP), increases the Macintosh® computer's ability to interoperate in a heterogeneous computer environment.

What's in this guide

This guide is divided into six chapters and one appendix that contain the following information:

- Chapter 1, "Introduction," provides an overview of the MacTCP architecture.
- Chapter 2, "Making MacTCP Calls," describes the PBOpen, PBControl, and PBClose calls.
- Chapter 3, "User Data Protocol," describes UDP routines.
- Chapter 4, "Transmission Control Protocol," describes TCP routines.
- Chapter 5, "Name-to-Address Resolution," describes how textual names are resolved to IP addresses.
- Chapter 6, "Miscellaneous Interfaces," describes the types that are found throughout the programmatic interfaces supplied with MacTCP.
- The Appendix, "Constants," presents command codes, UDP asynchronous event codes, TCP asynchronous event codes, and reasons for TCP termination.

Who should read this guide

This guide is designed both for Macintosh programmers who are not familiar with TCP/IP and for TCP/IP programmers who are not familiar with the Macintosh programming environment.

Document conventions

This document reflects the Macintosh Programmer's Workshop (MPW) conventions for types and sizes of variables and fields.

Related documents

You might find the following reference materials useful.

- The *MacTCP Administrator's Guide.*
- *Inside Macintosh*, Volumes I-V.
- Douglas Comer, *Internetworking with TCP/IP* (Prentice-Hall, 1988).
- MacTCP implements protocols that conform to the following Request For Comments (RFC) and Military Standards (MIL-STD):
  - RFC 791, 894; MIL-STD 1777 (Internet Protocol)
  - RFC 768 (User Datagram Protocol)
  - RFC 793; MIL-STD 1778 (Transmission Control Protocol)
  - RFC 826 (Address Resolution Protocol)
  - RFC 903 (Reverse Address Resolution Protocol)
  - RFC 792 (Internet Control Message Protocol)
  - RFC 951, 1048 (Bootstrap Protocol)
  - IDEA004 (Routing Information Protocol)
  - RFC 1034, 1035 (Domain Name Resolver)
  - RFC 950 (Internet Subnetting)
  - RFC 1010 (Internet Assigned Numbers)

Chapter 1 **Introduction**

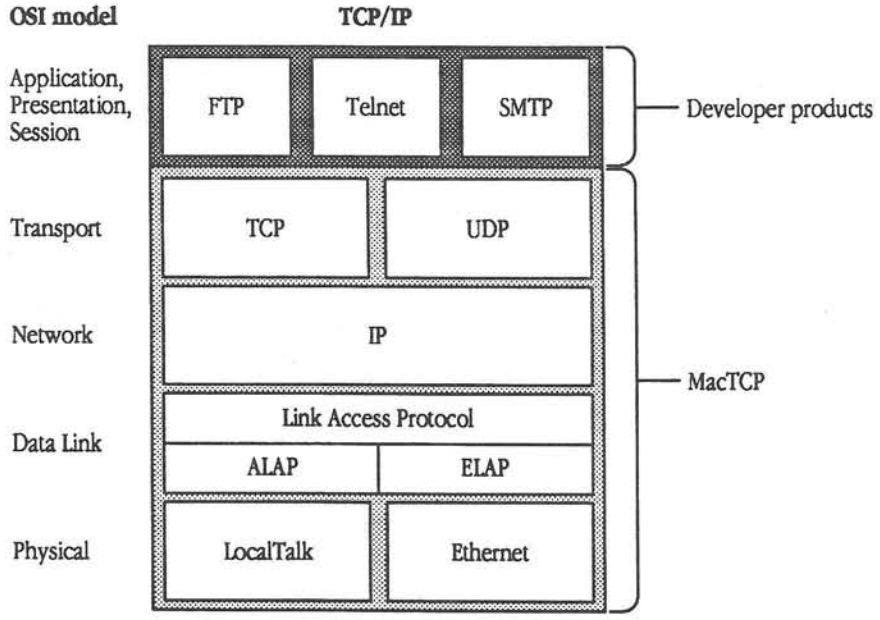
MACTCP™ IS A SOFTWARE DRIVER for the Macintosh® Operating System that allows developers to create Macintosh applications for network environments that use the Transmission Control Protocol/Internet Protocol (TCP/IP). TCP/IP is a widely used industry standard for connecting multivendor computers. The TCP/IP protocol layers are fully compatible across all implementations on different hardware platforms, allowing different vendors' computers that run TCP/IP to interoperate and share data and services.

This chapter provides an overview of the MacTCP architecture. ■

Architectural overview

MacTCP protocols partially conform to the International Standards Organization (ISO) Open Systems Interconnection (OSI) layers of networking functionality. Figure 1-1 shows a comparison of the OSI and MacTCP communications architecture.

■ **Figure 1-1** MacTCP protocols and OSI network layers



The TCP/IP protocols shown in Figure 1-1 are described in the following sections.

Application, presentation, and session layers

The services of TCP, the User Datagram Protocol (UDP), and IP are augmented by application services in higher-level protocols such as the File Transfer Protocol (FTP), Telnet, and the Simple Mail Transfer Protocol (SMTP). These application services correspond to the application, presentation, and session layers of the OSI model.

Transport layer

The UDP and TCP provide services at the transport layer of the OSI model.

User Datagram Protocol (UDP)

UDP provides unreliable transmission of data between processes. UDP's transport of data is unreliable because, unlike TCP, it does not provide error checking, it does not acknowledge that data has been successfully received, and it does not order incoming messages. UDP messages can be lost, duplicated, or arrive out of order. Like TCP, UDP messages are encapsulated in IP datagrams for delivery.

The advantage of UDP is that the overhead associated with establishing and maintaining an error-free TCP session is avoided. Upper-layer protocols that don't require reliability use UDP to transmit data. For instance, the domain name system uses UDP because reliability is not critical; if there is no response to a domain name query, the resolver simply retransmits.

Transmission Control Protocol (TCP)

TCP provides reliable transmission of data between processes. It ensures that data is delivered error-free, without loss or duplication, and in sequence.

Upper-layer protocols such as Telnet pass data to TCP for delivery to peer processes. TCP encapsulates the data into segments and passes the segments to IP, which puts the segments into datagrams and passes them across the internet. TCP at the receiving end checks for errors, acknowledges error-free segments, and reassembles the segments for delivery to upper-layer protocols. If a segment is lost or damaged, it will not be acknowledged, and the sending process will retransmit.

TCP has a flow control mechanism so that computers of different speeds and sizes can communicate. When TCP at the receiving end sends an acknowledgment, it also advertises how much data it is prepared to accept on the next transmission.

Network layer

The Internet Protocol (IP) provides services at the network layer of the OSI model. IP is responsible for sending data across multiple networks. IP accepts segments of data from TCP or UDP, places the data in packets called datagrams, and determines the correct path for the datagram to take. The datagrams are sent across the internet, through as many gateways as needed, until they reach the destination host.

IP provides an addressing mechanism that allows routing between networks. The header of an IP datagram contains source and destination internet addresses so that any host in a network can route a packet to a destination, either directly or through a gateway.

IP has the ability to fragment a datagram as it is transmitted across a network. Since IP can be used with many different physical network implementations that specify different sizes for physical data frames, datagrams can be fragmented to fit into a small data frame. Fragments are reassembled as they arrive at the destination.

IP is often referred to as an unreliable delivery system because it makes a best-effort attempt to deliver all datagrams, but delivery is not guaranteed (TCP guarantees delivery). It is also called a connectionless delivery system because it routes each datagram separately. When IP receives a sequence of datagrams from TCP or UDP, it routes each datagram in the sequence individually and each datagram may travel over a different path to the destination.

Link access layer

The AppleTalk® Link Access Protocol (ALAP) and Ethernet Link Access Protocol (ELAP) provide services at the link access layer of the OSI model. ALAP and ELAP provide best-effort delivery of information between devices. They provide the basic service of packet transmission between devices connected to a single physical network.

Physical layer

MacTCP operates networks in which the physical layer uses the LocalTalk™ cabling system or Ethernet networking standard.

Chapter 2 **The MacTCP Driver**

MACTCP IS A MACINTOSH Operating System driver that implements the Internet Protocol (IP), User Datagram Protocol (UDP), and Transmission Control Protocol (TCP). A part of the Macintosh Operating System called the Device Manager handles communication between applications and the MacTCP driver.

The MacTCP driver resides in the Macintosh System Folder. At startup, the MacTCP driver registers itself with the Device Manager as `.IPP`.

The MacTCP driver supports the following low-level Device Manager calls:

- PBOpen
- PBControl
- PBClose

The PBOpen call

Before the application can exchange information with the MacTCP driver, the driver must be opened using the Device Manager PBOpen call, which opens the driver and returns its reference number. All subsequent calls to the driver use this reference number.

On machines with 1 megabyte (MB) of memory or more, the MacTCP driver uses the INIT-31 mechanism to automatically install itself in the device table and load itself into the system heap each time the machine is restarted. At load time, all internal buffers needed by MacTCP are also allocated on the system heap. These buffers are used (among other things) to hold incoming fragments awaiting reassembly and outgoing fragments following fragmentation. The amount of buffer space allocated is based on the amount of memory in the machine.

On machines with less than 1 MB of memory, MacTCP will not load itself at system startup; rather, the driver must be installed in the System file using the provided Installer script. When the PBOpen call is made, the driver is loaded into the application heap on a Macintosh 512K enhanced computer by the Device Manager.

The MacTCP driver performs its initialization sequence at PBOpen time. The initialization sequence includes initializing the local network handler, setting its local address and subnet mask, verifying that this address is reasonable and unused, and starting up the internal TCP/IP/UDP protocol machinery. PBOpen returns with an error code if any step in this process fails. In almost all cases, if PBOpen fails, the driver must be reconfigured and the system then restarted to clear the problem. Very few circumstances exist where transitory problems cause PBOpen to fail.

Because of the complexity of initializing MacTCP, PBOpen can return errors from many parts of the system including the Resource Manager, Device Manager, File Manager, Slot Manager, and AppleTalk.

The PBControl call

The Device Manager PBControl call sends control information in a parameter block to the driver. The type of information sent is specified by csCode in the parameter block. The MacTCP driver decides what to do based on the procedure number specified by the csCode (the Appendix lists procedure numbers).

All UDP and TCP routines documented in this guide are implemented as Device Manager Control calls. All MacTCP routines support both synchronous and asynchronous modes.

If a synchronous call is made, the application can't continue until the call is completed. When control is returned to the application, the result code in the parameter block is set indicating the success or failure of the call.

If the call is asynchronous, the I/O request is placed on the driver I/O queue, and control is returned to the calling program almost immediately. While requests are taken from the driver I/O queue one at a time and processed, the application is free to perform other tasks. The application has two ways of knowing when the call has been completed:

- By polling the ioResult field in the parameter block. When this value changes from commandInProgress to some other value, the call has been completed.
- By specifying an I/O completion routine to be called when the requested operation is complete.

The ioCompletion field in the parameter block contains a pointer to a completion routine to be executed at the end of an asynchronous call. The ioCompletion field should be NIL for asynchronous calls with no completion routine, and is automatically set to NIL for all synchronous calls.

Each routine description includes a parameter block format. The number next to each field name indicates the byte offset of the field from the start of the parameter block pointed to by A0. An arrow next to each parameter name indicates whether it's an input, output, or input/output parameter:

Arrow	**Meaning**
→	Parameter is passed to the routine.
←	Parameter is returned by the routine.
↔	Parameter is passed to and returned by the routine.

All Device Manager Control calls return an integer result code of type OSErr in the ioResult field of the I/O parameter block. Each routine description lists all of the applicable result codes generated by MacTCP, along with a short description of what the result code means.

The PBClose call

The application does not need to issue a PBClose call to MacTCP. Because the driver can be shared by a variety of applications, once initialized, it never stops operation until the machine is restarted. PBClose returns an error code of closErr.

Chapter 3 **User Datagram Protocol**

THE USER DATAGRAM PROTOCOL (UDP) provides a low-overhead transaction service to allow upper-layer protocols (ULP) to send datagrams between one another. UDP is datagram oriented with best-effort delivery, but it does not use acknowledgments to make sure messages are delivered to the destination, does not order incoming messages, and does not provide feedback to control the rate at which information flows between machines.

Like the Transmission Control Protocol (TCP), UDP uses port fields to specify the source and destination processes of each transaction. An optional checksum is also used. ■

Data structures

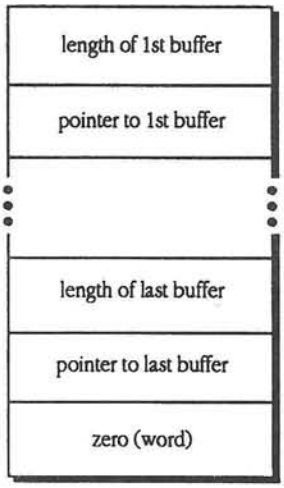
The UDP packet is composed of an 8-byte header, followed by up to 65,507 bytes of data. The UDP header has the following structure:

source port	word
destination port	word
length	word
checksum	word

Write Data Structures

To send a UDP datagram, the user must format a Write Data Structure (WDS), which has the format shown in Figure 3-1. A WDS specifies a list of buffers to write in a single operation.

- **Figure 3-1** UDP Write Data Structure



The simplest WDS describes a single buffer and is 8 bytes in length: a word length, a long pointer, followed by a terminating word of 0. The most complex WDS that can be used to send a UDP datagram describes 6 buffers and is 38 bytes in length.

Receive buffer area

The User Datagram Protocol (UDP) does not allocate memory for storing UDP stream databases or for buffering received datagrams. Instead, the user must pass UDP enough memory for these purposes in the UDPCreate call. This has two advantages:

- The buffer memory can be allocated off the application heap instead of the system heap, which is very limited.
- The user has control over the buffering provided by UDP and can allocate the appropriate amount of memory for the type of application and performance level desired.

This buffer area for incoming datagrams belongs to the MacTCP driver as long as the UDP stream is opened. When UDPRelease is called, this memory is returned to the user. The user can then reuse it or return it to the system.

Using UDP

To send or receive UDP datagrams, the user must first call UDPCreate to create a UDP stream, creating a port listener for the given UDP port and passing a memory block to the UDP driver to use in buffering incoming datagrams. UDPCreate also registers a routine that UDP uses to notify the user of asynchronous events associated with this UDP stream. MacTCP can support 64 open UDP streams simultaneously.

To receive a UDP datagram, call UDPRead. Then, when you finish with the buffer that holds the received datagram, call UDPBfrReturn. To send a UDP datagram, call UDPWrite. When you finish using a UDP port, call UDPRelease to close the UDP stream.

UDP routines

This section describes calls to the UDP driver. The C parameter block definitions are presented at the end of the chapter. Table 3-1 lists each UDP routine and its function.

■ **Table 3-1** UDP routines

Routine	Function
UDPCreate	opens a UDP stream
UDPRead	retrieves a datagram received on a UDP stream
UDPBfrReturn	returns a receive buffer to the UDP driver
UDPWrite	sends a datagram on a UDP stream
UDPRelease	closes a UDP stream
UDPMTU	returns the maximum size of UDP data that can be sent in a datagram without IP fragmentation

UDPCreate

Parameter block

→	26	word	csCode = UDPCreate
←	28	long	stream pointer
→	32	long	pointer to receive buffer area
→	36	long	length of receive buffer area
→	40	long	pointer to asynchronous notification routine (ASR)
↔	44	word	local UDP port
→	46	long	user data pointer

UDPCreate opens a UDP stream. It must be called before any UDP datagrams can be sent or received on a particular UDP port. UDPCreate returns a stream pointer, which must be used in all subsequent UDP calls that operate on this UDP stream.

The receive buffer area is a block of memory that the user must give to the UDP driver. UDP uses this memory to buffer incoming datagrams. This block of memory belongs to UDP while the stream is open; it cannot be modified or relocated until UDPRelease is called. The minimum allowed size of the receive buffer area is 2048 bytes, but it should be at least 2N + 256 bytes in length, where N is the size in bytes of the largest UDP datagram the user expects to receive.

The ASR is called by UDP to notify the user of asynchronous events such as data arrival and Internet Control Message Protocol (ICMP) messages. If the routine is 0, the user is not notified of asynchronous events. See the section "UDP Asynchronous Notification Routine" later in this chapter for more information.

If the local port is 0, UDP assigns an unused local port.

The user data pointer is returned in all ASRs for the created UDP stream.

Result codes

noErr	no error
streamAlreadyOpen	there is already an open stream using this receive buffer area
invalidLength	the receive buffer area is too small
invalidBufPtr	the receive buffer area pointer is 0
duplicateSocket	a stream is already open using this local UDP port
insufficientResources	there are already 64 UDP streams open

UDPRead

Parameter block				
	→	26	word	csCode = UDPRead
	→	28	long	stream pointer
	→	32	word	command time-out value in seconds (0 = infinite)
	←	34	long	remote IP address
	←	38	word	remote UDP port
	←	40	long	pointer to UDP data
	←	44	word	length of UDP data
	→	46	word	reserved; must be set to 0
	→	48	long	user data pointer

UDPRead retrieves a datagram that has been received on the UDP stream defined by the stream pointer. Some number of datagrams are buffered internally within UDP even when no UDPRead commands are outstanding, so it is not necessary to keep a UDPRead command outstanding at all times. The exact number of datagrams that can be buffered within MacTCP depends on the size of the receive buffer area given to MacTCP in the UDPCreate call and the size of datagrams received.

If a UDP datagram arrives on an open UDP stream and no UDPRead commands are outstanding, the user is given a data arrival notification as a hint that a UDPRead command should be issued. See the section "UDP Asynchronous Notification Routine" later in this chapter for more information.

The command time-out period is specified in seconds. If no datagram arrives within the time-out period, the UDPRead command is completed in error. The minimum allowed value for the command time-out is 2 seconds. A 0 command time-out means infinite; the UDPRead command is not completed until a datagram arrives.

The remote IP address and remote UDP port specify the source of the datagram.

It is possible that UDPRead returns successfully, yet the length of UDP data is 0. This happens when a UDP packet arrives that has the passed-in UDP stream's local UDP port value, but contains no data. Since the UDPSend command permits sending 0-length UDP datagrams, the UDPRead command must pass up 0-length datagrams for symmetry.

For every UDPRead command that is completed successfully and returns a nonzero amount of data, the user must call UDPBfrReturn with the same stream pointer and UDP data pointer to return the receive buffer to the UDP driver for reuse.

Result codes		
	noErr	no error
	invalidStreamPtr	the specified UDP stream is not open
	commandTimeout	no data arrived within the specified period
	connectionTerminated	the UDP stream was closed by a UDPRelease command

UDPBfrReturn

Parameter block	→	26	word	csCode = UDPBfrReturn
	→	28	long	stream pointer
	→	40	long	pointer to UDP data
	→	48	long	user data pointer

UDPBfrReturn returns a receive buffer to the UDP driver that had been passed to the user because of a successful UDPRead call that returned a non-zero amount of data.

Result codes	noErr	no error
	invalidStreamPtr	the specified UDP stream is not open
	invalidBufPtr	the UDP receive buffer is not owned by the user

UDPWrite

Parameter block				
	→	26	word	csCode = UDPWrite
	→	28	long	stream pointer
	→	34	long	remote IP address
	→	38	word	remote UDP port
	→	40	long	pointer to WDS
	→	44	byte	checksum flag
	→	46	word	reserved
	→	48	long	user data pointer

UDPWrite sends a datagram on a UDP stream.

The datagram's destination is specified by the remote IP address and remote UDP parameter ports. The buffers described by the WDS must not be modified or relocated until the command is completed. There can be up to 6 buffers described by the WDS. The total length of the UDP data described by the WDS must be between 0 and 8,192 inclusive. If the Checksum flag is not 0, UDP computes and transmits a checksum; otherwise, the checksum is transmitted as 0. The reserved field must be set to 0.

Result codes		
	noErr	no error
	invalidStreamPtr	the specified UDP stream is not open
	invalidWDS	too many buffers in WDS or WDS pointer is NIL
	invalidLength	the UDP data is too long to fit in one datagram
	insufficientResources	too many datagrams are outstanding in the transmit queue

UDPRelease

Parameter block	→	26	word	csCode = UDPRelease
	→	28	long	stream pointer
	←	32	long	pointer to receive buffer area
	←	36	long	length of receive buffer area
	→	46	long	user data pointer

UDPRelease closes a UDP stream. Any outstanding commands on that stream are terminated with an error. The ownership of the receive buffer area used to create the UDP stream passes back to the user.

Result codes	noErr	no error
	invalidStreamPtr	the specified UDP stream is not open

UDPMTU

Parameter block	→	26	word	csCode = UDPMaxMTUSize
	←	32	word	maximum datagram size
	→	34	long	remote address
	→	38	long	user data pointer

UDPMTU returns the maximum size of UDP data that can be sent in a single datagram without fragmentation. This number does not include the IP and UDP headers. The value is relative to the destination address. If the address is on the local network, the network maximum MTU size is returned; otherwise, the value of 576 bytes is returned.

Result codes	noErr	no error

UDP asynchronous notification routine

The asynchronous notification routine (ASR), which was registered with UDP in the UDPCreate call, is called by UDP to notify the user of asynchronous events relevant to a particular UDP stream.

Since this routine is called from the interrupt level, the user must not allocate or return memory to the system. Also, the user is not allowed to make further synchronous MacTCP calls from an ASR. The values of all registers must be preserved except registers A0–A2 and D0–D2.

The C description of the ASR is as follows:

```
typedef pascal void (*UDPNotifyProc) (StreamPtr udpStream,
unsigned short eventCode, Ptr userDataPtr, struct ICMPReport *icmpMsg);
```

NotifyProc is passed to the UDPCreate call for use on notification of data arrival and ICMP message reception. If this method of notification is not desired, no procedure should be passed to the UDPCreate call. Asynchronous notification will be used with the UDPRead command only; all other commands are completed in a finite amount of time and can be called synchronously.

At entry, A0 contains the stream pointer, A1 contains a pointer to the ICMP report structure if the event code in D0 is ICMP received, A2 contains the user data pointer, A5 is already set up to point to application globals, and D0 (word) contains an event code.

Event codes

UDPDataArrival	a UDP datagram has arrived on this stream but no UDPRead commands are outstanding
UDPICMPReceived	an ICMP has been received on this stream; register A1 points to the ICMP report

An ICMP message reports an error in the processing of a datagram that was sent on a UDP stream. When an ICMP message is received, a data structure is passed up by UDP to the client to describe the received message. This data structure, called an ICMP report, has the following format:

0	long	stream pointer
4	long	local IP address of stream
8	word	local UDP port of stream
10	long	remote IP address (destination of original datagram)
14	word	remote UDP port
16	word	ICMP message type
18	word	optional additional information
20	long	optional additional information pointer

The values for the ICMP message type are as follows:

0	net unreachable

1 host unreachable

2 protocol unreachable

3 port unreachable

4 fragmentation required

5 source route failed

6 time exceeded

7 parameter problem

8 missing required option

Codes 0-3 are defined as follows:

- Net unreachable indicates that, according to the information in a gateway routing table, the network specified in the IP destination field of a UDP datagram is unreachable.
- Host unreachable indicates that a gateway determined that the host specified in the IP destination field of a UDP datagram is unreachable.
- Protocol unreachable indicates that a UDP datagram was delivered to the destination host but UDP was not ready to receive any datagrams.
- Port unreachable indicates that a UDP datagram was delivered to the destination host but there was no UDP client listening on that particular port.

These ICMP messages may be received occasionally when the topology of the internet changes. A single destination unreachable message should not be taken too seriously; however, if several successive UDPSend commands each result in an ICMP report indicating that the destination is unreachable, the UDP client should assume that the remote host has either crashed or is no longer accessible.

The remaining codes (4–8) indicate problems in the format of the IP header on a UDP datagram. They are informational only. Since the UDP client has no access to the IP header, the user cannot correct the error.

C parameter block definitions

The following C type definitions are used with parameter-block Device Manager calls to UDP:

```
#define UDPCreate           20
#define UDPRead             21
#define UDPBfrReturn        22
#define UDPWrite            23
#define UDPRelease          24
#define UDPMaxMTUSize       25
#define UDPCtlMax           29

typedef enum UDPEventCode {
    UDPDataArrival = 1,
    UDPICMPReceived,
    lastUDPEvent = 65535
    };

typedef pascal void (*UDPNotifyProc) (
    StreamPtr udpStream,
    unsigned short eventCode,
    Ptr userDataPtr,
    struct ICMPReport *icmpMsg);

typedef void (*UDPIOCompletionProc) (struct UDPiopb *iopb);

typedef unsigned short udp_port;

typedef struct UDPCreatePB {  /* for create and release calls */
    Ptr                 rcvBuff;
    unsigned long       rcvBuffLen;
    UDPNotifyProc       notifyProc;
    unsigned short      localPort;
    Ptr                 userDataPtr;
} UDPCreatePB;

typedef struct UDPSendPB {
    unsigned short      reserved;
    ip_addr             remoteHost;
    udp_port            remotePort;
    Ptr                 wdsPtr;
    Boolean             checkSum;
    unsigned short      sendLength;
    Ptr                 userDataPtr;
} UDPSendPB;
```

◆ *Note:* The following parameter block is used with the UDPRead and UDPBfrReturn commands:

```
typedef struct UDPReceivePB
    unsigned short      timeOut;
    ip_addr             remoteHost;
    udp_port            remotePort;
    Ptr                 rcvBuff;
    unsigned short      rcvBuffLen;
    unsigned short      secondTimeStamp;
    Ptr                 userDataPtr;
} UDPReceivePB;

typedef struct UDPMTUPB {
    unsigned short      mtuSize;
    ip_addr             remoteHost;
    Ptr                 userDataPtr;
} UDPMTUPB;

typedef struct UDPiopb {
    char                        fill12[12];
    UDPIOCompletionProc         ioCompletion;
    short                       ioResult;
    char                        *ioNamePtr;
    short                       ioVRefNum;
    short                       ioCRefNum;
    short                       csCode;
    StreamPtr                   udpStream;
    union {
        struct UDPCreatePB      create;
        struct UDPSendPB        send;
        struct UDPReceivePB     receive;
        struct UDPMTUPB         mtu;
    } csParam;
} UDPiopb;
```

Chapter 4 **Transmission Control Protocol**

THE TRANSMISSION CONTROL PROTOCOL (TCP) is a highly reliable, connection-oriented byte-stream protocol. It is designed to operate over a wide variety of networks and to provide virtual circuit service with orderly transmission of user data. TCP serves as the basis for a reliable interprocess communication mechanism on top of the IP layer where loss, damage, duplication, delay, or misordering of packets can occur. ■

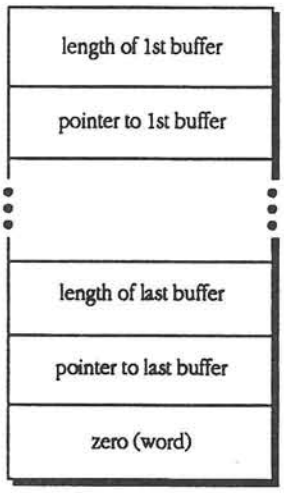
Data structures

This section describes TCP Read and Write Data Structures and the buffer area.

Read and Write Data Structures

The MacTCP driver uses Read Data Structures and Write Data Structures (RDS/WDS) to pass data between the user and TCP. These structures allow a single read or write operation to handle multiple blocks of data; that is, they allow scatter-read and gather-write capability. Figure 4-1 shows TCP Read and Write Data Structures.

■ **Figure 4-1** TCP Read and Write Data Structures



The simplest possible Read and Write Data Structures are 8 bytes in length: a word length, a long pointer, and a terminating word of 0. No maximum number of buffers can be described by a RDS/WDS used to transfer data between TCP and the user.

Receive buffer area

TCP does not allocate memory for storing TCP stream databases or for buffering received datagrams. Instead, the user is required to pass TCP enough memory for these purposes in the TCPCreate call. This has two advantages:

- The buffer memory can be allocated off the application heap instead of the system heap, which is very limited.
- The user has control over the buffering provided by TCP and can allocate the appropriate amount of memory for the type of application and performance level desired.

This buffer area for incoming datagrams belongs to the MacTCP driver as long as the TCP stream is open. When TCPRelease is called, this memory is returned to the user. The user can then reuse it or return it to the system.

Using TCP

This section describes how to send and receive TCP segments.

Streams and connections

A TCP stream supports one connection at a time. But a TCP connection on a stream can be closed and another connection opened without releasing the TCP stream. MacTCP can support 64 open TCP streams simultaneously.

Asynchronous notification routine

When a TCP stream is created, a routine must be registered that TCP can use to notify the user of asynchronous events relevant to that TCP stream. This routine is called the asynchronous notification routine (ASR).

Examples of events that MacTCP communicates to the user by means of an ASR include Time-out expired, Data arrived, and Connection terminated. Since the ASR is called from interrupt level, it cannot allocate or release memory. In addition, synchronous MacTCP driver commands cannot be issued from an ASR.

Connection opening

To listen for an incoming connection, use the TCPPassiveOpen command. This command can be used to specify whether any incoming connection will be accepted or only connections from a particular remote IP address and TCP port.

To initiate outgoing connections, use the TCPActiveOpen command.

Receiving data

Two methods are available for receiving data: a high-performance method and a simple method. The user can choose the method that best suits the application or combine the two methods if desired.

Using the TCPNoCopyRcv routine is the high-performance method. Data is delivered to the user directly from the internal TCP receive buffers and no copy is required. An RDS is formatted to allow many received segments to be delivered to the user in one TCPNoCopyRcv command. TCPBfrReturn must be called for every TCPNoCopyRcv that returns a nonzero amount of data in order to return the internal receive buffers to the MacTCP driver.

Using the TCPRcv routine is the simple method of receiving data. Data is copied from the TCP internal receive buffers into the user's buffer, which can be of arbitrary length and location. No RDS is used and no TCPBfrReturn call is required.

Sending data

To send data on a connection that is already established, use the TCPSend command.

Both commands use WDSs to implement a gather-write capability. This allows the user to send several discontiguous chunks of data in one operation.

Time-outs

TCP normally provides some time-out services to the upper-layer protocol (ULP). This time-out service is known as the ULP time-out. If there is data to send on a connection, but for some specified period of time either the data cannot be sent because the remote TCP is not set to receive any data or the data has been sent but the remote TCP has not acknowledged it, then the ULP time-out expires.

When the ULP time-out expires, TCP executes the specified ULP time-out action. There are two possible actions: abort the connection or report the timer expiration by means of an ASR and restart the timer.

In addition to this normal TCP timer, MacTCP provides command time-outs on commands that are not subject to the ULP timer. These commands are TCPPassiveOpen, TCPNoCopyRcv, and TCPRcv.

A TCPPassiveOpen command instructs TCP to wait for an incoming connection. The ULP timer is not started until the first connection initiation segment arrives. To prevent a TCPPassiveOpen command from waiting indefinitely for a connection initiation segment to arrive, a command time-out is provided. If no connection initiation segment is received within the specified time-out period, the TCPPassiveOpen command is completed with an error code.

Similarly, the TCPNoCopyRcv and TCPRcv commands are not protected by the ULP timer. In the absence of command time-outs, both commands could wait indefinitely for data to arrive. If a command time-out is specified on a TCPNoCopyRcv or TCPRcv command, the command is always completed within the specified time period. If no data has arrived at that point, the command is completed with an error. If some data has arrived, the command is completed successfully, returning the data that has arrived so far.

Pushed data

Normally, TCP is allowed to collect data sent by means of TCPSend commands and to send that data in segments, as convenient. By setting the Push flag in the TCPSend command, TCP can be instructed to send all unsent data at once.

Similarly, TCP is allowed to collect received data internally and deliver it with reasonable promptness to the user. However, when pushed data arrives, TCP immediately delivers all received data to the user.

Note that there is no necessary relationship between the Push flag in a TCPSend command and segment boundaries. The push function does not provide a method of marking message boundaries.

Urgent mode

TCP's urgent mechanism allows the sending user to prompt the receiving user to accept some urgent data and permits the receiving TCP to indicate to the receiving user when all currently known urgent data has been received.

TCP does not define what the user is required to do when in urgent mode, but the general practice is that the receiving user takes action to process the arriving data quickly.

A receiving user can be put in urgent mode in two ways: by means of an Urgent flag in a TCPNoCopyRcv or TCPRcv command, or by an urgent ASR notification. The user is always taken out of urgent mode by a Mark flag in the TCPNoCopyRcv or TCPRcv command that contains the last byte of urgent data.

To send data as urgent, the user must set the Urgent flag in the TCPSend command.

Connection closing

TCP closes communications gracefully. All outstanding Send requests are transmitted and acknowledged before the connection is allowed to close. A user can issue several TCPSend commands followed by a TCPClose command and expect that all the data will be sent successfully to the remote TCP.

A TCPClose command means "I have no more data to send," but it does not mean "I will receive no more data" or "shut down this connection immediately." A connection may remain open indefinitely after a TCPClose command is completed as the remote TCP continues to send data. When the remote TCP also issues a close command—and only then—the connection is closed. A TCPRcv command should be issued after a TCPClose command is completed to make sure that all the data is received.

If the desired effect is to break the connection without any assurance that all data in transit is delivered, use the TCPAbort command.

Network management information

TCP keeps two types of network management information: global TCP information and stream-specific information, which is relevant only to a particular stream.

The TCPGlobalInfo command makes global TCP information accessible. This command returns pointers to the actual structures where TCP stores this information. Thus, the user has read-write access to this information.

The TCPStatus command makes stream-specific information accessible. Most stream-specific information is copied into the TCPStatus I/O parameter block, giving the user read-only access. But a direct pointer to the traffic statistics allows the user read-write access to those counters (see the section "TCPGlobalInfo" later in this chapter).

Formatting MacTCP commands

In most cases, a 0 value for a parameter in an I/O parameter block means that TCP should use its default value. Thus, the user could initialize I/O parameter blocks and then fill in the required parameters, plus optional parameters.

TCP routines

This section presents calls to the TCP driver. Table 4-1 lists each TCP routine and its function. The C parameter-block definitions are presented at the end of the chapter.

■ **Table 4-1** TCP routines

Routine	Function
TCPCreate	opens a TCP stream
TCPPassiveOpen	listens for an incoming connection
TCPActiveOpen	initiates an outgoing call to a remote TCP
TCPSend	sends specified data on a connection
TCPNoCopyRcv	retrieves data that has been received on a connection
TCPBfrReturn	returns a set of receive buffers to the TCP driver
TCPRcv	retrieves data that has been received on a connection
TCPClose	signals that the user has no more data to send on this connection
TCPAbort	terminates the connection without trying to send outstanding data or deliver received data
TCPStatus	extracts information from TCP regarding a particular connection
TCPRelease	closes a TCP stream
TCPGlobalInfo	allows the user access to global statistics and parameters that affect the operation of TCP

TCPCreate

Parameter block

→	26	word	csCode = TCPCreate
←	28	long	stream pointer
→	32	long	pointer to receive buffer area
→	36	long	length of receive buffer area
→	40	long	pointer to ASR
→	44	long	user data pointer

TCPCreate opens a TCP stream. A TCP stream is not equivalent to a TCP connection. MacTCP returns a pointer to a stream database. The stream pointer is an input parameter in all subsequent commands affecting the stream.

The receive buffer area is a block of memory that TCP uses to buffer incoming segments. Ownership of this block of memory passes to TCP. The memory—a minimum of 4096 bytes—cannot be modified or relocated until TCPRelease is called. The size of the receive window that TCP offers is based on the size of the receive buffer area passed to TCP in the TCPCreate call. High-performance and block-oriented applications should provide TCP with a large receive buffer area: 16 KB (kilobytes) is recommended and up to 128 KB can be useful in certain applications. Character-oriented applications can use the minimum value of 4096 bytes; however, at least 8192 bytes is recommended.

An ASR must be provided. The ASR is called by TCP to notify the user of asynchronous events such as Data arrival, Urgent data outstanding, and Connection terminated.

Result codes

noErr	no error
streamAlreadyOpen	an open stream is already using this receive buffer area
invalidLength	the receive buffer area is too small
invalidBufPtr	the receive buffer area pointer is 0
insufficientResources	there are already 64 TCP streams open

TCPPassiveOpen

Parameter block	→	26	word	csCode = TCPPassiveOpen
	→	28	long	stream pointer
	→	32	byte	ULP time-out value in seconds; 0 = use default
	→	33	byte	ULP time-out action; 0 = report, nonzero = abort
	→	34	byte	validity bits for optional parameters
	→	35	byte	command time-out in seconds; 0 = infinity
	↔	36	long	remote IP address; can be 0
	↔	40	word	remote TCP port; can be 0
	←	42	long	local IP address
	↔	46	word	local TCP port; if 0, TCP assigns an unused port
	→	48	byte	type of service
	→	49	byte	precedence
	→	50	byte	don't fragment flag
	→	51	byte	time to live
	→	52	byte	security flag
	→	53	byte	IP option count
	→	54	var.	IP options
	→	94	long	user data pointer

TCPPassiveOpen listens for an incoming connection. The command is completed when a connection is established or when an error occurs.

Several fields in the TCPPassiveOpen command are optional. To indicate whether the user is including these optional parameters, a set of validity bits is defined as follows:

bit 4 precedence parameter is valid
bit 5 type of service parameter is valid
bit 6 ULP time-out action parameter is valid
bit 7 ULP time-out value parameter is valid

For example, a value of 0xC0 would mean that the ULP time-out value and action parameters are valid. If a validity bit is 0, TCP uses its default value for that parameter.

If the remote IP address and remote TCP port are 0, a connection is accepted from any remote TCP. If they are not 0, a connection is accepted only from that particular remote TCP. If the local TCP port is 0, TCP assigns an unused port value.

If a connection is partly established but cannot be completed within the ULP time-out period after the first connection opening segment arrives, the ULP action is taken. If the ULP time-out action is abort (the default value), the connection is broken and the TCPPassiveOpen command is completed in error. If the ULP time-out action is report, an ASR call informs the ULP, and the ULP timer is restarted. The minimum value of the ULP time-out is 2 seconds; 0 means that TCP should use its default value of 2 minutes.

If no connection opening packet arrives within the specified command time-out period after the TCPPassiveOpen command is issued, the command is completed in error. The minimum value of the command time-out is 2 seconds; 0 means infinite.

Users have control over many fields in the IP header of all segments sent on this connection. These fields can be set only when the connection is opened. They stay fixed during the connection.

The type of service is a 3-bit field:

bit 0 set for low delay
bit 1 set for high throughput
bit 2 set for high reliability

For example, a value of 0x02 means high throughput. The default value is 0.

Precedence has the following values:

0 routine
1 priority
2 immediate
3 flash
4 flash-override
5 CRITIC/ECP
6 internetwork control
7 network control

The default value for precedence is 0.

If the Don't Fragment flag is not 0, all segments sent on this connection are prohibited from being fragmented by the local IP or any intermediate IP. If a segment cannot be delivered without fragmentation, it is discarded.

The Time to live indicates the maximum time that segments on this connection are allowed to remain in the internet system. This value is decreased by every IP module that processes the segment; thus, it is effectively a maximum hop count (the number of times a segment can pass through a module). The minimum value is 2; 0 means TCP should use its default value (the default value is 60).

If the Security flag is not 0, TCP inserts its configured default IP security option into all segments sent on this connection. In addition, for all arriving segments that contain a security option, TCP verifies that the security matches the configured default security. Note that this flag is relevant only if no security option is present in the user-specified IP options.

Finally, the user can specify additional IP options to be sent with every segment. The option count is the number of long words in the IP option field. Pad bytes of 0 should be appended to the IP options so that the options are an integral number of long words. The maximum value of the option count field is 10 unless the Security flag is also not 0, in which case the maximum value is 9.

TCP does not perform any verification on the user-specified IP options, but simply inserts them into the IP header of every segment sent on the connection. If the user specifies an invalid list of IP options, the result cannot be predicted. See Request for Command (RFC) 894 for the proper format of IP options.

Result codes

noErr	no error
invalidStreamPtr	the specified TCP stream is not open
connectionExists	this TCP stream already has an open connection
duplicateSocket	a connection already exists between this local IP address and TCP port, and the specified remote IP address and TCP port
commandTimeout	no connection attempt was received in the specified time-out period
openFailed	the connection came halfway up and then failed

TCPActiveOpen

Parameter block				
	→	26	word	csCode = TCPActiveOpen
	→	28	long	stream pointer
	→	32	byte	ULP time-out value in seconds; 0 = use default
	→	33	byte	ULP time-out action; 0 = report, 1 = abort
	→	34	byte	validity bits
	→	35	byte	command time-out value
	→	36	long	remote IP address; cannot be 0
	→	40	word	remote TCP port; cannot be 0
	←	42	long	local IP address
	↔	46	word	local TCP port; if 0, TCP assigns an unused port
	→	48	byte	type of service
	→	49	byte	precedence
	→	50	byte	don't fragment flag
	→	51	byte	time to live
	→	52	byte	security flag
	→	53	byte	IP option count
	→	54	var.	IP options
	→	94	long	user data pointer

TCPActiveOpen initiates an outgoing call to a remote TCP. The command is completed when a connection is established or when an error occurs.

TCPActiveOpen accepts the same parameters as TCPPassiveOpen, except that the remote IP address and remote TCP port must be specified. In addition, no command time-out is provided; if the connection cannot be established within the ULP time-out period, the command is completed in error.

See TCPPassiveOpen for a description of other parameters.

Result codes

noErr	no error
invalidStreamPtr	the specified TCP stream is not open
connectionExists	this TCP stream already has an open connection
duplicateSocket	a connection already exists between this local IP address and TCP port, and the specified remote IP address and TCP port
openFailed	the connection came halfway up and then failed

TCPSend

Parameter block

→	26	word	csCode = TCPSend
→	28	long	stream pointer
→	32	byte	ULP time-out value in seconds
→	33	byte	ULP time-out action; 0 = abort, nonzero = report
→	34	byte	validity bits
→	35	byte	push flag
→	36	byte	urgent flag
→	38	long	WDS pointer
→	42	long	reserved
→	46	word	reserved
→	48	long	user data pointer

TCPSend sends the specified data over the connection. The command is completed when all data has been sent and acknowledged, or when an error occurs.

If all data cannot be sent and acknowledged within the ULP time-out period, then the ULP action is taken. If the ULP action is abort, the connection is broken, all pending commands are returned, and a terminate notification is given. If the ULP action is report, a ULP time-out notification is given. See TCPPassiveOpen for a description of the validity bits.

If the Push flag is not 0, TCP sends the data immediately without waiting to see if more TCPSend commands are issued. If the Urgent flag is not 0, TCP sends the data in urgent mode. In this case, all data described by the WDS is assumed to be urgent data.

The WDS can be arbitrarily complex; that is, there is no limit to the number of buffers that can be sent in a TCPSend command; however, the total number of data bytes described by the WDS must be between 1 and 65,535, inclusive. The WDS and the buffers it describes must not be modified or relocated until the TCPSend command is completed.

Result codes

noErr	no error
invalidStreamPtr	the specified TCP stream is not open
invalidLength	the total amount of data described by the WDS was either 0 or greater than 65,535 bytes
invalidWDS	the WDS pointer was 0
connectionDoesntExist	there is no open connection on this TCP stream
connectionClosing	a TCPClose command was already issued, so there is no more data to send on this connection
connectionTerminated	the connection went down; the reason will be given in a terminate ASR

TCPNoCopyRcv

Parameter block				
	→	26	word	csCode = TCPNoCopyRcv
	→	28	long	stream pointer
	→	32	byte	command time-out in seconds
	←	34	byte	urgent flag
	←	35	byte	mark flag
	→	42	long	RDS pointer; RDS is modified by TCP
	→	46	word	number of entries in RDS excluding terminating 0; not modified by TCP
	→	48	word	reserved
	→	50	long	user data pointer

TCPNoCopyRcv retrieves data that has been received on a connection. Data is not copied out of TCP's internal buffers; rather, an RDS is formatted to allow the user access to the TCP data in place. This command offers significant performance improvements over TCPRcv commands.

The command is completed when

- pushed data arrives
- urgent data is outstanding
- some reasonable period passes after the arrival of nonpush, nonurgent data
- the RDS is full; that is, the received data is in more noncontiguous chunks than the RDS can describe
- the amount of data received is greater than or equal to 25 percent of the total receive buffering for this stream
- the command time-out expires

The user must allocate memory for the RDS, which can contain an unlimited number of elements. The specified number of entries in the RDS does not include the terminating 0. For example, the simplest possible RDS has one element and is 8 bytes in length: a word length field, a long pointer field, and a terminating word length field of 0.

TCP formats the RDS to point to the received TCP data. The RDS may not be completely filled by TCP. For example, if the user passes an RDS three entries, TCP may complete the TCPNoCopyRcv command with the RDS describing only one buffer. TCP does not modify the field in the TCPNoCopyRcv command that indicates the number of entries in the RDS; rather, it places a terminating zero in the RDS itself.

The command time-out period starts when the receive command is issued, not when the first byte of data arrives. If no data arrives within this time-out period, the TCPNoCopyRcv command is completed in error. If some data has arrived when the command time-out expires, the TCPNoCopyRcv command is completed successfully, returning the data that has arrived so far. A command time-out of 0 indicates an infinite time-out period. In this case, the TCPNoCopyRcv command is not completed until some data has arrived.

If the Urgent flag is not 0, the data returned by this command is the beginning of the outstanding urgent data. This flag is one of two mechanisms that puts the user in urgent mode. The other mechanism is by means of an urgent ASR notification, which is used when there are no outstanding TCPNoCopyRcv or TCPRcv commands.

If the Mark flag is not 0, the data returned by this command ends the urgent data. Since TCP does not deliver urgent and nonurgent data together, the last byte of data described by the RDS is the last byte of urgent data. The Mark flag is the only mechanism for taking the user out of urgent mode.

The Urgent flag is set only on the first TCPNoCopy Rcv or TCPRcv command that contains urgent data. The Mark flag will be set only on the last TCPNoCopyRcv or TCPRcv command that contains urgent data. For example, if three TCPNoCopy Rcv or TCPRcv commands are required to deliver all urgent data, the settings of the Urgent and Mark flags will be as follows:

	Urgent flag	**Mark flag**
First TCPNoCopyRcv/TCPRcv command	1	0
Second TCPNoCopyRcv/TCPRcv command	0	0
Third TCPNoCopyRcv/TCPRcv command	0	1

Both the Urgent flag and the Mark flag can be set in a single TCPNoCopyRcv or TCPRcv command if all outstanding urgent data can be delivered in a single command.

The user is responsible for calling TCPBfrReturn after every TCPNoCopyRcv command that is completed successfully, in order to return the receive buffers owned by the TCP driver. The RDS must be returned unmodified so that the TCP driver can correctly recover the appropriate receive buffers.

Result codes

noErr	no error
invalidStreamPtr	the specified TCP stream is not open
connectionDoesntExist	this TCP stream has no open connection
invalidLength	the RDS has 0 entries
invalidBfrPtr	the RDS pointer was 0
commandTimeout	no data arrived within the specified time-out period
connectionClosing	all data on this connection has already been delivered
connectionTerminated	the connection went down; the reason will be given in a terminate ASR

TCPBfrReturn

Parameter block	→	26	word	csCode = TCPBfrReturn
	→	28	long	stream pointer
	→	42	long	RDS pointer
	→	50	long	user data pointer

TCPBfrReturn returns to the TCP driver a set of receive buffers that a successfully completed TCPNoCopyRcv command passed directly to the user. The RDS must be identical to the RDS given to the user when the TCPNoCopyRcv command is completed. TCPBfrReturn returns an error if the user attempts to return a set of buffers more than once.

Result codes		
	noErr	no error
	invalidStreamPtr	the specified TCP stream is not open
	connectionDoesntExist	this TCP stream has no open connection
	invalidBufPtr	the RDS pointer was 0
	invalidRDS	the RDS refers to receive buffers not owned by the user

TCPRcv

Parameter block				
	→	26	word	csCode = TCPRcv
	→	28	long	stream pointer
	→	32	byte	command time-out in seconds
	←	34	byte	urgent flag
	←	35	byte	mark flag
	→	36	long	receive buffer pointer
	↔	40	word	receive buffer length
	→	48	word	reserved
	→	50	long	user data pointer

TCPRcv retrieves data that has been received on a connection. Data is copied out of the TCP internal buffers into the user's buffer. The command is completed when

- enough data has arrived to fill the receive buffer
- pushed data arrives
- urgent data is outstanding
- some reasonable period passes after the arrival of nonpushed, nonurgent data
- the amount of data received is greater than or equal to 25 percent of the total receive buffering for this stream
- the command time-out expires

The command time-out period starts when the receive command is issued, not when the first byte of data arrives. If no data arrives within this command time-out period, the TCPRcv command is completed in error. If some data has arrived when the command time-out expires, the TCPRcv command is completed successfully. A command time-out of 0 means infinite; the TCPRcv command will not be completed until some data has arrived.

If the Urgent flag is not 0, the data returned by this command is the beginning of the outstanding urgent data. This flag is one of two mechanisms that puts the user in urgent mode. The other mechanism is by means of an urgent ASR notification, which is used when there are no outstanding TCPNoCopyRcv or TCPRcv commands.

If the Mark flag is not 0, the data returned by this command ends the urgent data. Since TCP does not deliver urgent and nonurgent data together, the last byte of data described by the RDS is the last byte of urgent data. The Mark flag is the only mechanism for taking the user out of urgent mode.

The Urgent flag is set only on the first TCPNoCopyRcv or TCPRcv command that contains urgent data. The Mark flag is set only on the last TCPNoCopyRcv or TCPRcv command that contains urgent data. For example, if three TCPNoCopy Rcv or TCPRcv commands are required to deliver all urgent data, the settings of the urgent and Mark flags will be as follows:

	Urgent flag	**Mark flag**
First TCPNoCopyRcv/TCPRcv command	1	0
Second TCPNoCopyRcv/TCPRcv command	0	0
Third TCPNoCopyRcv/TCPRcv command	0	1

Both the Urgent flag and the Mark flag can be set in a single TCPNoCopyRcv or TCPRcv command if all outstanding urgent data can be delivered in a single command.

If the TCPRcv command is completed in error, the receive buffer length is not modified by TCP, but no data is returned.

Result codes

noErr	no error
invalidStreamPtr	the specified TCP stream is not open
invalidLength	the receive buffer length was 0
invalidBufPtr	the receive buffer pointer was 0
commandTimeout	no data arrived within the specified time-out period
connectionDoesntExist	this TCP stream has no open connection
connectionClosing	all data on this connection has already been delivered
connectionTerminated	the connection went down; the reason will be given in a terminate ASR

TCPClose

Parameter block				
	→	26	word	csCode = TCPClose
	→	28	long	stream pointer
	→	32	byte	ULP time-out value in seconds
	→	33	byte	ULP time-out action; 0 = abort, nonzero = report
	→	34	byte	validity bits
	→	35	long	user data pointer

TCPClose signals that the user has no more data to send on the connection. It does not mean that the connection should be broken. The remote TCP must also issue a close before the connection can be gracefully closed, so a connection may stay up an arbitrary amount of time after a user has issued a TCPClose. To break a connection without ensuring that all data has been sent and acknowledged, use the TCPAbort command.

The command is completed when the FIN flag has been sent and acknowledged. If the FIN is not acknowledged within the ULP time-out period, the ULP time-out action is taken. If the ULP action is abort, the connection is broken, all pending commands are returned, and a terminate notification is given. If the ULP action is report, a ULP time-out notification is given. See TCPPassiveOpen for a description of the validity bits.

Result codes

noErr	no error
invalidStreamPtr	the specified TCP stream is not open
connectionDoesntExist	this TCP stream has no open connection
connectionClosing	one TCPClose command was already issued for this connection
connectionTerminated	the connection went down; the reason will be given in a terminate ASR

TCPAbort

Parameter block

→	26	word	csCode = TCPAbort
→	28	long	stream pointer
→	32	long	user data pointer

TCPAbort terminates the connection without attempting to send all outstanding data or to deliver all received data. TCPAbort returns the TCP stream to its initial state. The user is also given a terminate notification.

Result codes

noErr	no error
invalidStreamPtr	the specified TCP stream is not open
connectionDoesntExist	this TCP stream has no open connection

TCPStatus

Parameter block				
	→	26	word	csCode = TCPStatus
	→	28	long	stream pointer
	←	32	byte	ULP time-out value in seconds
	←	33	byte	ULP time-out action; 0 = abort, nonzero = report
	←	38	long	remote IP address
	←	42	word	remote TCP port
	←	44	long	local IP address
	←	48	word	local TCP port
	←	50	byte	type of service
	←	51	byte	precedence
	←	52	byte	TCP connection state
	←	54	word	send window
	←	56	word	receive window
	←	58	word	amount of unacknowledged data
	←	60	word	amount of unread data
	←	62	long	security option pointer
	←	66	long	send unacknowledged
	←	70	long	send next
	←	74	long	congestion window
	←	78	long	receive next
	←	82	long	smoothed round trip time in ms
	←	86	long	last round trip time in ms
	←	90	long	maximum segment size that can be sent
	←	94	long	pointer to statistics block
	→	98	long	user data pointer

TCPStatus allows the user to extract information from TCP regarding a particular connection.

See the TCPPassiveOpen command for a description of the usual open parameters (bytes 32–51).

The TCP connection state has the following values:

0	Closed	no connection exists on this stream
2	Listen	listening for an incoming connection
4	SYN received	incoming connection is being established
6	SYN sent	outgoing connection is being established
8	Established	connection is up
10	FIN Wait 1	connection is up; close has been issued

12	FIN Wait 2	connection is up; close has been completed
14	Close Wait	connection is up; close has been received
16	Closing	connection is up; close has been issued and received
18	Last Ack	connection is up; close has been issued and received
20	Time Wait	connection is going down

The send window is the amount of data the remote TCP is currently willing to accept from the local TCP. The receive window is the amount of data the local TCP is currently willing to accept from the remote TCP.

The statistics block has the following structure:

0	long	number of data segments received
4	long	number of data segments sent
8	long	number of data segments retransmitted
12	long	number of data bytes received
16	long	number of duplicate data bytes received
20	long	number of data bytes received beyond receive window
24	long	number of data bytes sent
28	long	number of data bytes retransmitted
32	struct	size histogram of sent segments

The histogram is a variable-length structure of the following format:

0	word	number of size buckets
2	word	value: smallest segment size
4	long	counter number: number of segments sent between this size and the next larger size
8	word	value: second smallest segment size
10	long	counter number: number of segments sent between this size and the next larger size

The user is free to update the statistics block. Only the counters in the segment-size histogram should be modified. The number of size buckets and the segment size value for each bucket cannot be changed.

Result codes

noErr	no error
invalidStreamPtr	the specified TCP stream is not open
connectionDoesntExist	this TCP stream has no open connection

TCPRelease

Parameter block

→	26	word	csCode = TCPRelease
→	28	long	stream pointer
←	32	long	pointer to receive buffer area
←	36	long	length of receive buffer area

TCPRelease closes a TCP stream. If there is an open connection on the stream, the connection is first broken as though a TCPAbort command had been issued.

The receive buffer area passed to MacTCP in the TCPCreate call is returned to the user. The user is now free to reuse or release this buffer area.

Result codes

noErr	no error
invalidStreamPtr	the specified TCP stream is not open

TCPGlobalInfo

Parameter block				
	→	26	word	csCode = TCPGlobalInfo
	←	32	long	pointer to TCP parameters
	←	36	long	pointer to TCP statistics
	←	40	long	pointer to CDB table
	→	44	long	user data pointer

The TCPGlobalInfo command allows the user access to

- parameters that affect the operation of TCP
- global statistics collected by TCP

The TCP parameters should not be modified by the user. The structure has the following format:

0	long	pointer to Pascal string describing retransmission time-out algorithm
4	long	minimum value of retransmission time-out (ms)
8	long	maximum value of retransmission time-out (ms)
12	long	maximum segment size this TCP can receive
16	long	maximum number of streams MacTCP can support
20	long	maximum value of receive window MacTCP can offer

The TCP statistics are the sum of traffic information for all streams that have been opened since system startup. These counters will wrap around to 0 when incremented past their maximum value. These statistics can be modified by the user. The TCP statistics structure has the following format:

0	long	number of outgoing connections attempted
4	long	number of outgoing connections opened
8	long	number of incoming connections accepted
12	long	number of connections that closed gracefully
16	long	number of connections that were aborted
20	long	number of data bytes received
24	long	number of data bytes sent
28	long	number of duplicate data bytes received
32	long	number of data bytes retransmitted
36	long	total number of segments received (includes acknowledgments)
40	long	total number of segments sent (includes acknowledgments)

	44	long	number of segments received that contained all duplicate data
	48	long	number of segments retransmitted

Result codes

noErr	no error

TCP asynchronous notification routine

The TCP asynchronous notification routine (ASR) is a user-supplied routine called by TCP to notify the user of asynchronous events relevant to a particular TCP stream. The user registers this routine with TCP in the TCPCreate call.

Since this routine is called at interrupt level, it cannot release or allocate memory. In addition, an ASR routine cannot issue additional MacTCP driver calls.

The C description of the ASR is as follows:

```
typedef pascal void (*TCPNotifyProc) (StreamPtr tcpStream,
unsigned short eventCode, Ptr userDataPtr, unsigned short terminReason,
struct ICMPReport *icmpMsg);
```

A1 contains a pointer to the Internet Control Message Protocol (ICMP) report structure if the event code in D0 is ICMP received, A2 contains the user data pointer, A5 is already set up to point to application globals, D0 (word) contains an event code, and D1 contains a reason for termination.

Event codes

closing	all data on this connection has been received and delivered
ULP time-out	no response from the remote TCP; reported only if TCP is configured to report a time-out instead of aborting the connection
terminate	connection no longer exists
data arrival	data arrived, no receive commands outstanding
urgent data outstanding	user should go into urgent mode
ICMP message received	an Internet Control Message has been received on the stream; register A1 points to the ICMP report

For the terminate event, a reason for connection termination is given in D1 (word):

remote abort	the remote TCP aborted the connection
network failure	to be supplied in a later draft of this guide
security/precedence mismatch	invalid security option or precedence level
ULP time-out	the ULP time-out expired; ULP time-out action is abort
ULP abort	the user issued a TCPAbort command
ULP close	the connection closed gracefully
service failure	unexpected connection initiation segment read

◆ *Note:* Refer to the Appendix for the values of these types.

A closing notification means that the remote TCP has sent all the data it intends to send on this connection and that all data has been delivered to the user. Closing notification does not mean that the connection has been broken. A user can continue to send data for an arbitrary length of time after a closing notification is given. Only when the user issues a TCPClose command will the connection be terminated.

The ULP time-out notification is given if the configured ULP time-out action for this connection is report, and the local TCP cannot get some data sent or acknowledged (or both), within the specified ULP time-out period. The ULP timer is restarted after notification and expires again in another ULP time-out period if nothing changes.

For each connection, TCP issues exactly one terminate notification when the connection is broken. This rule applies both when the connection closes gracefully and when the connection terminates in error.

Data arrival notification is given if a segment arrives and no receive commands are outstanding. Even if more data arrives later, TCP does not issue another data arrival notification until a receive command has been issued and completed. In other words, a data arrival notification is not given with every segment that arrives, but instead is intended to prompt the user to issue a receive command.

Urgent notification is given only if no receive commands are outstanding when TCP learns that there is outstanding urgent information on the connection. If there are outstanding receive commands, they are completed with the Urgent flag set to get the user into urgent mode.

An ICMP message reports an error in the processing of a datagram that was sent on a TCP stream. When an ICMP message is received, a data structure is passed up by TCP to the client to describe the received message. This data structure, called an ICMP report, has the following format:

0	word	stream pointer
2	long	local IP address of stream
6	word	local TCP port of stream
8	long	remote IP address (destination of original datagram)
12	word	remote TCP port
14	word	ICMP message type
16	word	optional additional information
18	long	optional additional information pointer

The values for the ICMP message type are as follows:

0	net unreachable
1	host unreachable
2	protocol unreachable
3	port unreachable
4	fragmentation required

5 source route failed

6 time exceeded

7 parameter problem

8 missing required option

Codes 0-3 are defined as follows:

- Net unreachable indicates that, according to the information in a gateway routing table, the network specified in the IP destination field of a TCP segment is unreachable.
- Host unreachable indicates that a gateway determined that the host specified in the IP destination field of a TCP segment is unreachable.
- Protocol unreachable indicates that a TCP segment was delivered to the destination host, but there was no process on that host that would receive TCP segments.
- Port unreachable indicates that a TCP segment was delivered to the destination host, but there was no client of TCP listening on that particular TCP port.

If the TCP stream is configured for the ULP time-out action of abort, the TCP client need not take any action in response to destination unreachable messages (they are informational only). TCP breaks the connection if there is data to send, but it is not acknowledged within the ULP time-out period. If, however, this TCP stream has been configured for the ULP time-out action of report, then the TCP user has taken responsibility for deciding when and if the remote host is no longer available and the connection should be broken. A single destination unreachable message should not be taken too seriously since such messages may be received occasionally when the topology of the internet changes. But if several successive TCPSend commands each result in an ICMP report indicating destination unreachable, the TCP client should assume that the remote host has either crashed or is no longer accessible, and should break the connection.

Codes 4-8 are explained in the following paragraphs.

Fragmentation required indicates that the TCP user has set the Don't Fragment flag in a TCPOpen command, yet a segment on that TCP connection could not be delivered to its destination without fragmentation. To avoid this, don't set the Don't Fragment flag on TCPOpen commands.

Source route failed indicates that the TCP user has specified the route this datagram should take in the IP options but that particular route was not available.

Time exceeded indicates that the Time to Live specified in a TCPOpen command was too short to allow a TCP segment on this TCP stream to be delivered through all the necessary gateways on the way to its destination. A longer Time to Live value should be specified in the TCPOpen command.

Parameter problem indicates that the IP header used on a TCP segment was not acceptable by either an intermediate gateway or the final destination. The additional information pointer in the ICMP report structure points to a static copy of the IP header used for sending segments on this TCP connection. The additional information value indicates a byte offset in the IP header where the parameter problem exists. For example, an offset of 1 indicates that the Type of Service field was invalid and an offset of 20 indicates that the first option present is invalid.

Missing required option means that the remote TCP requires a specific IP option to be present in the IP header. The IP option type code that is required by the remote TCP is passed in the additional information field.

Much of the information that asynchronous notifications pass to the user is also available in other ways. For example, a user can tell that the remote TCP has closed a connection either by waiting for a closing notification or by submitting TCPRcv commands until one is returned with a connectionClosing error.

C parameter block definitions

The following C type definitions are used with parameter-block Device Manager calls to TCP:

```
#define TCPCreate                30
#define TCPPassiveOpen           31
#define TCPActiveOpen            32
#define TCPSend                  34
#define TCPNoCopyRcv             35
#define TCPRcvBfrReturn          36
#define TCPRcv                   37
#define TCPClose                 38
#define TCPAbort                 39
#define TCPStatus                40
#define TCPExtendedStat          41
#define TCPRelease               42
#define TCPGlobalInfo            43
#define TCPCtlMax                49

typedef enum TCPEventCode {
    TCPClosing = 1,
    TCPULPTimeout,
    TCPTerminate,
    TCPDataArrival,
    TCPUrgent,
    TCPICMPReceived,
    lastEvent = 65535
};

typedef enum TCPTerminationReason {
    TCPRemoteAbort = 2,
    TCPNetworkFailure,
    TCPSecPrecMismatch,
    TCPULPTimeoutTerminate,
    TCPULPAbort,
    TCPULPClose,
    TCPServiceError,
    lastReason = 65535
};

typedef pascal void (*TCPNotifyProc) (
    StreamPtr tcpStream,
    unsigned short eventCode,
    Ptr userDataPtr,
    unsigned short terminReason,
    struct ICMPReport *icmpMsg);

typedef void (*TCPIOCompletionProc) (struct TCPiopb *iopb);

typedef unsigned short tcp_port;
```

```
typedef unsigned char byte;

enum {  /* ValidityFlags */
    timeoutValue = 0x80,
    timeoutAction = 0x40,
    typeOfService = 0x20,
    precedence = 0x10
};

enum {  /* TOSFlags */
    lowDelay = 0x01,
    throughPut = 0x02,
    reliability = 0x04
};

typedef struct TCPCreatePB {
        Ptr rcvBuff;
        unsigned long rcvBuffLen;
        TCPNotifyProc notifyProc;
        Ptr userDataPtr;
}TCPCreatePB;
```

◆ *Note:* The following parameter block is used with the TCPPassiveOpen and TCPActiveOpen commands:

```
typedef struct TCPOpenPB {
    byte ulpTimeoutValue;
    byte ulpTimeoutAction;
    byte validityFlags;
    byte commandTimeoutValue;
    ip_addr remoteHost;
    tcp_port remotePort;
    ip_addr localHost;
    tcp_port localPort;
    byte tosFlags;
    byte precedence;
    Boolean dontFrag;
    byte timeToLive;
    byte security;
    byte optionCnt;
    byte options[40];
    Ptr userDataPtr;
}TCPOpenPB;

typedef struct TCPSendPB {
        byte ulpTimeoutValue;
        byte ulpTimeoutAction;
        byte validityFlags;
        Boolean pushFlag;
        Boolean urgentFlag;
        Ptr wdsPtr;
        unsigned long sendFree;
        unsigned short sendLength;
        Ptr userDataPtr;
}TCPSendPB;
```

◆ *Note:* The following parameter block is used with the TCPRcv, TCPRcvBfrReturn, and TCPNoCopyRcv commands:

```
typedef struct TCPReceivePB
    byte commandTimeoutValue;
    byte filler;
    Boolean markFlag;
    Boolean urgentFlag;
    Ptr rcvBuff;
    unsigned short rcvBuffLen;
    Ptr rdsPtr;
    unsigned short rdsLength;
    unsigned short secondTimeStamp;
    Ptr userDataPtr;
}TCPReceivePB;

typedef struct TCPClosePB {
    byte ulpTimeoutValue;
    byte ulpTimeoutAction;
    byte validityFlags;
    Ptr userDataPtr;
}TCPClosePB;

typedef struct HistoBucket {
    unsigned short value;
    unsigned long counter;
};

#define NumOfHistoBuckets  7

typedef struct TCPConnectionStats {
    unsigned long dataPktsRcvd;
    unsigned long dataPktsSent;
    unsigned long dataPktsResent;
    unsigned long bytesRcvd;
    unsigned long bytesRcvdDup;
    unsigned long bytesRcvdPastWindow;
    unsigned long bytesSent;
    unsigned long bytesResent;
    unsigned short numHistoBuckets;
    struct HistoBucket sentSizeHisto[NumOfHistoBuckets];
    unsigned short lastRTT;
    unsigned short tmrSRTT;
    unsigned short rttVariance;
    unsigned short tmrRTO;
    byte sendTries;
    byte sourchQuenchRcvd;
}TCPConnectionStats;
```

```
typedef struct TCPStatusPB {
    byte ulpTimeoutValue;
    byte ulpTimeoutAction;
    long unused;
    ip_addr remoteHost;
    tcp_port remotePort;
    ip_addr localHost;
    tcp_port localPort;
    byte tosFlags;
    byte precedence;
    byte connectionState;
    unsigned short sendWindow;
    unsigned short rcvWindow;
    unsigned short amtUnackedData;
    unsigned short amtUnreadData;
    Ptr securityLevelPtr;
        /* HEMS-HEMP stats */
    unsigned long sendUnacked;
    unsigned long sendNext;
    unsigned long congestionWindow;
    unsigned long rcvNext;
    unsigned long srtt;
    unsigned long lastRTT;
    unsigned long sendMaxSegSize;
    struct TCPConnectionStats *connStatPtr;
    Ptr userDataPtr;
}TCPStatusPB;

typedef struct TCPAbortPB {
    Ptr userDataPtr;
}TCPAbortPB;

typedef struct TCPParam {
    unsigned long tcpRtoA;
    unsigned long tcpRtoMin;
    unsigned long tcpRtoMax;
    unsigned long tcpMaxSegSize;
    unsigned long tcpMaxConn;
    unsigned long tcpMaxWindow;
}TCPParam;
```

```
typedef struct TCPStats {
    unsigned long tcpConnAttempts;
    unsigned long tcpConnOpened;
    unsigned long tcpConnAccepted;
    unsigned long tcpConnClosed;
    unsigned long tcpConnAborted;
    unsigned long tcpOctetsIn;
    unsigned long tcpOctetsOut;
    unsigned long tcpOctetsInDup;
    unsigned long tcpOctetsRetrans;
    unsigned long tcpInputPkts;
    unsigned long tcpOutputPkts;
    unsigned long tcpDupPkts;
    unsigned long tcpRetransPkts;
}TCPStats;

typedef struct TCPGlobalInfoPB {
    struct TCPParam *tcpParamPtr;
    struct TCPStats *tcpStatsPtr;
    unsigned long *tcpCDBTable;
    Ptr userDataPtr;
}TCPGlobalInfoPB;

typedef struct TCPiopb {
    char                            fill12[12];
    TCPIOCompletionProc             ioCompletion;
    short                           ioResult;
    char                            *ioNamePtr;
    short                           ioVRefNum;
    short                           ioCRefNum;
    short                           csCode;
    StreamPtr                       tcpStream;
    union {
        struct TCPCreatePB          create;
        struct TCPOpenPB            open;
        struct TCPSendPB            send;
        struct TCPReceivePB         receive;
        struct TCPClosePB           close;
        struct TCPAbortPB           abort;
        struct TCPStatusPB          status;
        struct TCPGlobalInfoPB      globalInfo;
        } csParam;
}TCPiopb;
```

Chapter 5 **Name-to-Address Resolution**

TEXTUAL NAMES are resolved to IP addresses using internal caches and the domain name server. The AddressXlation interface accomplishes this task by searching an internal table originally derived from the static file Hosts, queries made to domain name servers, and information found in the internal cache of domain name server responses. ■

The AddressXlation.h header file

The AddressXlation.h interface supports the domain name resolver (DNR) function using the procedure calls described in the following sections.

The OpenResolver call

```
extern OSErr OpenResolver(char *fileName);
```

The OpenResolver call must be made before any resolver queries are attempted. The full path name of the default Hosts file must be passed in the fileName field. If the value NIL is passed, the resolver uses the filename Hosts in the default start up folder, which is typically the System Folder. The Hosts file is used to load the internal cache with name-to-address mappings and domain name server values. The syntax of this file is outlined on page 33 of RFC1035.

The StrToAddr call

```
#define NUM_ALT_ADDRS 4

typedef struct hostInfo {
    int rtnCode;
    char cname[255];
    unsigned long addr[NUM_ALT_ADDRS];
};
```

The hostInfo record is passed into the address translation routine in the StrToAddr call. The rtnCode indicates whether the result fields are valid for this call; a value of noErr indicates that the call has been successful. The cname is the official name of the hostName that was specified. The addr array is a list of addresses for the hostName specified. Multiple addresses are returned for hosts that are multihomed.

```
typedef pascal void (*ResultProcPtr)(
    struct hostInfo *hostInfoPtr,
    char *userDataPtr);

extern OSErr StrToAddr(
    char *hostName,
    struct hostInfo *hostInfoPtr,
    ResultProcPtr ResultProc,
    char *userDataPtr);
```

StrToAddr takes a string in one of two forms and translates it into a 32-bit IP address. The string can be in IP dot notation (that is, W.X.Y.Z) or in a valid domain name syntax. The translated address is returned immediately if the passed in host is in IP dot notation or if the matching address is contained in the local cache. If the address is not contained in the local cache, the rtnCode is cacheFault and the domain name server is contacted to resolve the address. When the response has been returned from the domain name server or the domain name query has completed unsuccessfully, the ResultProc is called with the appropriate rtnCode and return information. The StrToAddr procedure can be called with a userDataPtr, which is user-defined and not modified by the resolver. The userDataPtr is returned when the ResultProc is called. Domain names that contain no domain name delimiters, that is, "." (dot), are terminated with the domain name suffix specified for the default domain name server in the Control Panel or a dot if no default is specified.

The AddrToStr call

```
extern OSErr AddrToStr(unsigned long addr, char *addrStr);
```

The AddrToStr call takes an IP address and returns a string with the ASCII equivalent of the form W.X.Y.Z where W, X, Y, and Z are decimal numbers. The application must provide the storage for the return string. The string can have a maximum length of 16 bytes.

The EnumCache call

```
typedef enum AddrClasses {
    A = 1,
    NS,
    CNAME = 5,
    lastClass = 65535
};

typedef struct cacheEntryRecord {
    char *cname;
    unsigned short type;
    enum AddrClasses class;
    unsigned long ttl;
    union {
        char *name;
        ip_addr *addr;
    } rdata;
};

typedef pascal void (*EnumResultProcPtr)(
    struct cacheEntryRecord,*cacheEntryRecordPtr,
    char *userDataPtr);

extern OSErr EnumCache(
    EnumResultProcPtr enumResultProc,
    char *userDataPtr);
```

If the application wants to enumerate all the entries in the cache, EnumCache should be called. This procedure returns no errors and has completed ennumerating the cache when it returns. For every entry in the cache, the user's enumResultProc is called with a pointer to a cache entry. The fields and values returned in the CacheEntryRecord are as follows:

name	the name of the entry.
type	the type of the entry, where □ *A* (value = 1) is an address. The value of rdata is an addr. □ *NS* (value = 2) is a name server. The value of rdata is a name. □ *CNAME* (value = 5) is an alias for the canonical name (cname) found in the rdata field.
class	the class of the entry. The only class allowed is IN (value 1).
ttl	the time the entry has to live in the cache relative to the current time (GetCurrent).
rdata	the rdata field as determined by the type of entry specified in the type field.

The CloseResolver call

```
extern OSErr CloseResolver();
```

Before the application exits, the CloseResolver call must be made to release memory structures and terminate all outstanding domain name server calls. CloseResolver should not be called until all outstanding resolver calls have been completed.

The AddrToName call

```
extern OSErr AddrToName(ip_addr addr, struct hostInfo
*hostInfoPtr, ResultProcPtr ResultProc, char *userDataPtr);
```

The AddrToName call is used to acquire the canonical name of a host given its IP address. The domain name server is queried using an IN-ADDR query. The application passes to the AddrToName call the IP address of the host in question, a pointer to a hostInfo record, the result procedure to be notified with the result, and an optional user data pointer. The AddrToName call always returns cachefault for a query that will be sent. The passed in result procedure is called with the appropriate result code. If the result code is noErr, the cname field will contain the canonical name for the IP address passed to the AddrToName call.

Binding the DNR to the application

The DNR in MacTCP is implemented as a code resource in the MacTCP driver file.

A file called DNR.c in the Libraries Folder of the MacTCP release disk provides a working example of how to open the DNR. This file should be compiled and linked to your application. Procedural access to the DNR is provided by a file produced by MPW 2.0.2 called DNR.o in the Libraries Folder of the MacTCP release disk. Sources have also been provided if you need to port the functionality to another development system.

To use the DNR, the application must first find the procedure pointers that are part of the DNR resource. The DNR resource, named "dnrp," is attached to the driver file, which is in the System Folder of the startup disk. The resource can be opened using the Resource Manager routines in the Toolbox. Once the resource is opened, the first long word of the resource is a pointer to a procedure that jumps to the correct DNR procedure. Calls to this procedure should be made using the procedure index value as the first argument of the call, followed by the arguments for the procedure as they are specified in the AddressXlation.h interface. The procedure assignments are as follows:

OpenResolver	1
CloseResolver	2
StrToAddr	3
AddrToStr	4
EnumCache	5
AddrToName	6

Chapter 6 **Miscellaneous Interfaces**

THIS CHAPTER describes types that are found throughout the programmatic interfaces supplied with MacTCP. ■

MacTCPCommontypes

This file defines result code name-to-number mapping, Internet Control Message Protocol (ICMP) message report structures, and other miscellaneous types throughout MacTCP.

Result codes

The result codes in MacTCPCommontypes are described as follows:

- *Note:* Chapters 3, 4, and 5 contain specific occurrences of the result codes described in this section.

`#define inProgress`	1	When an IOPB is still pending, ioResult is set to inProgress.
`#define ipBadLapErr`	-23000	Unable to initialize the local network handler.
`#define ipBadCnfgErr`	-23001	The manually set address is configured improperly.
`#define ipNoCnfgErr`	-23002	A configuration resource is missing.
`#define ipLoadErr`	-23003	Not enough room in the application heap (Macintosh 512K enhanced only).
`#define ipBadAddr`	-23004	Error in getting address from a server or the address is already in use by another machine.
`#define connectionClosing`	-23005	A TCPClose command was already issued so there is no more data to send on this connection.
`#define invalidLength`	-23006	The total amount of data described by the WDS was either 0 or greater than 65,535 bytes.
`#define connectionExists`	-23007	The TCP or UDP stream already has an open connection.
`#define connectionDoesntExist`	-23008	This TCP stream has no open connection.
`#define insufficientResources`	-23009	There are already 64 TCP or UDP streams open.
`#define invalidStreamPtr`	-23010	The specified TCP or UDP stream is not open.
`#define streamAlreadyOpen`	-23011	An open stream is already using this receive buffer area.
`#define connectionTerminated`	-23012	The TCP connection went down; the reason will be given in a terminate ASR.

`#define invalidBufPtr`	-23013	The receive buffer area pointer is 0.
`#define invalidRDS`	-23014	The RDS refers to receive buffers not owned by the user.
`#define invalidWDS`	-23014	The WDS pointer was 0.
`#define openFailed`	-23015	The connection came up halfway and then failed.
`#define commandTimeout`	-23016	The specified command action was not completed in the specified time period.
`#define duplicateSocket`	-23017	A stream is already open using this local UDP port or a TCP connection already exists between this local IP address and TCP port, and the specified remote IP address and TCP port.
`#define ipDontFragErr`	-23032	The packet is too large to send without fragmenting and the Don't Fragment flag is set.
`#define ipDestDeadErr`	-23033	The destination host is not responding to address resolution requests.
`#define ipNoFragMemErr`	-23036	Insufficient internal driver buffers available to fragment this packet on send.
`#define ipRouteErr`	-23037	No gateway available to handle routing of packets to off-network destinations.
`#define nameSyntaxErr`	-23041	The hostName field had a syntax error. The address was given in dot notation (that is, W.X.Y.Z) and did not conform to the syntax for an IP address.
`#define cacheFault`	-23042	The name specified cannot be found in the cache. The domain name resolver will now query the domain name server and return the answer in the call-back procedure.
`#define noResultProc`	-23043	No result procedure was passed to the address translation call when the resolver must be used to find the address.
`#define noNameServer`	-23044	No name server can be found for the specified name string.
`#define authNameErr`	-23045	This domain name does not exist.
`#define noAnsErr`	-23046	None of the known name servers are responding.
`#define dnrErr`	-23047	The domain name server has returned an error.
`#define outOfMemory`	-23048	Not enough memory is available to issue the needed DNR query or to build the DNR cache.

Miscellaneous types

This section describes types that are common to all the programmatic interfaces in MacTCP.

```
#define BYTES_16WORD            2       /* bytes per 16 bit ip word */
#define BYTES_32WORD            4       /* bytes per 32 bit ip word */
#define BYTES_64WORD            8       /* bytes per 64 bit ip word */

typedef unsigned char b_8;      /* 8-bit quantity */
typedef unsigned short b_16;    /* 16-bit quantity */
typedef unsigned long b_32;     /* 32-bit quantity */

typedef b_32 ip_addr;    /* IP address is 32-bits */
typedef b_16 ip_port

typedef struct ip_addrbytes {
    union {
          b_32 addr;
          char byte[4];
          } a;
    } ip_addrbytes;

typedef struct wdsEntry {
     unsigned short length;       /* length of buffer */
     char * ptr;       /* pointer to buffer */
     } wdsEntry;

typedef struct rdsEntry {
     unsigned short length;       /* length of buffer */
     char * ptr;       /* pointer to buffer */
     } rdsEntry;

typedef unsigned long BufferPtr;

typedef unsigned long StreamPtr;
```

Internet Control Message Protocol report structures

In TCP and UDP, the ASR routine can be called with an Internet Control Message Protocol (ICMP) message. This section describes the types and structures of the ICMP messages.

```
typedef enum ICMPMsgType {
     netUnreach, hostUnreach, protocolUnreach, portUnreach, fragReqd,
     sourceRouteFailed, timeExceeded, parmProblem, missingOption,
     lastICMPMsgType = 65535
     } ICMPMsgType;

typedef struct ICMPReport {
     StreamPtr streamPtr;
     ip_addr localHost;
     ip_port localPort;
     ip_addr remoteHost;
     ip_port remotePort;
     enum ICMPMsgType reportType;
     unsigned short optionalAddlInfo;
     unsigned long optionalAddlInfoPtr;
     } ICMPReport;
```

Refer to the section "UDP Asynchronous Notification Routine" in Chapter 3 and "TCP Asynchronous Notification Routine" in Chapter 4 for details on how the ICMP report structure is used.

GetMyIPAddr

This section describes how an application obtains the IP address of the machine on which it is running. GetMyIPAddr describes the parameter block that makes the PBControl call that returns the IP address and subnet mask of the local host. The csCode for this call is 15 and the driver reference number is returned from the Open Driver call.

```
#define ipctlGetAddr        15      /* csCode to get our IP address */

#define ParamBlockHeader    \
        struct QElem *qLink;        \
        short qType;        \
        short ioTrap;       \
        Ptr ioCmdAddr;      \
        ProcPtr ioCompletion;       \
        OSErr ioResult;     \
        StringPtr ioNamePtr;        \
        short ioVRefNum;    \
        short ioCRefNum;    \
        short csCode

struct IPParamBlock {
        ParamBlockHeader;   /* standard I/O header */
        ip_addr ourAddress; /* our IP address */
        long ourNetMask;    /* our IP net mask */
        };
```

The IP address is returned in the field ourAddress and the subnet mask is returned in the field ourNetMask.

Appendix **Constants**

Command codes

UDPCreate	20
UDPRead	21
UDPBfrReturn	22
UDPWrite	23
UDPRelease	24
UDPMaxMTUSize	25
TCPCreate	30
TCPPassiveOpen	31
TCPActiveOpen	32
TCPSend	34
TCPNoCopyRcv	35
TCPBfrReturn	36
TCPRcv	37
TCPClose	38
TCPAbort	39
TCPStatus	40
TCPRelease	42
TCPGlobalInfo	43

UDP asynchronous event codes

data arrival	1
ICMP message received	2

TCP asynchronous event codes

closing	1
ULP timeout	2
terminate	3
data arrival	4
urgent data outstanding	5
ICMP message received	6

Reasons for TCP termination

remote abort	2
network failure	3
security/precedence mismatch	4
ULP timeout	5
ULP abort	6
ULP close	7
service failure	8

APPLE LICENSE AGREEMENT

PLEASE READ THIS DOCUMENT CAREFULLY BEFORE OPENING THE PACKAGE. BY OPENING THE PACKAGE, YOU ARE AGREEING TO BECOME BOUND BY THE TERMS OF THIS AGREEMENT, WHICH INCLUDES THE SOFTWARE LICENSE AND DISCLAIMER OF WARRANTY. IF YOU DO NOT AGREE TO THE TERMS OF THIS AGREEMENT, DO NOT OPEN THE PACKAGE. PROMPTLY RETURN THE UNOPENED PACKAGE TO THE PLACE WHERE YOU OBTAINED IT FOR A FULL REFUND.

The enclosed computer program ("Apple Software") is licensed, not sold, to you for use only under the terms of this license, and Apple reserves any rights not expressly granted to you. You own the disk on which the Apple Software is originally or subsequently recorded or fixed, but Apple retains ownership of all copies of the Apple Software itself.

1. License. This license allows you to:

(a) Use the Apple Software only on a single Apple computer. You must obtain a supplementary license from Apple before using the Apple Software in connection with systems, multiple central processing units, computer networks or emulations on a mainframe or minicomputer.

(b) Make one copy of the Apple Software in machine readable form solely for backup purposes. The Apple Software is protected by United States copyright law. You must reproduce on each copy the Apple copyright notice and any other proprietary legends that were on the original copy .

(c) Transfer the Apple Software and all rights under this license to another party together with a copy of this Agreement provided you give Apple written notice of the transfer and the other party reads and agrees to accept the terms and conditions of this Agreement.

(d) Configure the Apple Software for your own use by adding or removing fonts, desk accessories, and/or device drivers.

2. Restrictions. You may not distribute copies of the Apple Software to others or electronically transfer the Apple Software from one computer to another over a network. The Apple Software contains trade secrets and in order to protect them you may not decompile, reverse engineer, disassemble or otherwise reduce the Apple Software to a human perceivable form. YOU MAY NOT MODIFY, ADAPT, TRANSLATE, RENT, LEASE, LOAN, RESELL FOR PROFIT, DISTRIBUTE, NETWORK OR CREATE DERIVATIVE WORKS BASED UPON THE APPLE SOFTWARE OR ANY PART THEREOF.

3. Termination. This License is effective until terminated. This License will terminate automatically without notice from Apple if you fail to comply with any provision of this License. Upon termination you must destroy the Apple Software and all copies thereof. You may terminate this License at any time by destroying the Apple Software and all copies thereof.

4. Export Law Assurances. You agree and certify that neither the Apple Software nor any direct product thereof is being or will be shipped, transfered or reexported, directly or indirectly, into any country prohibited by the United States Export Administration Act and the regulations thereunder or will be used for any purpose prohibited by the same.

7. Government End Users. If this Apple Software is acquired by or on behalf of a unit or agency of the United States Government this provision applies. This Apple Software: (a) was developed at private expense, and no part of it was developed with government funds, (b) is a trade secret of Apple for all purposes of the Freedom of Information Act, (c) is "commercial computer software" subject to limited utilization as provided in the contract between the vendor and the governmental entity and (d) in all respects is proprietary data belonging solely to Apple Computer, Inc.

For units of the Department of Defense (DoD), this Apple Software is sold only with "Restricted Rights" as that term is defined in the DoD Supplement to the Federal Acquisition Regulations, 52.227-7013 (c) (ii) and:

> Use, duplication or disclosure is subject to restrictions as set forth in subdivision (c) (ii) of the Rights in Technical Data and Computer Software clause at 52.227-7013.
> Manufacturer: Apple Computer, Inc., 20525 Mariani Avenue, Cupertino, California 95014.

If this Apple Software was acquired under a GSA Schedule the Government has agreed to refrain from changing or removing any insignia or lettering from the Software or the documentation that is provided or from producing copies of manuals or disks (except one copy for backup purposes) and:

> (1) Title to and ownership of this Apple Software and documentation and any reproductions thereof shall remain with Apple Computer, Inc.
> (2) Use of this Apple Software and documentation shall be limited to the facility for which it is acquired; and
> (3) If the use of the Apple Software is discontinued to the installation specified in the purchase/delivery order and the Government desires to use it at another location, it may do so by giving prior written notice to Apple Computer, Inc., specifying the type of computer and new location site.

Governmental personnel using this Apple Software, other than under a DoD contract or GSA Schedule, are hereby on notice that use of this Apple Software is subject to restrictions that are the same as or similar to those specified above.

6. Disclaimer of Warranty. THE APPLE SOFTWARE IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION ANY WARRANTY WITH RESPECT TO ITS MERCHANTABILITY OR ITS FITNESS FOR ANY PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE APPLE SOFTWARE IS WITH YOU. SHOULD THE APPLE SOFTWARE PROVE DEFECTIVE, YOU (AND NOT APPLE OR AN APPLE AUTHORIZED REPRESENTATIVE) ASSUME THE ENTIRE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

APPLE DOES NOT WARRANT THAT THE FUNCTIONS CONTAINED IN THE APPLE SOFTWARE WILL MEET YOUR REQUIREMENTS OR THAT THE OPERATION OF THE APPLE SOFTWARE WILL BE UNINTERRUPTED OR ERROR FREE OR THAT DEFECTS IN THE APPLE SOFTWARE WILL BE CORRECTED.

SOME STATES DO NOT ALLOW THE EXCLUSION OF IMPLIED WARRANTIES, SO THE ABOVE EXCLUSION MAY NOT APPLY TO YOU. THIS WARRANTY GIVES YOU SPECIFIC LEGAL RIGHTS AND YOU MAY ALSO HAVE OTHER RIGHTS WHICH VARY FROM STATE TO STATE.

7. Limitation of Remedies and Damages. IN NO EVENT WILL APPLE BE LIABLE (i) TO YOU FOR ANY INCIDENTAL, CONSEQUENTIAL OR INDIRECT DAMAGES (INCLUDING DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION, AND THE LIKE) ARISING OUT OF THE USE OF OR INABILITY TO USE ANY APPLE SOFTWARE EVEN IF APPLE OR ANY AUTHORIZED APPLE REPRESENTATIVE HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES, OR (ii) FOR ANY CLAIM BY ANY OTHER PARTY.

BECAUSE SOME STATES DO NOT ALLOW THE EXCLUSION OR LIMITATION OF LIABILITY FOR CONSEQUENTIAL OR INCIDENTAL DAMAGES, THE ABOVE LIMITATIONS MAY NOT APPLY TO YOU.

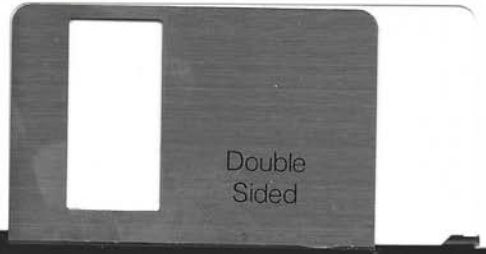
Apple's liability to you for actual damages for any cause whatsoever, and regardless of the form of the action, will be limited to the greater of $50 or the money paid for the Apple Software that caused the damages or that is the subject matter of, or is directly related to, the cause of action.

8. General. This Agreement will be construed under the laws of the state of California, except for that body of law dealing with conflicts of law. If any provision of this Agreement shall be held by a court of competent jurisdiction to be contrary to law, that provision will be enforced to the maximum extent permissible, and the remaining provisions of this Agreement will remain in full force and effect.

3/9/87

MacTCP™ Evaluation Kit
Sample Distribution Disk
Version 1.0; Disk 2 of 2
APDA # M0230LL/A

APDA™
© Apple Computer, Inc. 1987-89 All rights reserved.



MacTCP™ Evaluation Kit
Developer Disk
Version 1.0; Disk 1 of 2
APDA # M0230LL/A

APDA™
© Apple Computer, Inc. 1987-89 All rights reserved.







MacTCP™ v.1.0, Evaluation Kit

with Interim Man

M0260LL/A

This package contains:

2 Manuals:

Set of Release Notes

2 Disks

2 1-inch Binders

2 1-inch Binder Co

2 1-inch Spine I

If you have any que
1-800-282
1-408-56
1-800-e

Apple Programmers and
Developers Association

Dear MacTCP™ Customer:

Thank you for your interest in MacTCP. Enclosed is your purchase.

Please note that your product may have a different part number printed on it
the number you saw on a data sheet or read in the APDAlog. We needed to
the numbering system late in the production cycle for inventory purposes,
chose to ship the product anyway rather than cause delays in delivery. Plea
assured that you are in fact receiving the correct product you ordered.

Again, thank you for your support and for your patience. I hope this did not
you any undue confusion. If you have any questions, please call us toll-free
1-800-282-2732 or write us at:

APDA Customer Service
20525 Mariani Avenue, M/S 33G
Cupertino, California 95014.

Sincerely,

MacTCP™ Evaluation Kit
Sample Distribution Disk
Version 1.0, Disk 2 of 2

MacTCP™ Evaluation Kit
Developer Disk
Version 1.0, Disk 1 of 2

APDA™

Double Sided

MacTCP™ v.1.0,
Administrator's Guide

APDA # M0260LL/A

Apple Programmer
Developers Associa

Dear MacTCP™ C

Thank you for yo

Please note that y
the number you s
the numbering sy
chose to ship the
assured that you

Again, thank you
you any undue c
1-800-282-2732 or
AP
20
Cu

Sincerely,

THE APPLE PUBLISHING SYSTEM

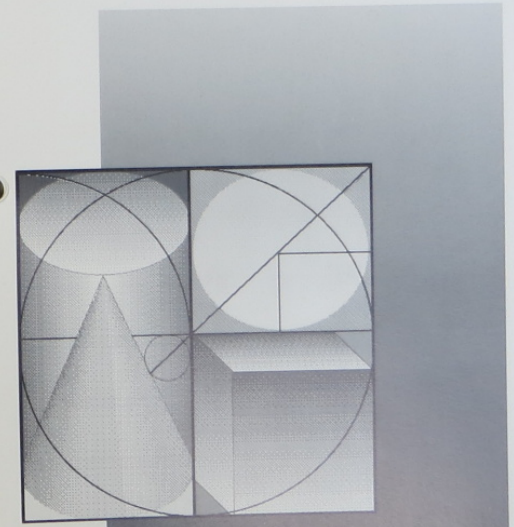
This Apple® manual was written, edited, and composed on a desktop publishing system using Apple Macintosh® computers and Microsoft® Word. Proof pages were created on the Apple LaserWriter® printers; final pages were printed on a Varityper® VT600™. Line art was created using Adobe Illustrator™. POSTSCRIPT®, the LaserWriter page-description language, was developed by Adobe Systems Incorporated.

Text type and display type are Apple's corporate font, a condensed version of Garamond. Bullets are ITC Zapf Dingbats®. Some elements, such as program listings, are set in Apple Courier, a fixed-width font.

MacTCP™ v.1.0, Programmer's Guide

with Interim Manual

APDA # M0260LL/A



MacTCP™ v.1.0, Evaluation Kit

with Interim Manual

APDA# M0230LL/A

MacTCP™ v.1.0, Evaluation Kit

with Interim Manual

APDA# M0230LL/A

