

alpha micro
P R O D U C T S

alpha micro
P R O D U C T S

alpha micro
P R O D U C T S

alpha micro
P R O D U C T S

alpha micro
P R O D U C T S

alpha micro
P R O D U C T S

alpha micro
P R O D U C T S

alpha micro
P R O D U C T S

alpha micro
P R O D U C T S

alpha micro
P R O D U C T S

alpha micro
P R O D U C T S

AlphaTCP® Administrator's Guide

© 2002 Alpha Micro Products

REVISIONS INCORPORATED	
REVISION	DATE
00	April 1993
01	January 1994
02	February 1995
03	July 1995
04	December 1995
05	September 1996
06	November 1996
07	May 1997
08	November 1977
09	March 1999
10	September 1999
11	April 2000
12	September 2002

AlphaTCP Administrator's Guide

To re-order this document, request part number DSO-00187-00.

This document applies to AlphaTCP version 1.5A and later.

The information contained in this manual is believed to be accurate and reliable. However, no responsibility for the accuracy, completeness or use of this information is assumed by Alpha Micro Products.

The following are registered trademarks of Alpha Micro Products, Irvine, CA 92614:

AMIGOS	AMOS	Alpha Micro	AlphaACCOUNTING
AlphaBASIC	AlphaCALC	AlphaCOBOL	AlphaDDE
AlphaFORTRAN 77	AlphaLAN	AlphaLEDGER	AlphaMAIL
AlphaMATE	AlphaNET	AlphaPASCAL	AlphaRJE
AlphaTCP	AlphaWRITE	OmniBASIC	VER-A-TEL

The following are trademarks of Alpha Micro Products, Irvine, CA 92614:

AlphaBASIC PLUS	AlphaVUE	AM-PC	AMTEC
AlphaDDE	AlphaCONNECT	DART	<i>inSight/am</i>
<i>inFront/am</i>	ESP	MULTI	

All other copyrights and trademarks are the property of their respective holders.

ALPHA MICRO PRODUCTS
17534 Von Karman
Irvine, CA 92614

Table of Contents

CHAPTER 1 -INTRODUCTION	1-1
GRAPHICS CONVENTIONS	1-1
TECHNICAL BACKGROUND INFORMATION	1-3
Internet Addresses	1-3
Network Classes	1-4
Subnetting	1-4
TCP/IP Protocols	1-5
BIBLIOGRAPHY	1-7
CHAPTER 2 -DOWNLOADING THE SOFTWARE	2-1
INSTALLATION OVERVIEW	2-1
PREREQUISITES	2-1
Getting Your Own Internet Network Number	2-2
DOWNLOADING THE SOFTWARE	2-2
Verifying the Software	2-3
ENTERING THE PRODUCT INSTALLATION CODE	2-3
WHAT'S NEXT?	2-4
CHAPTER 3 -MODIFYING THE ALPHATCP CONFIGURATION FILES	3-1
DETERMINING MEMORY REQUIREMENTS	3-1
TCPEMU Memory	3-1
The Shared Memory Area	3-2
NETLOG Memory	3-3
User Memory	3-3
CONFIGURING THE ALPHATCP SETUP FILES	3-3
Configuration File (CONFIG.)	3-6
TCPEMU Startup File (GOTCP.CMD)	3-10
Hosts File (HOSTS.)	3-11
DNS Resolver File (RESOLV.)	3-12
Local Host Name File (MYNAME.)	3-12
Networks File (NETWRK.)	3-13
Services Definition File (SERVIC.)	3-13
TCP Ersatz File (TCPIP.ERZ)	3-13
Time Zone File (TIMZON.)	3-13
WHOIS Server File (WHOIS.)	3-14
TCP Line Printer Definition File (LPR.)	3-14
If You Are Using Ethernet	3-14
Second Ethernet	3-15

If You Are Using SLIP	3-16
If You Are Using a Print Server	3-17
To Make AMOS Printers Accessible	3-17
If You Are Using SMTP MAIL	3-18
If You Are Using POP3 Mail	3-19
If You Are Using the Web Server	3-20
If You Are Using AlphaNET Tunneling	3-20
WHAT'S NEXT?	3-21
CHAPTER 4 -MODIFYING THE SYSTEM INITIALIZATION FILE	4-1
CHANGES TO THE SYSTEM INITIALIZATION COMMAND FILE	4-1
TROUBLESHOOTING STARTUP PROBLEMS	4-5
TESTING THE INSTALLATION	4-5
WHAT'S NEXT?	4-6
CHAPTER 5 -ALPHATCP SUPPORT AND SERVER PROCESSES	5-1
NETWORK LOGGER (NETLOG)	5-1
Requirements	5-1
Operation	5-2
Runtime Control	5-2
Security	5-2
TCP EMULATOR (TCPEMU)	5-2
Requirements	5-2
Operation	5-2
Runtime Control	5-3
Security	5-3
SLIP SERVER (SLIPD)	5-3
Requirements	5-3
Operation	5-3
Customizing	5-3
Runtime Control	5-4
Security	5-4
TELNET SERVER (TELNETD)	5-4
Requirements	5-5
Operation	5-5
Customizing	5-5
Telnet Source Host Information	5-7
Security	5-8
FTP SERVER (FTPD)	5-8
Requirements	5-8
Operation	5-9
Customizing	5-9
Security	5-11
Tree View	5-12
MAIL SERVER (SMTPD)	5-13

Requirements	5-13
Operation	5-13
Customizing	5-13
Kill File	5-17
Runtime Control	5-17
Security	5-17
POST OFFICE AND PASSWORD SERVERS (POP3D AND POPPWD)	5-18
Requirements	5-18
Operation	5-18
Customizing	5-19
Runtime Control	5-20
Security	5-20
LINE PRINTER CLIENT (LPR)	5-20
Requirements	5-20
Operation	5-21
Customizing	5-21
Printer Information	5-22
Security	5-23
LINE PRINTER SERVER (LPD)	5-23
Requirements	5-23
Operation	5-23
Customizing	5-24
Security	5-24
WEB SERVER (HTTPD)	5-24
Requirements	5-24
Operation	5-25
Default Documents	5-26
Special File Requests	5-26
Customizing	5-26
Specialized Filename mapping	5-27
Page Redirection	5-28
Publicly Accessible Devices	5-28
Runtime Control	5-28
Security	5-28
ITC TUNNELING SERVER (ITCD)	5-28
Requirements	5-29
Operation	5-29
Customizing	5-29
Replacing SerialNET Installations	5-30
Accessing Non-Tunneled AlphaNET	5-30
Runtime Control	5-31
Security	5-31
E-MAIL BASED FILE SERVER (BBSERV)	5-32
Requirements	5-32
Operation	5-32
Customizing	5-33
Security	5-33
TFTP SERVER (TFTPD)	5-33
Requirements	5-34

Operation	5-34
Security	5-34
RWHO SERVER (RWHOD)	5-34
Requirements	5-35
Operation	5-35
Runtime Control	5-35
Security	5-35
TAME SERVER (TAMED)	5-35
Requirements	5-35
Operation	5-36
Customizing	5-36
Security	5-36
SIMPLE NETWORK TIME PROTOCOL SERVER (SNTPD)	5-36
Requirements	5-36
Operation	5-37
Customizing	5-37
Security	5-37
BOOTP SERVER (BOOTPD)	5-38
Requirements	5-38
Operation	5-38
Customizing	5-39
Security	5-39
BOOTPC	5-39
Requirements	5-39
Operation	5-39
Customizing	5-40
Security	5-40
TDDD	5-40
Requirements	5-40
Operation	5-40
Customizing	5-40
Runtime Control	5-41
Security	5-41
Device Sharing	5-42
IPFW	5-42
Requirements	5-42
Operation	5-43
Customizing	5-43
Runtime Control	5-44
Security	5-44

CHAPTER 6 -ADMINISTRATIVE COMMANDS **6-1**

LOGCTL	6-2
LOGFMT	6-3
NETSTA	6-3
NETSTA Output	6-4
ROUTE	6-6

ARP	6-7
PS	6-8
UKILL	6-8
STARTD	6-9
NSLOOK	6-10
Non-Interactive Mode	6-10
Interactive Mode	6-10
Interactive Mode Commands	6-10
Error Messages	6-14
TCPMON	6-14
Understanding Packets	6-15
CVTNID	6-16
FNU2A	6-17
FTPLOG	6-17
TNWRAP	6-17
WHOIS	6-18
KLPR	6-18
IPCFG	6-18
AMPM	6-19
Synchronizing With an AMPM Package Repository	6-19
Running an AMPM Package Repository	6-20
AMPC Package Index	6-22
TROUTE	6-22
Successful Traces	6-23
Failed Traces	6-23
CHAPTER 7 -SLIP DIALUP SUPPORT	7-1
SERVER SCRIPT PROCESSING	7-1
SCRIPT CREATION	7-2
THE SLIP LOG FILE	7-3
SLIP CABLING	7-4
MODEM CONFIGURATION	7-4
CHAPTER 8 -TUNING ALPHATCP PERFORMANCE	8-1
IPCINI COMMAND	8-1
TCPEMU MEMORY	8-1
CONFIG. FILE	8-2
Ifconfig Setup	8-2
TCPEMU.MBD FILE	8-2
Determining Allocation	8-3
Determining Usage	8-3
CHAPTER 9 -ADVANCED WEB SERVER USAGE	9-1

HOW WEB DOCUMENTS ARE INDEXED	9-1
Absolute and Relative URLs	9-1
Same System, Different Directory	9-2
How AMOS Handles URLs	9-2
SUPPORTING IMAGE MAPS	9-3
SUPPORTING HIT COUNTERS	9-4
SUPPORTING FILL-OUT FORMS	9-5
Always Relative to Root	9-5
Application Definition File	9-6
UNIX and AMOS Differences	9-6
Filenames Appended to a URL	9-7
GET Method Arguments	9-8
POST Method Arguments	9-8
Supplying MIME Headers	9-8
Command Line Meta-Characters	9-9
PUT AND DELETE METHODS	9-9
SUPPORTING COOKIES	9-9
SUPPORTING AUTHORIZATION	9-10
SUPPORTING VIRTUAL DOMAINS	9-10
SAMPLE APPLICATIONS	9-11
SYSTAT Demo	9-11
BASIC Demo	9-12
<u>APPENDIX A -NETLOG ERROR MESSAGES</u>	A-1
<u>APPENDIX B -INSTALLATION AND CONFIGURATION WORKSHEETS</u>	B-1
<u>APPENDIX C -MIME TYPES FILE</u>	C-1
<u>APPENDIX D -BBSERV BBS INTERFACE</u>	D-1
TOPICS FILE	D-1
TOPIC RETRIEVAL	D-1
TOPIC POSTING	D-1
TOPIC SUBSCRIPTIONS	D-2
<u>APPENDIX E -APPLICATION TERMINATION</u>	E-1
TERMINATING APPLICATIONS ON VIRTUAL TERMINALS	E-1
Exit Request Generation	E-1
Handling Termination Requests in Assembler	E-2
Handling Termination Requests in AlphaBASIC	E-2

DOCUMENT HISTORY

INDEX

Chapter 1 - Introduction

This manual is for the AlphaTCP administrator—the person responsible for installing AlphaTCP and keeping it running properly. It includes the following information:




- Chapter 1 contains background information on the basic concepts of networking with TCP/IP, as well as a bibliography of books we found helpful.
- Chapters 2 through 4 contain information on installing AlphaTCP—downloading the software, configuring AlphaTCP setup files, and modifying the AMOS system initialization command file.
- Chapter 5 discusses the AlphaTCP emulator (TCPEMU). The TCPEMU process runs as a background task and spawns the servers necessary for the various AlphaTCP connections. This chapter also discusses the various servers supported by AlphaTCP, such as FTP, SMTP, POP3, LP, and HTTP.
- Chapter 6 describes the AlphaTCP administration commands. These commands let you set up and display network routing tables, check network status, and more.
- Chapter 7 covers using SLIP (Serial Line Interface Protocol) to make serial line (modem) connections.
- Chapter 8 discusses fine-tuning your configuration for optimum performance.
- Chapter 9 describes handling fill-out forms, image maps, virtual domains and cookies when using the web server.
- Appendix A lists the files included with AlphaTCP.
- Appendix B contains a set of worksheets you will find helpful when installing and configuring AlphaTCP. You can photocopy these worksheets for use whenever you need to set up a server.
- Appendix C discusses the Multipurpose Internet Mail Extensions (MIME) types file.
- Appendix D describes the BBSERV interface, allowing you to develop an electronic mail interface to a BBS or database running on the system.

For directions on the everyday use of AlphaTCP, see the *AlphaTCP User's Guide*, which describes how you can transfer files and set up virtual terminal connections between computers using AlphaTCP. That manual also lists the AlphaTCP error messages and includes some background material about networking, TCP/IP, AMOS, and UNIX.

GRAPHICS CONVENTIONS

This manual conforms to other Alpha Micro publications in its use of a standard set of graphics conventions. We hope these conventions simplify our examples and make them easier for you to use.

Symbol	Meaning
filespec	<p>An AMOS file specification that identifies a specific file within an account. A complete filespec for the local computer is made up of the device name, the file name, the file extension, and the account number. For example: DSK0:SYSTEM.INI[1,4].</p> <p>A file specification may also consist of an ersatz name, which specifies a particular disk account, and a file name, like this: TCP:NETLOG.LST.</p> <p>A file specification for a remote computer includes the host name of the remote computer. The exact format depends on the network protocol.</p>
[p,pn]	This abbreviation represents an account on a disk where you can store files and data. An actual disk account number looks like this: [100,2] or [1,4].
{ }	Optional parts of a command appear in braces. You can enter exactly what is in the braces or substitute the correct value if it's a parameter. Do not include the braces. For example: DIR { / <i>switch</i> } indicates / <i>switch</i> is not required.
TEXT	This bold typeface represents characters you type. Since many AlphaTCP commands are case-sensitive, the text may be either upper or lower case. Variable parts of the entry are in <i>italics</i> , as noted below.
<i>text</i>	We use this <i>bold italic</i> type for variable parts of command examples. Replace the text shown with the appropriate entry.
TEXT	Text in this TYPEFACE is used for characters the computer displays on your screen, command syntax, host names, and program examples. In command syntax, variable parts of the command line are in italics, as noted below.
<i>text</i>	This <i>italic</i> typeface indicates a variable part of a command line or program syntax.
<i>parameter</i>	Following a command line or program format sample, the first reference to each parameter in the sample is in this <i>bold italic</i> text. Succeeding references are not, unless they would be unclear otherwise.
COMMAND	AMOS command names and file names are given in all capitals. Most AMOS commands are not case-sensitive. However, some AlphaTCP commands on AMOS, such as TELNET, must be entered in lower case.
command	UNIX commands, directory paths, and file names are given in bold text. All UNIX commands and file names are case-sensitive, so they are given in the proper case (normally lower case). Since they are based on corresponding UNIX commands, the commands you use while "inside" AlphaTCP, such as the various FTP commands, are given in lower case bold.
RETURN	The key symbol indicates a reference to a key on your keyboard. The name of the key appears inside the key symbol.
CTRL -C	This indicates a control sequence you press on the keyboard. Press CTRL and hold it down while you press the indicated key.

Symbol	Meaning
^	When displayed before a capital letter, this means the letter is a “control character.” For example, when you press CTRL -C, it appears on your screen as ^C (^C is the control character that cancels most programs and returns you to AMOS command level).
	This symbol indicates an important note or warning you should read carefully before going further in the documentation. Usually, text next to this symbol contains instructions for something you <i>must</i> or <i>must not</i> do, so read it carefully.
	This symbol indicates a helpful bit of information, or a “short cut” that could save you time or trouble.
	This symbol indicates something you should keep in mind while following a set of instructions.

TECHNICAL BACKGROUND INFORMATION

This section covers the basic concepts of networking with TCP/IP. In order to install this package properly, you must understand these concepts and configure the software to fit your environment.

Internet Addresses

Each host on the network has its own address, unique to that computer. This is how data packets are delivered to the proper computer. Whenever you add another computer to the network, you must set it up with its own address.

Internet addresses are separate from Ethernet or AlphaNET addresses. Since TCP/IP is an internet protocol, it needs an address format which is consistent across differing network types.

The internet address is 32 bits (4 octets) long. It is usually displayed as four decimal numbers between 0 and 255, separated by periods, like this: 101 . 14 . 0 . 23. The first one, two or three numbers, depending on the class of the network (see below) represent the network. This is important if you are connected to the Internet. The rest of the address is the node number.

The network number portion of the internet address should be the same for all nodes on the local network. If you are connected to the Internet, use the network number assigned to you. If you have an isolated local network, any number of the proper format will do.



The node number portion of the internet address must be unique for each node on the local network. It *cannot* be all 0s or all 255s.

Gateway computers (those attached to two or more different networks) have separate internet addresses for each network they are connected to.



On some TCP/IP implementations, including AlphaTCP, interpretation of IP addresses may be performed in hex or octal as well as decimal. If any of the values is preceded with *0*, that value is interpreted as an octal number. If the value is preceded with *0x*, it is interpreted as a hexadecimal number. This applies to addresses entered from the keyboard, and those stored in configuration files. It also applies to subnet masks, which may be easier to understand in hexadecimal.

Network Classes

There are three primary network classes: A, B, and C. They differ in how much of the internet address makes up the network number and how much makes up the node number. There are relatively few class A networks and each network can have a large number of nodes, so class A networks have 8-bit network numbers. Class B networks have 16-bit network numbers. Class C networks have 24-bit network numbers, allowing a great many networks but relatively few nodes per network.

The first number of the internet address determines the class of network. Class A addresses are from 1 to 127, class B addresses from 128 to 191, and Class C addresses from 192 to 223. Numbers above 223 are reserved for other specialized classes and broadcast addresses. Selection of a specific class is determined by the maximum number of nodes you expect to have on a network. Class A addresses are reserved for very large institutions. For example:

- 67 . 0 . 3 . 25 is a class A address. 67 is the network number; the rest of the address specifies the node.
- 154 . 1 . 0 . 77 is a class B address. 154 . 1 is the network number; 0 . 77 is the node number.
- 201 . 3 . 18 . 205 is a class C address. 201 . 3 . 18 is the network number; the node number is 205.

Subnetting

Dividing a single internet network address into multiple physical networks is known as subnetting. A computer with multiple physical network connections acts as a gateway between the two networks, sending to each subnetwork messages addressed to its group of nodes. Each class of IP address has a default *net mask* which defines which bits specify network and which specify host. These are built into the TCP stack. A *subnet mask* overrides this default to let you redefine the boundary and use some of the host bits as additional network bits for your own subnet.

You can set up subnetting by defining subnet masks in the NETWRK. file. You can define routes between subnets using the ROUTE command. See Chapters 3 and 6.

As explained previously, a class A network uses the first byte as the network number, while a class B uses the first two bytes, and a class C uses the first three. The following values could be used as subnet masks for class A, B, and C, with no effect because the boundaries match the defaults:

- 255.0.0.0 Class A default mask
- 255.255.0.0 Class B default mask
- 255.255.255.0 Class C default mask

Using the default, the class C address 192.168.1.* is a single network with valid hosts in the range 1-254 (the all-zero and all-one addresses .0 and .255 cannot be used). It is helpful to understand that although the mask is written as decimal values, it is actually used as bits. Thus, you will usually set the desired bits as binary on paper and convert to decimal.

Now assume we need to break this into two different networks. The same rule about all-zero and all-one applies to subnetworks, for historical reasons. Since these two subnetworks cannot be used, we need to break the address into four subnetworks, two of them usable. This is done by specifying that the upper two bits of the host area should be used for the network number (decimal 128+64):

- 255.255.255.192 Class C with 4 subnets (2 usable)

This gives us four possible subnets. In binary, this is 00, 01, 10, and 11. Six bits to the right remain for the host. When you see a full IP address for a host on these subnets, realize that part of the byte defining the host also defines the subnet. In other words, 192.168.1.67 is really binary 0100 0011, or the third host on the '01' subnet. Further, 192.168.1.191 is binary 1011 1110, or the last usable host on the '10' subnet.

Host numbers 65-126 are on the first usable subnet, while host numbers 129-190 are on the second usable subnet. Host numbers 1-62 and 193-254 cannot be used because they would be on the all-zero and all-one subnets. The numbers 0, 63, 64, 127, 128, 191, 192, and 255 also cannot be used because they would be all-zero or all-one host numbers. Viewing these numbers in binary on paper should make this more obvious, especially with a bar between the network and host portions. There must be at least one 0 and one 1 on each side of the bar to be a valid host number.

TCP/IP Protocols

This section contains information on the TCP/IP family of protocols AlphaTCP is based on. Within the software, network protocols provide rules for the structure of the packets transported across the network, including header and data format. Addresses and control information are included. The suite of TCP/IP protocols lets user programs establish, use, and end sequenced, error-free connections. A brief overview of these protocols is given below.

Protocol	Function
ARP	The Address Resolution Protocol translates internet addresses into physical network addresses. The computer supports time-out of ARP information and also collects ARP information from request packets targeted at other stations—those not addressed to your computer.
BOOTP	The BOOTP protocol allows diskless hosts to retrieve their network parameters, including an optional boot file name, from a central server.
DHCP	An extension to the BOOTP protocol allowing temporary allocation of IP addresses
DNS	The Domain Name Service is a distributed database of host names and addresses. It replaces the HOSTS. file in large installations. AlphaTCP supports the Resolver portion of DNS.
HTTP	The HyperText Transfer Protocol provides the transport mechanism for delivering documents on the World Wide Web.
FTP	The File Transfer Protocol lets users on computers running under different operating systems access files. The underlying protocol layer is TCP/IP. The user interface is the program FTP, described in Chapter 2 of the <i>AlphaTCP User's Guide</i> . FTP offers a variety of services beyond simple file transfer, including: working with the directory on the remote computer, transferring groups of files, appending a local file to a file on the remote host, displaying FTP status, and so on. FTP requires you to have a login on the remote computer.
ICMP	The Internet Control Message Protocol, an adjunct to IP, provides a way to transmit and receive datagram error and control messages between the local host and remote hosts or gateways. For example, if a gateway's buffer is too full to forward a datagram, ICMP messages are sent.
IP	The Internet Protocol provides a simple host-to-host datagram service across a network. IP can deal with datagrams which are too long for the network by fragmenting them into datagrams of an acceptable size. Incoming datagram fragments are reassembled into whole datagrams.
LPR	The Line Printer protocol is used to transfer print data to a host with a printer attached. AlphaTCP supports the client side of the Line Printer protocol.
NTP	The Network Time Protocol is used to synchronize system clocks with very high precision. The SNTP server uses a subset of the full NTP protocol.
POP3	The Post Office Protocol handles the retrieval of mail by hosts which are usually not on the network or have limited memory resources.
SLIP	The Serial Line Internet Protocol lets you use the other TCP/IP protocols over hardwired serial lines or phone lines. Most SLIP implementations provide no direct modem support, but AlphaTCP has been enhanced to use script files to define the steps needed for automatic modem connections.

Protocol	Function
SMTP	The Simple Mail Transfer Protocol is used to pass electronic mail messages between hosts.
TCP	The Transmission Control Protocol is used for the transport level connection between your computer and the other computer. TCP gets the data to the right process on the right computer as a guaranteed reliable byte stream.
TELNET	TELNET provides virtual terminal connection between two computers. The systems can be of different types and operating systems, as long as both follow the TELNET protocol. The underlying protocol is TCP/IP; the user interface is the program TELNET, described in Chapter 3 of the <i>AlphaTCP User's Guide</i> .
TFTP	Like FTP, the Trivial File Transfer Protocol allows different operating systems to exchange data. The underlying protocol layer is UDP/IP. The user interface is the program TFTP, described in Chapter 2 of the <i>AlphaTCP User's Guide</i> . TFTP operates much like FTP, but is simpler to use because it doesn't offer as many options. It does not require you to have a login on the remote computer, and is limited to transferring files one at a time. It is often used to access databases that do not require user authentication. File transfer automatically translates the file into a format understood by the receiving computer. (That doesn't mean Excel spreadsheets automatically become AlphaCALC spreadsheets, but it does mean that a UNIX text file is transferred to an AMOS computer in AMOS file format.)
UDP	The User Datagram Protocol provides a simple, connectionless, non-guaranteed packet delivery service. It is used for file transfer using TFTP. When using UDP, it is up to the higher levels to provide reliability.

BIBLIOGRAPHY

Here is a list of some of the books that discuss TCP/IP and networking concepts that we have found helpful:

Albitz, Paul and Liu, C. *DNS and BIND in a Nutshell*. Sebastapol, CA: O'Reilly & Associates, 1992. ISBN 1-56592-010-4

All about the administration of the Domain Name Service and the most-often encountered server implementation, Berkeley Internet Name Domain (BIND). A must if you will be supporting a DNS server and changing name, address, and mail exchanger records.

Comer, Douglas E. *The Internet Book*. Prentice-Hall, Inc., 1994. ISBN 1-56276-192-7

Very good introduction to the Internet and how the protocols and applications work. More advanced and detailed than *How the Internet Works*.

Comer, Douglas E. and Steven, David L. *Internetworking with TCP/IP, Volumes 1-3*. Prentice-Hall, Inc., 1993. ISBN 0-13-474222-2

Three books that start with an overview of the TCP/IP protocol stack, then detail the algorithms involved.

Eddings. *How the Internet Works*. ZD Press, 1994. ISBN 0-13-151565-9

A very basic introduction to the Internet and how it all fits together. Many simple illustrations.

Ford, Andrew. *Spinning The Web*. International Thompson Publishing, 1995. ISBN 0-442-01996-3

An extensive and well written book on every aspect of providing information and services via the World Wide Web.

Hahn, Harley and Stout, Rick. *The Internet Yellow Pages*. Osborne-McGraw-Hill, 1994. ISBN 0-07-882023-5

Extensive listing of services to be found on the Internet.

Lowe, Doug. *Networking for Dummies*. IDG Books Worldwide, Inc., 1994. ISBN 1-56884-079-9

This book contains a friendly discussion of PC-oriented network basics.

Miller, Mark A. *Troubleshooting TCP/IP: Analyzing the Protocols of the Internet*. M&T Books, 1992. ISBN 1-55851-268-3

Many packet traces and descriptions intended to aid in the isolation and resolution of TCP/IP Networking problems.

Stevens, W. Richard. *TCP/IP Illustrated, Volume 1: The Protocols*. Addison-Wesley Publishing Co., 1993. ISBN 0-201-63346-9

Highly detailed description of the TCP/IP protocol and applications. Many packet traces and descriptions of most of the quirks of TCP/IP. The author participates in the comp.protocols.tcp-ip USENET newsgroup and is often found there answering TCP/IP questions.

Wiggins, Richard W. *The Internet for Everyone: A Guide for Users and Providers*. McGraw-Hill, 1995. ISBN: 0-07-067018-8

A detailed discussion of many of the concepts introduced in this manual, such as host names, telnet, assignment of IP addresses, etc.

Chapter 2 - Downloading the Software

This chapter describes the first step in installing AlphaTCP—downloading the software:

- Determining if your computer meets AlphaTCP's prerequisites
- Downloading the software
- Verifying the software downloaded correctly
- Entering the Product Installation Code (PIC)

INSTALLATION OVERVIEW

The AlphaTCP installation consists of three basic steps:

1. Downloading the software.
 - Determining if your computer meets AlphaTCP's prerequisites
 - Downloading the software
 - Verifying the software downloaded correctly
 - Entering the Product Installation Code (PIC)
2. Modifying the AlphaTCP configuration files.
 - Determining the memory requirements of the servers you are going to use
 - Modifying the AlphaTCP configuration files
3. Modifying the AMOS system initialization file.
 - Modifying your system initialization command file
 - Testing your network setup
 - Fine-tuning AlphaTCP performance by loading files into system memory

All three of these steps must be completed before AlphaTCP is up and running. Additional customization procedures for specific servers are found in Chapter 5, "AlphaTCP Support and Server Processes."

PREREQUISITES

Before you install AlphaTCP, make sure your computer has all of the software and hardware it needs to use this product. See the Release Notes for your version of AlphaTCP for a complete list of these requirements.

AlphaTCP requires a minimum of about 1MB of memory. We suggest you have 2MB or more if you use the TELNED and FTPD servers. The exact amount of memory required depends on what AlphaTCP features you use. There is a detailed discussion of memory requirements in the next chapter.



Processes such as TELNET and FTPD are called daemons under UNIX. Under AMOS, they are known as servers, and they are referred to as such throughout this manual. See Chapter 5 for more information on these servers.

Any computers you wish to TELNET into should have definitions for the type of terminal you are using on AMOS.

Getting Your Own Internet Network Number

With the explosive growth of the Internet over the past few years, the method of assigning network numbers has changed. Internet service providers are now assigned blocks of addresses by the Internet Network Information Center (InterNIC, or NIC for short). When you sign up with a provider, they will assign you a group of addresses from their pool. Unless your provider turns those addresses over to you, you will have to change the addresses on all your systems when you change providers. Choose your provider very carefully. If you have a large network, you may have quite a bit of work to do if you change.

If you won't be connecting to the Internet, you may use one of the following network numbers:

10 . * . * . *	Class A Network
172 . 16 . * . *	Class B Network
192 . 168 . 1 . *	Class C Network

The above addresses are a subnet-free set of the ones referenced by Request For Comments (RFC) number 1597. If you decide later to connect to the Internet, you will need to change your hosts to use an officially assigned range of addresses. Never allow these addresses to be routed to the Internet. RFC number 1627 describes the disadvantages of using these unofficial numbers in greater detail.

DOWNLOADING THE SOFTWARE

You can install part of the AlphaTCP software on any disk device, though some files should be on DSK0: for computer-wide access. The files which should be on DSK0: take up about 2600 blocks; those which can be on any device occupy about 3100 blocks.

To copy the AlphaTCP software to your hard disk, use the commands appropriate for your release medium. For example, if you received the software on AlphaCD, follow the instructions in the CD-ROM case; if you received AlphaTCP on streaming tape, use the MTURES command, and so on.

The AlphaTCP files are in their proper accounts on the release medium.

If you are installing all the software on DSK0:, you can LOG to OPR: and copy all the files directly to their proper accounts on DSK0:.

If you are installing the software on a disk other than DSK0:, log to the disk account where you want to install the majority of the software (usually [7,50] on the desired disk device), and copy all the files from [7,50] on the release medium to that account. All other files should be copied to DSK0:.

Verifying the Software

Before continuing with the installation, you should verify that all the files were copied to your disk correctly. To do so:

```
LOG OPR: [RETURN]
VERIFY TCP [RETURN]
```



If you installed the software somewhere other than DSK0:[7,50], you need to edit the OPR:TCP.DIR file and change the device name and/or account number for the [7,50] files to the correct one before you run VERIFY. When changing this information, be sure not to move the version or hash code information, since VERIFY will not work properly if this information is not in the correct columns.

If you receive any error messages, download the software again. If VERIFY still shows an error, contact your Alpha Micro dealer.

ENTERING THE PRODUCT INSTALLATION CODE



If you are installing AlphaTCP on a system running AMOS 2.3 or later, *skip this step*. Once you have entered the PIC for AMOS itself, AlphaTCP does not need to be encoded separately.

Like most Alpha Micro software products, AlphaTCP is keyed to the Software Security Device (SSD) in your computer, and requires you to enter the correct Product Installation Code (PIC) for your SSD before you can use the software.



Be sure your AlphaTCP PIC is for the same number of users as your AMOS license. Your AlphaTCP license should match the number of licensed AMOS users, not the anticipated number of AlphaTCP users.

Once you've verified the software has been downloaded correctly, follow this procedure to install the PIC:

1. Make sure your job has at least a 600KB memory partition. You cannot run the PIC installation program if you have less. Also, make sure the job you are using is *not* named TCPEMU.
2. Log to the account where you installed the AlphaTCP release (usually [7,50]):

```
LOG devn:[7,50] [RETURN]
```

3. Type this command:

```
TCPEMU [RETURN]
```

A screen will display requesting the PIC. As when entering all PICs, as long as this screen is displayed, other users on your computer are suspended. Type in your PIC, being sure to use the correct capitalization and including all punctuation marks. After your computer processes the PIC, you return to the AMOS prompt.



Do not continue with the installation until you've entered your PIC. If you configure AlphaTCP and modify your system initialization file without having entered the proper PIC, your computer will not boot properly!

WHAT'S NEXT?

You have completed the first step in installing AlphaTCP. Go on to the next chapter to find out how to set up the AlphaTCP configuration files.

Chapter 3 - Modifying the AlphaTCP Configuration Files

This chapter contains information on:

- Determining the memory requirements of the servers you are going to use.
- Modifying the AlphaTCP configuration files.

DETERMINING MEMORY REQUIREMENTS

The following sections discuss the four areas where AlphaTCP has specific memory requirements or may affect existing memory allocations. They are:

- The TCPEMU job. This determines the number of data paths, queued messages, and several other items which affect the performance of AlphaTCP.
- The shared memory area (SMEM). This is where AlphaTCP gets memory for jobs it must spawn to start servers or “fork” child processes. All virtual terminal sessions are assigned memory from here also, so you must decide how much memory you will allow users connected by TELNET.
- The NETLOG error logging job.
- User memory requirements may be affected by the memory needed to run the AlphaTCP client programs to make connections to other computers.

Of these areas, the first two are most involved and require the most memory.

TCPEMU Memory

The memory requirement for TCPEMU depends on many things, including the speed of your network and the amount of activity on it, the number of AlphaTCP connections you expect, and the number of data paths used (each FTP session often has two data paths).

TCPEMU requires a minimum 900KB of memory to run with a single user and no servers. You should add 20KB of memory for every expected connection, into or out of AMOS. If you have several TCP/IP computers on your network, you may want to increase your memory further.

The amount of memory you should assign to TCPEMU also depends on the quantity and makeup of traffic on your network, which is beyond the scope of this document. Once the AlphaTCP software is installed and operational, you can examine the data from the network logger (NETLOG). This data includes information about the memory layout AlphaTCP has created, as well as errors which may indicate poor performance due to insufficient memory. See Chapter 5 for further information on servers, and Chapter 8 for performance tuning.

The Shared Memory Area



AlphaTCP allocates memory for all its servers, and for the child processes created when a remote connection is made to your computer, from the AMOS shared memory area (SMEM). Also, all memory required by incoming virtual terminal (TELNET) sessions comes from SMEM. Place SMEM *immediately following* the last SYSTEM statement in your system initialization command file.

The AlphaTCP servers which use SMEM fall into two classes: those which use a fixed area for a single copy of themselves, and those which may need to “fork” one or more child processes. *When specifying JOB memory sizes, such as in TCP:CONFIG., use 6KB less than listed here.*

*ftpd	56KB
*lpr	41KB
*smtpd	131KB
*tftpd	41KB
*pop3d	66KB
poppwd	76KB
*lpd	41KB
*httpd	106KB
rwhod	81KB
slipd	91KB (for each SLIP interface)
telned	106KB
itcd	106KB
tamed	206KB
sntpd	91KB
bootpd	86KB
bootpc	86KB
tddd	66KB
ipfw	86KB

These values assume the programs with asterisks are loaded in system memory and don’t include the active “forked” children. For each connection expected, add the amount of memory listed for the server.



All of these memory requirements include 6KB overhead area required by AMOS for the job and terminal definitions. That is why the memory you specify per server in the TCP:CONFIG. file (as described later in this chapter), is 6KB less than shown in the table above.



Make sure CMDLIN.SYS is loaded into system memory.

After you’ve added up the requirements for all the servers, add the memory which will be needed by all incoming TELNET connections. Each connection will need at least 6KB overhead memory in addition to the job’s memory. TELNET connections will require more for terminal drivers (.TDVs). For example, a TELNET job using the PCI.TDV requires almost 9K of overhead. As described in Chapter 5, you can limit incoming connections to a specific amount of memory or a specific range of memory sizes.

The SMEM area for AlphaTCP is the total required for the servers you're starting, any child processes for the servers, and memory partitions for incoming TELNET connections. Make sure your SMEM area is large enough to handle the maximum number of simultaneous connections you expect.



If you are using AlphaNET version 2.4 or later, you may elect to have VTSER also use the SMEM area. See the *AlphaNET Administrator's Guide* for releases 2.4 or later.

NETLOG Memory

The network error logger requires 40K of memory to run.

User Memory

The client programs provided with AlphaTCP require the following minimum memory partitions. Users who want to make an outgoing connection must have at least this much memory in their partitions.

If the .LIT files are NOT in system memory, the requirements are:

ftp	155KB
tftp	124KB
telnet	124KB

If the .LIT files are in system memory, the requirements are:

ftp	77KB
tftp	59KB
telnet	63KB

The AlphaTCP maintenance programs are small enough that you should have no problem running any of them if you can run any of the client programs.

CONFIGURING THE ALPHATCP SETUP FILES

You now need to enter the information for your computer into the AlphaTCP configuration files. These files are discussed in this chapter. There are several files, all found in DSK0:[7,50], or whatever account you installed the software in:

File	Function
BOOTPD.	This is the configuration file used by the BOOTPD server. See Chapter 5.
CONFIG.	The CONFIG. configuration file tells AlphaTCP how to connect to the network and which servers to start. A CONFIG.NEW file is shipped with AlphaTCP and may be modified by you before being renamed to CONFIG.. Information on this file can be found both in this chapter and in Chapter 5.

File	Function
FTP.	The FTP. file contains the connection message for FTP. See Chapter 5 for information on this file.
FTPUSR.	The FTP users file lists the names, passwords, and permissions for all valid FTP users. See Chapter 5 for information on this file.
HOSTS.	The hosts file has a list of host names and corresponding internet addresses for all computers you may want to connect to. The HOSTS. file is not needed if you are using DNS (the Domain Name Service).
HTTPD.CVT HTTPD.HTM HTTPD.DEV	The HTTPD.CVT file provides a way to map long filenames which generate conflicting AMOS filenames. The HTTPD.HTM file is the default web document sent by the HTTPD server. It is sometimes known as the welcome or home page. The HTTPD.DEV file is used to list devices which have public access without requiring PUBLIC.PPN in every account.
ITCD.	The ITCD. file provides the mapping between AlphaNET CPU IDs and AlphaTCP IP addresses for the ITCD server.
LPR.	The line printer definition file lists TCP printer names along with their host and print queue definitions. An LPR.NEW file is shipped with AlphaTCP. You may modify this file, if setting up a TCP line printer, and rename it to LPR.. Information on this file can be found both in this chapter and in Chapter 5.
MIME.TYP	The MIME.TYP file provides mapping between AMOS file extensions and MIME content-type and content-encoding fields. It is placed in the LIB: account and is used by the HTTPD server. The SMTPD and POP3 servers may also use it in the future.
MYNAME.	The local host name file contains the fully qualified domain name for computers using DNS. (If you are upgrading from AlphaTCP 1.1 to later versions, note that this file is now very important.)
NETWRK.	The networks file contains the internet address of this computer and any sub-network definitions. A NETWRK.NEW file is shipped with AlphaTCP containing sample entries. You can edit NETWRK.NEW to reflect your own configuration and then rename it to NETWRK. during the AlphaTCP installation.
POPUSR.	The Post Office server file contains connection information for each user who will be using the post office. See “If You Are Using POP3 Mail” later in this chapter for information on this file.
RESOLV.	The RESOLV. file contains the list of name servers used when you are using Domain Name Service. A RESOLV.NEW file is shipped with AlphaTCP. You may rename it to RESOLV. during AlphaTCP installation and edit it for your configuration

File	Function
SERVIC.	The <code>SERVIC.</code> file contains a definition of the well-known port numbers associated with servers. A <code>SERVIC.NEW</code> file is shipped with AlphaTCP and you must rename it to <code>SERVIC.</code> during AlphaTCP installation. With each new AlphaTCP release, this file is updated with new services.
TCPEMU.MBD	This file alters the size distribution of the TCP/IP message blocks and is optional. See Chapter 8 for more information.
TDDD.	This file is used by the TCP Device Driver Daemon to map incoming TCP ports to non-filestructured devices (such as TRM), and names under the TDD: device to remote TCP ports.
TIMZON.	The <code>TIMZON.</code> file tells AlphaTCP the name of your time zone, along with the number of hours and minutes it is offset from universal time (GMT or UTC). See “Time Zone File (<code>TIMZON.</code>)” later in this chapter for information on the entry format. If you will be running the <code>SNTPD</code> time server you do not need to maintain this file as it will be created and adjusted for daylight saving time by <code>SNTPD</code> .
SMTPD.LOC SMTPD.ALI SMTPD.FOR SMTPD.FWD	These are configuration files used by the <code>SMTPD</code> server. See Chapter 5 for information on their contents.
SNTPD.	The <code>SNTPD.</code> file provides information needed by <code>SNTPD</code> to synchronize your system time and <code>TIMZON.</code> file with another host. See “If You Want to Synchronize Your System Time” later in this chapter for more information.
TCPIP.ERZ	The <code>TCPIP.ersatz</code> file defines the location of the ersatz device <code>TCP:.</code> A <code>TCPIP.NEW</code> file is shipped with AlphaTCP containing a sample ersatz definition, and may be renamed to <code>TCPIP.ERZ</code> during AlphaTCP installation. If you are not installing AlphaTCP in the default location of <code>DSK0:[7,50]</code> , edit <code>TCPIP.ERZ</code> to reflect the correct location. You may add additional entries when starting servers.
GOTCP.CMD	The <code>GOTCP.CMD</code> file is run by <code>TCPEMU</code> . This file must be edited to reflect your own computer’s configuration. A <code>GOTCP.NEW</code> file is shipped with AlphaTCP and may be renamed to <code>GOTCP.CMD</code> . (Remember to edit it for your computer first.)

Note that most of these files have null (`.`) extensions. If you use `DIR` to list these files, you will see only the file name. To edit these files using `AlphaVUE` or `AlphaXED`, you must include the period after the file name.

AlphaTCP includes prototypes for most of these files. The prototype files have the extension `.NEW` and need to be renamed if you decide to use them. The following sections describe how to modify the prototypes with the information you need, assuming you have renamed them to the proper extension. For instance, to rename a prototype `SERVIC.NEW` file to a null extension, enter:

```
LOG TCP: 
RENAME *. =SERVIC.NEW 
```

Configuration File (CONFIG.)

AlphaTCP reads the configuration file to determine how to connect to the local network interface and which servers to spawn. This file also allows you to change the time-out period TCP uses when closing a connection, as well as the time servers will have their connections dropped due to communication failure. The file name is CONFIG.. The prototype file, CONFIG.NEW, looks like this:

```
# attach the ethernet to the internet package (using AlphaNET NDVs):
# EXAMPLE: ifconfig <interface> <AlphaNet Network #> arpa <receive buffers>
#ifconfig ec0 101 arpa 100
# attach the ethernet to the internet package (using standalone LDVs):
# EXAMPLE: ifconfig <interface> <driver>
#ifconfig ec0 am319s
# attach the slip interface to a slip daemon:
# EXAMPLE: ifconfig <interface> <terminal> <dialup script>
#ifconfig asy0 modem1 dialup.slp
#ifconfig asy1 modem2
#ifconfig asy2 modem3
#ifconfig asy3 modem4
# change the tcp close timeout to fit the environment
# EXAMPLE: closetime <seconds>
#closetime 120
# change the tcp keepalive timeout to fit the environment
# EXAMPLE: keepalive <minutes>
#keepalive 15
# disable all packets unless ipfw (or tcpmon) is running
#mustfirewall
# start up the daemons (servers)
# EXAMPLE: start <daemon> <memory> <options...>
# (these assume ftpd, tftpd, smtpd, pop3d, lpr, lpd, and httpd are
# in system memory)
#
#start rwhod 75k
#start ftpd 50k -o
#start telned 100k
#start tftpd 35k
#start smtpd 125k -i
#start pop3d 60k
#start poppwd 70k
#start lpr 35k
#start lpd 35k
#start itcd 100k
#start httpd 100k
#start tamed 200k
#start sntpd 85k -b2 -p240
#start bootpd 80k
#start bootpc 80k -h -n -t
#start tddd 60k
# If you have a default router this is a simple way to set the default route.
# Fill in the IP address, once complete route will exit and free the memory.
# ** Don't start this if you are configuring with BOOTP or DHCP.
#start route 200k -s add xxx.xxx.xxx.xxx
```

All lines starting with # are comments.

The lines that start `ifconfig` tell AlphaTCP how to get to the network interfaces—`ec0` represents the Ethernet; `asy0` represents SLIP. For information about what changes to make to enable SLIP connections, see “If You Are Using SLIP Connections” later in this chapter. For information about what changes to make to enable Ethernet connections, see “If You Are Using Ethernet” later in this chapter.

To Change the TCP Close Time-out

The line starting `closetime` lets you change the TCP close time-out from its default of 600 seconds. The value on this line tells AlphaTCP how long to try to close a connection with data on it. If this period is exceeded, the connection is reset and all queued data is discarded. The period should be long enough to avoid discarding data on a working but slow path.

This value actually sets the maximum period of the `FIN_WAIT_2` state, where the connection has been closed locally but the remote end has not performed its close (or at least the close has not yet been received). You will not usually need to change this value.

To Change the TCP Keepalive Time-out

The line starting `keepalive` lets you change the TCP keepalive time-out from its default of thirty minutes. Most of the AlphaTCP servers enable keepalive polling, which causes AlphaTCP to test the data connection every two minutes when no other data is present. The value on this line tells AlphaTCP how long before a non-responding connection is terminated. The minimum value is four minutes. The value should be large enough to handle transient network and router failures, especially where background servers are involved.

Setting this value too low will result in connections being dropped due to temporary problems such as a modem redialing, a router rebooting, or a network cable being moved. It would be very annoying to have a long file transfer or mail delivery aborted simply because the keepalive time-out is too short to handle a temporary network failure.

If You Have Gateways



If you are configuring with DHCP or BOOTP, you may skip this section as the default gateway will automatically be configured.

You will need to perform `ROUTE` commands to define the gateways to the other networks. If there is a single gateway which handles all unknown addresses, this is specified on the `ROUTE` command with the `-s` smart gateway switch. The easiest way to perform a default `ROUTE` command is to add it to the `TCP:CONFIG.` file like a server. For example, adding the following line to `TCP:CONFIG.:`

```
start route 200k -s add 192.168.1.55
```

causes AlphaTCP to spawn a job executing the `ROUTE` command. After AlphaTCP is up, the `ROUTE` command will define 192.168.1.55 as the address to pass all unknown addresses to. It will then exit, having done its job, and cause the spawned job to disappear, thus freeing up the 200k. If you are using the example configuration file, the final line shows how this is set up.

If You Dynamically Configure Using BOOTP or DHCP

You need to start the BOOTP client, `BOOTPC`. Uncomment the `start bootpc` statement to enable it. After configuration, `BOOTPC` will exit unless it has acquired a temporary DHCP lease. If a temporary DHCP lease was acquired, `BOOTPC` will sleep until it is time to renew the lease.

Starting Servers

The lines beginning with `start` tell AlphaTCP to spawn the various servers used for accessing AMOS from remote computers, determine the memory allocated to each, and set any server specific options. Configuration information for most servers may be found in Chapter 5. The valid server names are:

Server	Function
<code>bbserv</code>	E-mail based file server. Requires configuration before enabling.
<code>bootpc</code>	The bootp/dhcp client. This allows AlphaTCP to automatically retrieve configuration parameters from a central server and to lease temporary IP addresses.
<code>bootpd</code>	Allows local storage of network parameters for terminal and print servers.
<code>ftpd</code>	The FTP server for TCP file transfers from remote hosts. FTPD requires a configuration file before enabling.
<code>httpd</code>	The World Wide Web server for HyperText documents
<code>itcd</code>	The AlphaNET ITC tunneling server. Passes AlphaNET messages over TCP/IP connections.
<code>lpd</code>	Line printer server. Requires an ersatz name before enabling.
<code>lpr</code>	Line Printer client. LPR requires a configuration file before enabling.
<code>pop3d</code>	The Post Office Protocol server. POP3D requires a configuration file before enabling.
<code>poppwd</code>	POP password modification server. Allows Eudora users to change their POP password.
<code>rwhod</code>	Allows users to see what computers are on the network.
<code>smtpd</code>	The Simple Mail Transfer Protocol server. SMTPD requires a configuration file before enabling.
<code>sntpd</code>	Simple Network Time Protocol. Allows you to synchronize your system time with that of a time server. It also adjusts your <code>TIMZON.</code> file as needed.
<code>tamed</code>	TCP Access Made Easy server. Allows AlphaBASIC or assembly language programs to access TCP functions.
<code>tddd</code>	Allows access to a TCP connection through the TDD: device, and allows incoming TCP connections to non-filestructured devices such as <code>TRM:</code> .
<code>telned</code>	The TELNET server for virtual connection into AMOS. There are several options available for the telned line.
<code>tftpd</code>	The TFTP server for UDP file transfers from remote hosts.

The server names in this file are case-sensitive—they *must* be lower case!



You don't need to start the SLIP servers. AlphaTCP starts them automatically if there are `if-config` lines for the `asy0`, `asy1`, `asy2`, and/or `asy3` devices.

Start the servers you want by removing the `#` at the beginning of the line and setting up any configuration files or options required by them. See Chapter 5 for details.

Memory Requirements—The memory required by each server is discussed earlier in this chapter and is allocated from the shared memory (SMEM) area. The memory size defined in the `CONFIG.` file represents a single instance of the server. It is assumed that the `.LIT` file is loaded in system memory for servers which “fork.” This value is also used as the memory requirement for spawned jobs of servers which “fork” copies of themselves. You must enter the memory in kilobytes and follow it with a `K`, as shown. The `K` can be either upper or lower case.

The `CONFIG.` values for the servers are lower than those given earlier in the chapter because the `CONFIG.` entries do not include the 6KB overhead automatically assigned to each server as required by AMOS.



Since TFTP provides no system security whatsoever, you should start it only if you are absolutely certain you can trust every user on every connected network. See Chapter 5 for more information.

If You Allow Incoming Virtual Terminal Connections

Delete the `#` symbol from the `#start telned 100k` line. There are a number of configuration options for this server. By default, incoming connections will be accepted on the standard TELNET port, memory and terminal type will be asked, memory size will default to 64K, and the session will begin with LOGON. For other options, see Chapter 5.



If your physical TRMDEFs plus VTSER type-2 TRMDEFs add up to your AMOS port license count, you will need to reduce one of them. TELNED does not use the “TRMDEF #” style ports at all.

If You Allow Incoming File Transfer Requests

If you want secure FTP sessions, delete the `#` from the `#start ftpd 50k` line. For information on the `-o` option that overrides AMOS file protection, see “FTP SERVER (FTPD),” in Chapter 5.

If you want TFTP sessions, delete the `#` from the `#start tftpd 35k` line. There is no security associated with TFTP, and many sites will not want to enable it.

If You Want System Uptime Information

Delete the `#` from the `#start rwhod 75k` line. This generates broadcast information on the network and is not always welcome at larger installations.

If You Want to Synchronize Your System Time

Remove the `#` from the beginning of the `start sntpd 85k -b2 -p240` line. The SNTPD server synchronizes your system time to match the time retrieved from the time server system specified on this

line. It also creates the TCP:TIMZON. file for you, and changes it for daylight saving time as needed. You must also set up information in the TCP:SNTPD. file. See Chapter 5 for more information on this.

If You Want to Use TCP Through Application Programs

Delete the # from the beginning of the `start tamed 200k` line. This allows you to write and run programs using the TAME monitor calls or AlphaBASIC XCALLs.

Finishing Up the CONFIG. File

You can now save the file and exit the text editor.



Modifying other AlphaTCP files is required to finish enabling some of the options discussed in this section (for example, you must modify the NETWRK. file to complete SLIP connection setup), so be sure to read the rest of the sections in this chapter before moving on!

TCPEMU Startup File (GOTCP.CMD)

The TCPEMU job runs the TCP:GOTCP.CMD file. The file is shipped with AlphaTCP as GOTCP.NEW, and should be edited to reflect your computer configuration and renamed to GOTCP.CMD. The GOTCP.NEW file looks something like this:

```

:R
LOG TCP:                ; LOGIN TCPEMU job
SYSTEM SERVICE
WAIT NETLOG             ; make sure NELOG at dot
RENAME NETLOG.OLD=NETLOG.LST/D ; get previous log file
FTPLOG NETLOG.OLD      ; process FTP activity
FORCE NETLOG           ; start up NETLOG job
LOG TCP:
SYSTEM SERVICE
NETLOG TCP:NETLOG.LST

WAIT NETLOG            ; wait for NETLOG to run
TCPEMU                 ; start up TCP/IP

```

The initial GOTCP.CMD file references the SYSTEM SERVICE user name to log the TCPEMU and NETLOG jobs into the TCP: account. If this is not a valid user name on your computer, either add it using MUSER or change GOTCP.CMD accordingly. Both jobs must be logged into TCP:. If logging in as SYSTEM SERVICE requires a password, enter it into GOTCP.CMD.

TCP:GOTCP.CMD also starts the NETLOG program and selects the NETLOG output file (TCP:NETLOG.LST).

Hosts File (HOSTS.)

AlphaTCP reads the hosts file to convert host names into internet addresses. You should enter host names in this file for all computers so you won't have to use internet addresses to communicate with other hosts.



If you are using DNS, move to the section on the RESOLV. file, because you don't need to use the HOSTS. file. If you are not using a name server (DNS), make sure there is no RESOLV. file.

Using your text editor, edit the file HOSTS.. The prototype file looks like this:

```
#Hostname file for AlphaTCP
#
192.168.1.1  am4000.company.com  am4000  #An AMOS AM-4000 with AlphaTCP
192.168.1.2  sco.company.com           sco      #A SCO UNIX system with TCP/IP
192.168.1.3  am920.company.com         am920   #An AM-920 with DOS TCP/IP
127.0.0.1    localhost.company.com    localhost #local loopback
```

Diagram labels with arrows pointing to the corresponding fields in the prototype file:

- Host internet address (points to 192.168.1.1)
- Host name (points to am4000.company.com)
- Alias (points to am4000)
- Comment (points to #An AMOS AM-4000 with AlphaTCP)

Each line begins with the internet address for a host, followed by its host name and any aliases it may be called, separated by spaces. Again, a # indicates a comment line.

Host names may contain any character except a space. Generally, a host name consists of a local node name, company name, and a primary domain name. For example: `steve.alphamicro.com`.



When host names are looked up by themselves (no domain name) with DNS, transparently the lookup gets the domain name appended. In order to make a future move to a DNS server simple, you should make the HOSTS file behave the same way. To do this, place the full name first (host name with domain name). Next, place just the host name as the first alias, as in the above examples (such as: `math101.uc.edu math101`). This allows HOSTS file lookups to work with full names (host name with domain name) as well as host names only, just like DNS.

The primary domain names in use in the United States are shown below. Usually you will associate your organization under the most appropriate one. International domain names have different conventions, and usually end in a two letter country abbreviation.

Domain Name	Description
COM	Company or corporation
EDU	Educational organization
GOV	Government office
MIL	Military
NET	Internet provider or administration
ORG	Organization (catch all)



Once you enter all hosts you may save the file and exit the text editor.

DNS Resolver File (RESOLV.)

If you are configuring via BOOTP or DHCP, you may skip this step. The file will be created automatically for you.

The resolver file tells AlphaTCP you are using the Domain Name Service and which computers to contact. Domain Name Service replaces the local HOSTS. file (or adds to it) with access to a distributed name database. This database is provided via servers (often UNIX computers running BIND software) all over the world.

You may use the RESOLV.NEW file as a sample file. Make your changes and rename it to RESOLV.. Delete the # at the front of a `nameserver` line to enable a name server.

Name servers are defined using the **nameserver** command. The **nameserver** command specifies the internet address of a host supplying Domain Name Service. Only one host per **nameserver** command may be used, to a maximum of three **nameserver** commands. Supporting multiple name servers provides redundancy should the primary server go down. An example of a **nameserver** command is:

```
nameserver 192.168.2.1
```



If you want AlphaTCP to check your local HOSTS. file before it checks the name server(s), remove the # from the front of the `#tryhostfile` line.

Your new host must be added to the DNS by the administrator of your domain. If you are setting up a new domain yourself, you may want to contact someone familiar with the procedure. The complexity of DNS administration is beyond the scope of this document. For further information, you may want to reference the book *DNS and BIND* (ISBN 1-56592-010-4).

Local Host Name File (MYNAME.)



If you are configuring via BOOTP or DHCP, you may skip this step. The file will be created automatically for you.

The local host name file (MYNAME.) resides in TCP:, and contains the full name of the host with its domain. It is displayed for file transfers, system uptime data, and is used for mail delivery. (Programs may get the host's Internet addresses from a name server or HOSTS. file.) This file is analogous to the UNIX *uname* command.



Earlier releases of AlphaTCP simply used MYNAME. to obtain the host name for display in various servers. Now, however, this file also specifies the **fully qualified domain name** for Domain Name Service and MAIL support. Therefore, you must be sure this file contains your fully qualified domain name as your computer is known to the DNS.

Edit the file MYNAME. and change the first line to your fully qualified domain name. Then save the file and exit the text editor. For example, assume your company has the domain `widgit.com` and your host is called `sales`. Your fully qualified domain name to be listed on the first line in the MYNAME. file must be:

```
sales.widgit.com
```

Networks File (NETWRK.)

AlphaTCP reads the networks file to determine the internet address of this computer for each network it is connected to, as well as other networks which may be routed to. This file is modified in the sections “If You Are Using Ethernet” and “If You Are Using SLIP”.

Services Definition File (SERVIC.)

The SERVIC. file contains definitions of well-known port numbers associated with different servers (such as TELNET). You probably won't need to modify this file unless you define your own server. A SERVIC.NEW file is shipped with AlphaTCP.



Because it may contain new definitions, you should always use SERVIC.NEW, renaming it to SERVIC.. But first look in your existing SERVIC. file to see if there are any changes you have made that you want to add to the SERVIC.NEW file before renaming it.

TCP Ersatz File (TCPIP.ERZ)

TCPIP.ERZ defines the TCP: ersatz device. A TCPIP.NEW file containing an ersatz definition for the default AlphaTCP location, DSK0:[7,50], is shipped with AlphaTCP. If you install AlphaTCP there, simply rename that file to TCPIP.ERZ. If you install AlphaTCP in a different location, modify TCPIP.NEW for the new location and then rename it to TCPIP.ERZ. If you need to define other TCP ersatz devices, place them in TCPIP.ERZ.

Time Zone File (TIMZON.)

The time zone file tells AlphaTCP the name of your time zone, along with the number of hours and minutes it is offset from universal time (GMT or UTC). It is very important to set this file up properly. It is currently used in SMTP mail and may be used by future applications.



If you are running the SNTPD server you don't need this file. When the SNTPD server synchronizes the system clock it creates this file for the local time zone. The file is updated when the time zone adjusts for daylight saving time. Because this file is important, you should verify SNTPD has been able to create it for you after the SNTPD “boot delay” period has elapsed.

Using your text editor, create and edit the file TIMZON.. There should be only one line in the file, with the format:

```
-hhmm (zone)
```

Hhmm is a four-digit number that tells AlphaTCP that the local time zone is hh hours and mm minutes from universal time. **zone** tells it which time zone you are in. The - sign indicates “behind” universal time. For example:

```
-0800 (PST)
```

tells AlphaTCP you are in the Pacific Standard Time zone, and that you are eight hours and zero minutes behind universal time. If the current UTC is 1400, then at -0800, PST is 0600 (1400-0800).

This table shows popular time zones and the correct entries for those zones:

<u>Time Zone Description</u>	<u>TIMZON. Entry</u>
Pacific Daylight Time	-0700 (PDT)
Mountain Standard Time	-0700 (MST)
Central Standard Time	-0600 (CST)
Eastern Standard Time	-0500 (EST)



If you live in an area with daylight savings time and are not using SNTPD, remember to change this file when you adjust your local time.

WHOIS Server File (WHOIS.)

The WHOIS. file contains the name of the server to contact when looking up domain information using the WHOIS command. AlphaTCP includes the file WHOIS.NEW, which contains the name of the Internet Network Information Center server, rs.internic.net. You can rename this file to WHOIS.

TCP Line Printer Definition File (LPR.)

The LPR. file contains the definition for any TCP line printers associated with AMOS print spoolers. An LPR.NEW file is shipped with AlphaTCP, which you may modify and rename to LPR.. See “If You Are Using a Print Server,” below, for more information.

If You Are Using Ethernet

The following sections tell you what changes need to be made to the CONFIG. and NETWRK. files to support Ethernet connections.

Changing CONFIG.

If you are going to configure the Ethernet interface using bootp or dhcp, uncomment the *start bootpc* line and set the switches on the command line for your preferences (see Chapter 5).

- **For AlphaNET NDVs:** Delete the # at the beginning of the first `ifconfig ec0...` line. Then, change the number between `ec0` and `arpa` (101 in the example file) to the *AlphaNET network number* of the Ethernet interface. You can find this number by using NTSTAT/F/P or examining the file specified in the NETINI command in your system initialization file. (Make sure you use the network number, not the entire `cpuID` displayed at the beginning of the NTSTAT/F display.)
- **For Standalone LDVs:** Delete the # at the beginning of the `ifconfig ec0 am319s` line. If you are using an LDV other than AM319S.LDV, change the `am319s` parameter to the LDV you are using. Be sure the LDV is installed properly in your system initialization file. See Chapter 4.



The interface provided by an LDV is much more efficient for AlphaTCP than an NDV. If you are not using AlphaNET you should use an LDV.

Changing NETWRK.



If you are planning on configuring from a BOOTP or DHCP server, you do not need to define an IP address or netmask for Ethernet. Instead, replace the ip address with the word `bootp` or `dhcp` and be sure you enabled the `bootpc` client.

Change the internet address on the line beginning `ethernet` to the proper address for this computer. (This is the 4-place dotted decimal address of the host. Enter it in place of the `192.0.2.1` value on the `#ethernet 192.0.2.1 ether eth net` line.) The network portion should be the same as all other nodes on this network, and the node number should be unique. Delete the `#` at the beginning of the line so `ethernet` starts in column 1, but do not delete the remainder of the line past the address.

Optional Subnet Mask—This value overrides the standard class-based mask. It is appended to the above address with a slash and placed in the same configuration file. For example, adding a subnet mask to the address in the sample file might look like this:

```
#ethernet 192.0.2.65/255.255.255.192 ether eth net
```

Chapter 1 has an explanation of subnetting.

Second Ethernet

AlphaTCP 1.5 supports dual Ethernet interfaces on 68060 and ColdFire based systems. AlphaTCP may be set up to route between the segments, but the segments must not be connected together and the network portion of the IP address must be different between interfaces. A second Ethernet is defined by adding lines to the `CONFIG.` and `NETWRK.` files.

The following table shows the supported Ethernet interface combinations:

System	Interface 1	Interface 2
AM-7000	PCILSI	AM319S
AM-6000	AM-176	AM319S
Eagle 450	AM-138	PCINIC

In `CONFIG.`, you may copy the LDV or NDV `ifconfig ec0` line. You should then change `ec0` to `ec1`, and change the LDV name or the NDV network number as needed.

In `NETWRK.`, copy the existing `ethernet` definition and add the number one to the end of all the names as below. Change the IP address as needed to cause the second interface to have a unique network portion of the address. Example:

```
ethernet1      192.168.2.250   ether1 eth1 net1
```



The only application which allows connecting two Ethernet interfaces to the same segment involves running AlphaNET on one interface (through an NDV) and AlphaTCP on the other (through an LDV). This configuration has distinct performance advantages for AlphaTCP when AlphaNET must be supported, since AlphaTCP will not need to go through the NDV. It does not cause any conflicts as each Ethernet interface is running a different protocol. For best perform-

ance, the interface in the leftmost column above should be used for AlphaNET, and the rightmost column should be used for AlphaTCP.

If You Are Using SLIP

The following sections contain information on what changes need to be made to the CONFIG. and NETWRK. files to support SLIP connections. See Chapter 5 for information on SLIP log and security files.

Changing CONFIG.

SLIP Connection 1—Remove the # at the beginning of the `ifconfig asy0...` line. Change `modem1` to the name of the terminal and change `dialup.slp` to the script file name (see Chapter 6). If the serial line is hardwired (that is, you are not using a modem for this connection), remove the script file name.

If this is a modem connection, you need to create a script file which will be used when AlphaTCP needs to establish communications over this link (see Chapter 6). You may use a communications package to manually perform the login sequence, then transcribe this session into a script file.

SLIP Connections 2-4—Make the above changes to the `asy1`, `asy2`, and `asy3` lines as required.

Changing NETWRK.

SLIP Connection 1—If you are using SLIP, change the internet address on the line beginning `slip` to the proper address for this computer. (This is the 4-place dotted decimal address of the host. Enter it in place of the `192.0.3.1` value on the `#slip 192.0.3.1 serial` line.) The network portion should be the same as the other node on this line, and the node number should be unique. Delete the # at the beginning of the line, so `slip` starts in column 1, but do not delete the remainder of the line past the address.

SLIP Connection 2—Make a copy of the line beginning `slip...`, and make the same changes as for SLIP Connection 1, but change the `slip` to `slip1` and `serial` to `serial1`.

SLIP Connection 3—Make a copy of the line beginning `slip...`, and make the same changes as for SLIP Connection 1, but change the `slip` to `slip2` and `serial` to `serial2`.

SLIP Connection 4—Make a copy of the line beginning `slip...`, and make the same changes as for SLIP Connection 1, but change the `slip` to `slip3` and `serial` to `serial3`.

Optional Subnet Mask—This value overrides the standard class-based mask. It is appended to the above address with a slash and placed in the same configuration file. For example, adding a subnet mask to the address in the sample file might look like this:

```
#ethernet 192.0.2.65/255.255.255.192 ether eth net
```

Chapter 1 has an explanation of subnetting.

If You Are Using a Print Server

Set up your AMOS spoolers as usual. In place of the line in the AMOS printer initialization file normally used to define the terminal name (for example, `DEVICE=TRM:SPOOL`) enter `DEVICE=TLP:name`. *name* may be any unique name up to six characters in length—write this name down for use later.

Changing CONFIG.

Edit the TCP:CONFIG. file and delete the # from the `#start lpr 35k` line.

Changing LPR.

The `DEVICE=TLP:name` definition is mapped to the TCP printer in the TCP:LPR. file. A sample file, LPR.NEW, is provided, which may be modified and renamed to LPR.. You may enter multiple lines to define multiple printers. The format of each entry is:

```
name=host,print-queue
```

name is the name specified in the `DEVICE=TLP:name` line in the printer initialization file, *host* is the remote host on which the printer resides, and *print-queue* is the name of the print queue for the printer on that host. Here is a sample LPR. file:

```
TCPPRT=host.company.com,PASSTHRU
```

If you want to define a dynamic-IP printer—a printer name that a remote user can assign to a local printer “on the fly”—enter an asterisk instead of the host name.

See the “Customizing” section in “Line Printer Client (LPR)” in Chapter 5 for details on this file.

Changing TCPIP.ERZ

Create an account which may be used by LPR for temporary storage of print files. Add a TCPLPR: ersatz definition for this account into the TCPIP.ERZ file.

To Make AMOS Printers Accessible

The “queue name” specified by remote hosts is used to select the desired AMOS printer. It should match the spelling and case of the AMOS spooler name.

Changing CONFIG.

Edit the TCP:CONFIG. file and delete the # from the `#start lpd 35k` line.

Changing TCPIP.ERZ

Create an account for LPD to store print files in until they are printed. Add a TCPLPD: ersatz definition for this account into the TCPIP.ERZ file.

If You Are Using SMTP MAIL

Creating TIMZON.

Create the file TCP:TIMZON. and enter information about your time zone. See “Time Zone File (TIMZON.)” in this chapter for information.

Creating Special User Names

You must set up a special user name on the system for TCP/IP mail. By default, this user is called “Internet.” Your mail package may have different requirements. See your mail documentation for details about TCP/IP or Internet mail. If the user name is different, you also need to notify the SMTP mail server of this through a start line switch in CONFIG.. TCP/IP mail is handled through this special user.

Another special user name used is called “Postmaster.” If this is not a real destination in your mail package, the name may be changed through a start line switch in CONFIG.. If this is done, mail sent through TCP/IP to “Postmaster” is forwarded to the user specified.

Changing CONFIG.

Edit the TCP:CONFIG. file and delete the # from the beginning of the `#start smtpd 125k -i` line. If you are not using a name server, remove the `-i` from the line.

There are many other options on the start line, some of which you need to set up if the above user names are different than the defaults. You should consult Chapter 5 before continuing to see if any other options apply to your installation.

Changing TCPIP.ERZ

Create an account which may be used by SMTPD for temporary storage of files. Add an ersatz called TCPMAL: for this account into the TCPIP.ERZ file.

Creating Special SMTPD Files

Several special SMTPD files allow you to:

1. Specify host names which should be treated as if they were local names (SMTPD.LOC)
2. Forward mail for unrecognized hosts to a remote computer that is using a name server, if you aren't using one (SMTPD.FOR)
3. Define alias addresses for mail recipients (SMTPD.ALI)

For information on creating these files, see the “Customizing” section in “Mail Server (SMTPD)” in Chapter 5.

Configuring the Name Server

It is very important when using SMTP with name servers to properly configure the name server A and MX records for your setup. This is beyond the scope of this document and should be done by someone familiar with DNS and BIND administration. A good book to reference is *DNS and BIND in a Nutshell*—see the bibliography in Chapter 1.

If You Are Using POP3 Mail

The POP3D server is used to allow hosts that cannot support a full SMTP implementation, such as PCs, to get mail. See Chapter 5 for more information on this server. The POPPWD server supports the *change password* function in Eudora and is optional.

Creating the POPUSR. File

Create the file TCP:POPUSR. and enter the connection information for each user who will be accessing the post office. Define each user on a separate line, with the login, password, and mailbox name of the user. The format of the user definition is:

```
login password user-name
```

The *login* and *password* should be unique to the Post Office, but the login does not need to be a real login name. No embedded spaces are allowed in the login or password. The mail name entry may contain embedded spaces, and must match the name used by the user when directly accessing mail on AMOS. Here's a sample line:

```
j_doe    jd_paswd1    John Doe
```

In the above example, the remote POP client would log into the Post Office server as `j_doe`. The Post Office server would then access mail in the local mail package for user `John Doe`. Though handling of embedded spaces may vary between local mail packages, in the above example using AlphaMAIL the user's Internet mail address would be `john_doe@{yoursite}`.

Changing CONFIG.

Edit TCP:CONFIG. and delete the # from the beginning of the `#start pop3d 60k` line. Many options can be included on this start line. Consult Chapter 5 before continuing to see if any apply to your installation.

If you want to allow Eudora users to change their POP password, edit TCP:CONFIG. and delete the # from the beginning of the `#start poppwd 70k` line.

Changing TCPIP.ERZ

Create an account which may be used by POP3D for storage of files. Add an ersatz definition for this account called TCPPOP: in the TCPIP.ERZ file.

If You Are Using the Web Server

There are a number of command line options you may need to use. See Chapter 5 for more information on this server.

Load the following files into system memory: HTTPD.LIT and HTTPD.RTI.

Verify that you have created the proper TCP:TIMZON. file. Details for this file may be found earlier in this chapter.

Create the file MIME.TYP in the LIB: account. The example file MIME.NEW may be renamed and used as-is. Details for this file may be found in Appendix C.

Creating the HTTPD.HTM File

The TCP:HTTPD.HTM file is the default document sent by the server when no other one is selected. The format of this file is HTML, HyperText Markup Language. Writing HTML documents is beyond the scope of this document. There are many books available on the subject of writing HTML and 'Web Publishing', one of which is referenced at the end of Chapter 1. A very simple example is provided, HTTPD.NEW, which may be renamed and modified. You will need to read the Web Server section in Chapter 5 to use the Web Server.



You can change the default document sent by using the `-m` option when starting HTTPD. See Chapter 5.

Changing CONFIG.

Edit the TCP:CONFIG. file and delete the `#` from the `#start httpd 100k` line.

If You Are Using AlphaNET Tunneling

The ITCD server automatically installs its own AlphaNET network driver interface when it starts. Therefore no NETINI line is needed in the system initialization file for this AlphaNET network. As this interface is different than the physical interface installed by NETINI, the AlphaNET CPU IDs accessed through the ITCD server must be different, too. For example, if AlphaNET nodes accessed directly through the Ethernet NDV are within AlphaNET Network 1, the CPU IDs through ITCD must *not* be within Network 1. Use the command CVTNID to convert a network, group, and node set into the AlphaNET CPU ID used within the ITCD. file. CVTNID may also be used to convert in the opposite direction by supplying the network, group, and node number separated by commas.

Creating ITCD.

The TCP:ITCD. file translates AlphaNET CPU IDs into AlphaTCP IP addresses. Each host which will be accessed, including the local one, needs to be defined on a line in the file.

Each line begins with an AlphaNET CPU ID (for example, 16842753) followed by the corresponding AlphaTCP IP address (for example, 192.0.2.1). The two entries may be separated by spaces or tabs.

You must also change your system initialization command file to set up ITC tunneling. See Chapter 4.

Changing CONFIG.

Edit the TCP:CONFIG. file and delete the # from the `#start itcd 100K` line.

WHAT'S NEXT?

You have completed the second step in installing AlphaTCP. Go on to the next chapter to find out how to change the AMOS system initialization command file.

Chapter 4 - Modifying the System Initialization File

This chapter contains information on modifying the AMOS system initialization file to install AlphaTCP:

- Modifying your system initialization command file.
- Testing your network setup.
- Fine-tuning AlphaTCP performance by loading files into system memory.

AlphaTCP requires several modifications to your system initialization command file (also called the “INI file”). This file is called either AMOSL.INI or AMOS32.INI, depending on your version of AMOS. These instructions assume you are familiar with this initialization file.

If you want more information, refer to the *System Operator’s Guide to the System Initialization Command File*.



Before you perform the following steps, you need to determine memory requirements for both the TCPEMU job and the shared memory area. The memory size depends on which servers you start and how many connections each will support. See the discussions of memory requirements for each server earlier in this book.



Never modify your system initialization command file directly. Always copy the file and modify the copy. Changing your initialization file directly could make it impossible to boot your computer!

CHANGES TO THE SYSTEM INITIALIZATION COMMAND FILE

Follow these steps to update your INI file for AlphaTCP:

1. Make sure the JOBS statement allocates enough jobs to allow for:
 - Two jobs for the AlphaTCP emulator.
 - One job for each server which will be spawned.
 - One job for each connection each server will handle.
 - One extra job per SLIP interface enabled.
 - Extras to cover extra jobs spawned by various tasks such as MTUSAV, MULTI, Metropolis, etc.

Since there is not much overhead associated with increasing the number of jobs, be generous in adding jobs; add at least 10 extra.

2. At the end of the JOBALC statements add the following line:

```
JOBALC NETLOG,TCPEMU
```

Because TCPEMU spawns many jobs, it is best to add it to the end of the JOBALC statements so the spawned jobs will appear by TCPEMU in a STAT display.

3. At the end of the TRMDEF statements, add the following two lines:

```
TRMDEF NETLOG,PSEUDO,NULL,100,100,100
TRMDEF TCPEMU,PSEUDO,NULL,100,100,100
```

4. If you are using SLIP to use AlphaTCP over a modem, add a TRMDEF statement for the port the modem is connected to. Make sure to set the baud rate correctly for the modem you are using. The other parameters of the modem TRMDEF are not important to AlphaTCP (though they may be to other applications). You can use the TELTYP terminal driver.
5. If you are using SLIP to use AlphaTCP over a serial line, add a TRMDEF for each serial connection. The parameters of the TRMDEF are not important to AlphaTCP (though they may be to other applications). You can use the TELTYP terminal driver.
6. Locate the ERSATZ statement(s) and add the following line there:

```
ERSATZ ERSATZ.NEW
```

This file defines the TCP: ersatz device as DSK0:[7,50]. If you installed the AlphaTCP software in another account, change this definition to the correct account.

If you are using the web server (HTTPD), you need an ersatz file to store references to accounts the web server will access. Add the following ersatz statement:

```
ERSATZ WEB.ERZ
```

Finally, if you are using the AlphaNET tunneling server (ITCD) you need an ersatz file to reference the new AlphaNET CPU IDS with meaningful names. Add the following ersatz statement:

```
ERSATZ ITCD.ERZ
```

7. If users of other computers will TELNET into AMOS, you need to add TRMDEF or TDVDEF statements to load terminal drivers which match those on the remote computers (The VT-100 is a popular terminal/emulation outside the AMOS environment). If you want the line editor available on those sessions, the line editor must already be defined on one of the permanent TRMDEF statements.
8. If you are using TCP/IP print servers, add a DEVTBL TLP line at the end of the DEVTBL statements.
9. If you want to use TCP device drive daemon, add a *DEVTBL TDD* line at the end of the DEVTBL statements.
10. If you are using AlphaMAIL, verify there is a LODEMD command before the first SYSTEM statement. This command should be loading the electronic mail driver for your mail package. If you don't see this statement, refer to your mail documentation for details or contact your mail provider.

11. If the TCP: account is not on DSK0:, add a MOUNT *devn:* command before the first SYSTEM statement. (*devn:* is the disk TCP: resides on).
12. If you will be using AlphaNET with AlphaTCP and an LDV (using ITC tunneling), you must add a NETFAM command to load NETFAM.SYS, the AlphaNET networking software. This replaces the NETINI command you would use if you were setting up an AlphaNET network using an NDV. NETFAM must come after the MSGINI command and before the first SYSTEM statement.
13. Next, modify the SYSTEM statement section.

Add the following lines anywhere within the SYSTEM statements, but before the final one:

```
SYSTEM TCP:IPCINI/N 100 100K
SYSTEM RTI.LIT
```

Unless it is already loaded (which is common), enter:

```
SYSTEM CMDLIN.SYS
```

If using TCP/IP print servers, add:

```
SYSTEM DVR:TLP.DVR
SYSTEM TCP:LPR.LIT
SYSTEM TCP:LPR.RTI
```

If using TCP device driver daemon, add:

```
SYSTEM DVR:TDD.DVR
SYSTEM TCP:TDDD.LIT
SYSTEM TCP:TDDD.RTI
```

If AMOS printers will be accessible via TCP/IP, add:

```
SYSTEM TCP:LPD.LIT
SYSTEM TCP:LPD.RTI
```

If accepting FTP sessions, add:

```
SYSTEM TCP:FTPD.LIT
SYSTEM TCP:FTPD.RTI
```

If accepting TFTP sessions, add:

```
SYSTEM TCP:TFTPD.LIT
SYSTEM TCP:TFTPD.RTI
```

If using SMTP mail, add:

```
SYSTEM TCP:SMTPD.LIT
SYSTEM TCP:SMTPD.RTI
```

If using POP3 mail, add:

```
SYSTEM TCP:POP3D.LIT
SYSTEM TCP:POP3D.RTI
```

If using the Web Server, add:

```
SYSTEM TCP:HTTPD.LIT
SYSTEM TCP:HTTPD.RTI
```

You may want to put other AlphaTCP programs in system memory to save memory or improve performance. See the next section.

14. If you are using an LDV network interface driver, you must add a SYSTEM statement for it:

```
SYSTEM name.LDV[1,6]/N
```

name is the name of the LDV for your Ethernet interface.

If you use an AM-362, you must specify the address of the AM-362 board in the SYSTEM statement. The format is:

```
SYSTEM AM362.LDV[1,6]/N addr
```

addr is the device address (from 1 - 6).

1. You must allocate shared memory for the servers using the SMEM statement. The SMEM statement *must* come right after the final SYSTEM statement. There must be only one SMEM line in the file. The line will look like this:

```
SMEM memory
```



You should already have calculated the memory requirements as described earlier. Enter that number for *memory* above. If shared memory is already defined, make sure it's in the proper place (not before or inside the SYSTEM statements) and specify enough memory for both AlphaTCP and whatever else is using the shared memory area.

15. To make sure the NETSER job is running before you start these jobs, place a WAIT NETSER statement above the following SETJOBS (assuming you are using AlphaNET).
16. Add the following two lines within the SETJOB statements. If you are using AlphaNET, they must come *after* the NETSER job is running:

```
SETJOB NETLOG,NETLOG,40K
SETJOB TCPEMU,TCPEMU,memory,TCP:GOTCP.CMD
```



You should already have calculated the memory requirements required for TCPEMU as described earlier. Enter that number for *memory* above.

17. If you have made AMOS printers accessible, the printers are task manager based, and the edit level of TSKINI.LIT is earlier than 1.2 (127), you must start the task manager *after* the

TCPEMU job. Also, if TSKINI.LIT is prior to 1.2 (127), you should place a WAIT TCPEMU, then a WAIT LPR statement, before to starting the task manager.

18. Save the file and exit the text editor, but don't reboot or use MONTST yet.



The TCP:GOTCP.CMD file which the TCPEMU job runs uses the user name SYSTEM SERVICE to log the job into the TCP: account. If this is not a valid user name on your computer, either add it using MUSER or modify the GOTCP.CMD file to use another user name. If logging in as SYSTEM SERVICE requires a password, add it to the GOTCP.CMD file.

You should now be ready to start AlphaTCP. Make sure no one else is using the computer and use MONTST to test your new initialization file.

To make sure the changes have had the desired effect, run STAT. The TCPEMU job should be in SI (software interrupt) state, and all servers requested should be spawned and also in SI state (this can take several seconds). If this is not the case, you may not have given the TCPEMU job enough memory, or not provided enough shared memory to start the servers you requested.

TROUBLESHOOTING STARTUP PROBLEMS

If the AlphaTCP package, or one server, does not start up, you can enter:

```
VUENL (RETURN)
```

This executes CMD:VUENL.DO which executes LOGCTL to get a copy of NETLOG.LST, and then VUES it. It will show TCPEMU initialization statistics and error and other messages from all of the servers and TCPEMU.

If you have specified bootp or dhcp configuration of the Ethernet, and the servers stay in S1 state, it is likely that the bootp server isn't configured to recognize the Ethernet's MAC address, the dhcp server isn't available, or the BOOTPC client was not started in the TCP:CONFIG. file

See Appendix B for error messages displayed in the NETLOG.LST file.

TESTING THE INSTALLATION

In addition to using STAT to make sure the servers are in the proper state, you can test the AlphaTCP stack by connecting to your own computer through TCP.

To test, use the **telnet** command with localhost as the host name. For example:

```
telnet localhost (RETURN)
```

establishes a TELNET connection to the local computer, and makes sure the TELNED server and TCP stack are up and running properly.

WHAT'S NEXT?

You have completed the final step in installing AlphaTCP. The rest of this book tells you how to use the AlphaTCP features designed specifically to help the network administrator. You should now turn to Chapter 8 for information on tuning up your installation.

Chapter 5 - AlphaTCP Support and Server Processes

The AlphaTCP emulator (TCPEMU) is the process responsible for creating the UNIX Streams and TCP/IP environment under AMOS. It also spawns the servers requested by the administrator at startup and handles the spawning and despawning of all AlphaTCP processes. Since most of AlphaTCP operates in the background without physical terminals, there is also a network error logger (NETLOG).

AMOS server processes are the equivalent of UNIX daemons. Server processes allow remote computers to request local actions. They are responsible for allowing a virtual terminal connection into the computer, or accessing local files for transfer. For users of AlphaNET, VTSER and NETSER could be considered servers.

The AlphaTCP servers are spawned by the TCPEMU job, using memory within the shared memory (SMEM) area.

Spawning and memory allocation for the various servers are specified by commands in the TCP:CONFIG. file.

This chapter describes special requirements, setup, and security concerns for each of the AlphaTCP server processes. See Chapter 3 for information on TCP:CONFIG., setting up the TCPEMU job, and memory requirements.



Servers may be brought up after TCPEMU is running without rebooting. If you want to start a server which was not in CONFIG., simply type the `start` line at the AMOS prompt as you would have entered it into CONFIG., but replace the word `start` with `startd`. See Chapter 6.

NETWORK LOGGER (NETLOG)

As most of the AlphaTCP package operates in the background or under spawned jobs, there is rarely a terminal attached where network problems may be displayed. For this reason, the NETLOG program intercepts these messages and logs them to disk. You can then examine the log file when you suspect a network problem. Alternatively, NETLOG can send the messages to a terminal attached to the NETLOG job.

Requirements

The NETLOG program must run on a job called NETLOG and must be started before the rest of the AlphaTCP package. You do not need to attach it to a real terminal, though you may want to do so for the reasons discussed below. Starting up NETLOG is usually done in the GOTCP.CMD file discussed in Chapter 3.

Operation

Messages normally output to the screen are redirected by AlphaTCP as AlphaNET messages to the NETLOG job. NETLOG prepends the job name and time, and then sends the message to the disk. This command starts NETLOG:

```
NETLOG {file}
```

If you don't specify a *file*, NETLOG sends the message to its own terminal. If this is what you want, be sure to attach a real terminal to the NETLOG job before starting NETLOG; otherwise messages are discarded.

Runtime Control

The NETLOG program is controlled by the LOGCTL program. LOGCTL can tell NETLOG to exit, close the file, open a new file, or log a user-defined message in the log file. You can split the log file into separate files, one for each job which logged a message, using the program LOGFMT. LOGCTL and LOGFMT are described in Chapter 6.

Security

There are no security concerns for the network logger.

TCP EMULATOR (TCPEMU)

The TCP emulator (TCPEMU) provides the UNIX Streams and TCP/IP environment the application programs and servers use to converse with each other. In addition, TCPEMU handles the spawning and de-spawning of the server jobs and their children, along with various other process control functions.

Requirements

The TCPEMU program must run on a job called TCPEMU. If it is executed on a job other than TCPEMU it merely displays the Product Installation Code (PIC) and the license size.



As TCPEMU spawns many jobs, you may want to place the TCPEMU job at the end of your JOBALC statements. That way, TCPEMU and its spawned jobs will be grouped together in a STAT display.

Operation

When the TCPEMU job starts, it reads the NETWRK. file and builds a UNIX Streams-like environment. It then reads the CONFIG. file. It uses this to link itself into the AlphaNET NDV or standalone LDV driver path so it can send and receive non-AlphaNET packets. Continuing, it starts up any requested servers by spawning a job within shared memory, with the same name as the server, and forcing the job to execute the server program.

Once everything is settled, a STAT should reveal the TCPEMU job and the servers in SI (software interrupt) state.

Runtime Control

TCPEMU will respond to a 'usr1' signal sent using UKILL. The process id (pid) of TCPEMU is 1. This will cause TCPEMU to write out a usage distribution of message blocks to the network logger. This may be useful for tuning AlphaTCP performance as detailed in Chapter 8. For information on UKILL and sending signals, see Chapter 6.

Security

There are no security concerns for the AlphaTCP emulator.

SLIP SERVER (SLIPD)

The SLIP server (SLIPD) handles the interface between the serial line and the IP data stream, performing the required conversions on the data packet to conform to the SLIP protocol. In addition, it handles the dialing and connection time-out for a dialup SLIP connection via modems.

Requirements

The SLIP server is *not* started as a normal server process. It is started automatically by the TCP emulator in response to configuration of the serial interface (asy0-asy3) by the `ifconfig` command. One server is started per *asn* device.

Operation

The SLIP server watches both the serial line and the IP stream for data. It performs the appropriate conversions and transfers one to the other. In addition, if IP data becomes available and the connection needs to be started over a modem, the SLIP server reads the specified script file and attempts to establish the connection. Whenever modem connections are established, the SLIP server records the source and destination Internet addresses of the message which initiated the connection in the file `TCP:SLIPD n .LOG` (where n is the number of the *asn* device the SLIP server is controlling). When the connection is terminated, the time it was active is logged to the same file.

Customizing

You may customize the SLIP server to limit incoming packets to certain secure, local computers. By default, all incoming packets are allowed. Once enabled, all incoming packets are discarded unless they are destined for one of the listed internet addresses. No other computers will be accessible or have access to computers over the SLIP connection.

To enable this feature, log into TCP: and create a file for each serial device you have enabled (asy0-asy3) and wish to secure. The name of these files is `SLIPOK. n` , where n is the number of the *asy* device (0,1,2,

or 3) you are controlling. Inside these files, place the internet address (not the domain name) of each local computer you want to have access.

For example, assume you have two SLIP links (asy0 and asy1). Assume asy0 is a hardwired line to a local computer. In this case, you would *not* create a SLIPOK.0 and allow all traffic to flow. Now assume asy1 is a dialup connection to the Internet. You would likely want to disable access to most of your critical systems. In this case, you would create a SLIPOK.1 file.

On separate lines in the SLIPOK.n file, list the internet address of the local system and each secure system you have. When AlphaTCP is started up again, nobody but the listed computers will be accessible through the asy1 SLIP interface.



Changes only take place once AlphaTCP is restarted. All lines in the security files which start with ; or # are treated as comments and are ignored.

Runtime Control

You may manually force SLIPD to hang up a modem by sending it a ‘term’ signal using UKILL (the default signal sent by UKILL). Note: if there are TCP/IP packets flowing through SLIPD, after hanging up it will reestablish the connection automatically and continue. For information on UKILL and sending signals, see Chapter 6.

Security

Once a SLIP connection has been established, remote nodes look no different than a local node on Ethernet (just slower). SLIP in itself has few security concerns—it simply provides a door to the world. If there are security holes accessible from any of your Ethernet nodes and you have a dialup SLIP interface running, those same holes are available to anyone running TCP/IP.

If you are concerned about this, you may want to disable auto-answer on your modem. Although this prevents users from calling into your computer, there would still be a window while a connection is active which you established. Remember, SLIP is a network connection. Once the physical connection is established, data can flow freely in both directions between any two computers where routing can be established.



For information on limiting incoming packets to certain secure, local computers, see “Customizing,” above.

TELNET SERVER (TELNETD)

The TELNET server (TELNETD) provides virtual terminal access into AMOS from any other computer running the TCP/IP protocol. The connection begins with a negotiation process where each end finds out what level of compatibility the other end provides. Once this is accomplished, the remote user is presented with the standard login screen for the AMOS computer, or whatever the system operator has specified to run (see “Customizing” below). You can activate multiple servers to start different applications on different ports.

Requirements

The connections established are Type 2 virtual connections, the same style AlphaNET CONECT creates over Ethernet. These connections have special requirements for the terminal driver, so verify all of your terminal drivers are compatible. (If they don't have an "output" routine, or they operate properly over AlphaNET Type 2 connections, they are probably fine.)



TELNETD does not use the VTSER "TRMDEF #" ports. The maximum number of incoming connections allowed is your AMOS license, less physical ports, less VTSER "TRMDEF #" ports. Reduce the number of "TRMDEF #" ports to get more TELNETD ports.

As stated in Chapter 4, the AlphaNET NETSER job should be running before AlphaTCP is started. This is because the TELNETD server registers itself with NETSER so TELDMP may contact it for connection display purposes. If you choose not to start NETSER, TELNETD will log a warning stating that it couldn't register with NETSER, and TELDMP will not work.

If you start multiple servers, each must be assigned a unique port number and have a unique description which will be registered with NETSER. The job names of subsequent servers will be bumped from TELNETD to TELNEE, TELNEF, etc. You may load TELNETD.LIT into system memory and reduce the memory allocated by the size of the program.

Operation

The TELNETD server listens for connection requests. When one is received, by default it asks the remote user for terminal type and memory size. You may predefine either or both of these, and/or set a range of valid memory sizes. After this, local and remote compatibility levels are negotiated, a job is spawned for the user, and the virtual connection is established. Usually the AMOS LOGON screen displays, but you can override this and start an application directly.

When a TELNET connection is started TELNETD sends the following message to the remote user:

```
***** Welcome to AMOS *****
```

You can customize this message by creating the file TCP:TELNET. and placing your own message in it. TELNET will display the contents of this file instead of the standard message. Following this message, when memory or terminal type needs to be determined, TELNET prompts the user for either or both.



Appendix E discusses what your application may need to do when a TELNET job running it is shut down.

Customizing


Many options are available at startup as defined on the `start` line of the CONFIG. file. By default, TELNETD starts on the primary TELNET port (23), requests terminal type and memory size from the user, then enters the AMOS LOGON program. You can modify this procedure by entering any or all of these parameters on the `start` line:

```
start telned 100k {-options} {port} {memory} {tdv} {"desc"} {"command"}
```

The available *options* are:

- asec* Access time-out in seconds. Connection is dropped if the job is still not logged in after this period.
- cmin* Connection time-out in minutes. Connection is dropped after this amount of time no matter what is being done.
- imin* Inactivity time-out in minutes. Connection is dropped after no output has occurred in this time period.
- j* When TELNED terminates a job, any pending command file or JOBCMD forced commands are normally discarded. The -j switch keeps them in force for the first termination attempt. This may be useful if your application uses the JOBCMD field for cleanup, but should generally be avoided.
- k* Disables keepalives. When keepalives are on (the default), TCP sends a query over an unused connection every two minutes. If a certain time (set in the TCP:CONFIG. file) goes by without a response, it closes the connection. This lets TCP recover the system resources used by a connection if the remote system is rebooted or the connection is lost. The only reason to disable keepalives is if you have a problem with users leaving dialup sessions open after they are done. In this case, disabling keepalives, combined with the -i switch, lets TCP close a connection after a certain amount of inactivity. Using -k to disable keepalives is not recommended.

Remember that, even if you disable keepalives, the computer at the other end of the connection may still send them, keeping the connection open.
- llogin* Allows you to override the normal use of LOGON.LIT, instead using the command specified in place of **login**. If the command is multiple words, place the whole entry in quotes. For example: "`-lLOG SYS :`"
- n* Disables special processing of FF and isolated CR characters, thus behaving like pre-AlphaTCP 1.4A TELNED. This may be needed if you are using InSight, which cannot be made RFC-854 compliant. The RFC-compliant behavior which confuses InSight is active with a combination of AlphaTCP 1.4A and AMOS 2.3A, or later.

 If you have a mix of clients to support on one server, you may leave this switch off the TELNED command line, and instead allow each client needing it to append -n to the end of the memory / terminal driver prompt. For example: "`>am65 200k -n`"
- p* Allows dynamic job priority scheduling. By default, the TELNED job excludes itself from SET DYNAMIC. This switch may be used if your particular environment operates better with TELNED receiving a variable priority.
- tt dv* Allows you to change the displayed example terminal driver from AM65A to something else, such as VT100.
- wsiz e* Sets the size of the input buffer and line width, which otherwise defaults to 100. Use this option with caution, as it increases the memory allocated for the line editor buffers.

port allows you to set up TELNED with a TCP port number different from the default of 23, thus allowing multiple servers with their own specific applications. When using a port number other than 23, use one not defined in *SERVIC.*, preferably 3000 or greater. If you want to use the default port but define other options on the line, use 0 for the port number as a placeholder.

memory has three possible formats:

- xxk* Sets all connections to **xxk** memory. Does not prompt anyone making a connection for the memory size desired. For example, 80k.
- xxk?* Changes the default memory to **xxk**, but asks the person making the connection for the desired size. For example, 100k?.
- xx-yyk* Prompts the user for the memory size, but only accepts answers in the range **xxk** to **yyk**. For example, 100-200k.

tdv lets you pre-define the terminal driver to use. If memory is also pre-defined, the user is not prompted for anything when establishing a session. If you don't want to define a terminal driver but do want to use the rest of the command line, use an asterisk (*) here as a placeholder.

description is the heading which will be shown in TELDMP for this server. It must be less than 32 characters, contained within quotation marks, and different than any other TELNED server started. This entry is mandatory when the next option is present.

command specifies an AMOS command (normally starting an application) to execute in place of the default AMOS LOGON. The command and any command line parameters for it must be enclosed in quotation marks. When TELNED is set up to start an application automatically, the connection will be terminated when the job exits to command level. Normally, the connection is terminated only when the user logs off or disconnects.

The following example line automatically executes the BASIC program PARTS upon connection to port 3000. It so happens that PARTS requires 150K of memory, and will only be requested by users with VT100 emulations, such as UNIX or PC users:

```
start telned 100k 3000 150k vt100 "Parts Lookup" "RUN PARTS"
```

The next example does the same thing, except the service is provided on the default TELNET port, and the user will be asked for his or her terminal type (but not the memory requirement):

```
start telned 100k 0 150k * "Parts Lookup" "RUN PARTS"
```

Telnet Source Host Information

Three utilities let you access information about the source of a TELNET connection. The VTID command prints this information to the screen, while the GETVTI.SBR subroutine provides it to an AlphaBASIC application. The VTIDS command operates much like the VTID command, however it prints information for all virtual connections on the system to the screen.



These utilities function only under AMOS 2.3A and later. They can also return information for AlphaNET VTSER connections, with content specific to the AlphaNET protocol.

To use VTID, simply type the command on a virtual terminal session.

For information on using the GETVTI.SBR, consult the AlphaBASIC Subroutine Users Guide. There is also a GETVTI monitor call which is documented in the AMOS Monitor Calls Manual.

To use VTIDS, just type the command on any terminal.

Security

Users connected via TELNET have all of the same privileges as a hard-wired terminal. The login process is handled by the normal AMOS log on procedures by default. User names and passwords, therefore, are retrieved from the USER.SYS file.

All users in USER.SYS should have passwords. Leaving a user name without a password makes it easy for anyone to log in and do anything they wish, including, for example, changing their FTP privilege level or formatting a disk.

Passwords should not be easily guessable and should preferably contain letters and numbers to make it difficult for a program accessing a dictionary to guess a password. Upper and lower case letters are considered identical in USER.SYS.

As this is a virtual terminal connection, setting a device to no access has no effect on the logged-in TELNET user. There is no way to prevent access to the TCP:FTPUSR. file via a TELNET login, other than possibly password protection on the TCP: account and/or LOKSER read protection, or some other AMOS security package.

There is a window of vulnerability when an automatically started application exits to the AMOS command prompt. There are two ways to prevent this. The application may log itself off (by clearing JOBUSR) and exit once it receives an exit request (SI\$EXI software interrupt, or a SIGTERM signal). There is also a wrapper program supplied, TNWRAP, which runs your program in a shell protected by the above method. Simply place the TNWRAP command ahead of your normal command line. See Appendix E for more information on properly shutting down an application when a connection is closed.

FTP SERVER (FTPD)

The FTP server (FTPD) provides a TCP based file transfer capability from a remote computer running TCP/IP. It also supports other functions such as directory listing, renaming and erasing files, account logging, and so on.

Requirements

The file MYNAME. must exist in the TCP: account and contain the name of the local host as you want it known to other hosts. See Chapter 3.

The file `RPC:FTPDEV.RPC` may be placed on other AlphaNET computers to allow FTP to list devices on those computers. (RPC: is traditionally located in `DSK0:[7,34]`.)

The files `FTPD.LIT` and `FTPD.RTI` should be loaded in system memory.

Operation

The FTP server listens for connection requests. When one is received, it forks by telling `TCPEMU` to spawn a job to handle the rest of the connection, then goes back to listening. The spawned job performs a simplified login process and handles requests from the remote user.

When an FTP connection is established, FTP sends the message:

```
220 hostname FTP server (AMOS TCP/IP v1.x) ready.
```

to the remote user. You can customize this message by creating the file `TCP:FTP..` FTP will display the contents of this file before the standard message. Keep each line to 75 characters or less as the FTP server will prepend four characters of its own.

Customizing

AlphaTCP reads the FTP users file, `FTPUSR.`, to verify login information when a remote user connects to the local computer using the FTP server. The FTP server does not read user names from the `USER.SYS` file (TELNET, however, does). This gives you greater control over who can use FTP.

Edit the file `FTPUSR.` The prototype file looks like this:

```
samantha 1PaSwOrD2 dsk5:[101,0] 3
william 3pAsWoRd4 dsk5:[102,0] 3
root 1d2IF3icu2LT dsk0:[1,2] 15
anonymous * dsk5:[377,376] 1
```

There are four or five parts to each line (each line in the prototype file has only the first four):

1. The user login name.
2. The user login password.
3. The account to log into upon a successful login.
4. The user privilege level.
5. An optional file protection flags override.

Each part must be delimited by a single space only. If you require multiple word entries, separate the words with something other than a space, such as an underline or period.

All names and passwords in `FTPUSR.` must be a single word with no spaces, though you can include punctuation characters. Entries are case-sensitive—**A** is different than **a**.

As in the other files, a `#` marks a comment line.

The password should contain case differences and a combination of letters and numbers. This makes it more difficult to guess the password through use of a program feeding dictionary words or random numbers to the software. A password of * tells AlphaTCP to accept any password. Any login with this password should have a very restricted privilege level. Common login names to use are *ftp* and *anonymous*.

The user login account can be either a device name and account number, as shown, or an ersatz device name. If you enable the new tree view, the login account does not need to be a ppn. It may be a disk device or an AlphaNET CPU ID (including 0-, which will present all local disk devices).

The user privilege level is determined by adding the flags for each of the functions you want to allow to the user. The privilege flags are:

- 1 = User can read files
- 2 = User can write and delete files
- 4 = User can log to other accounts
- 8 = User can access remote AlphaNET files
- 16 = User can access accounts containing PUBLIC.PPN

For example, in the prototype file, user “root” has a privilege level of 15. Since this equals 1+2+4+8, this user can read or write files (including remote AlphaNET files), and log to other accounts. The users with a privilege level of 3, however, can only read and write local files but cannot log to other accounts nor access remote AlphaNET files. Users with a privilege level of 17 can only access files and log to accounts containing the file PUBLIC.PPN. Privilege flags 4 and 16 should never be used together.

The optional file protection flags override allows you to override the default of 0505051717 on extended logical devices. Files created with 0505051717 cannot be deleted or overwritten via a network, including AlphaNET and AlphaTCP file utilities. See the *AMOS Monitor Calls Manual* for more information.

Once you have entered all valid FTP users, save the file and exit the text editor.

Many options are available on the start line of the CONFIG. file, as follows:

```
start ftpd 50k {-e} {-l} {-m} {-n} {-o} {-p} {-Pport} {-r} {-tseconds} {-u}
```

The optional switches have the following meanings:

- e Electronic mail address should be supplied as password for guest logins. By default, guest logins can type anything as a password. Using this switch, they must type something at least 5 characters long containing an “@”.
- l All logins and file transfers are sent to the network error logger. The FTPLOG program may be used to process the network error log and record the activity. See Chapter 6.
- m Causes multiline responses, such as used to display the TCP:FTP. file, to have a status code prepended to every line. Though a violation of the FTP specification, many servers do this and many clients now require it.
- n Changes the format returned by a LIST command (FTP dir command) to be more usable by a graphical FTP client. Although LIST is defined to be human-readable while NLST is machine readable, many graphical FTP clients try to process the output of a LIST operation. The default output of a LIST does not match any system type selectable.
- o Overrides the AMOS file protection flags. By default, connections are treated as “network connections”. This prevents files on traditional directories from being deleted or overwritten. This switch causes connections to be treated as logins to the host, thus allowing traditional files to be deleted or overwritten.
- p Changes the PWD command to return ERSATZ names whenever possible. By default the PWD command always returns device and PPN even if an ersatz exists.
- Pport Sets the port number for this FTP server to *port*. This lets you start multiple instances of FTPD on the same computer, each listening for connections on a different port.

Remember, each server you start adds to AlphaTCP’s overall memory requirements.
- r Prevents the reverse-lookup of hostnames. With this switch, only the IP address of connecting clients will be logged.
- tseconds Sets the number of seconds FTP will wait for a command before disconnecting the session. By default, no time-out is active.
- T Enable tree view. Also requires -n and an ersatz, TCPFTP:, where temp files may be written.
- u Allow read/write of OPR:USER.SYS and TCP:FTPUSR. (security risk)

Once you’ve entered all valid FTP users, save the file and exit the text editor.

Security

FTP login names are handled separately from other logins. Where TELNET by default uses LOGON and USER.SYS, FTP keeps the login names and passwords in the file TCP:FTPUSR.. Along with name, password, and account information there is a user privilege level. This determines user capabilities, so set it properly.

Privileges for users connected via FTP are limited to those defined for the user name they are logged in under. FTP has these general restrictions:

- FTP cannot execute programs and is limited primarily to file transfers.
- Files cannot be copied into any account other than the login one no matter what privilege level the user is.
- File “get” is limited to the login account if the user does not have log privileges, and accounts containing a PUBLIC.PPN file if granted that privilege.
- Get is allowed over AlphaNET only if the user has net privileges.
- If a device has no access set, FTP cannot copy files to or from it. It may display a directory, however. Failure because of no access is displayed as `device does not exist`.
- File protection flags are checked unless the `-o` switch is used. TCP/IP is treated as group 5. See the *AMOS Monitor Calls Manual* for more information.

FTP cannot copy the USER.SYS file or the TCP:FTPUSR. file. This provides some security for TELNET and FTP user names. However, if there are copies of these files with different names, such as backup files, FTP will happily transfer those and give away the whole show. Also, these files are accessible through a TELNET login (see the section on the TELNET server).

Passwords should not be easily guessable and should preferably contain both upper and lower case letters, and numbers, to make it difficult for a program accessing a dictionary to guess a password.

Tree View

FTPD may now display the AMOS filesystem as a heirarchical tree. This will allow someone with a graphical FTP client to traverse the accounts and disk devices on a system.

The following requirements must be met to enable tree view:

- AlphaNET NETSER must be running
- RPC.SYS must be loaded in system memory with /N
- The ersatz TCPFTP: must reference a ppn for temp files
- The `-T` and `-n` switches must be enabled on FTPD

Only users with log or PUBLIC.PPN privileges are allowed to move around the directory tree.

Some graphical FTP clients do not support moving up from the login directory. In this case, the users login should be changed to a "virtual account". For instance, if the login is DSK0:, the user will see a list of ppns on DSK0:. If the login is 0- (the AlphaNET ID for "this machine"), then the user will see a list of disk devices.

MAIL SERVER (SMTPD)

The MAIL server (SMTPD) handles the transfer of electronic mail both into and out of AMOS. Access to the local mail package is provided through an electronic mail driver provided with the mail package.

Requirements

If you are using DNS (Domain Name Service), the MX (Mail Exchanger) records must be properly setup to route mail to your host. This must be done by the administrator of your domain. If you are not using DNS, all hosts must be listed in the TCP:HOSTS. file. It is *very* important this is done correctly.

The ersatz TCPMAL: must be created for an account where SMTPD may write temporary files.

A special user must be set up in the mail package. By default, this special user is *Internet*. The Internet user name may be overridden if the mail package uses a different naming convention.

There must be a postmaster defined, someone who is responsible for administration of the mail and can answer questions about mail users and the site. By default, SMTPD will attempt to send the mail to *Postmaster*. You may override this and forward mail destined for the postmaster to a real user in the mail package. If you chose not to do this, then you *must* have a user Postmaster in the mail package.

You should load the SMTPD.LIT and SMTPD.RTI files in system memory.

Operation

SMTPD operates as both a client and a server. When a connection is requested, SMTPD forks a copy of itself to handle the incoming mail. Also, periodically SMTPD forks a copy of itself to bridge mail between AlphaTCP and the local mail package. After this completes, SMTPD checks to see if there is now mail to be sent out. If so, SMTPD forks copies of itself to handle the outgoing mail. If you are using DNS, multiple hosts may be accessed (based on MX records) until a connection is made. In any case, should all connection attempts fail the delivery will be retried in varying intervals for up to 32 hours. After this, the mail will be bounced back to its sender.

Customizing

SMTPD may be customized through command line options as well as configuration files. There are four configuration files SMTPD recognizes, all of them in the TCP: account. The files are SMTPD.LOC, SMTPD.ALI, SMTPD.FWD, and SMTPD.FOR.

Defining Host Names as Local Names

SMTPD.LOC lists additional host names, besides the host's fully qualified domain name, which represent local mail. For example, assume your host is called `mail.widgit.com` and is the primary mail host for your company. To allow people to send mail to a generic address like `user@widgit.com` you must set up an MX record in DNS to pass mail for `widgit.com` to `mail.widgit.com`. You would also list the name `widgit.com` in the SMTPD.LOC file. This tells SMTPD to pretend all incoming mail addressed to `widgit.com` was really meant for itself and don't try forwarding it elsewhere.

Create the file TCP:SMTPD.LOC and enter the host names to be treated as local, one per line.

Forwarding Mail to a Remote Computer that Uses a Name Server

If you are not using DNS, but access a computer that does, you can forward mail for unrecognized hosts to that remote computer. Create the file TCP:SMTPD.FOR and enter the host name of the computer to forward to on the first line. If you wish to use the host's IP address instead, place it inside square brackets. Example: [192.168.0.1]

When the server is able to resolve a hostname (it's in TCP:HOSTS. or you really are using DNS) it will usually deliver the message directly rather than use the relay in SMTPD.FOR. You may force the server to always use the relay by specifying the -q switch. This can be handy if you always want mail delivered through a single host (for instance, if the AMOS host is behind a firewall).

Limiting Hosts and Users That Can Use SMTPD to Forward Mail

To prevent malicious users from forwarding their junk mail through your server, specify the -o switch on the SMTPD command line and create the file TCP:SMTPD.FWD. List valid destination and source lines in the file as follows:

```
TO host-or-domain-name
```

This allows any connecting host to send mail to the specified destination. If your site is listed as a backup SMTP host for other hosts or domains, they must be specified on TO lines in the file.

```
FROM ip-address
```

This allows only certain hosts, or groups of hosts, to use SMTPD to forward mail. Portions of the IP address may be replaced with an asterisk as a wildcard. Usually you would place a wildcard IP address for your entire domain on a FROM line in the file. If you have remote users with static addresses, you could also allow them to use SMTPD by placing their address on a FROM line. Your remote users with dynamic address may always forward mail to any address if they login with proper authentication. Please remember this is for outgoing client mail only and is independent of the POP3 username and password used to receive mail. Non-authenticated users can still connect to and use the mail server, but mail forwarding is restricted to the rules contained in the SMTPD.FWD file. Contact Alpha Micro Technical Support for Username/Password information.

Client email programs which support outgoing mail authentication have various methods of implementations. Some implementations may be incompatible with AlphaTCP. Please consult the instructions for your mail client for configuration information. Here is a Microsoft Outlook Express version 5.5 example:

From the tools menu select Accounts. Then select the Mail tab. Next, select the desired email account and click on Properties. Select the Servers tab, then check the box under Outgoing mail server that says "My server requires authentication." Next, click on Settings, then select Logn using and enter the account name and password.

Defining Aliases for Users with Mail Addresses on Other Hosts

SMTPD.ALI maps user name aliases with actual mail addresses on other hosts. This allows a consistent name and address format to be used outside a company, while allowing differing names and hosts inside. For example, assume you have two users on different computers in the company: John Doe's real mail address is `johnd@sales.widgit.com` while Jane Doe's is `jdove@corp.widgit.com`.

You would place the following entries in the SMTPD.ALI file:

```
John_Doe    johnd@sales.widgit.com
Jane_Doe    jdove@corp.widgit.com
```

Now, users outside the company will be able to send mail to `John_Doe@widgit.com` and `Jane_Doe@widgit.com`. This is also very handy for field personal. Using an alias, they may be provided with a corporate address which actually forwards to their mailbox at a national Internet service provider.

Normally SMTPD will check for a mailbox in the local mail system first (for example, in USER.SYS when using AlphaMAIL). If a user has a local mailbox, the SMTPD.ALI file won't be used. You can force SMTPD to check the SMTPD.ALI file first with the `-a` switch. This can be handy if you want to temporarily override a users local mailbox and forward their mail elsewhere. Note, however, that any mail sent between local mail users (such as two users running AlphaMAIL locally) can only be delivered to the users local mailbox.

CONFIG. Startup Options

The following command line options are available on the start line for `smtpd` in TCP:CONFIG.:

```
start smtpd 125k {-a} {-b} {-ccount} {-d} {-fsize} {-h} {-i} {-lname}
{-mcount} {-n} {-o} {-pname} {-q} {-scount} {-tminutes} {-u} {-xcount}
{-z}
```

These options perform the following functions:

- `-a` Forces SMTPD to check the alias file (SMTPD.ALI) before looking for a local mailbox. Normally, a local mailbox takes precedence.
- `-b` Tells SMTPD not to perform uudecode on incoming uuencoded binary files.
- `-ccount` Changes the default client limit to *count*. The default is ten, which means a maximum of ten outgoing mail connections are allowed at any one time. If more outgoing mail exists, it will wait until other clients exit.
- `-d` Turns on debug mode. In this state, information about SMTPD operation and connection traces are sent to the network logger. You may then view the network error log to diagnose delivery problems. As much information is logged by this mode it is advised to leave it off for normal operation. The status of debug may be changed by sending a `usr2` signal to the primary SMTPD process. See UKILL in Chapter 6 for further detail.
- `-fsize{K}{M}` Places a file size limit on incoming mail. After a mail message reaches this

- size, the rest of the message is truncated and a line stating this is appended to the message. By default, no limit is imposed.
- h Tells SMTPD to strip the host name when appending a source address to a local address. This is used when you are setting up to address mail to the domain name. For example, if the full name of the system is `sales.widgit.com`, using the `-h` switch causes outgoing mail to appear to come from `widgit.com`.
 - i This switch should be enabled when using DNS. It causes mail delivery to be retried for failures to read the host's MX records. When using a host file, do not include this switch.
 - lname Overrides the default Internet special user name. If *name* is multiple words, enclose the entire option in quotes. Example: `"-lOther Name"`.
 - mcount Sets a recipient limit. Older versions of SMTPD had a build-in limit of 6.
 - n Notifies the postmaster when mail bounces by sending a courtesy copy of the bounce notice.
 - o Causes SMTPD to check TCP:SMTPD.FWD before accepting mail destined for another site.
 - pname Changes the name of the postmaster to *name*. By default, the postmaster name is `Postmaster`. If the name contains multiple words, place the entire option in quotes. Example: `"-pJohn Doe"`. All incoming mail addressed to postmaster will be forwarded to the specified user.
 - q Forces SMTPD to pass all mail through the relay host in SMTPD.FOR. Without this switch, SMTPD will only pass messages to the relay which it couldn't resolve a destination for itself.
 - scount Changes the default server limit to *count*. The default is ten, which means a maximum of ten connections will be accepted. If more connections are attempted, they will be refused with a request to try again later.
 - tminutes Changes the default time-out limit to *minutes*. The default is four minutes. This time-out is used in the client to determine how long to wait for a response to a command. It is used in the server to determine how long to wait for a command request.
 - u Enables the user list mailback option. When this option is active, any incoming mail addressed to users will generate a return message. This message will contain the names of all users in the mail package. It will also list all users in the TCP:SMTPD.ALI file.
 - xcount Changes the maximum *Received* header lines from the default of 20.
 - z Attempt to map internal mail addresses to aliases. This scans the SMTPD.ALI for the local mailbox name. If found, the From: address is converted to the alias on outgoing mail.

Kill File

In some circumstances, such as harassment or the activities known as “mail bombing” or “spamming,” it can be useful for the mail administrator to refuse mail from an individual or a site. Listing users or sites in the kill file causes SMTPD to refuse all mail from them.

To refuse mail, edit or create the TCP:SMTPD.KIL file and enter the e-mail addresses, one per line. To refuse mail for an entire site, list the at-sign (@) and site only. For example, the following two lines would refuse mail from a harassing user and a site mailing unsolicited ads. These are both hypothetical:

```
naughty@someisp.net
@fastmoneyspam.com
```

If you add items but e-mail continues to arrive from the sites, the mail headers may have been modified. In this case, you can enable debug mode on the mail server until one of the unwanted messages arrives. You can then examine the network error log and locate the incoming session. The address you want to add to the kill file will be found in the logged MAIL FROM: line. Don't forget to disable debug mode again or your network log file could fill up rapidly.

Runtime Control

The primary SMTPD job will respond to ‘usr1’ and ‘usr2’ signals sent using UKILL. The ‘usr1’ signal will force SMTPD to wake up early and attempt a delivery cycle.

The ‘usr2’ signal toggles the debug mode on or off (the same mode which may be enabled using the -d switch on the SMTPD command line). This can be useful if you are having delivery problems with a message and only want to log activity for the duration of the delivery attempt. For information on UKILL and sending signals, see Chapter 6.

Security

The primary security concerns with SMTP are “name spoofing,” message modifications along the delivery path, junk email, and privacy. These are concerns with most implementations of electronic mail today.

“Name spoofing” is faking the name of the sender. Hackers may use this to get passwords, pretending to be someone of authority. The same risk is present on the telephone. Never give out private information in response to a mail message, just as you shouldn't in response to a phone call.

Modifications can be made to the message by a dishonest system operator. Be sure your MX records point to trusted computers.

Anyone running an SMTP server on the Internet needs to prevent it from being used as a distribution point for junk email, also known as *spam*. If your site is found, your server will be used to mail tens of thousands of messages which often results in a backlash from angry recipients. Older versions of SMTPD had a small recipient limit, which made it a poor target site. The latest SMTPD no longer has this recipient limit, but also has options to prevent such misuse. Be sure to use the new security features, and test your installation!

Finally, as messages are transferred in plain text it is possible the message could be read by more than the intended recipient. In addition, messages could be stored on backup tapes for years. Don't pass anything embarrassing or incriminating over electronic mail.

POST OFFICE AND PASSWORD SERVERS (POP3D AND POPPWD)

NOTE: A new server, QPOP3D, is included in the AlphaTCP 1.4A release. The section below applies equally to both the original POP3D and new QPOP3D servers. The new server estimates the message sizes it reports to the client. As a result, it avoids the lengthy login and occasional timeouts when a mailbox contains many messages or large attachments. Since this behavior is not standard, it may not work with all POP clients. As an added benefit, the new server supports UIDL, allowing a clients "leave on server" option to work.



On March 31, 2004, the original Post Office server, POP3D, was discontinued and is no longer supported or maintained. As a convenience to existing installations, current releases of TCP contain a POP3D that is identical to the QPOP3D post office server so that either name can be used. Additionally, the older, unsupported version of the post office server is still available as OPOP3D.

The Post Office Server (POP3D) handles the retrieval of electronic mail by hosts which are usually not on the network or have limited memory resources. (A typical example of such hosts is a PC.) For these hosts it may be impractical to support a full SMTP implementation. Using the Post Office Server, the mail is stored on the AMOS host in the local mail package for later retrieval. Access to the local mail package is performed through an electronic mail driver provided with the mail package. The Post Office Server is a retrieve-only server. To send mail, the remote host should contact an SMTP server.

The Password Server (POPPWD) supports the Eudora *change password* feature, and usage is optional.

Requirements

An electronic mail driver for the local mail package must be loaded in the system initialization file. The POP3D.LIT and POP3D.RTI files must be loaded in system memory.

The POP3D server requires 60K of memory to run, and should be started in the TCP:CONFIG. file using a **start** command. (See "Customizing," for options that can be used.)

You must create the ersatz TCPPOP: for an account where POP3D may write temporary files.

The POPPWD server requires 70K of memory to run and should be started in the TCP:CONFIG. file using a **start** command. There are no custom options for this server.

Operation

POP3D operates as a server only. When a connection is requested, POP3D forks a copy of itself to handle the session. After the remote computer performs a POP3 login, POP3D queries the local mail package for a list of messages in the new mail folder. It then retrieves all of the messages and places them in the temporary account. From there, the remote host may retrieve various messages and op-

tionally flag them for deletion. Once the remote host closes the session, any messages flagged for deletion are then removed from the local mail package.

POPPWD listens for connections from Eudora clients. Upon validation of the current login and password, the new password is written into the POPUSR. file. Successful and failed attempts are logged to the TCP:POPPWD.LOG file.

Customizing

You can customize POP3D by using command line options as well as a user login file.

The file TCP:POPUSR. contains the login, password, and mail name of each Post Office user. Each user is defined on a separate line. The login and password should be unique to the Post Office. Do not use the real user login on the computer. No embedded spaces are allowed for these two entries. The mail name entry may contain embedded spaces and must match the name used by the user when directly accessing the local mail package. An example line would be:

```
j_doe    jd_paswd1    John Doe
```

The remote computer would be configured to login as `j_doe`, with a password of `jd_paswd1`. POP3D will then access mail for user John Doe.

The following command line options are available on the start line for POP3D in TCP:CONFIG.:

```
start pop3d 60k {-d} {-h} {-scount} {-tminutes}
```

These options perform the following functions:

- d Turns on debug mode. In this state, information about POP3D operation and connection traces are sent to the network logger. You may then view the network error log to diagnose problems. As this mode logs a lot of information, we suggest you leave it off for normal operation. The status of debug may be changed by sending a “usr2” signal to the primary POP3D process. See the section on UKILL in the *AlphaTCP Administrator’s Guide* for further detail.
- h Causes the host name portion of the full domain name to be stripped. This changes the source address in messages which have not traversed SMTP. Normally, the full domain name of the local computer running POP3D is appended to the user name (i.e., John Doe appears as `John_Doe@foo.company.com`) if `foo` is the host running POP3D. Using the `-h` switch causes `foo` to be stripped (i.e., the address is now `John_Doe@company.com`). This is useful if your site is running POP3 servers on multiple hosts but has a primary host handling all SMTP mail for your domain.
- scount Changes the default server limit to *count*. The default is twenty, which means a maximum of twenty connections will be accepted. If more connections are attempted, they will be refused.
- tminutes Changes the default time-out limit to *minutes*. The default is four minutes.

This time-out is used to determine how long to wait for a command request.

Runtime Control

The primary POP3D job will respond to a 'usr2' signal sent using UKILL. The 'usr2' signal toggles the debug mode on or off (the same mode which may be enabled using the -d switch on the POP3D command line). This can be useful if you are having problems with a session and only want to log activity for the duration of the session. For information on UKILL and sending signals, see Chapter 6.

Security

The primary security concerns with POP3D are privacy and passwords.

As all data is sent in plain text, it is possible for a host along the connection path to trap and read the mail or login sequence.

The privacy concern is no different than over the SMTP delivery path which delivered the mail in the first place.

If you use logins and passwords in the TCP:POPUSR. file which have nothing to do with the real login (or the FTP login), trapping of POP3D logins should be of little use for breaking into a computer.

The primary security concern with POPPWD is the ability of users to change their own password to something less secure, or to one matching their real login password.

LINE PRINTER CLIENT (LPR)

The line printer client provides access through the standard AMOS print utilities to printers on UNIX systems and TCP/IP based print servers.

Requirements

You must have a spooler defined in the system initialization file for each TCP/IP based printer you wish to access.

The TLP.DVR driver must be included in the system initialization file. It should be on a DEVTBL line, as well as placed in system memory with a SYSTEM command. LPR.LIT and LPR.RTI should also be loaded into system memory.

The ersatz TCPLPR: must be created for an account where LPR may write temporary files.

To use dynamic printers, AMOS must be version 2.3A or later.

Operation

The spoolers output print files through the TLP driver. In turn, LPR writes the file to a temporary file in the TCPLPR: account. When the file is complete, LPR forks a copy of itself to contact the print server and transfer the file to it. Only one server per TCP/IP printer is active at any time.

Active print requests are stored in the TCPLPR: account. Two files are created for each print request: a .CF and a .DF file. The .CF file contains LPR protocol information and the .DF file contains the print data. If the system is booted while print files are queued, they are automatically resent when LPR is started.

Customizing

The following command line options are available on the start line for LPR in TCP:CONFIG.:

```
start lpr 35k {-d} {-f} {-p} {-t} {-u} {-x}
```

- d Enables debug mode. This causes transient failure messages to be sent to the network error log.
- f Passes forms information to the remote. This is non-standard and only useful when the remote is the AlphaTCP LPD Server. You must also be running the latest spoolers configured with the “REMOTEFORMS” option enabled.
- p Limits source port numbers to the range 721-731 for hosts which require it. Avoid using this option as it impacts the ability to print documents rapidly in sequence.
- t Disables timeout on print requests. Normally a print file will be abandoned if a connection to the print server cannot be established within one hour. Using this switch causes the print file to be attempted forever.
- u Allows customization of the “user” name passed in the printer control file. This may be required for some hosts. By default the user name passed is “AlphaTCP”.
- x Exchanges the order of the control and data file transfer (the control file is created by lpr; the data file is the file being printed). By default, the control file is sent to the print server first. Using the -x switch causes lpr to send the data file first, which may be required by some print servers.

In the spooler configuration file, you need to define the TLP device and a unique name of up to six characters. For example, where you normally find `DEVICE=TRM:SPOOL`, place the line `DEVICE=TLP:SPOOL`. The name `SPOOL` can be anything as long as no other spooler using TLP: also uses the name `SPOOL`. If you enabled the -f LPR option, also add the line “`REMOTEFORMS = TRUE`” to the spooler configuration file.

In the file `TCP:LPR`, you need to list the above name, along with the host name and printer queue name to access. Each printer must be placed on a separate line. The format of the line is as follows: `NAME=hostname, queuename` where `NAME` matches the name after TLP: in the spooler. (Note that `NAME` must be all uppercase.) For example:

```
SPOOL=sales.widgit.com,orders
```

There is one other argument on the printer definition line. It is optional and likely only to be needed for IBM mainframe computers. It allows you to specify the class name passed in the control file. Simply append a comma and the exact class name string you wish to use to the end of the printer definition line.



If the print server is listening on a port other than the default of 515, the port number may be overridden by following the host name with a colon and the port number. This is very rare. If the device is not using port 515, it's likely not using the LPR protocol. If the print server in the previous example was listening on port 3000, the following line would be used:

```
SPOOL=sales.widgit.com:3000,orders
```

Defining Dynamic-IP Printers

Many users of dialup TCP/IP services are assigned a different IP address each time they connect to the Internet. This is known as dynamic IP addressing. Obviously, if someone has a different IP address each time he or she connects to your computer, you cannot assign his/her local printer an AMOS printer name in LPR. To allow these users to print to their local printers, you can assign one or more printer names to be used as dynamic-IP printers. A dynamic-IP printer is basically a printer “slot” which has a name but no physical printer or IP address defined. The remote user can use the MYLPR command (discussed in the *AlphaTCP User's Guide*) to assign the printer name to his/her local printer temporarily.

To define a dynamic-IP printer:

1. Create a spooler initialization file defining the printer name and device name, as described above. As with a permanent TCP printer, the device name is TLP:*SPOOL*, and *SPOOL* must be a unique name to use for this printer. For example: DEVICE=TLP:TEMPPC.
2. In the LPR. file, add a line for this printer name. Instead of the host name, enter an asterisk. For example: TEMPPC=*, queue1

That's all there is to it. Any remote user can now use the MYLPR command with the device name you've assigned the dynamic-IP printer (in our example, TEMPPC), then print reports from your AMOS computer to a local printer.

You can assign multiple dynamic-IP printers by repeating the above process with different printer and device names.

If anyone prints to a dynamic-IP printer, but the printer is not currently assigned, LPR queues the print request until someone uses MYLPR to assign the printer. If someone changes the printer assignment by using MYLPR while a file is printing, the current file finishes printing on the currently assigned printer; any other files in the queue, and new print requests, go to the newly assigned printer.

Printer Information

You can retrieve various printer information by opening the TLP device for input and reading. You can also use the TYPE command with the TLP device name in place of a file name. For example, to print the status of a TLP device named SPOOL (as in the previous section) to the screen, enter:

```
TYPE TLP0:SPOOL 
```


This displays five pieces of information:

Host	The name or IP address of the remote printer as listed in the LPR. file.
Queue	The name of the print queue as listed in the LPR. file.
Active	The number of seconds a server has been spawned working on one or more files.
Files	The number of files currently being processed by the spawned server above.
Queued	The number of files waiting to be processed. These will start processing once the current spawned server finishes.

Security

The only security concern for the line printer client is related to dynamic-IP printers. Because you don't know at any given time where the physical printer with this printer name is located, it's possible to print a report on a printer halfway around the world. This is especially a concern because of the dynamically assigned IP addresses from dialup services. Consider this scenario:

1. One of your remote users accesses your computer through a dialup service, with a dynamic IP address. She uses MYLPR and prints a report to her local printer.
2. She forgets to use MYLPR/U to unassign the printer name, and logs off. The printer is still defined at the IP address she had been using.
3. Someone else using the same dialup service gets assigned the same IP address your user had been using.

If this new, unknown user is running a print server, his printer is now defined as a TCP printer on your computer! If anyone prints to this printer name, the report will print on this unknown printer.

To help avoid this problem, LPR unassigns the printer itself upon any error that might mean the user has dropped off, such as host unreachable, port unreachable, or TTL exceeded.

LINE PRINTER SERVER (LPD)

The line printer server allows TCP/IP based hosts to access AMOS printers using the Line Printer protocol.

Requirements

The ersatz TCPLPD: must be created for an account where LPD may write temporary files. The LPD.LIT and LPD.RTI files should be loaded into system memory.

Operation

Hosts wishing to print a file contact the LPD server. The LPD server forks a copy of itself to handle the connection. A control file and data file are then transferred and stored in the TCPLPD: account. Once

this is done, the file is sent to the AMOS spooler matching the printer queue name in the control file and the connection is closed.

Customizing

The following command line options are available:

```
start lpd 35k {-foption} {-r}
```

- fD Submits files with the default form feed setting for the printer. This is the same as not specifying any switches at all.
- fN Submits files with the/NOFF (no formfeed) option.
- fY Submits files with the /FF (formfeed) option.
- r Causes LPD to ignore formatting options in the control file, treating the print file as raw data. Useful for misconfigured LPR clients, but rarely needed.

Security

The only security concern for the line printer server is the ability for any TCP/IP host to access your printers. If your network is large enough that this is a problem, or you are attached to the Internet, you may want to set up a firewall router and filter packets to the LPD server port.

WEB SERVER (HTTPD)

The web server allows graphical web browsers such as NCSA Mosaic to retrieve HyperText documents, sound, and graphics files from AMOS using the HTTP protocol.

Requirements

The HTTPD.LIT and HTTPD.RTI files should be loaded into system memory. An ersatz file, such as WEB.ERZ, should be defined which contains all accounts the web server will reference. The web server cannot use account numbers directly.

The TIMZON. file must be properly created in the TCP: account.

The MIME.TYP file must be properly created in the LIB: account. This file is available as the example file MIME.NEW in the release. The format of this file is documented in Appendix C.

The HTTPD.HTM primary document must be present in the TCP: account.



Any accounts referenced by documents must contain the file PUBLIC.PPN to be accessible. The content of this file is unimportant. If a request is made for a file and the PPN does not contain this filename, the request will be refused. Note this does NOT include the primary document, TCP:HTTPD.HTM. You should NOT place a PUBLIC.PPN file in the TCP: directory as it would allow retrieval of your FTP password file! With AlphaTCP 1.5, you may now open entire devices for public access by listing them in HTTPD.DEV. This eliminates the need to have PUBLIC.PPN in any accounts on that device.

Operation

When a web browser contacts the web server it forks a copy of itself to handle the session. The browser then passes a request for a file, possibly an HTML document or a referenced image, which the server returns to it. The connection is then closed.

A browser will often establish a connection to read an HTML document, then open one or more connections to retrieve any images referenced by the document.

Documents reference images and other documents via a Universal Resource Locator (URL). The URL is a path to the document or image specifying the transfer method, the host, and a UNIX-style path. A typical URL looks like this:

```
http://www.widgit.com/sales/products/prices.html
method- ^ ^--- hostname -- ^---UNIX-style path-----^
```

The AMOS web server uses the same format, using ersatz names to hide the AMOS directory structure. For example, the following URL references the SALES:NEPRIN.HTM file on an AMOS system:

```
http://www.widgit.com/sales/neprin.htm
      ^-ersatz-^AMOS file name
```

The filename conversion function described in Chapter 2 of the *AlphaTCP User's Guide* is also used. This allows your URL to look more descriptive than the AMOS filename it references. For example, the above example filename is rather meaningless. However, the following URL references the exact same file:

```
http://www.widgit.com/sales/new-products-info.html
```

You may choose to understand the name conversion function, or you may simply use the FNU2A program to determine the resulting AMOS filename for you when creating URLs. For example, run the following command:

```
FNU2A sales/new-products-info.html
```

This displays the resulting AMOS file specification: `sales:neprin.htm`. This is where you should place the document under AMOS.

Details on creating documents are beyond the scope of this document. There are many books about writing HTML and Web publishing. For information on supporting fill-out forms and image maps, see Chapter 9.

Default Documents

There are two defaults which the web server uses when portions of the URL are missing.

The first default is used when a URL does not contain any path information beyond the host name. For example:

```
http://www.widgit.com/
```

To the AMOS web server, this means no directory or filename information was provided. The web server will return the TCP:HTTPD.HTM file as the master web page. You can change this default file using the `-m` option, described below.

The second default is used when a URL contains path information but no filename. For example:

```
http://www.widgit.com/sales/
```

The AMOS web server assumes the *sales* portion should be used as an ersatz name and returns the HOME.HTM file in the SALES: directory. If the URL does not end in a forward slash (/), the browser is redirected with one that does. This allows relative references to function. If you advertise a URL without a filename, showing the URL with a terminating forward slash will reduce your system's overhead.

Special File Requests

There are a couple of files in the root directory sometimes requested by a client which have a special meaning. Normally the only file which the web server allows access to in TCP: is the primary HTTPD.HTM file. The web server also allows access to the following files in the TCP: account:

robots.txt

This file is usually requested by search engines. The format of this file, the Robot Exclusion Protocol, is documented in various HTML books and on the Internet.

favico.ico

This file is requested by Internet Explorer when a user adds your site to their *Favorites* list. It allows your site to replace the default icon IE5 places next to the link. The actual filename requested in the URL is favicon.ico, which is truncated to six characters for AMOS.

Customizing

There are a number of command line options which may be placed on the start line for httpd in TCP:CONFIG.:

```
start HTTPD 100k -aaddress -d -l -mersatz -scount -tminutes
```

- address* Overrides the default e-mail address for the Web administrator, which is 'postmaster' at the local host name. For example:
- ```
postmaster@www.widgit.com
```
- If there is a problem, users may send an e-mail message to this address notifying you.
- d* Enables debug mode. This causes connection information to be written to the network logger.
- l* Writes access information to HTTPD.LOG. This logfile is in standard NCSA format and may be processed by tools and services supporting this standard. It only records IP addresses, not hostnames. When activated, this file can grow rapidly.
- ersatz* Lets you change the default “home page” sent when a request does not include a file name or ersatz name. The *-m* option changes the default file sent from TCP:HTTPD.HTM to HOME.HTM in the account with the ersatz name *ersatz*.
- scount* Specifies the maximum number of servers which may be active; by default it is 20. If more connection requests arrive while there are this many active sessions they will be refused.
- tminutes* Specifies the time-out in minutes in which a request must be made; by default it is four minutes. If a request has not been issued by this time, the connection will be dropped.



There is a great advantage to placing your web documents and images on an extended directory device, which contains date stamps on the files. When a request arrives for a file on an extended directory, the server is able to examine the modification date. If it has not been modified since the browser last requested it, the browser is told to use its own copy. This avoids transferring the file again and can be very nice for users with slow connections.

## Specialized Filename mapping

HTTPD typically converts filenames to an AMOS format automatically. Usually you can manipulate a URL to generate a unique AMOS filename. There are certain cases, however, when you do not have control over the URL. One example would be a library of Java code. In this case, it is quite possible for the library to use unique URLs which map to the same AMOS filename.

For this reason, HTTPD supports a file, TCP:HTTPD.CVT, which may be used to manually map URL filenames to unique AMOS filenames. HTTPD will strip the filename from the URL and attempt to locate it in this file. If found, the entire URL is replaced with the AMOS filespec. The filename is scanned for in a non case-sensitive manner. An example TCP:HTTPD.CVT file follows; all examples would normally access the same AMOS file: **really.cla**

Example:

```
Convert canned URL filenames into unique AMOS filenames
ReallyBigFirstName.class rbfnam.cls # example 1
ReallyBigOtherName.class rbonam.cls # example 2
ReallyReallyBigName.class rrbnam.cls # example 3
```

## Page Redirection

If you move a page to another account or host, but need to support the previous URL, you may redirect the page to the new URL. This is done by replacing the old page with a file containing the following line:

```
Redirect: {new url}
```

## Publicly Accessible Devices

If you wish to open a device for public access without adding PUBLIC.PPN to every account, create the file HTTPD.DEV. Place each public device, for instance **sub4:**, on a line by itself. The web server does not need to be restarted.

## Runtime Control

The primary HTTPD job will respond to a 'usr2' signal sent using UKILL. The 'usr2' signal toggles the debug mode on or off (the same mode which may be enabled using the -d switch on the HTTPD command line). This can be useful if you are having problems with a session and only want to log activity for the duration of the session. For information on UKILL and sending signals, see Chapter 6.

## Security

By manually entering a URL, users may request any file they wish. The web server limits this by verifying the requested account contains the file PUBLIC.PPN. If this file does not exist, the request is refused. Do not place a file called PUBLIC.PPN in any account which should not be accessible to the public. This is especially true of any account which might contain passwords, such as [1,2], [1,4], and [7,50].

Since fill-out forms run external programs, there are security issues with those programs. Only administrator-configured programs may be run, so the risk is limited to the programs themselves. See Chapter 9 for more information on fill-out forms.

## ITC TUNNELING SERVER (ITCD)

The ITC Tunneling Server (ITCD) allows AlphaNET communications to be performed over a reliable TCP/IP connection. This allows native AMOS applications access to the reliability of TCP/IP without modification.

## Requirements

Only one copy of ITCD should be started. Do not attempt to start multiple ITCD servers. All systems which will be accessible via ITCD, including the local one, must be listed in the TCP:ITCD. file. This file provides the link between AlphaNET CPU IDs and the TCP/IP IP address.

The CPU IDs used by ITCD must be part of a unique AlphaNET network number. They must not use the same network number as an interface started using the NETINI command.

ITCD does not support the broadcast feature of AlphaNET. Broadcast sends are refused with 'node does not exist' error codes.

## Operation

When the ITCD server starts up it scans the ITCD. file to determine its AlphaNET address. It then creates a network entry with a pointer to an internal NDV interface, like NETINI usually sets up. Whenever an ITC message is passed to the NDV it either gets sent down an existing TCP connection, or the connection is then established. After a period of time with no data, the TCP connection is shut down.

## Customizing

There are a number of command line options which may be placed on the start line for ITCD in TCP:CONFIG.:

```
start ITCD 100k -d -pport -tseconds
```

- |                        |                                                                                                                                                                                                                                         |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>-d</code>        | Enables debugging mode. When set, information about connection establishment or shutdown is written to the network logger.                                                                                                              |
| <code>-pport</code>    | Allows you to select a port number for the server. By default, the server looks for the 'amos' service in the TCP:SERVIC. file.                                                                                                         |
| <code>-tseconds</code> | Overrides the default connection time-out value of 120 seconds. This is the number of seconds of inactivity between hosts before the TCP connection will be shut down. Should more data appear, the connection would be re-established. |

AlphaNET CPU ID to IP address translations should be listed in the ITCD. file. This must include the local host as well.

Each line of the file should begin with an AlphaNET CPU id (such as 16842753). This should be followed with the corresponding IP address for that host (such as 192.0.2.1). Lines beginning with '#' or ';' are treated as comments and skipped. You may rename the example file TCP:ITCD.NEW and use that as a starting point.

## Replacing SerialNET Installations

When replacing a SerialNET installation with tunneled AlphaNET over SLIP you may want or need to change the AlphaNET addressing used. This is especially important when multiple serial lines are involved.

SerialNET requires a different network number for each serial line. Thus, if you have three SerialNET links, you have three individual network numbers. You may also have a number of gateway definitions on other systems to access these routes.

With tunneling, this requirement disappears. Tunneling needs its own network number, but not separate ones for each interface. To a program using tunneled AlphaNET, the world is one big AlphaNET network. The fact that one system is on the local Ethernet, and another is across the world over a dialup Internet account is buried in the TCP routing and hidden from AlphaNET.

On a system with multiple SerialNET connections being replaced (a hub system), set up the TCP:ITCD. configuration file with addresses for all the systems. You should use the Ethernet IP address to define the hub system rather than any of its SLIP IP addresses if possible.

The TCP:ITCD. file on the non-hub systems should match the hub system except for one item. The IP address for the hub system should match the SLIP IP address it will be using to communicate from. This is because the hub system will identify itself with a different IP address on each interface, and that IP address must be found in the TCP:ITCD. file for proper mapping.

You should also be able to remove any AlphaNET gateway definitions on the non-hub systems, since TCP handles all of the routing issues. There is a special case where you might still need to set up gateways. This is described in the next section.

## Accessing Non-Tunneled AlphaNET

You may have systems on your network which are unable to use tunneling. They could use AMOS 1.X, or simply lack the resources to run AlphaTCP. If you are using serial links or LDVs, this can pose a problem as you cannot simply define the non-tunneled network on a NETINI statement.

To solve this, use the gateway ability of AlphaNET. This involves adding NETINI statements with FORWARDTO= declarations in the setup file. This requires NETINI version 2.4(123)-2 or later on the tunneled systems. You must also pick a system which has direct access to both the tunneled and non-tunneled AlphaNET networks. This system will be used as the gateway.

As an illustration, assume network #1 is non-tunneled Ethernet, and network #2 is a tunneled network. Also assume three systems: system A is an AMOS 1.4 system with an AlphaNET ID on network #1 of 16842753-; system B is running TCP only and has a tunneled AlphaNET ID on network #2 of 33619969-; system C is using an NDV and TCP with two AlphaNET IDs, one on each network: 16842754- and 33619970-.

System C will be the gateway system. It should require no special configuration to support this as long as both networks are available to it.



On system A, you need to define the path to network #2. This is done by adding a NETINI statement after the existing one. For example:

```
NETINI NET002.NIN
```

The contents of this file would look much like this:

```
NETWORK=2
NAME=AlphaNET via AlphaTCP
FORWARDTO=16842754-
```

On system B, you need to define the path to network #1. This is done with a slightly different form of the NETINI statement, as network #2 won't exist yet as far as NETINI is concerned (it appears after AlphaTCP is started). Add this line after any existing NETINI statements, or in place of any NETFAM statement you might have. If you have neither, add it anywhere before the first SYSTEM statement:

```
NETINI/T NET001.NIN
```

The /T switch tells NETINI the gateway system is tunneled and not to complain about the lack of the destination network. The contents of the file would look like this:

```
NETWORK=1
NAME={whatever you call network #1}
FORWARDTO=33619970-
```

This completes the setup of the example networks. System A should now be able to access System B, and vice-versa. NTSTAT on either system should show a gateway to the opposite network.

## Runtime Control

ITCD will respond to 'usr1' and 'usr2' signals sent using UKILL.

The 'usr1' signal will force ITCD to disconnect all active connections. If further AlphaNET traffic arrives, the connection will be reestablished automatically.

The 'usr2' signal toggles the debug mode on or off (the same mode which may be enabled using the -d switch on the ITCD command line). This can be useful if you are having connection problems and only want to log activity for a short time. For information on UKILL and sending signals, see Chapter 6.

## Security

As ITC messages are being transferred, ITCD is open to the same security issues as AlphaNET. If you have an Internet connection, the server may be accessed by anybody.

Limiting this is the requirement that all hosts connecting have an AlphaNET address listed in the ITCD configuration file. If no entry exists for the connecting host, the connection will be dropped. IP spoofing attacks may be a concern, although most AlphaNET-based communication performs a lot of handshaking which would also have to be spoofed.

If you are concerned about IP spoofing attacks on the ITC server you may want to set up a firewall router and filter packets to the ITC server port. By default the server port is TCP port 2001, although this is not a standard and may be different at some sites.

## E-MAIL BASED FILE SERVER (BBSERV)

The e-mail-based File/BBS Server (BBSERV) allows users to request files via electronic mail. Also provided is an interface which allows a programmer to link a BBS-type service to BBSERV. This interface is described in Appendix D.

### Requirements

Only one copy of BBSERV should be started. Do not attempt to start multiple BBSERV servers.

BBSERV is set up so it can run without AlphaTCP. As such, it does not use the network logger like most servers. If you have problems setting up BBSERV, start it up on a normal job with a real terminal to see any error messages. BBSERV requires about 100K of memory to run.

BBSERV requires an account where temporary files may be created. You must define an ersatz account called BBSERV:

A file called BBSERV.HLP should be created in the BBSERV: account. This file should contain instructions on using the server. It will be mailed to any user requesting help, or whenever no other requests can be located in a mail message. The sample file BBSERV.NEW may be renamed and used as a starting point.

A mailbox must be created. This mailbox will be accessed using the electronic mail driver supplied with your mail package. By default, the mailbox name will be 'bbs'. The name 'info' is also a popular one and may be set up instead. See the following section 'Customizing' for how this is done.

If you have a BBS interface to BBSERV, there are a number of other items to set up. Refer to the documentation for the BBS for the additional requirements. If you are developing a BBS interface, see Appendix D.

### Operation

Users send requests via electronic mail to the server's mailbox. These requests are in the form of English sentences. Periodically the server reads the mail in its mailbox and carries out the requests. Any information requested is electronically mailed back using the source address of the original message. Multiple requests may be made in a single mail message, allowing many actions to be performed in a single request.

BBSERV itself understands requests for help, a session transcript, and files (items with a colon and a period).

When using BBSERV with a BBS, BBSERV also understands requests to read or post BBS topics (items without a colon or period), and to subscribe and unsubscribe from automated mailing of those topics.

Usage instructions for BBSERV are located in the BBSERV.NEW file. This file may be printed if you are configuring BBSERV. You should customize it and rename it to BBSERV.HLP so it may be requested by BBSERV users. All BBSERV actions are requested through electronic mail messages. There is no direct user interface to BBSERV itself.

## Customizing

There are a number of command line options available. If you don't have a BBS interface, enable the file retrieval option; otherwise the server will have no useful function to perform.

- f Tells BBSERV to allow file retrieval requests. File requests must contain a colon and a period to be recognized as such. Only accounts which contain the file PUBLIC.PPN are accessible.
- l *name* Changes the accessed mailbox from 'bbs' to *name*. If the mailbox name you want to use contains multiple words, place the entire option in quotes. For example: "-lINFO SERVER".
- p Enables posting to the BBS. Without this switch the BBS is read-only. This option is only valid when you have a BBS interface.
- r Tells BBSERV to append a line to the file BBSERV.ACT for each incoming message. This will contain the date, time, and mail address of the incoming message. The mail address may be either local or internet format, depending on the source.
- s Enables the mailing list subscribe/unsubscribe actions. This option is only valid when you have a BBS interface, and it supports automated digest mailing.
- t A transcript of the session will be mailed back to the user in addition to any requested information. The transcript describes each action performed and any errors encountered. By default no transcript is created unless the user requests it via a TRANSCRIPT request in the mail message.

## Security

File access is controlled by requiring the file PUBLIC.PPN in the requested account. The contents of this file are not important. If this file does not exist in the account of the requested file the request will be refused.

## TFTP SERVER (TFTPD)

The TFTP server (TFTPD) provides a UDP based file transfer capability from a remote computer running TCP/IP. No user login or authentication is performed, so this server should only be started on a secure network. TFTP is a simple protocol and lends itself to use in a command file; however, it can have some strange limitations on the UNIX side due to its lack of security.

## Requirements

TFTP.LIT and TFTP.RTI should be loaded into system memory.

## Operation

The TFTP server listens for connection requests. When one is received, it forks by having TCPEMU spawn a job to handle the rest of the connection, then goes back to listening. The spawned job performs the remote user's requested transfer and exits.

TFTPD's child processes are logged into DSK0:[1,2] and can transfer files or create PPNs anywhere. If no device or account information is provided on incoming file names, they are written to the login account (DSK0:[1,2]).

To correct a problem with duplicated requests in the TFTP specification, a subsequent access of the same file, from the same host, on the same port number, within 20 seconds is discarded. This should not be a problem if the remote host selects a unique port number for a second user request. If the same port number is selected for a second try at the same file within 20 seconds, the remote will hang until it times out, or a retry attempt arrives after the 20-second window has expired.

## Security

The spawned TFTP servers log themselves into DSK0:[1,2] and perform no user login or authentication. This means a computer running the TFTP server has NO security for its files beyond setting a device to noaccess.

Remote users may copy files from and to anywhere. This includes a remote AlphaNET computer. You should only execute the TFTP server on a node attached to a secure network. This means a network where everyone is trusted, and no other networks—where there may be unknown users—are connected. You may want to set noaccess on devices where sensitive files are stored. A failure due to noaccess displays as `device does not exist`.

File protection flags are checked. TCP/IP is treated as group 5. Files created by TFTP have a file protection value of 0505051717. See the *AMOS Monitor Calls Manual* for more information.

TFTP cannot copy the USER.SYS file or the TCP:FTPUSR. file. This provides some security for TELNET and FTP user names. However, if there are copies of these files with different names, such as backup files, TFTP will happily transfer those and give away the whole show. Also, these files are accessible through a TELNET login (see the section on the TELNET server).

## RWHO SERVER (RWHOD)

The RWHO server (RWHOD) provides computer uptime information among all computers running an RWHO server on the network.



The AlphaTCP version of the RWHO server provides only uptime information and not user login information as found under UNIX. Much of the login data used is unavailable or meaningless when related to an AMOS computer, and support of the user login information would greatly increase both overhead on the network and local disk storage.

## Requirements

The file MYNAME. must exist in the TCP: account and contain the name of the local host as you want it to be known to other hosts. See Chapter 3.

The disk containing the TCP: account must have at least 13 blocks of contiguous free space for the contiguous file where the computer uptime information is stored. This provides space for uptime information for 100 hosts.

## Operation

RWHOD listens for broadcasts from other hosts notifying everybody they are up. The uptime portion of these broadcasts is stored in the TCP:RWHOD.DAT file. RUPTIM accesses this file to display up/down information on a host-by-host basis.

Every two minutes, RWHOD broadcasts an “up” report for itself, which other hosts running an RWHO server receive and process.

## Runtime Control

If for some reason the information in the RWHOD.DAT file becomes stale, you can erase it before starting AlphaTCP, or stop it with UKILL, erase the file, and restart it with STARTD.

## Security

As no user login information is kept or transmitted, there are no security related concerns for the RWHO server.

## TAME SERVER (TAMED)

The TAME (TCP Access Made Easy) server allows you to write and run AlphaBASIC and assembly language programs which use TCP functions. For example, you can write a server program which listens for and accepts incoming connection requests. This lets you incorporate TCP functionality directly into your application programs.

## Requirements

There are no special requirements for the TAME server other than a minimum AMOS version of 2.3. If an application wishes to identify virtual terminal users the minimum AMOS version is 2.3A.

## Operation

For instructions on incorporating TCP functions into your programs through TAME, see the *AlphaBASIC XCALL User's Manual*, DSO-00066-00, Rev. 03 or later, or the *AMOS Monitor Calls Manual*, DSO-00040-00, Rev. 06 or later. Also, see Appendix E for a discussion of issues relating to application exit on spawned jobs.

## Customizing

There is one option available when you start the TAMED server. The `-d` option enables debug output. It should not be needed during normal operation.

## Security

There are no special security concerns for TAMED. However, remember that the client and server programs you write may give your users access to other systems, and users on other systems access to yours. Be sure to follow proper security precautions in writing TCP application programs.

## SIMPLE NETWORK TIME PROTOCOL SERVER (SNTPD)

The SNTPD server lets you synchronize the clock on your computer to the clock on an Internet time server you choose. It also creates the `TCP:TIMZON.` file and adjusts it for daylight saving time as needed.

## Requirements

You must create the file `TCP:SNTPD.` This file contains the name of the time server you want to use and other information controlling SNTPD. The `SNTPD.NEW` file included with the software looks like this:

```
NOTE: A current list of public timeservers can be found
at http://www.eecis.udel.edu/~ntp/
#
The hostname or address of the server we sync to
#timeserver=[fill in]
The name of the standard timezone, followed by (summer timezone) if used
#zone=PST (PDT)
DST determines when and how the summer timezone is calculated
format: [starts]-[ends], as (which)(weekday)(inmonth)(at24hour)
#dst=1SunApr2-LSunOct2
Offset in hours from Coordinated Universal Time, H:MM if minutes needed
#utc=-8
```

Make these changes to this file:

1. Remove the `#` from the beginning of the `timeserver` line, and replace `[fill in]` with the name of the server you will synchronize to. You can pick a server from the list at the address shown in the file. If BOOTPC is configured to setup the timeserver address, you may remove this line from the file. BOOTPC will add the line to the file itself. The rest of the file still needs to be configured, however.

2. Remove the # from the beginning of the zone line and, if needed, change the time zone abbreviations to the correct ones.
3. If you use daylight saving time, remove the # from the beginning of the dst line and, if needed, set the begin and end dates. The dates shown are correct for the U. S. The format of the dates is:

Week-of-month day-of-week month time

For example, the start date in the SNTPD.NEW file is 1SunApr2. This means daylight saving time starts on the first Sunday in April, at 2 A.M. Use standard three-letter abbreviations for the day and month. For the week, you can enter 1, 2, 3, 4, or L (last).

4. Remove the # from the utc line. This line contains the time difference between your standard time zone and Coordinated Universal Time (Greenwich Mean Time). You can enter just the hours, as shown (-8 is correct for Pacific Time), or, if necessary, hours and minutes separated by a colon.

Remove the # from the `start sntpd` line in the TCP:CONFIG. file. See “Customizing,” below.

## Operation

At the interval you define, SNTPD contacts the Internet time server and synchronizes your AMOS system clock to the time on the server.

SNTPD handles daylight saving time (DST) automatically. It adjusts the time when it next contacts the time server, after the time set to start or end DST in the SNTPD. file. This means there is a short delay between the exact beginning or end of DST and when your system time and TIMZON. file change to reflect it.

## Customizing

The command to start the SNTPD server has this format:

```
start sntpd 85k -bmin -pmin
```

**-bmin** sets the number of minutes after booting the server first contacts the time server. **-pmin** sets the interval it waits before contacting the server again. For example, this is the default line in the CONFIG.NEW file:

```
start sntpd 85k -b2 -p240
```

The server waits two minutes after booting before checking with the time server, then checks every four hours (240 minutes) thereafter.

## Security

There are no security concerns for the SNTPD server.

## BOOTP SERVER (BOOTPD)

The BOOTPD server allows you to maintain network parameters locally for certain devices, such as terminal and print servers. When these devices boot they can query bootpd for their configuration rather than maintain it locally.

### Requirements

You must create the TCP:BOOTPD. file. This file contains the Ethernet address and startup parameters for clients configuring via the bootp protocol. The BOOTPD.NEW file included with the software looks like this:

```
Sample BOOTP configuration file
[00-00-b1-00-00-00]
addr=192.168.1.1
mask=255.255.255.0
nameserver=ns1.widget.com,ns2.widget.com
gateway=192.168.1.254,192.168.1.253
hostname=sample.widget.com
bootfile=boot:sample.bin
```

Each definition begins with the clients Ethernet address in square brackets, and continues to the next Ethernet address or the end of the file.

*addr* specifies the clients IP address.

*mask* specifies the subnet mask the client should use. If you don't use subnetting on your network, this line should be removed so the client will use the default for the class of network you are on.

*nameserver* specifies the address of the domain name server on your network. If you use TCP:HOSTS. instead, this line should be removed. If no nameserver is specified, the client will only be able to contact hosts by their IP address.

*gateway* specifies the IP address of the default router. If your client will not be communicating across your router, this line should be removed.

*hostname* specifies the name the client should use, which should be its *fully qualified domain name*. This is its hostname along with your domain.

*bootfile* is an optional line for clients which will load their boot file via TFTP. This specifies the location and filename of the bootfile located on the AMOS host.

### Operation

When a client wishes to use bootp to configure itself, it broadcasts a bootp request on the network. The bootp server receives this request and attempts to locate the clients Ethernet address in the TCP:BOOTPD. file. If the bootp server locates the Ethernet address, it loads the configuration and returns a boot reply to the client.



## Customizing

There is one option available when you start the BOOTPD server. The `-d` option enables debug output. It should not be needed during normal operation.

## Security

There is nothing in the BOOTP protocol to prevent a rogue BOOTP server from supplying clients with false configuration, including false routing and DNS information.

## BOOTPC

BOOTPC is a BOOTP client which also supports DHCP. BOOTP and DHCP allow a host to self-configure its network parameters at boot time from a central server. DHCP also allows temporary allocation of IP addresses, simplifying a network administrators job.

## Requirements

If you have configured any of your interfaces in `TCP:NETWRK.` to use BOOTP or DHCP you should start BOOTPC. There are also a number of optional switches which you will probably want to set below. In order for the timeserver to work, you must create the `TCP:SNTPD.` file and provide all fields except for `timeserver=`. This line will be added or updated by BOOTPC before `SNTPD.` reads the file.

## Operation

For interfaces specified as BOOTP, a *bootrequest* is broadcast on the local Ethernet. When a server recognizes the hosts Ethernet address, it returns a *bootreply* with all configuration information.

For interfaces specified as DHCP, a few extra steps are performed. DHCP uses extensions to the *bootrequest* and *bootreply* packets. These identify a bootrequest as *dhcdiscover* or *dhcprequest*, and a bootreply as *dhcponffer* or *dhcpack*. There are additional types, but these are the primary ones. A *dhcdiscover* is broadcast on the network. When a dhcp server hears the request, it broadcasts a *dhcponffer*. There may be multiple servers which answer. A *dhcprequest* is then broadcast, thus notifying the entire network which server the dhcp client has chosen to configure with. The offering dhcp server, upon hearing the request, then reserves the settings and sends the client a *dhcpack*.

BOOTPC will continue running after configuring the interfaces if it must renew a temporary DHCP lease. It will terminate if there are no DHCP configured interfaces, or if the DHCP lease is a permanent one.

Until BOOTPC has finished configuring the network parameters, most other AlphaTCP servers will wait in **SI** state. If a dhcp lease expires, active connections will be terminated and these same servers will go back to sleep until a new address has been acquired.

## Customizing

There are a number of switches which may be used with BOOTPC:

- d* Enables debug mode, writes additional information to the network logger.
- f* When creating the TCP:RESOLV. file, adds a *tryhostfile* line.
- h* Creates the TCP:MYNAME. hostname file from the bootp host and domain name fields.
- n* Creates the TCP:RESOLV. nameserver file from the bootp nameserver field.
- t* Updates the time server address in the TCP:SNTPD. file. All other fields are preserved.

## Security

Since BOOTPC broadcasts a request for configuration information on the network, it's possible for someone to operate a rogue BOOTP/DHCP server on the network. This server can give incorrect parameters to the client.

## TDDD

The TCP Device Driver Daemon is a two-way server which provides access from a device driver to an outgoing TCP connection, plus access from an incoming TCP connection to a device. This is primarily used to give a spooler access to a raw printer port on a print server, or to give other hosts access to a serial or parallel port on AMOS.

## Requirements

The TDD.DVR device must be defined, and the driver must be loaded in system memory. Also, the TDDD.LIT and TDDD.RTI files must be loaded in system memory. All incoming and outgoing mappings must be defined in the TCP:TDDD. file.

## Operation

Upon startup, TDDD reads the TCP:TDDD. file and sets up incoming and outgoing maps. Outgoing maps define a name which can be opened through the TDD driver, along with the host and port that will be contacted when that name is opened. Incoming maps open a TCP port which will accept connections, along with a device and optional name which will be opened upon receiving a connection.

## Customizing

All connection relationships are mapped in the TCP:TDDD. file. Outgoing mappings provide a link between a name which may be opened through the TDD: device and a port number on a host. Incoming mappings provide a link between a local TCP port number and a non-filestructured device or a filename.

## Outgoing

An outgoing mapping is defined as follows:

```
OUTGOING {name}={ip or hostname},{port or service}
```

The *{name}* may be any name up to six characters, and represents the filename used when opening the TDD: device.

The *{ip or hostname}* may be the hosts IP address, or it's hostname. If you specify a hostname, it is resolved once when the TDDD. file is processed.

The *{port or service}* may be a TCP port number from 1 to 65535, or a service name in TCP:SERVIC., such as **telnet**.

Here are a couple of examples:

```
OUTGOING PRTSVR=192.168.1.2,3001
OUTGOING EMAIL=mail.widget.org,smtp
```

## Incoming

An incoming mapping is defined as follows:

```
INCOMING {port or service}={device or filename}
```

The *{port or service}* may be a TCP port number from 1 to 65535, or a service name in TCP:SERVIC., such as **telnet**.

The *{device or filename}* can be a device name such as LPR0:, a device and name such as TRM0:PRINTR, or a filename such as DSK0:LOG.TXT[100,100]. If you specify a file, incoming data will be appended to it.

Here are a few of examples:

```
INCOMING 3001=LPR0:
INCOMING 2000=TRM0:PRINTR
INCOMING amos=DSK6:LOG.LST[100,100]
```

## Runtime Control

If you modify the TDDD. file, you may request TDDD to reread the file by sending it a *usr1* signal as follows:

```
.ukill -usr1 pid
```

Doing this will cause any open connections to be terminated, thus it is a good idea to terminate any applications using the TDD device, and any with open incoming connections first.

## Security

TDDD will only open connections to hosts, and allow access to devices, which are configured in the TCP:TDDD. file. Once you provide a link between a tcp port and a device, anyone can open a TCP connection to that device and send it characters. This might be a concern if you have a printer or fax connected, as anyone with access to the defined port has access to the printer or fax.

## Device Sharing

Devices accessed by incoming connections under TDDD need to be defined as non-sharable by placing a forward-slash in front of them on the DEVTBL line in the System Initialization file. This will prevent output from multiple TDDD sessions, and from the spooler, from being written to the device simultaneously.

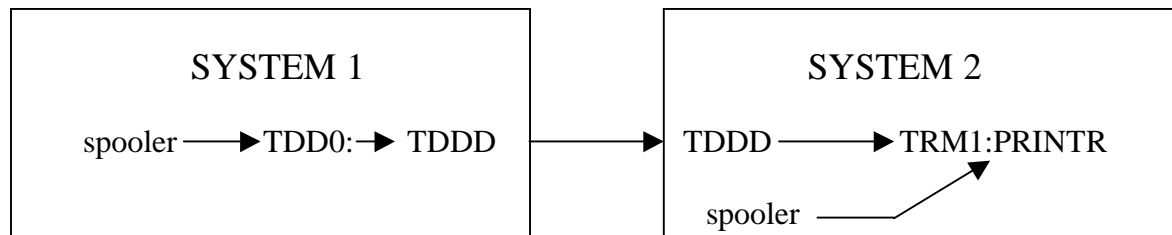
You should NOT set the TRM0: device as non-sharable, as this will limit access to only one printer at a time through the TRM device, thereby locking out access to other printers. Instead, define multiple non-sharable logical devices; /TRM1, /TRM2, etc. Change your spoolers and TDDD definitions to access each printer on a different logical. For instance, use TRM1:SHIPNG and TRM2:SALES in place of TRM0:SHIPNG and TRM0:SALES. Doing this will prevent conflicts between two jobs accessing TRM1:SHIPNG without affecting TRM2:SALES.

If you are using the task manager spooler, make sure you are using the following or later edit levels:

|            |            |
|------------|------------|
| SUBMIT.LIT | 1.1(113)-5 |
| TSKINI.LIT | 1.2(128)-5 |
| TSKINI.OVR | 1.2(128)-5 |
| TSKMAN.OVR | 1.2(128)-5 |

Earlier versions keep the device assigned and will prevent TDDD from gaining access to it.

The diagram below shows a TDDD-based print environment flow.



## IPFW

IPFW is a packet filtering firewall which can be used to enhance security of your network. Incoming packets are accepted or discarded based on a set of rules you specify.

## Requirements

IPFW uses the TCPMON packet monitoring hook to intercept packets. While in use, the TCPMON program cannot be used. If you terminate IPFW and start TCPMON, all packets will be accepted while TCPMON is active. The behavior of TCPEMU when neither IPFW nor TCPMON are active depends on the presence of *mustfirewall* in the TCP:CONFIG. file.

## Operation

Upon startup, the rulefile is read and the packet hook is set. From that point on, every packet entering the TCP stack is examined by IPFW and optionally discarded by the stack. If IPFW is terminated, a message is sent to the master terminal notifying the operator that the firewall has been disabled and the stack returns to normal operation. If *mustfirewall* is specified in the TCP:CONFIG. file the stack will not accept incoming packets except BOOTP/DHCP responses while IPFW is disabled.

## Customizing

IPFW acts only on incoming packets. Incoming packets are checked against a set of rules in the TCP:IPFW. file. The first matching rule determines if the packet will be accepted or discarded. If no rules match the packet, the packet will be discarded.

Rules are specified, one rule per line, in one of the following two formats:

```
{ALLOW, DENY} ICMP {type}
{ALLOW, DENY} {IP, UDP, TCP, OPENTCP} {from} {to}
```

*ALLOW* will keep the packet, while *DENY* will discard it. *IP*, *UDP*, *TCP*, and *ICMP* are all protocol names, while *OPENTCP* refers to a specific packet used by the TCP protocol to open a new connection.

For ICMP filters, *type* is the value of the ICMP message. The most likely ICMP message type you will want to filter is **5**, *redirect*. You may also want to filter out type **8**, *echo request*.

The *from* and *to* addresses may consist of dotted decimal ip and mask, as well as a port or port range. It may also be one of several keywords as explained further below. The format to specify an address is:

```
ip{/mask}{:port}{-endport}
```

The *ip* and optional *mask* are dotted decimal values. The mask defaults to 255.255.255.255, which means *compare all bits in the address*. The *port* and *endport* are optional TCP or UDP port numbers. Using *endport* allows you to specify a range of port numbers.

Instead of specifying addresses and port numbers, you may use any of the following keywords:

|               |                                                 |
|---------------|-------------------------------------------------|
| <b>any</b>    | Match any address                               |
| <b>me</b>     | Match any address assigned to a local interface |
| <b>local</b>  | Match any address on the local network          |
| <b>remote</b> | Match any address not on the local network      |

The following is an example rule file:

```
Rules involving ICMP have a different format: action icmp type
'type' if the icmp type value (i.e., for "ICMP redirect" use 5)
#
allow ip me me ; allow loopback packets
deny ip me any ; disallow spoofed packets
allow udp any:53 me:1024-65535 ; allow DNS responses
deny icmp 5 ; disallow ICMP redirects
allow icmp ; allow all other ICMP
deny opentcp remote any ; disallow non-local access
allow tcp any any ; allow all other TCP activity
```

## Runtime Control

After modifying the rules file you may tell IPFW to reload its rules by sending it a 'usr1' signal with UKILL. You may also toggle debug mode on and off with a 'usr2' signal. See Chapter 6 for more information on UKILL and sending signals.

## Security

Attempts to break into a system can be very clever, and setting up a packet filter to allow certain access while preventing unwanted access is difficult. Someone very familiar with TCP protocols and security issues should set up the filter rules, then they should be tested very carefully.

# Chapter 6 - Administrative Commands

This chapter describes in detail the available AlphaTCP administrative commands. These commands can be divided into two categories: error reporting commands and network status and control commands.

The network error reporting commands are related to the network error logging job, NETLOG:

- LOGCTL lets you open or close the log file or log user messages.
- LOGFMT processes the error log file and creates separate files containing the errors from each job.

The network status and control commands are:

- NETSTA displays the network status—what connections are active and which protocol they are using.
- ROUTE displays and controls internet routing.
- ARP displays and controls Ethernet routing.
- PS displays the status of jobs under the control of AlphaTCP
- UKILL sends signals to jobs under the control of AlphaTCP for cleanup purposes.
- STARTD allows you to start a server under the control of AlphaTCP, or restart one you performed a UKILL on.
- NSLOOK allows you to query a name server for information.
- TCPMON displays packets moving through TCP/IP.
- CVTNID converts between AlphaNET CPU IDs and network, group, and node values.
- FNU2A converts UNIX-style file names to AMOS file names.
- FTPLOG reads and sorts FTP activity from the error log and stores it in FTPLOG.LOG.
- TNWRAP prevents a TELNED started application from reaching the command prompt.
- WHOIS looks up Network Information Center records.
- KLPR provides limited print termination capabilities for print servers.
- IPCFG displays address, network mask, and optional DHCP lease period assigned to TCP interfaces.
- AMPM allows you or your customers to keep your products up to date by contacting a company's web server and retrieving a package directory.
- TROUTE displays the routers in the path to a host.

The following sections describe each of these commands in detail. Other network information commands, such as RUPTIM, PING, and TELDMP, are described in the *AlphaTCP User's Guide*.

Error messages which could be returned by these commands are described in the *AlphaTCP User's Guide*.

## LOGCTL

If you follow the installation instructions in Chapters 2-4, when you boot your computer the NETLOG job runs the NETLOG program using TCP:NETLOG.LST as the output file (the TCP:GOTCP.CMD file starts the program and selects the output file). All AlphaTCP error messages are sent through NETLOG to the NETLOG.LST file.

LOGCTL talks to the NETLOG job and allows you to perform functions such as closing the log file, opening a new one, or logging a user message. Once a log file is closed, you can examine it using a text editor or the TYPE command. You may also process the log file using the LOGFMT program (see below). The syntax for LOGCTL is:

```
LOGCTL { -option }
```

**-option** tells LOGCTL what you are requesting. Possible options are:

- a *file* Appends to an existing log file without erasing the contents.
- c Closes the current log file. Further output goes to the terminal (if any) attached to the NETLOG job until you use LOGCTL with the -o option.
- e Exits. This closes any open log file and stops the NETLOG program. You can restart it using the NETLOG command, as described in Chapter 5.
- m *text* Logs *text* as a message to the current output file. If you want to include spaces in your message, you must enclose the entire message in quotes.
- o *file* Opens *file* as the new log file. The default disk account is OPR:. If another log file is open, LOGCTL closes it before opening the new file.
- w *file* Writes the active log file to the readable *file*. The default disk account is OPR:. The active log file remains in use.

For example:

```
LOGCTL -o TCP:LOG2.LST [RETURN]
```

This closes the currently open log file, if there is one, and opens the file TCP:LOG2.LST as the new log file.

As another example:

```
LOGCTL -w TCP:LOG.TXT [RETURN]
```

This writes the contents of the currently active log file to TCP:LOG.TXT without disabling the active log. You can now examine the new file using TYPE, VUE, or a similar method.



## LOGFMT

LOGFMT reads a log file created by NETLOG and breaks it into individual files, each containing the messages from a single job. The syntax for LOGFMT is:

```
LOGFMT filename
```

*filename* is the name of the log file created by NETLOG. The file must be closed. You can use the LOGCTL command, described above, to close the log file.

LOGFMT reads the log file and writes each message to a file with the name of the job which sent the message and an extension of .LOG. For example, all messages logged by TCPEMU are sent to TCPEMU.LOG, while messages logged by TELNED are written to TELNED.LOG. The files are in the TCP: account.

You do not need to create the individual files written to by LOGFMT. LOGFMT creates the files it needs to write to. Any existing file of that name is erased.

## NETSTA

The NETSTA command displays information about the current state of the network. Depending on the options you choose, NETSTA can show information on:

- Each current network connection, with or without the server connections.
- General statistics about one or more of the network protocols.
- The network routing table.

The contents and format of the displays are described below. NETSTA's format is:

```
netsta {options} {interval}
```



The UNIX equivalent of NETSTA is **netstat**.

The available options are:

- A Displays an additional column, headed PCB, which is the address of the appropriate Protocol Control Block. This is used for debugging purposes only.
- a Shows the state of all connections; server connections are not shown otherwise.
- e Displays statistics relating to the Ethernet. You can combine this with the -s option.
- n Displays addresses and port numbers in numeric form rather than as host names and port names.
- p *proto* Shows connections for protocol *proto* only; *proto* can be either `tcp`, `udp`, or, with the -s option only, `ip`.
- R Displays an additional column, headed Rtx, which shows the number of re-transmissions performed on each connection.
- r Displays the contents of the routing table, in the same format as ROUTE. See the ROUTE command section for a description.
- s Displays per-protocol statistics (error occurrences, etc.) instead of connection states. The output format is described below.



You cannot combine options to show both connection data and overall network statistics. For example, you cannot enter `netsta -a -s [RETURN]`. You can, however, use both -s and -e on the same command.

The optional *interval* entry must be a number. If you include it, NETSTA continuously redisplay its information, pausing that many seconds between each display.

## NETSTA Output

If you don't use any options, the NETSTA display looks like this:

```
Active connections
Proto Recv-Q Send-Q Local Address Foreign Address (state)
tcp 0 0 local:telnet 89.0.4.1:5137 ESTABLISHED
tcp 0 0 local:1024 drs3:telnet ESTABLISHED
```

The list includes all active connections (server connections are shown only if you use the -a option). The columns are:

|        |                                                                                                        |
|--------|--------------------------------------------------------------------------------------------------------|
| Proto  | The name of the protocol used by the connection.                                                       |
| Recv-Q | The amount of data (in bytes) received but not processed on the connection.                            |
| Send-Q | The amount of data (in bytes) waiting to be transmitted, including data sent but not yet acknowledged. |

|                 |                                                                                                                                                                                                                                                                                                                                                                                                                                           |            |           |
|-----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------|-----------|
| Local Address   | The internet address of the local host and the port number the connection is using. If the local host has an entry in TCP:HOSTS., its name is shown instead. Likewise, if the port has a name listed in TCP:SERVIC., its name is listed. If the port is not yet established, the port number is shown as *. If you use the -n option, the host address and port number are shown as numbers even if there are names established for them. |            |           |
| Foreign Address | The internet address and port number of the remote host to which the socket is connected.                                                                                                                                                                                                                                                                                                                                                 |            |           |
| State           | This column applies to TCP connections only. The possible states are:                                                                                                                                                                                                                                                                                                                                                                     |            |           |
|                 | CLOSED                                                                                                                                                                                                                                                                                                                                                                                                                                    | LISTEN     | TIME_WAIT |
|                 | SYN_SEND                                                                                                                                                                                                                                                                                                                                                                                                                                  | SYN_RECV   | LAST_ACK  |
|                 | FIN_WAIT_1                                                                                                                                                                                                                                                                                                                                                                                                                                | FIN_WAIT_2 | CLOSING   |
|                 | ESTABLISHED                                                                                                                                                                                                                                                                                                                                                                                                                               | CLOSE_WAIT |           |

With the -A option, there is an additional column to the right, showing the address of the appropriate protocol control block.

When you use the -s option, the output looks like this:

```

tcp:
 0 incomplete headers
 0 header checksum errors
udp:
 0 incomplete headers
 0 header checksum errors
 0 bad data length fields
ip:
 0 incomplete headers
 0 header checksum errors
 0 bad data length fields

```

By default, statistics are shown for TCP, UDP and IP; you can use the -p option to display only one protocol.

The -e option output is:

```

ether:
 2560 packets successfully received
 4655 packets successfully transmitted
 0 packets lost
 0 receive errors
 0 transmission errors
 0 collisions detected

```

## ROUTE

ROUTE lets you see and change the network routing table. AlphaTCP uses the routing table to send packets to remote destinations (ones not on the local network). For each remote destination, the routing table assigns a gateway (a host on the local network which is also attached to one or more other networks). Packets are sent to the gateway, which sends them on to their destinations. Your local computer need not know the whole route. It only needs to know which gateway a packet should be sent to.

For a discussion of subnetting, see Chapter 1.



The routing table is stored in memory while AlphaTCP is running, not on the disk. This means you need to recreate it each time your computer is rebooted.



Only one copy of ROUTE may be run at a time. If you have multiple routes to set up, create a command file to run the ROUTE commands one after the other. If you have only a single default route, the simplest method is to spawn the ROUTE command with 200K as if it is a server within the CONFIG. file. Once the command has done its work, the job will be deleted and the memory freed. Again, do NOT spawn more than one copy of ROUTE.

The format for ROUTE is:

```
ROUTE {option} {command} {destination} {gateway}
```

*option* may be:

- f Deletes (flushes) the routing table. If this is used with one of the commands described below, the table is deleted prior to the command's application.
- s The operation refers to the "smart" gateway. You may designate one gateway as the smart gateway; packets for a destination which is not found in the routing table are sent to the smart gateway. Do not include a destination when you use this option.

The *command* may be:

|        |                              |
|--------|------------------------------|
| add    | Adds a route.                |
| change | Modifies an existing route.  |
| delete | Deletes a route.             |
| print  | Displays one or more routes. |

For add and change, you must include both a *destination* and a *gateway*, unless you are using the -s option. For gateway, you can enter either an internet address or a host name from the TCP:HOSTS. file. For destination, you can enter either an internet address or a network name from the TCP:NETWRK. file.

With the delete command, if you leave out the gateway specification, or enter \* for it, it deletes all routes to the given destination.

With the print command, if you leave out both the destination and the gateway, ROUTE displays the entire routing table. If you enter only a destination, all routes to that destination are displayed. Here is a sample route display:

| Network address | Gateway address | Redirect | Life   |
|-----------------|-----------------|----------|--------|
| 73.0.0.0        | 83.0.0.1        |          |        |
| 79.0.0.0        | 83.0.0.3        | 83.0.0.6 | 5 mins |
| 59.0.0.0        | 83.0.0.1        |          |        |

The second entry indicates packets addressed to network 79.0.0.0 are normally sent to the gateway with internet address 83.0.0.3. However, they are currently being diverted to 83.0.0.6, and the diversion will last for a further five minutes. Routes entered using ROUTE are considered permanent. Only routes from network ICMP redirects have a life span.

## ARP

ARP displays the table which maps Ethernet Addresses to internet addresses. This table is updated automatically by the ARP protocol while AlphaTCP is running. You can also manually change the table using the ARP command. There are three different formats for ARP:

```
ARP -g {in_addr} {-N network-number}
ARP -d in_addr
ARP -a in_addr ether_addr
```

With the **-g** option, ARP displays current ARP entries by interrogating the current TCP/EMU data. If you don't include any parameters, it displays all ARP entries. If more than one network interface uses ARP, entries for each ARP table are displayed.

If you include **in\_addr** (an internet address), only that entry is displayed.

If you use the **-N** option, ARP displays only the ARP entries for the network interface with network number **network-number**.

Using the **-d** option deletes the entry for the host with the internet address **in\_addr**, specified in standard "." notation.

The **-a** option creates an ARP entry associating the host with the internet address **in\_addr** with the Ethernet address **ether\_addr**. The Ethernet address is given as six hexadecimal bytes separated by hyphens. The entry is permanent.

Here is a sample ARP -g display:

| Internet Address | Ethernet Address      | Type | Life   |
|------------------|-----------------------|------|--------|
| 79.0.21.4        | 08-00-39-00-21-70     | temp | 1 mins |
| 79.0.3.3         | 08-00-4E-01-1E-85     | temp | 9 mins |
| 79.0.20.96       | 08-00-39-00-20-96     | temp | 8 mins |
| 79.0.21.1        | 08-00-39-00-21-F1     | temp | 1 mins |
| 79.0.20.9        | 08-00-39-00-20-49     | temp | 7 mins |
| 79.0.0.4         | 08-00-2B-06-6B-2B     | temp | 4 mins |
| 79.0.21.2        | (ARP request pending) |      |        |

ARP request pending indicates an ARP request was transmitted for this internet address, but no reply was received.

## PS

PS displays information about local jobs under the control of AlphaTCP.

As AlphaTCP provides a somewhat UNIX-like environment to its applications, this information is very UNIX-like. Information shown includes the process id (pid), parent process id (ppid), and group id (gid). The syntax is:

```
PS {-p}
```

The PS display looks like this:

| PID  | PPID  | PGRP  | TRM     | JOB     | COMMAND |
|------|-------|-------|---------|---------|---------|
| ---- | ----- | ----- | -----   | -----   | -----   |
| 24   | 1     | 24    | one     | one     | ps      |
| 3    | 1     | 1     | ftpd    | ftpd    | ftpd    |
| 2    | 1     | 1     | telnetd | telnetd | telnetd |
| 1    | 0     | 1     | tcpemu  | tcpemu  | tcpemu  |

If your process list exceeds one screen, you can have PS pause after every screenful of information by including the **-p** switch.

## UKILL

UKILL is the AMOS equivalent of the UNIX **kill** command. UKILL sends signals to jobs under control of AlphaTCP for the purpose of terminating old sessions when needed. As in UNIX, the syntax of UKILL is:

```
UKILL {-signal} pid {pid...}
```

**-signal** is optional. By default a SIGTERM signal is sent to the job (a non-AlphaTCP program will see an S\$EXI software interrupt). The only meaningful signal to specify is -9, which is a SIGKILL. Avoid sending SIGKILL signals to permanent AMOS processes as the actual SIGKILL translates to a software interrupt abort, which does not allow the release of TCPEMU resources.

For processes spawned by AlphaTCP, the SIGKILL signal is handled in a special way. AlphaTCP places the process in an internal “kill queue,” then performs various operations to get it to the command prompt over a period of time, before resorting to actually performing a software interrupt abort.

If you are running SLIP over a modem and want it to hang up before its normal period, you should use UKILL without the `-signal` option. This causes SLIP to hang up the phone. If another TCP/IP packet arrives to send it will redial the connection.

You can find the process id (pid) of the job by using the PS command. You can specify multiple process ids on the command line. The pid for TCPEMU itself is 1.



AlphaTCP is compatible with the AMOS command KTASK, which allows servers to be terminated by job name.

While `-signal` may be the signal number, the following names are also recognized and translated to the appropriate numbers. Most of the signals you won't ever use—they are internal.

| Name    | Value | Signal Function |
|---------|-------|-----------------|
| *-hup   | 1     | graceful exit   |
| *-term  | 1     |                 |
| *-int   | 2     | ^C interrupt    |
| *-quit  | 2     |                 |
| *-kill  | 9     | forced exit     |
| *-abrt  | 9     |                 |
| *-abort | 9     |                 |
| -pipe   | 13    | connection lost |
| -alm    | 14    | alarm ticked    |
| -alarm  | 14    |                 |
| -cld    | 18    | child exited    |
| -chld   | 18    |                 |
| -child  | 18    |                 |
| *-usr1  | 30    | user defined #1 |
| *-user1 | 30    |                 |
| *-usr2  | 31    | user defined #2 |
| *-user2 | 31    |                 |

\*These signals are mentioned in the server descriptions in Chapter 5.

## STARTD

The STARTD command starts or restarts a server controlled by TCPEMU. You can use it to start a server you forgot to add to the CONFIG. file without rebooting.

STARTD uses the same format as the start lines in the CONFIG. file (except the keyword `start` is replaced by the command STARTD). In general, that format is:

```
START server memory {arg arg ...}
```

See chapters 1 and 2 for the proper memory and argument list for the server you are starting.

## NSLOOK

The NSLOOK command queries name servers for information. It operates in interactive and non-interactive modes. Interactive mode lets you query name servers for information about various hosts and domains or print a list of hosts in a domain. Non-interactive mode prints just the name and requested information for a host or domain.

### Non-Interactive Mode

To perform a single non-interactive query, the format of the command line is:

```
NSLOOK {-option} hostname {- server}
```

The options available are the same as those listed for the set command in interactive mode, below. Each option must be preceded by a hyphen. For example, to change the default query type to host information, and the initial time-out to 10 seconds, enter the command:

```
NSLOOK -query=hinfo -timeout=10 hostname RETURN
```

### Interactive Mode

You can enter interactive mode by using the following format:

```
NSLOOK {- server}
```

*Server* is the name server you want to query. If you don't include a server name, the default name server for your domain is used.

In interactive mode, you can enter either host names to look up or commands. Any entry which isn't a valid NSLOOK command is interpreted as a host name. If you want to query a host name which is the same as an NSLOOK command, precede it with a backslash.

To leave interactive mode, type **exit**. You can interrupt a command at any time by pressing **CTRL-C**.

### Interactive Mode Commands

NSLOOK interprets the following words as commands. (Any other word is interpreted as a host name to look up.) If the word is an internet address and the query type is A or PTR, the name of the host is returned. If the word is a name and does not have a trailing period, the default domain name is appended to the name. (This behavior depends on the state of the set options domain, srchlist, defname, and search). To look up a host not in the current domain, append a period to the name. Words recognized as commands are:

```
set option{=value}
```

The **set** command lets you change the operation of lookups. It has many available options, as listed below. These are the same options you can enter on the command line in non-interactive mode.



|                                                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>all</code>                                           | Prints the current values of the frequently-used options to <b>set</b> , and information about the current default server and host.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| <code>class=<i>value</i></code>                            | Change the query class to one of:<br><b>IN</b> the Internet class.<br><b>CHAOS</b> the Chaos class.<br><b>HESIOD</b> the MIT Athena Hesiod class.<br><b>ANY</b> wildcard (any of the above).<br>The class specifies the protocol group of the information. (Default = <b>IN</b> , abbreviation = <b>c1</b> )                                                                                                                                                                                                                                                                                                                              |
| <code>{no}debug</code>                                     | Turn debugging mode on or off. Much more information is printed about the packet sent to the server and the resulting answer when debugging mode is on. (Default = <b>nodebug</b> , abbreviation = <b>{no}deb</b> )                                                                                                                                                                                                                                                                                                                                                                                                                       |
| <code>{no}d2</code>                                        | Turn exhaustive debugging mode on or off. Essentially, all fields of every packet are printed when on. (Default = <b>nod2</b> )                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| <code>domain=<i>name</i></code>                            | Change the default domain name to <b><i>name</i></b> . The default domain name is appended to a lookup request depending on the state of the <b>defname</b> and search options. The domain search list contains the parents of the default domain if it has at least two components in its name. For example, if the default domain is <b>CC.Berkeley.EDU</b> , the search list is <b>CC.Berkeley.EDU</b> and <b>Berkeley.EDU</b> . Use the <b>set srchlist</b> command to specify a different list. Use the <b>set all</b> command to display the list. (Default = domain substring from <b>TCP:MYNAME.</b> , abbreviation = <b>do</b> ) |
| <code>srchlist={<i>name1</i>/<br/><i>name2</i>/...}</code> | Change the default domain name to <b><i>name1</i></b> and the domain search list to <b><i>name1</i>, <i>name2</i></b> , etc. A maximum of 6 names separated by slashes can be specified. For example:<br><br><pre>set srchlist=lcs.MIT.EDU/ai.MIT.EDU/MIT.EDU</pre> sets the domain to <b>lcs.MIT.EDU</b> and the search list to the three names. This command overrides the default domain name and search list of the <b>{set domain}</b> command. Use the <b>set all</b> command to display the list. (Default = domain substring from <b>TCP:MYNAME.</b> , abbreviation = <b>srchl</b> )                                              |

|                                                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|---------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>{no}defname</code>                                | If set, append the default domain name to a single-component lookup request (one that does not contain a period). (Default = <code>defname</code> , abbreviation = <code>{no}def</code> )                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| <code>{no}search</code>                                 | If the lookup request contains at least one period but doesn't end with a trailing period, append the domain names in the domain search list to the request until an answer is received. (Default = <code>search</code> , abbreviation = <code>{no}sea</code> )                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| <code>port=value</code>                                 | Change the default TCP/UDP name server port to <i>value</i> . (Default = 53, abbreviation = <code>po</code> )                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| <code>querytype=value</code><br><code>type=value</code> | <p>Change the type of information query to one of:</p> <ul style="list-style-type: none"> <li>A the host's Internet address.</li> <li>AFSDB AFS distributed file system database.</li> <li>CNAME the canonical name for an alias.</li> <li>GID the group identifier of a group name.</li> <li>HINFO the host CPU and operating system type.</li> <li>MB a mailbox domain name.</li> <li>MG a mail group member.</li> <li>MINFO the mailbox or mail list information.</li> <li>MR a mail rename domain name.</li> <li>MX the mail exchanger.</li> <li>NS the name server for the named zone.</li> <li>PTR the host name if the query is an Internet address, otherwise the pointer to other information.</li> <li>SOA the domain's "start-of-authority" information.</li> <li>TXT the text information.</li> <li>UID user identifier.</li> <li>UINFO the user information.</li> <li>WKS the supported well-known services.</li> <li>ANY any matching records.</li> <li>AXFR all the information held on a zone.</li> </ul> <p>More information on the types can be found in RFC-1035. (Default = A, abbreviations = <code>q</code>, <code>ty</code>)</p> |
| <code>{no}recurse</code>                                | Tell the name server to query other servers if it does not have the information. (Default = <code>recurse</code> , abbreviation = <code>{no}rec</code> )                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <code>retry=number</code>                               | Set the number of retries to <i>number</i> . When a reply to a request is not received within a certain amount of time (changed with <b>set timeout</b> , the time-out period is doubled and the request is re-sent. The retry value controls how many times a request is re-sent before giving up. (Default = 4, abbreviation = <code>ret</code> )                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <code>root=host</code>                                  | Change the name of the root server to <i>host</i> . This affects the <b>root</b> command described below. (Default = <code>ns.nic.ddn.mil.</code> , abbrev-                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |

|                                                                                                                                                                                                                                                                                                                                |                                                                                                                                                                                                         |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                                                                                                                                                                                                                                                                                                | viation = ro)                                                                                                                                                                                           |
| timeout= <i>number</i>                                                                                                                                                                                                                                                                                                         | Change the initial time-out interval for waiting for a reply to { <i>number</i> } seconds. Each retry doubles the time-out period. (Default = 5 seconds, abbreviation = ti)                             |
| {no}vc                                                                                                                                                                                                                                                                                                                         | When set, always uses a virtual circuit (TCP) when sending requests to the server. Otherwise, UDP is used unless a response is too large to fit within a packet. (Default = novc, abbreviation = {no}v) |
| {no}ignoretc                                                                                                                                                                                                                                                                                                                   | Ignore packet truncation errors. (Default = noignoretc, abbreviation = {no}ig)                                                                                                                          |
| server <i>domain</i>                                                                                                                                                                                                                                                                                                           |                                                                                                                                                                                                         |
| lserver <i>domain</i>                                                                                                                                                                                                                                                                                                          |                                                                                                                                                                                                         |
| Changes the default server to <b>domain</b> . <b>lserver</b> uses the initial server to look up information about <b>domain</b> while <b>server</b> uses the current default server. If an authoritative answer can't be found, the names of servers that might have the answer are returned.                                  |                                                                                                                                                                                                         |
| root                                                                                                                                                                                                                                                                                                                           |                                                                                                                                                                                                         |
| Changes the default server to the server for the root of the domain name space. Currently, the host ns.nic.ddn.mil is used. (This command is the same as lserver ns.nic.ddn.mil.) The name of the root server can be changed with the <b>set root</b> command.                                                                 |                                                                                                                                                                                                         |
| finger <i>name</i> {> <i>filename</i> }                                                                                                                                                                                                                                                                                        |                                                                                                                                                                                                         |
| finger <i>name</i> {>> <i>filename</i> }                                                                                                                                                                                                                                                                                       |                                                                                                                                                                                                         |
| Connects with the finger server on the current host. The current host is defined when a previous lookup for a host was successful and returned address information (see the <b>set querytype=A</b> command). <b>name</b> is optional. > sends the output to <b>filename</b> , while >> appends the output to <b>filename</b> . |                                                                                                                                                                                                         |
| ls { <i>option</i> } <i>domain</i> {> <i>filename</i> }                                                                                                                                                                                                                                                                        |                                                                                                                                                                                                         |
| ls { <i>option</i> } <i>domain</i> {>> <i>filename</i> }                                                                                                                                                                                                                                                                       |                                                                                                                                                                                                         |
| List the information available for <b>domain</b> , optionally creating (>) or appending to (>>) <b>filename</b> . The default output contains host names and their internet addresses. When output is directed to a file, hash marks print for every 50 records received from the server. <b>option</b> can be one of:         |                                                                                                                                                                                                         |
| -t { <i>querytype</i> }                                                                                                                                                                                                                                                                                                        | lists all records of the specified type (see <b>querytype</b> , below).                                                                                                                                 |
| -a                                                                                                                                                                                                                                                                                                                             | lists aliases of hosts in the domain. Synonym for <b>-t CNAME</b> .                                                                                                                                     |
| -d                                                                                                                                                                                                                                                                                                                             | lists all records for the domain. Synonym for <b>-t ANY</b> .                                                                                                                                           |
| -h                                                                                                                                                                                                                                                                                                                             | lists CPU and operating system information for the domain. Synonym for <b>-t HINFO</b> .                                                                                                                |
| -s                                                                                                                                                                                                                                                                                                                             | lists well-known services of hosts in the domain. Synonym for <b>-t WKS</b> .                                                                                                                           |
| view <i>filename</i>                                                                                                                                                                                                                                                                                                           |                                                                                                                                                                                                         |

Allows you to examine the output from previous commands sent to a file. Uses the AlphaVUE text editor.

help  
?

Lists a brief summary of commands.

exit

Exits the program.

## Error Messages

If a lookup request is not successful, an error message is printed. Possible errors which you may see include:

Timed out

Server did not respond to a request after a certain amount of time (changed with **set timeout=value**) and a certain number of retries (changed with **set retry=value**).

No response from server

No name server is running on the server machine.

No records

The server does not have resource records of the current query type for the host, although the host name is valid. The query type is specified with the **set querytype** command.

Non-existent domain

The host or domain name does not exist.

Connection refused

Network is unreachable

The connection to the name or finger server could not be made at the current time. This error commonly occurs with **ls** and **finger** requests.

Server failure

The name server found an internal inconsistency in its database and could not return a valid answer.

Refused

The name server refused to service the request.

Format error

The name server found that the request packet was not in the proper format. It may indicate an error in NSLOOK.

## TCPMON

The TCPMON command allows the system administrator to see data flowing in and out of the AlphaTCP TCP/IP stack. Data is displayed for Ethernet and SLIP network paths. This may help in isolating connection problems. It is very limited in its display and abilities and does not take the place of a true network monitor. TCPMON is invoked as follows:

```
tcpmon {options}
```

The available options are:

- `-address` Displays only packets containing the specified IP *address*.
- `-lfile` Sends all displayed data to the specified log *file*. The file is stored in a compressed form for speed.
- `-pport` Limits the display to only packets containing the specified TCP or UDP *port* number. In place of a port number, you may use a service name. For example, `-ptelnet` or `-psmtp`.
- `-q` Disables the scrolling packet display and may avoid trace overruns caused by the display. This switch must be used only when using the `-l` switch.
- `-t` When running under a telnet session, forces display of packets related to the telnet session. Normally, TCPMON suppresses these packets on AMOS 2.3A and later.

## Understanding Packets

The TCP/IP protocol suite is very complex. You should have a good TCP/IP reference book when working at the packet level.

When you run TCPMON, a screen much like the following one will appear:

```

 IP Packet Monitor for AMOS TCP/IP V1.x
 IP ICMP UDP TCP LOST
Incoming: 2 0 1 1 ----
Outgoing: 1 1 0 0
TRACING address=192.0.2.1

Sent: at 0.00 seconds (offset +0.00 seconds)

 ICMP 192.0.2.1 -> 192.0.2.2 ttl 32 len 92 id 01ba
 Echo Request
Rcvd: at 2.04 seconds (offset +2.04 seconds)

 UDP 192.0.2.1 <- 192.0.2.2 ttl 64 len 122 id 1f3c
 Port 1303 <- domain Len: 112
Rcvd: at 6.10 seconds (offset +4.06 seconds)

 TCP 192.0.2.1 <- 192.0.2.2 ttl 64 len 40 id 1f40
 Port 1139 <- telnet Wnd: 4096 SYN ACK
 Seq: 1386048000 Ack: 1284352001

```

The TRACING line will appear only if you have specified the `-a` and/or the `-p` switch. If you specify the `-p` switch, the line will also contain the port number being traced.

Everything above the dashed line should be self-explanatory except for the LOST column. An incrementing value under LOST means packets are being transferred faster than TCPMON can buffer them. The packets are dropped from the TCPMON buffers, but still pass through the TCP/IP stack. Try moving to a faster terminal, or using the -q switch or a faster disk when logging. You may also try limiting the address or port being monitored if you have not already done so.

The area below the dashed line is reserved for scrolling packet information. The above example contains the three primary packet types you might see: TCP, UDP, and ICMP.

All packets displayed contain timing information, the direction of the packet (Sent : or Rcvd : ), the source and destination IP address (with an arrow pointing source to destination), the packet time-to-live (ttl : ), the packet length (len : ), and the packet id (id). The packet id will be followed by punctuation in the case of fragments. A + indicates a starting or intermediate fragment, while a . indicates the final fragment.

The first packet displayed or logged starts the elapsed time counter. The offset timing is the value from one packet to the next. Granularity of the timer is rather coarse and cannot be used to determine precise packet timing as one would do with a true network analyzer.

The first example is an ICMP message being sent out. ICMP information includes the ICMP message type. The example shows a packet from the PING program.

The second example is a UDP message being received. UDP information includes the port numbers, with an arrow pointing source to destination. If the port number is one of the common well-known port numbers, the name will be displayed, otherwise the number itself is shown. Also displayed is the UDP packet length. The example shows a response to a domain name query.

The third, and most complex, example is a TCP message being received. TCP information includes the port numbers, again with an arrow. It also includes the TCP window size and a breakdown of the flags. The following line contains the TCP sequence number and, if the ACK flag is set, the Acknowledgment number.

Not all fields are displayed in the packet traces. Others might be added as the need arises, however larger screen traces and processing time could increase the chance of packets being lost. The information provided should be helpful in isolating simpler problems without needing a network analyzer.

## CVTNID

The CVTNID command converts AlphaNET CPU IDs to network, group, and node values. It can also convert those values into a CPU ID. A CPU ID is entered with a trailing dash. Network, group, and node numbers are entered separated by commas or spaces.

The following is an example of converting a CPU ID into its individual parts:

```
.CVTNID 16842753-
16842753- translates to Network 1, Group 1, Node 1
```

The following is an example of converting the individual parts into a CPU ID:

```
.CVTNID 1,1,1
Network 1, Group1, Node 1 translates to 16842753-
```

## FNU2A

FNU2A converts long UNIX file names into AMOS file names. Its primary use is to determine the AMOS file name for a given Web URL you wish to use. In this way, your URL may be more meaningful than a 6 by 3 AMOS file name. The function used to convert the name is the same one described in the *AlphaTCP User's Guide* section "Converting UNIX File Names to AMOS."

For example, assume the sales department at widgit.com has their own account and wants to list their latest pricing. They could advertise the following URL:

```
http://www.widgit.com/sales/latest-prices.html
```

The portion after the host name will be used by the AlphaTCP HTTPD to locate the ersatz and file name for the document. To determine the file HTTPD will be looking for, use the FNU2A command:

```
FNU2A sales/latest-prices.html
```

FNU2A returns the following file name, which is the file name HTTPD will look for when the above URL is requested:

```
sales:latpri.htm
```

## FTPLOG

FTPLOG reads the network error log and writes FTP activity to the file TCP:FTPLOG.LOG. The FTPLOG program should only be run once each time TCP/IP is started. You should make a backup copy of the error log which FTPLOG may read. The example startup command file, GOTCP.NEW, does this and runs FTPLOG. The format for FTPLOG is:

```
FTPLOG LOGFILE
```

**LOGFILE** is the backup error log to read. FTP must have logging enabled to provide any useful information. See Chapter 5.

## TNWRAP

The TNWRAP program prevents an application automatically started by TELNED from reaching the AMOS command prompt. Normally, the session is disconnected when the application exits to AMOS. If the window between application exit and session disconnected is a security concern for you, use TNWRAP. Simply place TNWRAP prior to the command itself. For example, assume you have the following TELNED start line:

```
start telned 100k 2000 10k vt100 "Parts Lookup" "RUN PARTS"
```

PARTS.RUN will exit, and TCPEMU will disconnect the session. To prevent use of the connection during the brief period between the two events, use the following TELNET start line:

```
start telnet 100k 2000 10k vt100 "Parts Lookup" 'TNWRAP RUN PARTS'
```

## WHOIS

WHOIS lets you query the Internet Network Information Center (NIC) for information about a particular domain. The information returned includes administrative and technical contacts, name servers, and more. You can specify a particular server to query for the information.

The TCP:WHOIS. file contains the default server to contact for the information. The release includes a WHOIS.NEW file which you can rename to a default extension. This file contains the name of the default NIC server, rs.internic.net. If this file is not set up you will need to use the -h switch every time.

Type WHOIS and the name of the domain you want information about. For example:

```
WHOIS alphamicro.com
```

To contact a server other than the default, add the -h switch and server name before the domain name. The format to use is:

```
WHOIS -h server domain
```

## KLPR

The KLPR command terminates the currently transferring print request for a printer. KLPR is only useful for print servers which print as the file is being transferred to them. For hosts which accept the whole file then print, you need to terminate the printing on the destination host instead. KLPR must be run from the OPR: account and has the following format:

```
KLPR printer
```

*printer* is the name to the left of the equal sign in the TCP:LPR. file.

## IPCFG

IPCFG allows you to see the address, network mask, and optional DHCP lease period assigned to TCP interfaces. This is most useful when an interface has been configured with BOOTP or DHCP. When you enter IPCFG, something like this will be displayed:

| Mux | Type | Life | IP Address  | Net Mask      |
|-----|------|------|-------------|---------------|
| 1   | perm |      | 192.168.1.1 | 255.255.255.0 |
| 2   | perm |      | 127.0.0.1   | 255.0.0.0     |



The *mux* value is an internal reference inside TCPEMU which indirectly relates to the hardware interface, there is currently no method for relating it to a device name. *Type* may be permanent, bootp, or dhcp. The *life* column only pertains to interfaces configured with dhcp.

## AMPM

The Alpha Micro Package Manager (AMPM) lets you easily keep a product up to date, or allow your customers to keep your product up to date. AMPM does this by contacting a companies web server on the Internet and retrieving a package directory. It then resolves differences and updates files from the package repository which have changed. Your AMOS host must have Internet connectivity for this to work, unless you are running an AMPM package repository on your intranet.

## Synchronizing With an AMPM Package Repository

AMPM must be executed from the OPR: account. The format of the AMPM command is:

```
.AMPM{/switches} {/l:login /p:password} company:package devn:
```

- *Company* is a company listed in the LIB:AMPM.TXT file, for example: **alphamicro**
- *Package* is a package or product for which the company maintains a repository, for example: **tcp15**
- *devn:* is the disk on which you maintain your local copy of the package.

The following switches may be specified on the AMPM command line:

- /a* Execute any files specified in the package directory as autorun programs.
- /d* Output additional debug information
- /i* Ignore newer local versions, allowing unconditional synchronization
- /n* No prompting for [return] before performing actual file transfer
- /v* Verify only, create update command file but do not run it.

The optional login and password arguments may be used to access a package repository requiring authorization. On the AMPM command line, if the login or password contains spaces, the entire switch must be enclosed in quotation marks.

For example:           "/l: john doe"

If you attempt to access files in a private repository and have not specified the login and password arguments, AMPM will ask you for them if it finds files which need updating. When asked for the login and password information, do not enclose this information in quotation marks.

AMPM locates the company you specify in the LIB:AMPM.TXT file. The URL (*Universal Resource Locator*) associated with the company is then used to contact the site and request the directory file for the package you are updating.

Once the package directory is retrieved, it is adjusted to match the device you specified and checked for differences. Any differences are recorded and a command file is built to retrieve the updated files. If the command file is too large for AMOS to process, it will be split into multiple command files. The command file is then executed.

Once the updates are retrieved, the updated files are rechecked to verify they were transferred properly. Upon a successful synchronization, a line is then added to the OPR:AMPM.HST file noting the date, package, device, and number of files updated. This makes for a handy reference to identify the revisions of packages and when they were updated on the system.

There are a few directives which may be present in the package directory. One of these directives will automatically place you in VUE for reading a text file, which will likely contain instructions or a description of the update.

Another directive can automatically execute a downloaded program to perform configuration. Since this is a security concern, AMPM will only perform this step if you have specified the *-a* switch. Since an autorun program may do anything it wishes, it's best to enable this option *only* when accessing a highly trusted package repository such as one on your own intranet. If an autorun file is present, and you have not specified the *-a* switch, the name of the file will be displayed for you.



To retrieve a list of packages at the company, only specify the company name. The package list will be retrieved and will be displayed in VUE. For example: `.ampm alphamicro`

## Running an AMPM Package Repository

If you maintain a product and your customers systems are connected to the Internet, they can easily keep their version current if you operate an AMPM package repository. This is done using an ordinary web server on the Internet. You may use the AlphaTCP web server, or a non-AMOS web server.

If you wish to use a non-AMOS web server it will need to allow subdirectory names which look like a PPN specification, such as `[1,4]`. In addition, the following MIME type *must* be added to the server so AMPM can properly handle AMOS contiguous files. Add any additional extensions which are contiguous in your package:

| MIME Content-Type             | Suggested Extensions |
|-------------------------------|----------------------|
| application/x-amos-contiguous | DAT IDX IDA          |

The remainder of this section assumes you are configuring your package repository on an AMOS host. You must use AlphaTCP 1.5 - earlier versions of the web server are not compatible with AMPM.

There are two distinct areas to setup; an area to store a specific package, and an area to store directory files for all of your packages. Although both will usually reside on the same system, it is not necessary to keep them on the same system. The LIB:AMPM.TXT file on your customer's system also needs a line added which identifies where your company keeps its directory files.

You will probably want to store your package on a unique device or in a *folder* accessed with a device name. Files should reside in their normal ppns, as if it were loaded from a release tape. To eliminate the need to add a *public.ppn* file in every account, place the device in TCP:HTTPD.DEV. Each device should be placed on a line by itself. Note that you must change the host configuration file to add the HTTPD server to allow remote access to your package repository. Also, TAMED must be running on both the host and remote system.

The second area to setup will be the package directory area. This will be an account with an ersatz name, **pkgdir**: for example, which contains directory files for all of your packages. These directory files are simply standard VERIFY files (*DIR/D/H/V*) with a header line added. Individual files in the directory may have directives added which are described later. The package directory needs a *public.ppn* file present to make it accessible. The LIB:AMPM.TXT file on your customers system should reference this package directory something like this:

```
WidgetCo=http://www.widgets.com/pkgdir/
```

At the start of the package DIR file a one-line header is added which contains a package name, date of modification, and a url where the package is stored. It also contains an accessibility description, either

```
Access="public" or Access="private"
```

“Private” access simply tells AMPM that the server will be expecting a login and password to access the files in the package.

For example, assume this header is added to the top of the WID11.DIR file:

```
; Package="Widget 1.1" Date="1999/06/22" Location="http://www.widget.com/sub25/" Access= "public"
```

Files in the package directory may be flagged for special handling by adding a two character directive to the start of the filename. Insert the characters at the start of the line, but *don't* attempt to line up the version numbers or hash totals which get shifted over. The following directives may be used:

- \*a** Autorun file. Executes after update if *-a* was specified on the AMPM command line. Only one autorun file may be specified.
- \*c** Configuration file. Will only be transferred if it doesn't already exist. Hash mismatches will be ignored.
- \*p** PIC overlay file. Will only be transferred if a newer version and will notify user of the overwrite. Hash mismatches will be ignored.
- \*s** Show this file. File will be displayed using VUE after an update completes. Multiple show files may be specified, they will be displayed in the same order listed in the directory file.

Using the above examples, a customer interested in updating his package on DSK3: would log into OPR: and type:

```
.ampm widgetco:wid11 dsk3:
```

The AMPM program would locate *widgetco* in the LIB:AMP.M.TXT file and request the following URL:

```
http://www.widgets.com/pkgdir/wid11.dir
```

Upon verifying certain files needed updating, AMPM would then use the header information in the WID11.DIR file, plus the filename and ppn listed for the file, to build a url somewhat like this:

```
http://www.widgets.com/sub25/[p,ppn]/filnam.ext
```

## AMPC Package Index

You should maintain an index of packages, called AMPM.LST, in the package directory account. This file will be requested whenever a customer runs AMPM with only a company name. The AMPMX utility is provided to create this package index. AMPMX creates an AMPM.LST file in the current login directory, using the package account specified on the command line.

For example: `.ampmx pkgdir:`

You should then move the AMPM.LST file to the package directory account.

## TROUTE

TROUTE displays the routers in the path to a specified host, along with the reply time for each. This can be useful for identifying connectivity problems.

TROUTE works much like the TRACERT facility in Microsoft Windows. It sends up to three "ping" packets per hop, starting the packet time-to-live" at 1 and increasing it up to the maximum hop count. At each router, the time-to-live is decremented, and the router decrementing it to zero responds with an ICMP "time exceeded" message.

The Unix 'traceroute' command does much the same thing, but with UDP packets. The reason TROUTE (and Windows TRACERT) use ping packets (ICMP ECHO REQUEST) instead is because UDP is typically blocked at most firewalls now. It is more common to be able to send a ping through a firewall than a UDP packet.

TROUTE uses the following format:

```
TROUTE {-hhops} {-n} {-r} {-Tseconds} host
```

You can specify the maximum number of routers (hops) to attempt using the -h switch. The default is 30 hops.

The `-n` switch forces TROUTE to display IP addresses instead of the routers hostnames. This speeds up the trace, and is especially useful if reverse DNS lookups are failing and must time out.

TROUTE attempts to check the chain of responding routers for routing loops. The `-r` switch disables this. For various reasons false routing loops can be detected. One of the most common is the usage of the same private IP address space in different domains. If you get a "routing loop detected" error on a trace, you should rerun the trace with the `-r` switch.

Normally TROUTE waits for a response for up to 3 seconds. You can change this timeout with the `-t` switch.

## Successful Traces

Here is a sample of a successful TROUTE trace:

```
Tracing route to ns0.verio.net : 129.250.15.61
Maximum of 30 hops, 3 second timeout
Detecting routing loops, printing hostnames

 1 2 ms 2 ms 2 ms 192.168.1.254 (private)
 2 130 ms 126 ms 129 ms ewok.continet.com
 3 * * * Request timed out
 4 148 ms 145 ms 144 ms 192.168.255.1 (private)
 5 161 ms 150 ms 142 ms fa-6-0-0.a01.eugnor01.us.ra.verio.net
 6 146 ms 142 ms 149 ms pdx12-t3-0-1-0.or.nw.verio.net
 7 152 ms 166 ms 139 ms ge-1-0-0.r00.ptldor01.us.bb.verio.net
 8 159 ms 143 ms 139 ms ns0.verio.net

Trace completed in 8 hops
```

Note that the local router is using the private IP address of 192.168.1.254, and one of the routers at the ISP is using the private IP address 192.168.255.1. Had the ISP's router been using 192.168.1.254 also, then TROUTE would have falsely detected a routing loop at hop 4. In that case, the `-r` switch would have allowed the trace to continue.

Hop 3 is a router which doesn't respond to ping requests.

## Failed Traces

If TROUTE receives an error from a router other than "time exceeded", it will display the error and stop the trace at that hop. Here is a sample of a failed trace to a private IP address (the `-n` switch was also used to illustrate printing addresses instead of names):

```
Tracing route to 10.0.0.1 : 10.0.0.1
Maximum of 30 hops, 3 second timeout
Detecting routing loops, printing addresses

 1 3 ms 2 ms 2 ms 192.168.1.254 (private)
 2 137 ms 126 ms 122 ms 206.58.169.33
 3 * * * Request timed out
 4 141 ms 142 ms 144 ms 192.168.255.1 (private)
 5 146 ms 141 ms 139 ms 206.58.214.193
```

```
6 unr:host * unr:host 206.58.214.193
```

```
Trace failed at 6 hops
```

Note that the router at hop 6 responded with a "host unreachable" error. Although it appears there is a routing loop at hop 5, this is not the case. At hop 5, the router decremented the time-to-live to zero and discarded the packet. At hop 6, the time-to-live was still above zero, so the router attempted to route the packet and didn't find a route.

# Chapter 7 - SLIP Dialup Support

SLIP is the Serial Line Internet Protocol, and provides one of the more popular methods for using TCP/IP over serial lines or modems. Most SLIP implementations provide no modem support, requiring the user to establish the connection outside of SLIP. AlphaTCP provides additional capabilities within the SLIP server which allow it to handle the connection process automatically. Since the connection process may be as simple as dialing the modem or as complex as performing a login and requesting conversion to a SLIP connection, the SLIP server reads a script file. You define all the functions needed to establish the SLIP connection in the script file. If your installation uses a hardwired serial interface, you do not need to create a script file.



See the end of this chapter for information on building cables to make SLIP connections between AMOS computers and other computers.

## SERVER SCRIPT PROCESSING

Your modem should be set up so DTR (Data Terminal Ready) will hang up an active connection and DCD (Data Carrier Detect) will indicate a connection exists. Of course, these lines must also be properly connected to your computer (see the section “SLIP Cabling,” below).

The first action performed by the SLIP server before processing the script file is to toggle the DTR line and verify the DCD line indicates no connection exists. With the processing of the first line in your script file, the modem should be ready to accept commands.

The SLIP server then scans through the script file, performing the requested operations. While SLIP processes the script file, the line number is placed in the STAT program name area. For example, if you are looking at another screen running STAT you may see `DIAL7` under the `prog` area. This says the SLIP server is waiting for the seventh line in the script file to complete. If one of the operations fails, the modem is hung up and the failing line is reported to the network error logger. Usually, higher TCP/IP protocols will resend the data, causing another attempt to connect.

Once the script file has been successfully processed, the DCD line is checked for a connection. Provided DCD says a connection still exists, the SLIP server begins passing TCP/IP data. The program area in a STAT display will show that the connection is on and the number of minutes left until inactivity will cause a disconnect. For example, `ON3` means there can be three more minutes of inactivity before disconnect. This value will decrement unless there is activity. If it reaches zero, the connection is severed and STAT will show `OFF`. If STAT shows `ONLINE`, SLIP is active due to a specified connection time in the script file.



If STAT shows the SLIP server sitting in `HANGUP` constantly, then the server is having trouble hanging up the modem. Since DCD is used to detect when the modem is off-line, this usually indicates a cabling problem or modem setup error. It can also indicate an old IDV, incompatible serial port, or serial port failure.

## SCRIPT CREATION

Scripts are created as normal text files using a text editor. They should be placed in the TCP: account. The script file is specified to AlphaTCP via the `ifconfig` command in the TCP:CONFIG. file. See Chapter 3 for a description of the TCP:CONFIG. file.

You may place comments in your script file by starting the line with a semicolon or a pound sign. Comment lines are skipped over. The SLIP server recognizes seven commands: IDLE, REPLY, WAIT, DELAY, SHARED, DISCARD, ONLINE, and CONTINUOUS.

IDLE allows you to specify the number of minutes a connection may be inactive before it is terminated. This lets you override the default of five minutes.

REPLY takes a line of text and sends it to the modem or remote computer. It is used to dial the modem or log into the remote computer.

WAIT waits a number of seconds for a specified line of text to be received from the modem or remote computer.

DELAY simply waits the specified number of seconds before continuing with the script processing.

SHARED causes SLIP to allocate the port only during connections, allowing the modem to be shared with other applications. In this mode, SLIP cannot answer incoming calls.

DISCARD causes any buffered input from the modem or remote computer to be discarded.

ONLINE specifies certain time periods for SLIP to establish and hold a connection with a remote gateway. This is useful if you want to access your computer remotely—for example, over the Internet—but do not need the constant access (or expense) of a leased-line connection.

CONTINUOUS causes the modem to be kept on-line, redialing if the connection is dropped. One side of the connection should be set up as if hardwired (no script). this will prevent both hosts from trying to dial each other at the same time.

Any text specified in the WAIT and REPLY commands must be within quotation marks. Also, carriage returns and line feeds are not implied. If you need these or other embedded characters such as a quotation mark, specify them in a manner similar to the C programming language, with a backslash (\) and a character. For example:

|                 |                          |
|-----------------|--------------------------|
| <code>\n</code> | new line (linefeed)      |
| <code>\r</code> | return (carriage return) |
| <code>\t</code> | tab                      |
| <code>\"</code> | embedded quotation mark  |
| <code>\\</code> | embedded backslash       |



ONLINE commands are processed only once, when AlphaTCP is initiated. You can have multiple ONLINE commands in a script, and list multiple connection times per command, but there is a maximum of ten time periods in a script. You specify a time period by listing one or more two-letter abbreviations for the days of the week followed by the desired time period in parentheses in military (24-hour) format. The first letter of the day abbreviation must be capital and the second must be lower case, so the allowable codes are: Mo Tu We Th Fr Sa Su. If you include multiple connection times in a single command, you must separate them with a space.

Here is a sample ONLINE command:

```
ONLINE MoWeFr(08:00-09:00) TuTh(18:00-18:30)
```

This command causes SLIP to make a connection from 8:00 to 9:00 AM on Monday, Wednesday, and Friday, and from 6:00 to 6:30 PM on Tuesday and Thursday. During these times, the computer is available on the network. At the end of the time period, the standard IDLE countdown begins; the connection is ended after the line has been quiet for the IDLE period.

The following is an example of a script file used to dial into a Dial n' CERF gateway computer via a Hayes modem:

```
; AlphaTCP SLIP dialup script for Dial n' CERF
IDLE 4
REPLY "ATE0Q0\r"
WAIT 1 FOR "OK"
REPLY "ATDTxxxxxxx\r"
WAIT 30 FOR "CONNECT"
WAIT 15 FOR "Username:"
REPLY "xxxxxxx\r"
WAIT 3 FOR "Password:"
REPLY "xxxxxxx\r"
WAIT 5 FOR "pipeline.cerf.net>"
REPLY "slip\r"
WAIT 3 FOR "hostname:"
REPLY "xxxxxxx\r"
WAIT 3 FOR "Password:"
REPLY "xxxxxxx\r"
WAIT 5 FOR "bytes"
DELAY 1
DISCARD
```

The above script sets the idle disconnect time to four minutes, gets the modem's attention, connects to the Dial n' CERF gateway (as a standard terminal), logs in, requests SLIP mode, waits for the last word sent by the gateway in terminal mode, then allows a settle time and cleans up the input buffer before the SLIP connection is officially established.

## THE SLIP LOG FILE

The SLIP server maintains a log file called SLIPDn.LOG where *n* specifies the number of the asyn device the SLIP server is controller. This file logs whenever SLIP answers an incoming call or places an outgoing one. It also logs all disconnects and the length of the connection.

## SLIP CABLING

When constructing a cable to use for a SLIP connection to a modem, you must attach the following pins at the AMOS end of the cable. Note that *direction* is relative to AMOS:

| PIN | SIGNAL | DIRECTION |
|-----|--------|-----------|
| 2   | TXD    | IN        |
| 3   | RXD    | OUT       |
| 4   | RTS    | IN        |
| 5   | CTS    | OUT       |
| 7   | SG     | --        |
| 8   | DCD    | IN        |
| 9   | DTR    | OUT       |

Pins 8 and 9 (DCD and DTR) must be connected! Unlike many other communications packages, AlphaTCP requires these signals. If you don't connect them, AlphaTCP will be unable to tell if there is a connection, and won't be able to hang up the phone. Usually, you can see this in STAT as the server remaining in HANGUP state forever.



The correct pin numbers to wire at the remote end of the cable depend on the type of modem you are connecting to. There are too many possibilities to list here. Refer to the documentation for your modem and connect the pins for the appropriate signals.

If you are constructing a cable for a direct serial connection to another computer, you do not need to wire pins 8 and 9 at the AMOS end. Connect the remaining pins indicated above to the appropriate pins on the other computer.



The SLIP cable *must* use hardware handshaking! If you attempt to use XON/XOFF handshaking the connection will not work.

## MODEM CONFIGURATION

The modem must be configured for proper operation. The following attributes should be set up and saved as the modem power-up defaults:

- Serial interface speed should be set to a fixed rate, matching the baud rate of the port the modem is attached to.
- Hardware handshaking should be enabled and XON/XOFF handshaking disabled.
- XON/XOF characters should be passed to the system.
- When DTR is dropped, the connection should be dropped.
- When carrier is present, the DCD line should indicate this.
- Processing of +++ should be disabled, if possible.

# Chapter 8 - Tuning AlphaTCP Performance

AlphaTCP's performance relies on proper tuning to handle the characteristics of your network. This tuning must be done in a number of places: the IPCINI command, the TCPEMU job's memory assignment, and within the CONFIG. file.

## IPCINI COMMAND

The IPCINI command sets up a message area for communication between an AlphaTCP application and the TCPEMU job. The first value in the command is the number of message queues to allocate; the second is the size of the free buffer area.

You need one message queue for each data path an application opens. Certain applications, such as FTP, open multiple data paths. Increasing the number of message queues has a minimal effect on memory size, so it is a good idea to allow two or three for every AlphaTCP session which will be running, including servers. If an application runs out of message queues you may get a `device full` message (error number 6), although other failure messages are possible.

Space for holding messages passed between the application and TCPEMU is assigned from the free buffer area. These messages are usually under 100 bytes each. Each session usually has only one outstanding message at a time. If an application runs out of message buffer area you may get an `insufficient free memory` message (error number 2), although other failure messages are possible.

## TCPEMU MEMORY

The amount of memory you assign to the TCPEMU job determines the quantity of internal resources AlphaTCP can assign. AlphaTCP uses these resources in many ways, including holding messages in transit, time-out entries, and session state information.

When AlphaTCP sets up memory it outputs the memory allocations to the network logger for you to examine. The most important values are the number of streams and the layout of the "mblks" (message blocks). You change these values by adjusting the amount of memory assigned to the TCPEMU job.

The number of streams determines the maximum number of data paths available. There is a close relationship between this value and the number of message queues set up by IPCINI, as each used message queue will likely have a stream associated with it. TCPEMU uses about ten streams for itself, so this value should optimally be ten higher than the IPCINI message queue count.

The message block allocations are discussed in the next section.

## CONFIG. FILE

### Ifconfig Setup



This does not apply to `ifconfig` lines dealing with standalone LDVs, only those for AlphaNET NDVs. The interface to an LDV is different and does not use the setup below.

If you are using AlphaTCP on an Ethernet connection, the `ifconfig ec0` command in CONFIG. is very important. The value at the end of the line pre-allocates message blocks (called mblks in the NETLOG file) for use by the interrupt-level Ethernet driver. These message blocks store incoming TCP/IP packets. If this value is too small, you drop packets and performance is poor. If this value is too large, you rob the rest of AlphaTCP of needed resources, resulting in poor performance or failure.

The total number of message blocks assigned to the `ec0` device is broken down into three sizes: 64, 512, and 2048 bytes. 20% of the specified value is assigned as 2KB buffers, 5% are 512 byte buffers, and the remaining 75% are 64 byte buffers. This breakdown is output to the network logger so you can compare it to the breakdown of message blocks available.

If the number of buffers assigned exceeds 1/3 to 1/2 of any message block group you may want to reduce the buffer count or increase the memory assigned to TCPEMU.

You can verify that the `ec0` device has enough buffers available by typing:

```
NETSTA -e RETURN
```

If the packets lost value is non-zero then messages are arriving with no message block to place them in. If so, increase the buffer count. Be aware that you may need to increase the TCPEMU memory at the same time.

### TCPEMU.MBD FILE

Messages pass through AlphaTCP in message blocks, known as 'mblks'. These are created in eight different sizes when AlphaTCP is started. The number of mblks assigned to a given size depend on memory available and a preset distribution pattern. Preference is given to highly used sizes, although usage patterns may vary from those assumed by the default distribution.

The TCPEMU.MBD file lets you change the distribution pattern. A new distribution pattern is created by listing quantities separated by commas for all eight mblk sizes. These go from small to large mblks. You may use the file TCPEMU.NEW as a sample. This file contains the same numbers used internally as the default distribution.

Note that the numbers do not define the actual number of mblks created. TCPEMU uses the numbers and mblk sizes represented as a baseline. This baseline is then multiplied by the amount of memory available to TCPEMU to actually create the mblks. When adjusting the distribution, you may want to decrease some values when you increase others, otherwise the minimum memory requirements for TCPEMU will increase.

## Determining Allocation

The actual number of mblks created in each size is written as a table to the network log file at startup. See Chapter 6 for information on controlling the network logger. You may use this information to see what the effect of changing the distribution is.

## Determining Usage

You may determine the highest and current mblk usage at any time. When you send the TCPEMU job (pid #1) a 'usr1' signal, it outputs this information to the network log file. See Chapter 6 for information on UKILL and sending signals. The lines list highest and current use percentages for each of the mblk sizes from small to large.

If any of these values approach 70%, you may want to increase the distribution for that mblk size. At the same time you may wish to decrease the distribution for sizes with very little usage.

Actual usage of the various mblk sizes depend greatly on the network paths in use (Ethernet, SLIP, or a routed connection). It also depends on the applications run (telnet, ftp). It is best to examine the distribution after the system has been active for awhile.



# Chapter 9 - Advanced Web Server Usage

This chapter reviews how web documents are indexed, then continues with information on supporting image maps, Web page “hit counters,” fill-out forms, virtual domains, and cookies with the AlphaTCP World Wide Web server.

## HOW WEB DOCUMENTS ARE INDEXED

Web documents are indexed using a Uniform Resource Locator (URL). The URL defines exactly where the document may be located and how it should be retrieved. A URL looks like a UNIX path as in the following example:

```
http://www.widgit.com/sales/new-product-info.html
```

It is important to understand how each section of the URL is handled by a web browser in order to create proper URLs for the AlphaTCP web server. The URL is broken down as follows:

***http:*** is the transport method and stands for HyperText Transport Protocol. This is how the document is to be retrieved. This is the most common method, though others, such as `ftp:` and `telnet:`, may be seen on occasion.

`//` specifies that the next field is the host name of the system which has the document. In this case, the host name is `www.widgit.com`.

The remainder of the URL is a UNIX style path name to a file. As the World Wide Web originated on UNIX hosts, this method was adopted. How this is handled by AMOS is explained further on. Usually web pages will be stored in files with a `.html` or `.htm` extension. You will also see `.gif` and `.jpg` files, which are images for the browser to display. Browsers handle most types of files, either directly or by starting a program which understands the file type.

## Absolute and Relative URLs

Examine the example URL once more:

```
http://www.widgit.com/sales/new-product-info.html
```

The above URL is known as an absolute reference because it is complete. References which are incomplete are known as relative references. With relative references, the browser fills in the missing pieces based on the location of the previously accessed document.

For example, if the above referenced document contains the following link:

```

```

The browser assumes the link is to the same directory on the same system, thus generating the complete URL:

```
http://www.widgit.com/sales/new-product-prices.html
```

## Same System, Different Directory

To designate a file in a different subdirectory branch, place a forward slash before the filename. This makes the reference relative to root. The browser will not then assume the new subdirectory is part of the current one. For example, if the above link were changed to:

```

```

The browser will access the complete URL:

```
http://www.widgit.com/support/patches.html
```

## How AMOS Handles URLs

Though AMOS does not have paths like UNIX, the web server handles URLs as if it did, with a few limitations. The transfer method and host name are exactly the same as a URL for a UNIX system; it is the path and filename which require some explanation. Again, let's refer to the original example URL:

```
http://www.widgit.com/sales/new-product-info.html
```

When the AlphaTCP web server processed this URL, it converts sales/ into the ersatz name SALES:. This is important to note: The AlphaTCP web server must deal in ersatz names only: you cannot use a PPN within a URL. Besides being easier for the web server to deal with, many browsers do not handle PPNs within a URL well.

The remainder of the URL is treated as a filename. It is converted into a usable AMOS filename using the method described in the *AlphaTCP Users Guide* section, "Converting UNIX Filenames to AMOS." You can use the program FNU2A.LIT to see how a UNIX path will be converted. In this case the converted ersatz and filename yield SALES:NEPRIN.HTM.

## Single Level Subdirectories

It is important to note that the above processing generates the illusion of subdirectories. While it is possible to have nested subdirectories on UNIX, this cannot occur on AMOS. Therefore, subdirectory references must only exist one level deep, which allows a single ersatz name.

Further references which supply only a filename will cause the web browser to access the same account as the document containing the reference. This is handy when the files referenced by a document are in the same account as the document itself.



When files are located in a different account, the reference must be made relative to root using a leading forward slash. For instance, refer to the original example URL again:

```
http://www.widgit.com/sales/new-product-info.html
```

If the above document contains the link:

```

```

The web browser will improperly attempt to access the image file as:

```
http://www.widgit.com/sales/images/bumper.gif
```

Obviously, accessing SALES:IMAGES:BUMPER.GIF is an error. The proper reference to the file IMAGES:BUMPER.GIF (where the file is actually stored) would be:

```

```

### Limiting Access to Directories

By manually entering URLs, malicious users may attempt to request files they are not entitled to. To prevent this, the AlphaTCP web server only allows access to files if the account contains the file PUBLIC.PPN.

The contents of the public.ppn file are unimportant—only the filename itself is searched for. If this file exists, files in the account may be read by the web server and a number of other AlphaTCP servers. Do not place this file in any account containing passwords, such as OPR: or TCP:, or any account containing sensitive information.

## SUPPORTING IMAGE MAPS

Image maps allow an image which contains hot-spots to be displayed by a web browser. When the user clicks on different portions of the image, the coordinates are passed to a special application on the server's system. This application returns the URL defined for the selected hot-spot.

Image maps are created using an existing image, usually a GIF file, and a hot-spot editor. This is usually done on a PC or other graphical system. Most web guidebooks have information on creating and referencing image maps. AlphaTCP supports NCSA-format image maps. Special AlphaTCP web server requirements follow:

All links to the map file for the image should begin with the word **/imagemap/**. For example, the proper reference to the map file for the image web:doors.gif (assuming the map file is in the same directory as the image) would be:

```

```

This tells the AlphaTCP web server to start the imagemap application and pass it the filename WEB:DOORS.MAP.

The imagemap application needs to be described to the AlphaTCP web server. This is done by placing the following definition in the TCP:HTTPD.APP file:

```
#Image mapping function
Application=imagemap
Command=IMGMAP %e %o %a
Memory=50k
Kill=Yes
Decode=No
```

The program IMGMAP.LIT should be loaded into system memory. This program handles the coordinate conversions. The application name and the actual command executed do not have to match, as can be seen above. If the program is not loaded into system memory, the *memory=* line must be doubled and the program will need to be loaded from disk every time it's needed.

## SUPPORTING HIT COUNTERS

Hit counters let you monitor the number of times a web page has been accessed and optionally display the count on a page. AlphaTCP includes a very simple hit counter which returns a basic monochrome XBM image. The count can increment to a maximum of 999999, after which it wraps back to 1.

To add a counter to a page, use this image tag format:

```

```

*ersatz* is the ersatz name for the directory where the web page is located.

*page* is the name of the web page being counted; it must contain a .HTM extension.

*options* is a list of display options. You must supply at least one option, and may list several together. The available options are:

N	No increment (display only)
P	Print count on the page
Q	Quietly increment (no display)
R	Reverse the image colors
S	Print small digits
Z	Display leading zeros

For example:

```

```

This counts the hits to the Web page file WPAGE2.HTM in the account with the ersatz name WEB: and displays the results with reversed image colors and leading zeros.

To use hit counters, you must describe the hit counter application to the AlphaTCP web server. Place the following definition in the TCP:HTTPD.APP file:

```
Page access counter
Application=ticker
Command=WEBCNT %o %f %a
Memory=5K
Kill=Yes
Account=SYS:
Decode=No
```

The program WEBCNT.LIT must be loaded into system memory. This program increments the counter and returns the graphical representation of it.



If you want to use the counts in a program, the following information may be useful: Counts are stored in one block random files in the same directory and with the same name as the page being counted, with the extension .CNT. The count is stored in the first longword of the block; the rest of the file is unused and undefined. If you access this value in AlphaBASIC using a 4-byte binary variable you will need to swap the upper and lower words.

## SUPPORTING FILL-OUT FORMS

Fill-out forms provide a way to submit user-supplied information to a program for processing, or starting a program which generates a dynamically changing document. Under the UNIX environment these programs are known as CGI-BIN scripts.. They will be referred to from this point on simply as applications. The image map program described above is one such application.

The syntax for laying out a form is described in a number of web guidebooks. The actual application dealing with the form data will be different, as AMOS is a different environment than UNIX. This is what we will focus on.



Appendix E discusses issues relating to shutting down spawned jobs which are running applications. You need to keep these issues in mind when writing a forms application.

## Always Relative to Root

All URLs dealing with form applications on AMOS must begin with a forward slash. This prevents the browser from prepending path information relative to the form. All applications are defined in a single place, so the directory the form is located in should not be used in the application selection.

## Application Definition File

All applications referenced by the web server must be defined in the application definition file, TCP:HTTPD.APP. Each application should begin with a comment (a line starting with a hash mark or a semicolon), then the `application=keyword`. The format of an application definition is as follows:

```
Comment to describe the application
Application=url-name
Command=actual command with options
Memory=memory needed (K for kilobytes)
Account=Ersatz or dev:[ppn] to run in
Kill=Yes or No
Decode=Yes or No
Mime=type/code
```

The line ***application=*** contains the application name as found in a URL. Although you always place a forward slash before the application name in a document, you do not place the forward slash on this line.

The line ***command=*** defines the actual AMOS command line to run. There are a number of special *meta-characters* available on this line. They are listed below after some further explanation of why they're needed.

The line ***memory=*** specifies the amount of memory required by the application. Follow the number with a *K* (no space) to specify kilobytes.

The line ***account=*** is the account the program will be executed in. This may be an ersatz name or a device and ppn specification.

The line ***kill=*** says whether or not the application should be terminated if the browser disconnects. Usually this should be *Yes*.

The line ***decode=*** tells the web server whether or not to convert browser-encoded data into plain text. Set this to *Yes* if you want your fields returned on individual CRLF terminated lines.

The line ***mime=*** tells the server to prepend its own headers and the type of data the application is outputting. Usually this will be *text/plain* for raw text, or *text/html* for a formatted web document. If this line does not exist, your output must contain an HTTP status line, MIME headers, and a blank line before the actual data begins.

## UNIX and AMOS Differences

Due to differences between the AMOS and UNIX environments, applications supporting forms will also be different. The interface is flexible enough to allow applications to be written in any language: assembler, C, or BASIC. This section explains the need for the command line meta-characters.

## Environment Variables

In UNIX, many parameters are passed in *environment variables*. These are not a standard part of AMOS. Instead, the same information is written to a file. If an application needs the information, it looks it up in the file. The filename may be passed to the application on the command line using the *%e* meta-character.

## Application Input

Input is usually passed to an application via the *stdin* stream and *pipes* on UNIX. These are not available on AMOS. Instead, the application has two choices how it wants to retrieve its input. By default, input will be forced to the application as if a keyboard was attached. As an alternative, input may be written to a file which the application may read. The filename may be passed to the application on the command line using the *%i* meta-character.

## Application Output

Output is usually returned from an application via the *stdout* stream and *pipes* on UNIX. These are not available on AMOS. Instead, the application has two choices how it wants to return its output. By default, output to the screen will be trapped by the server and returned to the web browser. As an alternative, output may be written to a file and the screen information will be ignored. The filename which must be used as the output file is passed to the application on the command line using the *%o* meta-character.



If your application traps output rather than writing to a file, the TELTYP terminal driver *must* be loaded in system or user memory. It's usually easiest to load this driver into system memory in your system initialization file.



If you would prefer to have the web server return an existing page to the user rather than generating one yourself, output the following as the only line in the output file:

```
sendfile:filespec
```

where *filespec* is the page or file to be returned. The file should not reside on a publicly accessible device nor in an account with the PUBLIC.PPN file, as the file would then be accessible without running the application

## Filenames Appended to a URL

When an application is referenced by a URL in a document, the application name may be followed by a filename simply by continuing the path. This filename may be passed to the application on the command line using either the *%f* or *%v* meta-characters.

Both meta-characters pass a name which has been converted with the standard filename conversion function. The *%v* meta-character validates the name as a file which actually exists in an account with public access (one with a public.ppn file in it). If validation fails, the application will not be started and the browser will be sent an error.

As an example of this filename usage, assume the following link occurs in a document:

```
<FORM ACTION=" /demo-program/demos/sample.txt ">
```

The *demo-program* application will be started, and an occurrence of *%f* or *%v* on the command line will be replaced by the filename DEMOS:SAMPLE.TXT.

## GET Method Arguments

Forms have two methods of submission: *GET* and *POST*. When the *GET* method is used, the web browser will encode the fields and append them to the URL following a question mark. These arguments may be passed to the application on the command line using the *%a* meta-character.

The web server usually determines that a web application should be run, rather than a web page retrieved, by the existence of a *query string* at the end of the URL. This is a question mark, followed by optional arguments. There are cases where you may want to invoke an application without requiring a question mark. For instance, if a user is expected to enter the URL, and it has no arguments, they are very likely to forget to type the question mark. For this reason, HTTPD recognizes a very common directory in the path; *cgi-bin*. Both of the following URLs will invoke the *basic-demo* application:

```
http://www.widget.com/basic-demo?
http://www.widget.com/cgi-bin/basic-demo
```

## POST Method Arguments

When forms are submitted using the *POST* method, the fields are sent encoded as part of the input stream. Depending on existence of the command line meta-character *%i*, this information may be forced to the application as keyboard input, or passed in a file. If your application would rather not deal in raw encoded information, place the following line in your application definition:

```
Decode=Yes
```

This causes the server to decode the fields and turn them into individual CRLF terminated lines.

## Supplying MIME Headers

If you specify a MIME type in the application definition file, the server handles the details involved in returning a proper completion code and MIME header. If you wish to supply your own MIME header, you also must supply the normal server completion code with it. The completion code must be the first line of output, immediately preceding the MIME header. It consists of the server protocol (for instance, HTTP/1.0), the code value, and a string. Sample completion code lines include:

HTTP/1.0 200 OK	Requested information follows
HTTP/1.0 302 Found	Redirect to another URL
HTTP/1.0 304 Not Modified	Use local copy, unchanged since last access
HTTP/1.0 400 Bad Request	Unrecognized input provided
HTTP/1.0 401 Unauthorized	Authorization required for document
HTTP/1.0 403 Forbidden	Not allowed to provide document

HTTP/1.0 404 Not Found	Document was not located
HTTP/1.0 500 Server Error	Catch-all error
HTTP/1.0 501 Not Implemented	Requested operation not implemented
HTTP/1.0 6992 No Memory	Ran out of memory

## Command Line Meta-Characters

The following meta-characters may be used on the command= line. The actual value of the meta-character will be substituted at the position it occurs on the command line.

%%	Place a percent sign at this point on the command line.
%a	Place GET method arguments here.
%e	Place the environment variable filename here.
%f	Place the filename from the URL here.
%i	Place the input filename here.
%o	Place the output filename here.
%v	Same as %f, but validate the filename first.

## PUT AND DELETE METHODS

HTTPD supports two methods which are often used to update web pages and remove files; *PUT* and *DELETE*. Neither of these methods is directly handled by HTTPD. Instead, either cause the web application called *http-publish*. By adding an *http-publish* definition to TCP:HTTPD.APP, you may start your own program to handle the transaction. Your program should contain functions to request and check authorization before performing the PUT or DELETE. See [Supporting Authorization](#) below for how authorization verification is done.

## SUPPORTING COOKIES

Cookies are used to store small bits of information in the browser. Cookies are often used to hold user preferences and shopping cart selections. Cookies are created and processed by web applications. A sample program, BAS:WEBCKY.BAS, is provided in the AlphaTCP release.

Cookies contain a name, value, path, and expiration. The path and expiration are optional.

When a path exists, it defines a subset of the path portion of the URL to which the cookie is valid for. Usually you will set the path to the web application the URL runs. In other words, if the web application *cookie-demo* is usually invoked using a query string (arguments following the URL, separated by a question mark) then the path would be */cookie-demo*. If the web application is invoked with the special prefix *cgi-bin*, then the path would be */cgi-bin/cookie-demo*. If the web application could be invoked by either method, set the path to */* and the browser will supply the cookie in all requests to the system regardless of the path.

Expiration is tricky, as it must be supplied relative to the GMT timezone. If no expiration is given, the cookie will expire when the browser exits.

## SUPPORTING AUTHORIZATION

Authorization allows you to request a user name and password before supplying a document. The document may be located in a directory without PUBLIC.PPN thus preventing direct access to it, or it may be generated in response to the request.

Authorization is done with a user-supplied program. A skeleton AlphaBASIC application, BAS:WEBAUT.BAS, is provided in the AlphaTCP release to assist with this. Three functions must be supplied in the program to make it useful. These functions convert a device/ppn string to a user-friendly form, validate the user name and password, and generate the actual page. The comments at the start of the skeleton program describe a .DO file which must be created, and the application definition which must be added to TCP:HTTPD.APP.

Each link to a protected page should take the following form:

```
/cgi-bin/auth/page.htm
```

The path */cgi-bin/auth* starts the authorization program. The filename *page.htm* would be replaced by the protected document name.



If you would prefer to have the web server return an existing page to the user rather than generating one yourself, output the following as the only line in the output file:

```
sendfile:filespec
```

where *filespec* is the page or file to be returned. The file should not reside on a publicly accessible device nor in an account with the PUBLIC.PPN file, as the file would then be accessible without running the application

## SUPPORTING VIRTUAL DOMAINS

There are two methods servers may use to support virtual domains. The original method involved assigning multiple IP addresses to a server, which wasted address space. The newer method was defined in HTTP 1.1, and is the method used by HTTPD. Browsers supporting HTTP 1.1 send a header line identifying the name of the host in the URL. This line is used to identify which web space to display.

The first step is to define the following application in TCP:HTTPD.APP:

```
Virtual domain redirector
Application=virtual
Command=VDOM %e %o
Memory=20K
Kill=No
Account=TCP:
Decode=No
```

Second, move your TCP:HTTPD.HTM page to an account used to hold your primary web domain. You should probably also rename it to HOME.HTM. Replace the contents of the TCP:HTTPD.HTM file with the following line:



```
Redirect: /cgi-bin/virtual
```

Next, create the file TCP:VDM.CFG and place the hostname-to-url mapping lines in it. The format of these lines is: *{host}={url}*. For example, you might have a home page for your company which sells widgets, and a different one for your school of widget design. Here is an example configuration file for this:

```
www.widgets.com=/sales/home.htm
www.widget-design.edu=/teach/home.htm
```

This should allow display of the proper home page for a domain. The rest of the pages in the domain should not need redirection.

Finally, you might want to handle old browsers which don't supply a Host: line. Normally, VDOM will return an error. You might prefer to display your own page, which may contain a selection of the domains you are hosting instead. This is done by creating a web page with the name TCP:VDM.HTM. If this file exists, VDOM will return it instead of an error message.

## SAMPLE APPLICATIONS

These sample applications demonstrate a couple of simple uses of the web server's form support.

### SYSTAT Demo

The following simple application performs a SYSTAT. As there is no *%o* meta-character on the command line, the program output is trapped and used. In the application definition file, add the following definition:

```
#Demo application to run SYSTAT
Application=get-system-status
Command=SYSTAT
Memory=50k
Kill=Yes
Decode=No
Account=SYS:
Mime=text/plain
```

In a web document, add the following lines in the body:

```
<FORM ACTION="/get-system-status" METHOD=GET>
<INPUT TYPE="submit" VALUE="Go">
</FORM>
```

Accessing this page and clicking on the Go button should result in the output from a SYSTAT appearing on your screen.

## BASIC Demo

The following application is a bit more complicated. It runs a BASIC program which repeats back some of its configuration as a web document. As both the %i and %o meta-characters are present, all I/O is performed through files. It also demonstrates a way to handle the command line parameters without needing an XCALL, through the use of a DO file.

In the application definition file, add the following definition:

```
#Demo application to run a BASIC program
Application=basic-demo
Command=GORUN.DO WEBDMO %i %o %e %f
Memory=100k
Kill=Yes
Decode=Yes
Account=BAS:
Mime=text/html
```

In the CMD: account, create the GORUN.DO file and enter into it the following six lines:

```
:R
RUN $0
$1
$2
$3
$4
```

In the BAS: account, create the file WEBDMO.BAS and enter the following program:

```
MAP1 A$,S,100
MAP1 F$,S,100
INPUT LINE "",I$
INPUT LINE "",O$
INPUT LINE "",E$
INPUT LINE "",F$
OPEN #1,O$,OUTPUT
PRINT #1,"<HTML><HEAD>"
PRINT #1,"<TITLE>Form Demo for BASIC</TITLE>"
PRINT #1,"</HEAD><BODY>"
PRINT #1,"<H1>BASIC Form Demo</H1>"
PRINT #1,"<H2>Command Line Parameters</H2>"
PRINT #1,"Input filename: <I>"I$;"</I>
"
PRINT #1,"Output filename: <I>"O$;"</I>
"
PRINT #1,"Environment variables filename: <I>"E$;"</I>
"
PRINT #1,"URL embedded filename: <I>"F$;"</I>"
PRINT #1,"<HR><H2>Environment Variables</H2>"
OPEN #2,E$,INPUT
E'LOOP:
INPUT LINE #2,A$
IF EOF(2)=0 PRINT #1,A$;"
" : GOTO E'LOOP
CLOSE #2
PRINT #1,"<HR><H2>Input File Contents</H2>"
OPEN #2,I$,INPUT
I'LOOP:
INPUT LINE #2,A$
```

```
IF EOF(2)=0 PRINT #1,A$;"
" : GOTO I'LOOP
CLOSE #2
PRINT #1,"</BODY></HTML>"
CLOSE #1
END
```

Once this program is entered, compile it and verify there are no errors. In a web document, add the following lines in the body:

```
<FORM ACTION="/basic-demo/sys/ver.lit?testing" METHOD=POST>
Last Name:
<INPUT TYPE="text" NAME="last name" SIZE=20 VALUE="Doe">
<P>First Name:
<INPUT TYPE="text" NAME="first name" SIZE=20 VALUE="John">
<P>Deceased:
<INPUT TYPE="radio" NAME="dead" VALUE="yes">Yes
<INPUT TYPE="radio" NAME="dead" VALUE="no" CHECKED>No
<P><INPUT TYPE="submit" VALUE="Send">
<INPUT TYPE="reset" VALUE="Reset">
</FORM>
```

Accessing this page and clicking on the *Send* button should result in the output from the program appearing on your screen as a formatted web document. Compare the output with the source document above to see how some of the features work in real life. The program will return a document which looks something like this:

## BASIC Form Demo

### Command Line Parameters

```
Input filename: W00011.WAI
Output filename: W00011.WAO
Environment variables filename: W00011.WAE
URL embedded filename: DSK0:VER.LIT[1,4]
```

### Environment Variables

```
Query-string: testing
Path-info: sys/ver.lit
Path-translated: DSK0:VER.LIT[1,4]
Remote-addr: 192.245.218.73
Request-method: POST
Script-name: basic-demo
Server-protocol: HTTP/1.0
Server-software: AMOS-HTTPD/1.3(106)
Server-name: stevel.alphamicro.com
Server-port: 80
End-trusted-environment:
Content-length: 37
Content-type: application/x-www-form-urlencoded
```

### Input File Contents

```
last name=Doe
first name=John
dead=no
```

Some interesting features may be observed in the above output. For example:

```
URL embedded filename: DSK0:VER.LIT[1,4]
```

This item was provided to the program via the *%f* meta-character. The source of this filename is the remainder of the path in the URL:

```
/basic-demo/sys/ver.lit?testing
```

This feature may be used anywhere you need to provide a filename for an application to process. Had the *%v* meta-character been used, the existence of the file and accessibility (via PUBLIC.PPN) would have been verified first, thus preventing a malicious user from entering arbitrary file names. The same information is also available in raw and processed forms via the following two lines in the environment variables file:

```
Path-info: sys/ver.lit
Path-translated: DSK0:VER.LIT[1,4]
```

Another item of interest in the environment variables is this:

```
Query-string: testing
```

The source of this string is the remainder of the URL beyond the path:

```
/basic-demo/sys/ver.lit?testing
```

This may be used to pass arguments to the application and is the standard argument passing mechanism of the GET method. The arguments must be in *URL encoded* format. The same query string may be passed to the application on the command line using the *%a* meta-character.

The input file contains lines for each field on the form. The name of the field as defined on the form is followed by an equal sign and the value of the field.

# Appendix A - NETLOG Error Messages

The following messages may be seen in the network error log due to startup problems. Other messages which may appear in the log file or be output by client programs are listed in the *AlphaTCP User's Guide*.

xxx/xxxxx: unknown service

A message of this form indicates the TCP:SERVIC. file could not be accessed, there is a problem with it, or it is old and the requested service is not listed in it.

ARP entry received from address xxxxxxxx on net xxxxxxxx

Indicates one or more systems are on the network with an IP address outside of the network range. The hex values in the message are the IP addresses of the remote and local systems. These can be translated into a real IP address by converting each byte into decimal.

Can't get xxx host id in TCP:NETWRK. file, exiting

There is a problem with the format of the line for device xxx in the NETWRK. file.

?cannot load TCP100.OVR

The file TCP100.OVR was not found in DSK0:[1,4].

?cannot open package location file LIB:AMPM.TXT

AMPM.TXT maps company names to the web address of their package directories. An example file, LIB:AMPM.NEW, is provided in the release.

?command line too long, increase terminal buffer sizes

The line length of your terminal, as set up by TRMDEF, is too short to accept the command AMPM is generating.

ed\_putmsg : transmit error status = x

A packet could not be transmitted on the Ethernet. Usually this is followed by an ETHER TX failure message. The status code is returned from the NDV or LDV. A status code 5 usually indicates excessive Ethernet collisions.

ftpd: tcp:ftpusr. not present

The FTP user login file has not been created.

httpd: could not open mime types file LIB:MIME.TYP

The MIME types file does not exist. A sample file, MIME.NEW, was provided with AlphaTCP and likely was not renamed during the installation process.

?Improper SSD

The PIC entered does not match the SSD installed in the computer, or the TCP100.OVR file is mismatched or corrupted.

itcd: bad translation line in file tcp:itcd. for xxxxxxxx-  
There is a syntax error within the TCP:ITCD. file on the line which should map the displayed AlphaNET address into an IP address.

itcd: remote connect failed: no AlphaNET mapping for xxx.xxx.xxx.xxx  
There is no entry in the remote hosts TCP:ITCD. file for the local host's IP address. The TCP:ITCD. files on both systems must contain references to each system.

itcd: remote connect failed: node insertion failure  
The remote ITCD probably ran out of memory.

itcd: specified address already defined  
There is already a network defined with the number ITCD is trying to use. You may be trying to run ITCD more than once. Most likely, you have the network number also used by a driver invoked with NETINI. They must have different NETWORK numbers.

itcd: unable to resolve local address  
There is no entry in the TCP:ITCD. file for any of the local host's IP addresses. ITCD looks there to determine the AlphaNET CPU ID (and therefore the Network, Group, and Node numbers) it should assign itself.

lpd: bad -f option (valid options: -fD -dN -fY)  
You specified a formfeed option on the start line in CONFIG., but there is a syntax error with it.

lpd: cannot log to TCPLPD:  
The LPD server was unable to log to the TCPLPD: ersatz. Either the ersatz name or the account does not exist.

lpd-c: connection lost reading control / data file  
The spawned LPD server was receiving a control or data file, but the remote shut down the connection.

lpd-c: device full while reading control / data file  
The spawned LPD server was receiving a control or data file and the device where TCPLPD: resides became full.

lpd-c: failed to read request line  
The spawned LPD server was reading a request but the remote shut down the connection.

lpd-c: failed to spool to printer 'XXXXXX'  
The file was sent to the spooler but it was rejected. The spooler requested by the remote may not exist.

lpd-c: filtering cannot open data file  
LPD was attempting to filter the temp file for a print request submitted with format 'f' but the file is no longer there.

lpd-c: filtering cannot open temp output file  
There is a directory or disk problem on the device where TCPLPD: resides

lpd-c: invalid command 0xXX  
The spawned LPD server received an unrecognized request from the remote.

lpd-c: no valid print format in control file

The control file does not contain one of the supported format types. These are 'f', 'l', 'o', or 'v'.

lpd-c: unable to create temp control / data file

There is a directory or disk problem on the device where TCPLPD: resides

lpr: bad printer line: xxxxx

The displayed line has a format problem.

lpr: cannot determine hostname

The TCP:MYNAME. file is empty or the host name defined in it is invalid.

lpr: cannot log to TCPLPR:

The LPR client was unable to log to the TCPLPR: ersatz. Either the ersatz name or the account does not exist.

lpr: gethostname failed

There is no TCP:MYNAME. file or there is a problem with it.

lpr: no configuration file

There is no TCP:LPR. file to define the printers.

lpr: no printers defined

The LPR client processed the LPR. file but found no valid printer definitions.

lpr: unable to register with TLP device

The TLP device driver is not defined by a DEVTBL statement, or is not loaded into system memory.

?Need AT LEAST nnnK more memory to run!

This indicates enough memory was available in the TCPEMU job to begin initialization, but there is not enough to configure a minimal installation.

?Not enough memory to run TCPEMU

This indicates there is not enough memory allocated to TCPEMU to begin configuring AlphaTCP at all. You need to add quite a bit more memory.

pop3d: cannot access TCPPOP:

The POP server was unable to log to the TCPPOP: ersatz. Either the ersatz name or the account does not exist.

pop3d: electronic mail driver not installed

There is no LODEMD in the system initialization file to load the mail driver, or it failed to install.

rwhod: can't create TCP:RWHOD.DAT

There was a problem allocating the host data file. There must be at least 100 contiguous blocks on the device containing the TCP: account.

smtpd: cannot access TCPMAL:

The SMTP server was unable to log to the TCPMAL: ersatz. Either the ersatz name or the account does not exist.

smtpd: cannot allocate email driver impure memory

The mail driver needs more memory than the SMTP server has available.

smtpd: cannot begin email driver access  
The mail driver has refused to grant access to the mailbox of the SMTP server. By default this is the user Internet.

smtpd: could not resolve local addresses, exiting  
A failure occurred retrieving the local hosts internet addresses from TCPEMU or no networks are defined.

smtpd: could not resolve local hostnames, exiting  
The SMTP server tried to resolve the host name in TCP:MYNAME. but was unable to. Either the name is bad or there was a problem contacting a domain name server.

smtpd: daemon mailbox 'xxx' invalid  
The SMTP server mailbox name was not found. By default this is the user Internet.

smtpd: electronic mail driver not installed  
There is no LODEMD in the system initialization file to load the mail driver, or it failed to install.

smtpd: postmaster mailbox 'xxx' invalid  
The postmaster mailbox name was not found. By default this is the user Postmaster.

smtpd: user list unsupported by email driver, disabled  
The user list mailbox option was enabled, but the mail driver does not support it.

sntpd: remote host unsynchronized  
The server was contacted but is temporarily unsure about its accuracy. SNTPD will try later.

sntpd: trouble resolving timeserver name 'xxx'  
The server's IP address could not be determined. Either there is a problem with the name or the name server was temporarily unreachable. Check the name; SNTPD will try again later.

Start up procedure failed  
There is a problem with one of the start lines in the CONFIG. file.

stream xx, error 531 (Connection reset by peer)  
The remote process closed the TCP connection but didn't use the standard handshaking. Either the connection was hung, or the remote simply chose this method. Web browsers typically will do this in the normal course of operation to avoid creating TIME-WAIT sessions on the server. Sometimes this message is followed by messages from the server indicating a socket read error.

?System size exceeds package license.  
The AMOS monitor indicates it is licensed for a larger number of users than AlphaTCP. The AlphaTCP license is based on total operating system license size, not the number of AlphaTCP users.

TCP access layer is not running  
The TAMED server is not running.

telnetd: Unable to register service with NETSER  
The NETSER job is not responding to name registry requests or the name is already registered by another server. If NETSER is not being used, ignore this message.



?unable to locate 'company' in location file LIB:AMPM.TXT

Either the company you specified on the command line is not listed in the map file, or you mistyped the command line.



# Appendix B - Installation and Configuration Worksheets

The following worksheets will help you install and configure AlphaTCP. The worksheets included are:

- Calculating AlphaTCP Memory Requirements
- Initial AlphaTCP Setup
- Setting Up AlphaTCP Post Office Server
- Setting Up AlphaTCP Telnet Sessions
- Setting Up AlphaTCP File Transfer
- Setting Up AlphaTCP Unsecure File Transfer
- Setting Up AlphaTCP For Use With Print Servers
- Setting Up the AlphaTCP Web Server
- Setting Up AlphaTCP For Use With AMOS Printers
- Setting Up ITC Tunneling through AlphaTCP

Feel free to photocopy these worksheets for your use.



# Calculating AlphaTCP Memory Requirements

## CALCULATING TCPEMU MEMORY REQUIREMENTS

Functions	Number	X	Memory (in KB)	Total (in KB)
Base	1	X	700	700
Total Possible Connections		X	20	
Total ( <i>tcpmem</i> )				

## CALCULATING SMEM MEMORY AND ADDITIONAL JOB REQUIREMENTS

Functions	Number	X	Memory (in KB)	Total (in KB)
File Transfer Protocol (ftpd) <sup>1</sup>	+1 <sup>2</sup>	X	56	
Line Printer Protocol (lpr) <sup>1</sup>	+1 <sup>3</sup>	X	41	
Simple Mail Transfer Protocol (smtpd) <sup>1</sup>	+2 <sup>4</sup>	X	131	
Trivial File Transfer Protocol (tftpd) <sup>1</sup>	+1 <sup>2</sup>	X	41	
Network Computer Status (rwhod)	0 or 1	X	81	
Serial Line Internet Protocol (slipd)	interfaces	X	81	
Virtual Terminal Connection (telned)	0 or 1	X	106	
TELNET User Application Memory		X	appl. req.	
Postoffice Server (pop3d)	+1 <sup>2</sup>	X	66	
Postoffice Password Server (poppwd)	+1 <sup>2</sup>	X	76	
ITC Tunneling (itcd)	0 or 1	X	106	
WEB Server (httpd)	+1 <sup>2</sup>	X	106	
Line Printer Server (lpd)	+1 <sup>2</sup>	X	41	
Totals for SMEM and Jobs	( <i>tcpjob</i> )			( <i>smem</i> )

*tcpmem* is used when setting up the TCPEMU jobs memory allocation.

*tcpjob* is the additional jobs to add to the JOBS statement.

*smem* is the additional memory to add to the SMEM statement. If no other shared memory applications exist then this *is* the SMEM memory allocation.

<sup>1</sup>These memory requirements assume the associated programs are loaded into system memory.

<sup>2</sup>When using this service, enter the maximum number of simultaneous users allowed, plus one. If this service is not used, leave the memory requirements blank.

<sup>3</sup>When using this service, enter the maximum number of TCP based printers, plus one. If this service is not used, leave the memory requirements blank.

<sup>4</sup>When using this service, enter the maximum number of incoming and outgoing connections allowed, plus two. If this service is not used, leave the memory requirements blank.

# Initial AlphaTCP Setup

## SYSTEM INFORMATION

---

Customer/System Name: \_\_\_\_\_

SSD Number: \_\_\_\_\_

AMOS User Count: \_\_\_\_\_

## ALPHANET SETUP

---

*(Skip if using LAN drivers (\*.LDV) in place of AlphaNET)*

Product Installation Code: \_\_\_\_\_

AlphaNET Network Number(for network TCP) (*netno*): 100

## TCP SETUP

---

Product Installation Code<sup>1</sup>: \_\_\_\_\_

Host Internet Address (*hia*): \_\_\_\_\_

Subnet Mask (*snmsk*)<sup>2</sup>: \_\_\_\_\_ (*optional*)

Host Internet Name (*hin*): \_\_\_\_\_  
(host.company.grp grp=.com, .edu, .gov, .mil, .org, .uk, etc...)

Root TCP: PPN (*dev:[ppn]*): DSK0: [ 7, 50 ]

Ethernet Buffers (*ethbuf*): 100

TCPEMU Memory  
(*tcpmem*): 600 KB<sup>3</sup>

SMEM Memory (*smem*): 200 KB<sup>3</sup>

Time Zone: (*-hhmm*) \_\_\_\_\_, (*zone*) \_\_\_\_\_  
Pacific Standard Time (-0800, PST)      Mountain Standard Time (-0700, MST),  
Central Standard Time (-0600, CST)      Eastern Standard Time (-0500, EST).

<sup>1</sup>Installation of an AlphaTCP PIC is not required when running on AMOS 2.3 and later.

<sup>2</sup>The subnet mask is optional. For networks containing subnets contact the network administrator for the subnet mask information. This is necessary when the AMOS system must access systems or printers in a subnet.

<sup>3</sup>Use these default values for simple, lower use systems and for initial test. Use the attached worksheets to calculate exact memory requirements.

## SLIP CONNECTIONS (Define if Using Serial Line Interface to TCP)

---

SLIP Connection	TRMDEF (trmnam)	IDV (idv)	Port # (port)	Baud (baud)	Dialup Script (script) (optional)
asy0 (0)	ASY0				ASY0.SLP
asy1 (1)	ASY1				ASY1.SLP
asy2 (2)	ASY2				ASY2.SLP
asy3 (3)	ASY3				ASY3.SLP

## NETWORK SYSTEMS

---

Does network have a Domain Name Server (*hdns*)?:  Yes  No

### If *hdns* = Yes:

Domain Name Server (*dns*): \_\_\_\_\_  
(IP address)

Redundant Domain Name Server: \_\_\_\_\_  
(IP address)

Redundant Domain Name Server: \_\_\_\_\_  
(IP address)

### If *hdns* = No:

Add to TCP:HOSTS.

	<u>IP address</u>	<u>Host name</u> (host.company.grp)	<u>Alias</u> (host)
Host:	_____	_____	_____
Host:	_____	_____	_____
Host:	_____	_____	_____
Host:	_____	_____	_____
Host:	_____	_____	_____
Host:	_____	_____	_____
Host:	_____	_____	_____
Host:	_____	_____	_____
Host:	_____	_____	_____

(Use additional pages if you need more room for network hosts)

## INITIAL ALPHA TCP/IP SETUP CHECK LIST

---

For installation details see Chapters 2, 3, and 4.

- Verify host meets minimum requirements for OS version, memory, and hardware.
- Download and install AlphaNET 2.3 or later. (*Skip when using LDVs and AlphaNET is not wanted*).
- Download and verify AlphaTCP software.
- Enter the PIC by LOGing to the TCP account and entering TCPEMU. (*Skip for AMOS 2.3 or later*).
- Contact the network administrator or the Network Information Center at SRI International to determine the host internet.
- Add all the requirements for TCP services to determine TCPEMU memory requirements. Use the worksheet provided.
- Add all the requirements for TCP services to determine the Shared Memory (SMEM) area memory requirements. Use the worksheet provided.
- Rename TCP:SERVIC.NEW to TCP:SERVIC.
- Rename TCP:NETWRK.NEW to TCP:NETWRK.
- Rename TCP:CONFIG.NEW to TCP:CONFIG.
- Rename TCP:GOTCP.NEW to TCP:GOTCP.CMD.
- Create and edit TCP:MYNAME. As the only line in this file; enter *hin*.
- Create and edit TCP:TIMZON. As the only line in this file; enter *hmm (zone)*.

### For Ethernet Networks

- Edit TCP:CONFIG. to reflect the following:

#### **If you are using an AlphaNET NDV:**

- Remove the # in front of the **ifconfig ec0...** line containing the word *arpa*.
- Change the network number value. Use the following syntax:  
**ifconfig ec0 netno arpa ethbuf**



## If you are using a LAN driver (LDV):

- Remove the # in front of the **ifconfig ec0 am319s** line.
- Change **am319s** to the name of the LDV you are using.
- Edit TCP:NETWRK. to reflect the following:
  - Remove the # from the **ethernet...** line.
  - Replace the IP address on this line with this systems address. Use the following syntax:  
**ethernet *hia* ether eth net**  
(Note: Subnetted connections are beyond the scope of this quick installation guide).

## For Slip Connections

- Create a dialup script in TCP: for each SLIP connection that requires one. Use the file names from the table above (*script(0-3)*).
- Edit TCP:CONFIG. to reflect the following:
  - Remove the # in front of the **ifconfig asy0...** line.
  - Change the serial port and optional dialup script to the values from the table for asy0. Use the following syntax :  
**ifconfig asy0 *trnam(0)* {*script(0)*}**
  - Copy the **ifconfig asy0..** line and modify accordingly for each additional SLIP connection.
- Edit TCP:NETWRK. to reflect the following:
  - Remove the # from the **slip...** line.
  - Replace the IP address on this line with this systems address. Use the following syntax:  
**slip *hia* serial**  
(Note: Subnetted connections are beyond the scope of this quick installation guide).
  - Copy the **slip..** line and modify accordingly for each additional SLIP connection. Change **slip** to **slip1**, **slip2...** Change **serial** to **serial1**, **serial2...**

## For All Networks

### If *hdns* = Yes:

- Rename TCP:RESOLV.NEW to TCP:RESOLV.
- Edit TCP:RESOLV. and do the following:
  - Remove the # from the front of the **nameserver...** line.
  - Replace the sample nameserver IP address with ***dns*** using the following syntax :  
**nameserver *dns***

- If redundant Domain Name Servers exist (up to 2), add additional **nameserver** lines.
- To have AlphaTCP also search TCP:HOSTS., remove the # from the **tryhostfile** line.
- Add **hia** and **hin** to the Domain Name server by contacting the network administrator.

**If *hdns* = No:**

- Add the network system specifications to the **TCP:HOSTS.** file as needed. Follow the existing syntax for the additions.

**For All Networks**

- Verify that the user System Service exists on the system and does not require a password. If it does not exist or it is undesirable to use this configuration, add the password or a new user by changing the file TCP:GOTCP.CMD to reflect any changes to the LOGin portions (both NETLOG and TCPEMU LOGins) of the command file.
- Rename SYS:TCPIP.NEW to SYS:TCPIP.ERZ. If you are using an account other than DSK0:[7,50] for TCP:, modify TCPIP.ERZ to reflect the change. Change the ersatz to **TCP:=-dev:[ppn]** for the TCP: master account.
- Log to SYS:. Create and edit a test system initialization file. Use TCPIP.INI for simplicity.
  - Increase the job count on the JOBS line by 10 for initial testing. Calculate the actual number of additional jobs by using the attached worksheet and determining *tcpjob*.
  - Add the following after the last JOBALC line.  
**JOBALC NETLOG, TCPEMU**
  - Add the following after the last TRMDEF line.  
**TRMDEF NETLOG, PSEUDO, NULL, 100, 100, 100**  
**TRMDEF TCPEMU, PSEUDO, NULL, 100, 100, 100**
  - For SLIP connections, add a TRMDEF statement for each SLIP port. Use the following syntax for each of the defined ports:  
**TRMDEF trmnam(x), idv(x)=port(x):baud(x), NULL, 100, 100, 100**
  - Add the following after the last ERSATZ line.  
**ERSATZ TCPIP.ERZ**
  - If the TCP: account is *not* located on DSK0:, add the MOUNT statement for the TCP: device prior to the first SYSTEM command.
  - Add the following before the last SYSTEM command:  
**SYSTEM TCP:IPCINI/N 100 100K**  
**SYSTEM RTLLIT**
  - Immediately following the *last* SYSTEM command, add the following line:  
**SMEM smemK**

*(Note: If SMEM already exists, simply increase the available memory by **smem**).*

- Following the NETSER job initialization, add the following lines:  
**WAIT NETSER**  
**SETJOB NETLOG, NETLOG, 40K**  
**SETJOB TCPEMU, TCPEMU, *tcpmem*K, TCP:GOTCP.CMD**

*(Note: Only add the WAIT statement if AlphaNET is installed).*

- Reboot the computer using TCPIP.INI. and verify TCPEMU is running in a software interrupt state (RN SI) using STAT.
- Use **PING** to access another system on the network.
- If the network seems to be responding, set up additional servers by referencing the individual worksheets.

# Setting Up AlphaTCP Postoffice Server

## ALLOWED MAIL USERS:

---

Login Name ( <i>lognam</i> ) <sup>1</sup>	Login Password ( <i>logpsw</i> ) <sup>1</sup>	User Name ( <i>usrnam</i> ) <sup>2</sup>
<i>J_Smith</i>	<i>js_paswd</i>	<i>Joe Smith</i>

<sup>1</sup>Login names and passwords are case sensitive, cannot contain spaces, and should be unique to the postoffice. If they are the same as the Login for FTP, gaining one will get the other. Remember once connected to the net there is world wide access to this system.

<sup>2</sup>The user name may contain spaces and *must* match the name used by the user when directly accessing the local mail package.

## TCP SETUP

---

Local mail package driver and options<sup>1</sup> (*maildvr*):   *AMAIL*

POP3D file account (**TCPPPOP:**)(*dev:[ppn]*):       : [        ,        ]

Use debug mode (*dbgmod*):   *No*

Strip host name from domain name<sup>2</sup> (*strip*):   *No*

Maximum connections (*maxcon*):   *20*

Default time out limit (*timeout*):   *4*

<sup>1</sup>Refer to your mail package for the correct mail driver and any option switches that may apply.

<sup>2</sup>Setting this option to yes is useful if your site is running POP3 servers on multiple hosts, but has a primary host handling all SMTP mail for your domain.

## POSTOFFICE SERVER (POP3D) CHECK LIST

---

- Create the account *dev:[ppn]* for the temporary files.
- Add the ersatz TCPPOP:=*dev:[ppn]* to SYS:TCPIP.ERZ for the temporary file account.
- Copy your main system initialization file to a temporary .INI file (TCPIP.INI)
  - Increase the number of **JOBS** by 1 plus *maxcon*.
  - Add the **LODEMD** command before the first **SYSTEM** command using the following syntax:  
**LODEMD maildvr**
  - Load the following into system memory; TCP:POP3D.LIT, TCP:POP3D.RTI and SYS:CMDLIN.SYS. Use the following syntax:  
**SYSTEM TCP:POP3D.LIT**  
**SYSTEM TCP:POP3D.RTI**  
**SYSTEM SYS:CMDLIN.SYS**
  - Increase SMEM memory by  $(maxcon + 1) * 66K$ .  
*See memory requirements worksheet.*
- Edit the TCP:CONFIG. file.
  - Remove the # from the **#start pop3d...** line.
  - If *dbgmod* is yes, add **-d** to the end of the **start pop3d...** line.
  - If *strip* is yes, add **-h** to the end of the **start pop3d...** line.
  - If *maxcon* is not 20, add **-smaxcon** to the end of the **start pop3d...** line.
  - If *timeout* is not 4, add **-timeout** to the end of the **start pop3d...** line.
- Create or edit TCP:POPUSR. Add the valid postoffice users using the following syntax :  
*lognam(1) logpsw(1) usrnam(1)*  
*lognam(2) logpsw(2) usrnam(2)*  
*lognam(x) logpsw(x) usrnam(x)*

# Setting Up AlphaTCP Telnet Sessions

## TELNET CONFIGURATION:

---

Number of simultaneous sessions (*sesno*): \_\_\_\_\_

Application memory requirements (*apmem*): \_\_\_\_\_K

Terminal Drivers needed  
(enter \* for all) (*tdvs*): \_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_

Logon command (*logcmd*): \_\_\_\_\_ (optional — defaults to LOGON.LIT)

TCP port number (*tcpprt*): \_\_\_\_\_ (optional — defaults to 23, use 0 as placeholder)

Job memory  
(optional) (*jobmem*): \_\_\_\_\_ k or \_\_\_\_\_ k? or \_\_\_\_\_—\_\_\_\_\_ k  
fixed memory default memory memory range

Fixed terminal driver (*tdv*): \_\_\_\_\_ (optional — prompts as default, use \* as placeholder)

Description (*desc*): \_\_\_\_\_  
(Optional, but mandatory if using *stcmd* below)

Application startup command (i.e. RUN PARTS)  
(overrides *logcmd*) (*stcmd*): \_\_\_\_\_ (Optional)

## TELNET CHECK LIST

---

- Copy your main system initialization file to a temporary .INI file (TCPIP.INI).
- Increase the number of **JOBS** by 1, plus the number of possible sessions (*sesno*).
- Using TDVDEF add the possible terminal drivers to the system, use the following syntax:  
**TDVDEF \***  
(To load all the \*.TDV's in DVR)  
or  
**TDVDEF *tdvs, tdvs, tdvs...***
- Increase SMEM memory by (*sesno* \* (*apmem* + 6)+106)KB.
- Edit the TCP:CONFIG. file and do the following:
  - Remove the # from the #start telned... line.
  - If one or more of *logcmd*, *tcpprt*, *jobmem*, *tdv*, *desc* or *stcmd* are not default values, use the following syntax for the start telned line. If using all default values, ignore the following:  
**start telned 100k {"-logcmd"} tcpprt jobmem tdv "desc" "stcmd"**

# Setting Up AlphaTCP File Transfer

## FTP ALLOWED USERS

User Name ( <i>usrnam</i> ) <sup>1</sup>	User Password <sup>1</sup> ( <i>usrpsw</i> )	Initial Login ( <i>usrlog</i> )	R <sup>2</sup>	W <sup>3</sup>	L <sup>4</sup>	N <sup>5</sup>	Privileges <sup>6</sup> ( <i>usrprv</i> )
<i>JoeSmith</i>	<i>1blcYcLe1</i>	<i>DSK0:[100,0]</i>	1	2	4		7
			1	2	4	8	
			1	2	4	8	
			1	2	4	8	
			1	2	4	8	
			1	2	4	8	
			1	2	4	8	
			1	2	4	8	
			1	2	4	8	
			1	2	4	8	
			1	2	4	8	
			1	2	4	8	
			1	2	4	8	
			1	2	4	8	
			1	2	4	8	
			1	2	4	8	
			1	2	4	8	

Override AMOS file protection flags<sup>7</sup> (*ovrprt*)      Yes or No

<sup>1</sup>User names and passwords are case sensitive and cannot contain spaces. Passwords *must* be unique and difficult to guess. Remember, once connected to the net, there is world wide access to this system.

<sup>2</sup>User can **R**ead files (enter 1 if this is desired).

<sup>3</sup>User can **W**rite and delete files (enter 2 if this is desired).

<sup>4</sup>User can **L**og to other accounts (enter 4 if this is desired).

<sup>5</sup>User can access remote AlphaNET files (enter 8 if this is desired).

<sup>6</sup>Add up the values in columns R + W + L + N for privilege value.

<sup>7</sup>Yes connections are treated as local logins to the host, thus allowing traditional files to be deleted or overwritten. No connections are treated as "network connections." This prevents files on traditional directories from being deleted or overwritten.

## FTP CHECK LIST

---

- Copy your main system initialization file to a temporary .INI file (TCPIP.INI).
  - Load the following files into system memory:  
**SYSTEM TCP:FTPD.LIT**  
**SYSTEM TCP:FTPD.RTI**  
**SYSTEM SYS:CMDLIN.SYS**
- Edit the TCP:CONFIG. file and do the following:
  - Remove the # from the **#start ftpd...** line.
  - If *ovrprt* = YES then add **-o** to the end of the **#start ftpd...** line.
  - Create or edit TCP:FTPUSR. and add the valid FTP users. Use the following syntax :  
*usrnam(1) usrpsw(1) usrlog(1) usrprv(1)*  
*usrnam(2) usrpsw(2) usrlog(2) usrprv(2)*  
*usrnam(x) usrpsw(x) usrlog(x) usrprv(x)*



# Setting Up AlphaTCP Unsecure File Transfer

## TFTP CHECK LIST

---

- Read the section on security before setting up TFPD.
  
- Copy your main system initialization file to a temporary .INI file (*TCPIP.INI*).
  - Load the following files into system memory:  
**SYSTEM TCP:TFTPD.LIT**  
**SYSTEM TCP:TFTPD.RTI**  
**SYSTEM SYS:CMDLIN.SYS**
  
- Edit the TCP:CONFIG. file and do the following:
  - Remove the # from the **#start tftpd...** line.

# Setting Up AlphaTCP For Use With Printer Servers

## AMOS NETWORK PRINTERS:

---

Item	Printer (1)	Printer (2)	Printer (3)
Printer IP Address ( <i>prt<sub>ip</sub></i> )	. . .	. . .	. . .
Host Name of Printer (host.company.grp) ( <i>pr<sub>thnm</sub></i> )	. . .	. . .	. . .
Printer Alias (host) ( <i>pr<sub>tali</sub></i> )	<i>prt<sub>tcp01</sub></i>	<i>prt<sub>tcp02</sub></i>	<i>prt<sub>tcp03</sub></i>
TLP Device Name ( <i>name after DEVICE=TLP: (UCS) (pr<sub>tdev</sub>)</i> )	<i>PTCP01</i>	<i>PTCP02</i>	<i>PTCP03</i>
Printer Queue <sup>1</sup> ( <i>pr<sub>tque</sub></i> )			
Printer Initialization File ( <i>pr<sub>tini</sub></i> )	<i>ptcp01.pin</i>	<i>ptcp02.pin</i>	<i>ptcp03.pin</i>
AMOS Printer Name ( <i>pr<sub>tnam</sub></i> )	<i>ptcp01</i>	<i>ptcp02</i>	<i>ptcp03</i>
Is this printer a default ( <i>pr<sub>tdflt</sub></i> )	YES or NO	YES or NO	YES or NO
AMOS Job to Receive Printer Messages ( <i>msg<sub>job</sub></i> )			
Spooler Job Name <sup>2</sup> ( <i>spl<sub>job</sub></i> )	<i>ptcp01</i>	<i>ptcp02</i>	<i>ptcp03</i>
Spooler TRMDEF Name <sup>2</sup> ( <i>spl<sub>trm</sub></i> )	<i>ptcp01</i>	<i>ptcp02</i>	<i>ptcp03</i>

<sup>1</sup>Printer queue name (for UNIX, etc. type hosts) or **PASSTHRU** (for most print servers).

<sup>2</sup>For memory based spooler.

(Add *pr<sub>t<sub>ip</sub></sub>*, *pr<sub>thnm</sub>*, *pr<sub>tali</sub>* to **TCP:HOSTS**. or the Domain Name Server (*dns*)).  
(Use additional pages for more TCP printers).

## TCP SETUP

---

Temporary print file account (**TCPLPR**):

(*dev:[ppn]*) \_\_\_\_\_:[ \_\_\_\_\_ , \_\_\_\_\_ ]

## LINE PRINTER CHECK LIST

---

- Create the account *dev:[ppn]* for the temporary print files.
- Add the ersatz TCPLPR:=*dev:[ppn]* to SYS:TCPIP.ERZ for the temporary print file account.
- Create and edit each printer initialization file *prtini(x)*. Use the following syntax:
  - DEVICE=**TLP:*prtdev(x)*
  - NAME=***prtnam(x)*
  - DEFAULT=***prtdflt(x)*
  - OPERATOR=***msgjob(x)*
  - FORMFEED=**TRUE or FALSE
  - FORMS=**NORMAL
  - BANNER=**TRUE or FALSE
  - HEADER=**TRUE or FALSE
  - LPP=**60
  - WIDTH=**80
  - CLUSTER<sup>1</sup>=**TRUE or FALSE
  - INFORM=**<sup>1</sup>TRUE or FALSE
  - RESTART=**<sup>1</sup>TRUE or FALSE
- Copy your main system initialization file to a temporary .INI file (TCPIP.INI).
  - Increase the number of JOBS by 1 plus (the number of TCP printers times 2).
  - Define the TLP device in a DEVTBL statement using the following syntax:
    - DEVTBL TLP0**
  - Load the following into system memory; SYS:LPR.LIT, SYS:LPR.RTI and DVR:TLP.DVR. Use the following syntax:
    - SYSTEM SYS:LPR.LIT**
    - SYSTEM SYS:LPR.RTI**
    - SYSTEM DVR:TLP.DVR**
  - Increase SMEM memory by (# spoolers + 1) \* 41K.  
*See memory requirements worksheet.*

### If Memory Based Spooler

- Add the spooler job names to the JOBALC line:
  - JOBALC** *spljob(1), spljob(2)...spljob(x)*
- Add the spooler TRMDEFs:
  - TRMDEF** *splrm(1), PSEUDO, NULL, 50, 50, 50*
  - TRMDEF** *splrm(2), PSEUDO, NULL, 50, 50, 50*
  - TRMDEF** *splrm(x), PSEUDO, NULL, 50, 50, 50*
- Start the spooler jobs by using the SETJOB statement as follows:
  - SETJOB** *spljob(1), splrm(1), 8K, LPTINI prtini(1)*
  - SETJOB** *spljob(2), splrm(2), 8K, LPTINI prtini(2)*
  - SETJOB** *spljob(x), splrm(x), 8K, LPTINI prtini(x)*

<sup>1</sup>For task manager based spooler

Increase JOBS by one for task manager.

Add TASK job to JOBALC line:

**JOBALC TASK**

Add MANAGR to TRMDEF line:

**TRMDEF MANAGR, PSEUDO, NULL, 100, 100, 100**

Within the SETJOB area add the following line:

**SETJOB TASK, MANAGR, 400K, TASK.JIN**

Create and edit TASK.JIN and enter the following lines:

**TSKINI**

**S prtini(1)**

**S prtini(2)**

**S prtini(x)**

**G**

From SYS: execute the MAKQUE file and create a (S)pooler queue file with approximately 100 or so records.

## For All Printer Types

Edit the TCP:CONFIG. file. Remove the # from the #start lpr... line.

Rename TCP:LPR.NEW to TCP:LPR.

Map AMOS spoolers to network printers in the TCP:LPR. file. Use the following syntax for each defined printer:

***prtdev(1)=prtali(1), prtque(1)***

***prtdev(2)=prtali(2), prtque(2)***

***prtdev(x)=prtali(x), prtque(x)***

Add network printer specifications to the TCP:HOSTS. file or to the Domain Name Server (dns) as needed. Use the following syntax for the TCP:HOSTS. file:

***prtip(1) prthnm(1) prtali(1)***

***prtip(2) prthnm(2) prtali(2)***

***prtip(x) prthnm(x) prtali(x)***

# Setting Up the AlphaTCP Web Server

## WEB CONFIGURATIONS

---

Web Support Accounts <sup>1</sup> ( <i>ppn</i> )	Associated Ersatz ( <i>erz</i> )

Maximum simultaneous connections (*maxuse*): \_\_\_\_\_

<sup>1</sup>Define all the accounts that will contain accessible Web related files (HTML, graphics, forms...). Do **not** include TCP: in this list.

## WEB SERVER CHECK LIST

---

- Log to SYS:, create WEB.ERZ, and add all of your accounts and associated ersatz to this file. Use the following syntax:

*erz(1)=ppn(1)*

*erz(2)=ppn(2)*

*erz(x)=ppn(x)*

- In each account above (*ppn(x)*), create the file PUBLIC.PPN. This file can be empty so a program like MAKE will be sufficient. Do *not* create this file in TCP: or in any account that contains sensitive information.

- Create and edit a test system initialization file. Use TCPIP.INI for simplicity.

- Increase the number of JOBS by 1 + *maxuse*.

- After the last ERSATZ line add the following new ersatz statement:

**ERSATZ WEB.ERZ**

- Load TCP:HTTPD.LIT and TCP:HTTPD.RTI into system memory, using the following syntax:

**SYSTEM TCP:HTTPD.LIT**  
**SYSTEM TCP:HTTPD.RTI**

- Immediately following the *last* SYSTEM command, add the following line.

**SMEM (existing + (106K\*maxuse))**

*(Note: If SMEM already exists, simply increase the available memory by 106K \* 1\* maxuse. See memory requirements worksheet.)*

- Log to LIB: and rename LIB:MIME.NEW to LIB:MIME.TYP.

- Log to TCP: and verify that TCP:TIMZON. is properly set up.

- Edit the TCP:CONFIG. file. Remove the # from the **#start httpd 100k** line or add this line if it does not exist.

- Rename TCP:HTTPD.NEW to TCP:HTTPD.HTM.

*(Note: This is a sample file and must be customized to your operation once the server is operational).*

# Setting Up AlphaTCP For Use With AMOS Printers

## LPD SETUP

---

Max simultaneous print requests<sup>1</sup> (*maxrqs*) \_\_\_\_\_

Form feed options (*ff*)     Default (*def*)  
                                   Force Form feed (*ff*)  
                                   Force No Form feed (*noff*)

Temporary print file account (**TCPLPD:**). Be sure to select a drive with plenty of disk space:

(*dev:[ppn]*)    \_\_\_\_\_:[ \_\_\_\_\_ , \_\_\_\_\_ ]

<sup>1</sup> Enter the total number of incoming print requests you might expect at any single moment.

## LINE PRINTER CHECK LIST

---

- Create the account *dev:[ppn]* for the temporary print files.
- Add the ersatz **TCPLPD:=*dev:[ppn]*** to SYS:TCPIP.ERZ for the temporary print file account.
- Copy your main system initialization file to a temporary .INI file. Use TCPIP.INI.
  - Increase the number of JOBS by 1 plus *maxrqs*.
  - Load TCP:LPD.LIT and TCP:LPD.RTI into system memory. Use the following syntax:  
**SYSTEM TCP:LPD.LIT**  
**SYSTEM TCP:LPD.RTI**
  - Increase SMEM memory by (*maxrqs* + 1) \* 41K.  
*See memory requirements worksheet.*
- Edit the TCP:CONFIG file. Remove the # from the **#start lpd...** line. If *ff* is *ff* then add **-fY** to the end of this line, if *noff* then add **-fN** to the end.
- Any remote system requiring access to these printers setup, use the following information:
  - LPR Printer Format:    **Unfiltered**
  - LPR Printer Host:    **Address or name of AMOS system**
  - LPR Printer Queue:    **Spooler name on AMOS system**

# Setting Up ITC Tunneling Through AlphaTCP

## AMOS ITC IDs AND ASSOCIATED IP ADDRESSES

---

Ersatz( <i>erz</i> )	Network	Group	Node	AlphaNet CPU ID ( <i>netid</i> ) <sup>1</sup>	IP Address ( <i>netip</i> ) <sup>2</sup>

<sup>1</sup>Use the AMOS program **CVTNID**. Enter **Network**, **Group** and **Node** to calculate the **AlphaNet CPU ID** for each system.

<sup>2</sup>Enter the IP address for each system (now, each is a single system).

## ITC TUNNELING CHECK LIST

---

- If you have an *existing* AlphaNET network defined, insure that the network IDs (Network column) are **not** the same as any existing AlphaNET network numbers.
  
- Log to SYS: and create ITCD.ERZ. Add all of your system references to this file. Once complete, copy this same file to every system that will be using ITC tunneling. Use the following syntax:
  - erz(1)=netid(1)*
  - erz(2)=netid(2)*
  - erz(x)=netid(x)*



- Create and edit a test system initialization file. Use TCPIP.INI for simplicity.
  - Increase the number of JOBS by 1.
  - After the last ERSATZ line, add the following new ersatz statement:  
**ERSATZ ITCD.ERZ**
  - Immediately following the *last* SYSTEM command, add the following line:  
**SMEM (existing + 106K)**  
*(Note: If SMEM already exists, simply increase the available memory by 106K. See memory requirements worksheet).*
- Log to TCP: and create the file ITCD.
- Add or create the ITC to IP mapping in the file TCP:ITCD. Once complete, copy this same file to every system that will be using ITC tunneling. Use the following syntax for the TCP:ITCD. file:  
***netid(1) netip(1)***  
***netid(2) netip(2)***  
***netid(x) netip(x)***
- Edit the TCP:CONFIG. file. Remove the # from the **#start itcd 100k** line or add this line if it does not exist..

# Appendix C - Mime Types File

The MIME types file, LIB:MIME.TYP, maps AMOS file extensions to MIME content keywords. MIME stands for “Multipurpose Internet Mail Extensions” and provides a method of describing the contents of an e-mail or other message. HTTPD uses this file, as will SMTPD in the future. The file contains multiple lines defining a content-type and code, an optional content-encoding keyword within angle brackets, and file extensions to which the MIME definition applies:

```
content-type/code <encoding> ext ext ...
```

For example, the MIME type line for web pages (\*.HTM) is as follows:

```
text/http <7bit> HTM
```

The currently defined MIME types may be found in Request-For-Comments (RFC) #1521, available via anonymous FTP from ds.internic.net.

Most files you want to have treated as non-specific binary data would be added to the extensions list for the *application/octet-stream* type.

To specify file extensions to treat as pure text, add the extensions to the *text/plain* type. If this type doesn't exist yet, you may add it. For example:

```
text/plain <7-bit> TXT
```



# Appendix D - BBSERV BBS Interface

The BBSERV server by itself allows e-mail based file retrieval. By using the interface provided by BBSERV, you may develop an e-mail interface to a BBS or database running on the system. This section describes the interface provided by BBSERV.

The BBS interface programs are executed within the memory of the BBSERV job. The BBSERV job should have 60K MORE memory than the interface programs may require for themselves.

## TOPICS FILE

Topics within the BBS must be listed in a topics file. The topics file must be referenced by the ersatz name BBSTOP:.

The topics are listed one topic per line. Topic names must not contain any spaces. If you want to use multiple words for topics you must connect the words with some character such as a '.' or '-'.

You may optionally place a line at the end of the topics file consisting of the statement: .END .

## TOPIC RETRIEVAL

When BBSERV recognizes requests for BBS topics in a mail message, it invokes the command BBSGET to retrieve the information. The BBSGET function must be provided by the BBS.

Two filenames are passed on the BBSGET command line. The first is an input file and the second is an output file.

The input file should be read for the starting date and the requested topics. The first line is the desired starting date in MM/DD/YY format. The rest of the file contains the topics of interest, one per line.

The output file should contain any requested BBS postings in readable form. This file will be mailed back to the person requesting the information. If no postings are available, you may leave this file alone and BBSERV will return an appropriate message.

## TOPIC POSTING

When BBSERV recognizes a posting request for one or more BBS topics in a mail message, it invokes the command BBSPUT to post the information. The BBSPUT function must be provided by the BBS.

An input filename is passed on the BBSPUT command line. This file contains information about the posting, including the message itself.

- The first line begins with the keyword “. CATEGORY” and one or more BBS topics to which the message be posted. If more than one is listed they will be separated with commas.
- The second line begins with the keyword “TITLE” and contains the subject line of the original mail message. This may be used as the subject of the posting.
- The third line begins with the keyword “. USERNAME” and contains the e-mail address of sender. This may be a local or internet style address depending on the mail messages source.
- The fourth line is blank, followed by the text of the message to be posted. The rest of the file should be read in as lines of the message.

## TOPIC SUBSCRIPTIONS

BBSERV recognizes requests to subscribe to or unsubscribe from the various BBS topics. These are appended to the file BBSERV:BBSERV.SUB for processing by an external program at some time in the future. No other processing is done by BBSERV. Subscriptions must be enabled using the -s switch. Each line in the file consists of the following three fields separated by a space:

1. The character ‘S’ or ‘U’. This stands for ‘subscribe’ and unsubscribe.
2. A single topic name to be acted upon.
3. The mail address of the requester. This may be a local style address (as usually encountered in the local mail package) or an internet style address (`user@host`). It should be noted that local-style addresses may contain spaces, such as when first and last names are in use. Once reaching the third field, be sure to process the remainder of the line as a single field even if spaces are encountered between words.

It is expected that the program responsible for performing the batch BBS mailings will process this file periodically, possibly once per day. This program is not supplied and must be written by you. This program is known as the ‘list server.’

As an example, the following sequence of events should probably be performed by your program when it begins a mailing session:

1. Rename the subscriptions file and process the days subscriptions and/or unsubscriptions. If the file does not exist, there were none received.
2. Read the list server’s incoming mailbox for bounced mail (not the mailbox used by BBSERV—they should be different). Some suggestions for determining the message that refers to a bounce include:
  - The source address includes the word DAEMON somewhere on the left side of the ‘@’. (example, `MAILER-DAEMON@mail.widgit.com`).
  - The subject includes one of the following words: FAILED, RETURNED, BOUNCE, UNABLE, CANNOT, UNKNOWN. The comparison should not be case-sensitive.



Don’t drop subscribers right away, just bump a counter and update a ‘last bounce’ date. Drop them after at least a week’s worth of bounces. This will handle transient mail failures. Finding the original send address in the message may prove a bit tricky and will require an analysis of various bounce message formats to determine the best method.

3. Clear the bounce counter for all subscribers without a bounce in over a week. This keeps the counter from adding up over an extended period and causing a subscription to be improperly dropped.
4. Process each user left in the database, creating a file of all messages since the last mailing in all topics they are subscribed to, then mailing it to them.

If you don't already have a user interface for your mail package, you may need to access an electronic mail driver directly. For more information on this, refer to the document *Electronic Mail Driver Programmer's Guide*, DSO-00206-00.



# Appendix E - Application Termination

In an ideal world, all TCP-generated virtual terminal sessions would return to AMOS command level before the connection was closed and the job was despawned. However, due to lost connections, user error, etc., there will be times when AlphaTCP needs to close a session and despawn a job which is currently running an application. When writing applications which may run under AlphaTCP control, even if they are not specifically “network” applications, you need to keep this in mind, and design your programs so AlphaTCP, or any other controlling process, can properly shut down your application and reclaim the resources it was using before it despawns a spawned job.

The following sections discuss why this is necessary, how AlphaTCP attempts to terminate applications, and how you can make sure this process works smoothly with your application.

## TERMINATING APPLICATIONS ON VIRTUAL TERMINALS

Traditionally, jobs and terminals were allocated at boot time and existed until the next reboot. An application could assume the only reason to exit was because either the program or user wanted to. If the application didn't want the user to exit, it simply trapped the `CTRL/C` key. In fact, `CTRL/C` is used in various applications to terminate an activity while allowing the program to stay in control.

With the proliferation of network-based and task-swapping virtual sessions, this is no longer true. Jobs and terminals are created on the fly, and task server jobs are shared between different users. Applications must allow for a third source of exit requests: the controlling job. Under AlphaTCP, TCPEMU can only close a session and reclaim its resources if the job is at AMOS monitor level.

The next section describes how AlphaTCP requests application exit; after that, we discuss how to process this request properly in your assembler or AlphaBASIC application. This information should be useful even if you use another language for your application.

## Exit Request Generation



Since this process uses the software interrupt feature, this information applies only to AMOS 2.X. Software interrupts are not available in AMOS 1.X.

There are two software interrupt features to terminate a job:

- `SI$EXI` (SI-EXIT). This is a standard, maskable software interrupt, which allows the job to clean up after itself.
- `SI$ABT` (SI-ABORT). This is a nonmaskable termination which aborts the job upon its next monitor call. AlphaTCP uses this as a last resort.



When TCPEMU wants to free one of its controlled jobs, it sets an SI-EXIT request and sends a ^C to the job to wake it up. If necessary, it repeats this several seconds later, followed by an SI-ABORT and ^C after a few more seconds.



This method (SI-EXIT and ^C) is also used by other AMOS packages which use controlling jobs, such as AlphaNET, MULTI, and FLiP. However, the timing and implementation of further steps after the first exit request differs between packages.

In general, your application must close in 20 seconds or less after the first ^C to avoid being terminated by SI-ABORT. In addition, it should properly handle receiving additional SI-EXIT/^C requests if the termination process takes more than about 10 seconds. If your application generally exits quickly upon receiving a ^C, as do most AMOS .LIT files, you will probably not have any problems.



In the case of an inactive or dropped connection, it may be up to 30 minutes, or longer, before this shutdown process begins. See the section “To Change the TCP Keepalive Time-out” in Chapter 3 and the discussion of keepalives in the TELNED section in Chapter 5.

## Handling Termination Requests in Assembler



If you are unfamiliar with software interrupts, you should review the section on them in the *AMOS Monitor Calls Manual*. Software interrupts are actually not that difficult to use.

Place an index to your handler into SI.EXI of the software interrupt vector table and register the table. It is registered and enabled using the SIMSK monitor call and the SI\$EXI bit.

Your handler may do whatever you choose, from terminating the application to simply setting a flag and performing an SIRTN.

## Handling Termination Requests in AlphaBASIC

In a TCP-based application, you can enable SI-EXIT processing using a standard activity flag. See the chapter on the TCP XCALL in the *AlphaBASIC XCALL Subroutine User's Manual*, DSO-00066-00, revision 03 or later.



AlphaBASIC does not handle SI-EXIT requests itself. Newer implementations of AlphaBASE/Metropolis and ESP might support it; however, standard AlphaBASIC applications do not.

If you are not developing a TCP-based application, or want to retrofit an existing application, you can write your own XCALL to handle SI-EXIT processing. There is an unsupported XCALL, SIEXIT, you can use as an example of one way to do this. Both the XCALL and its source code are available in the file SIEXIT.CMP on our FTP site at <ftp://ftp.alphamicro.com/anon/>

# Document History

REVISION	RELEASE	DATE	DESCRIPTION
00	1.0	4/93	New document for AlphaTCP 1.0; adapted from Spider TCP documentation.
01	1.1	1/94	Added STARTD, PS, and UKILL commands. Added SLIP ONLINE command. Completely rewrote memory requirements. Added subnetwork information.
02	1.2	2/95	Added information on Mail server (SMTPD), Domain Name Service (DNS), Line Printer Client server (LPR), Post Office server (POP3D), new NSLOOK command, and expanded installation instructions.
03	1.3	7/95	Added information on AlphaNET ITC tunneling server (ITCD), World Wide Web server (HTTPD), Line Printer server (LPD), and e-mail based file server (BBSERV). and expanded installation instructions.
04	1.3A	12/95	Added information on supporting Eagle 550. Added new chapter about image maps and fill-out forms.
05	1.3B	8/96	Added information on supporting single PIC, POPPWD, and new server switches.
06	1.3B	11/96	Corrected Page 5-4. Changed SLIPD to SLIPOK in paragraph 2 and 3.
07	1.3C	5/97	Chapter 5: added information on TELNED -j and -w options, VTID command and GETVTI.SBR, e-mail blocking, restarting queued print files, and retrieving printer status. Chapter 6: added PS -p option. Chapter 9: added hit counting.
08	1.4	8/97	Added Appendix E. Ch 1: Added subnetting discussion. Ch 3: Updated CONFIG.NEW, added sections for TAMED and SNTPD. Ch 5: Added SNTPD, TAMED, FTPD -p option and HTTPD -m option. Ch 6: Added WHOIS.
09	1.4A	3/99	Added BOOTP client. Added POST method support to URL command and xcall. Enhanced FTPD server. Enhanced Web server to support PUT and DELETE methods. Improved SMTP server. Added LPR -t switch and TELNED -n switch. Added QPOP3D server.
10	1.5	9/99	Added BOOTPC client, which supports DHCP. Added TDDD and IPFW, as well as AMPM (AM packet manager)
11	1.5A	4/00	Ch 5: Added tree view capability for FTPD. Ch 6: Added TROUTE to display router information
12	1.5A	9/02	Ch 5: Expanded information on limiting hosts and users that can use SMTPD to forward mail



# Index

---

## A

absolute reference · 9-1  
AlphaNET ITC tunneling server · 3-8  
AlphaNET tunneling · 3-20  
AMOS printer · 5-23  
AMOS printers · 3-17  
AMPM command · 6-19  
Application Definition File · 9-6  
Application termination · E-1  
ARP command · 6-7  
ARP protocol · 1-6  
AUTHORIZATION · 9-10

---

## B

BBSERV · 5-32  
    security · 5-33  
bootp / dhcp client · 3-8  
BOOTP protocol · 1-6  
BOOTP server · 5-38  
BOOTPC server · 5-39  
BOOTPD file · 3-3  
bootpd server · 3-8

---

## C

Cabling for SLIP · 7-4  
CGI-BIN scripts · 9-5  
Changing the TCP close time-out · 3-7  
Checking local job status · 6-8  
Closing sessions · E-1  
Comment lines in setup files · 3-6  
CONFIG. file · 3-3, 3-6  
    and system performance · 8-2  
Configuration file · See CONFIG. file  
CONTINUOUS command for SLIP · 7-2  
Converting CPU IDs · 6-16  
COOKIES · 9-9  
CVTND command · 6-16

---

## D

Daemons, defined · 2-2  
DELAY command for SLIP · 7-2  
DHCP protocol · 1-6  
DISCARD command for SLIP · 7-2

Disk space required · 2-2  
Displaying network statistics · 6-3  
displaying routers · 6-22  
Displaying the routing table · 6-3, 6-6  
DNS · See Domain Name Service (DNS)  
Domain Name Service (DNS)  
    RESOLV. file · 3-12  
Domain name service protocol · 1-6  
Downloading the software · 2-2  
dynamically configuring using BOOTP or DHCP · 3-7  
Dynamic-IP printers · 5-22

---

## E

Electronic mail · 3-18  
Electronic Mail server · 3-8  
E-mail File Server · 5-32  
Entering the PIC · 2-3  
Error logging · See NETLOG  
Error messages · A-1  
Ersatz names · 3-5  
ERSATZ statement · 4-2  
Ethernet  
    address  
        in NETWRK. file · 3-15  
        mapping to internet address · 6-7  
    second segment support · 3-15  
    setting up in CONFIG. file · 3-14

---

## F

Fill-Out Forms · 9-5  
Firewalls · 5-42  
FNU2A command · 6-17  
FTP · 1-6  
    security · 5-11  
    server (FTPD) · 5-8  
    spawning FTPD server · 3-8  
    tree view · 5-12  
    user names and passwords · 5-9  
    user privilege level · 5-10  
FTP. file · 5-9  
FTPDEV.RPC file · 5-9  
FTPLOG command · 6-17  
FTPUSR. file · 3-4, 5-9

---

## G

GET Method Arguments · 9-8

GOTCP.CMD file · 3-5, 3-10, 4-5

---

## **H**

Handshaking for SLIP · 7-4

Hit counters · 9-4

Host names

  defining in HOSTS. file · 3-11

  localhost · 4-5

HOSTS. file · 3-4, 3-11

HTTP protocol · 1-6

HTTPD · 3-20

  security · 5-28

HTTPD.APP file · 9-4

HTTPD.CVT file · 3-4

HTTPD.DEV file · 3-4

HTTPD.HTM file · 3-4

HTTPD: · 5-24

---

## **I**

ICMP protocol · 1-6

IDLE command for SLIP · 7-2

Image Maps · 9-3

Installation

  AlphaNET number for Ethernet · 3-14

  checking setup · 4-5

  CONFIG. file · 3-6

  configuring SLIP · 3-9, 3-16

  downloading the software · 2-2

  gateways · 3-7

  memory requirements · 3-1, 3-9

  modem port name · 3-16

  not on DSK0: · 2-2, 2-3

  prerequisites · 2-1

  setup files · 3-3

  spawning servers · 3-6, 3-8

  system initialization file · 4-1

    ERSATZ statement · 4-2

    JOBALC statement · 4-1

    JOBS statement · 4-1

    LDV setup · 4-4

    MONTST · 4-5

    NETFAM statement · 4-3

    SETJOB statement · 4-4

    SMEM statement · 4-4

    SYSTEM statement · 4-3

    testing · 4-5

    TRMDEF statement · 4-2

  tcp close time-out · 3-7

  terminal drivers · 4-2

  testing the servers · 4-5

  verifying the software · 2-3

Internet · 1-3

Internet address

  and host names · 3-11

  defined · 1-3

  for this computer · 3-13

  format · 1-3

  in HOSTS. file · 3-11

  in NETWRK. file for Ethernet · 3-15

  in NETWRK. file for SLIP · 3-16

  mapping to Ethernet address · 6-7

Internet network classes · 1-4

Internet Network Information Center (INIC) · 6-18

IP protocol · 1-6

IPCFG command · 6-18

IPCINI command · 8-1

IPFW firewall · 5-42

ITC tunneling

  and SerialNet · 5-30

ITC Tunneling Server · 5-28

ITCD · 3-20, 5-29

  security · 5-31

ITCD. file · 3-4

---

## **J**

Job status, checking · 6-8

JOBALC statement · 4-1

JOBS statement · 4-1

---

## **K**

Keepalives

  disabling · 5-6

  setting time-out · 3-7

KLPR command · 6-18

---

## **L**

LDV · 4-4

Line printer · 3-8, 3-14

  setting up · 3-17

Line printer definition · 3-4

Line printer server · 3-8, 5-20

Line Printer Server · 5-23

Local host name file · See MYNAME. file

localhost, as host name · 4-5

Log file for SLIP · 7-3

LOGCTL · 4-5, 5-2, 6-2

LOGFMT · 6-3

LPD

  Security · 5-24

  spawning LPD server · 3-8

LPD. file · 3-17

LPR

  security · 5-23

  spawning LPR server · 3-8

LPR protocol · 1-6

LPR. file · 3-4, 3-14, 3-17

---

## M

Mail · 3-18  
Mail server · 5-13  
Memory requirements · 3-1  
    in CONFIG. file · 3-9  
    NETLOG · 3-3  
    overall · 2-1  
    RWHO · 5-35  
    servers · 3-2  
    shared memory area · 3-2  
    TCPEMU · 3-1, 8-1  
    user memory · 3-3  
MIME Headers · 9-8  
MIME.TYP file · 3-4  
mlblks · 8-2  
Modem  
    cabling for SLIP · 7-4  
    port name for SLIP · 3-16  
    using AlphaTCP with · 4-2  
    using with SLIP · 7-1  
MONTST · 4-5  
    during installation · 4-5  
MYNAME. file · 3-4

---

## N

Name servers, querying · 6-10  
NETFAM statement · 4-3  
NETLOG  
    allocating job · 4-1  
    closing a log file · 6-2  
    creating individual error files · 6-3  
    defined · 5-1  
    defining job · 4-4  
    defining terminal · 4-2  
    LOGCTL command · 6-2  
    LOGFMT command · 6-3  
    opening a log file · 6-2  
    operating · 5-2  
    security · 5-2  
    sending an error message · 6-2  
NETLOG error messages · A-1  
NETSTA command · 6-3  
Network classes · 1-4  
Network statistics, displaying · 6-3  
Networks file · 3-4  
NETWRK. file · 1-4, 3-4, 3-13  
NSLOOK command · 6-10

---

## O

ONLINE command for SLIP · 7-2  
Optimizing performance · 8-1

---

## P

package manager · 6-19  
Performance, optimizing · 8-1  
PIC (Product Installation Code) · 2-3  
POP3 mail · 3-19  
    spawning POP3D server · 3-8, 3-19, 5-19  
POP3 protocol · 1-6  
POP3D  
    security · 5-20  
POP3D server · 5-18  
POPUSR. file · 3-19  
POST Method Arguments · 9-8  
Prerequisites · 2-1  
Printer initialization file · 3-17  
Printers  
    dynamic-IP · 5-22  
Privilege level for FTP users · 5-10  
PS command · 6-8  
PUBLIC.PPN · 9-3  
publicly accessible devices · 5-28  
PUT and DELETE Methods · 9-9

---

## R

reading the network error log · 6-17  
relative references · 9-1  
REPLY command for SLIP · 7-2  
Requirements  
    disk space · 2-2  
    memory · *See* Memory requirements  
RESOLV. file · 3-4, 3-12  
Restarting servers · 6-9  
ROUTE command · 6-6  
Routing table  
    changing · 6-6  
    displaying · 6-3, 6-6  
RPC: · 5-9  
RWHO  
    information provided · 5-34  
    memory requirements · 5-35  
    security · 5-35  
    spawning RWHOD server · 3-8  
RWHOD server · 5-34

---

## S

Script files for SLIP · 3-16, 7-1  
Security  
    BBSERV · 5-33  
    FTP · 5-11  
    HTTPD · 5-28  
    ITCD · 5-31  
    LPD · 5-24  
    LPR · 5-23  
    NETLOG · 5-2

- POP3D · 5-20
- RWHO · 5-35
- SLIP · 5-4
- SMTPD · 5-17
- TAMED · 5-36
- TCPEMU · 5-3
- TELNET · 5-8
- TFTP · 5-34
- SerialNet
  - replacing · 5-30
- Servers · 5-1
  - after booting · 4-5
  - AlphaNET tunneling · 5-28
  - BBSERV · 5-32
  - BOOTP · 5-38
  - BOOTPC · 5-39
  - configuring · 3-6
  - defined · 2-2
  - E-mail · 5-32
  - FTPD · 5-8
  - HTTPD · 5-24
  - in CONFIG. file · 3-8
  - ITCD · 5-28
  - LPD · 5-23
  - LPR · 5-20
  - Mail · 5-13
  - memory requirements · 3-9
  - POP3D · 5-18
  - printer · 5-20
  - printers · 5-23
  - RWHOD · 5-34
  - SLIP · 5-3
  - SMTPD · 5-13
  - SNTPD · 5-36
  - TAMED · 5-35
  - TDDD · 5-40
  - TELNET · 5-4
  - TFTPD · 5-33
  - Web · 5-24
- Servers, restarting · 6-9
- SERVIC. file · 3-5, 3-13
- Services · 3-5
- Sessions, terminating · 6-8, 6-18, E-1
- SETJOB statement · 4-4
- Setup files · 3-3
  - comments · 3-6
- SHARED command for SLIP · 7-2
- Shared memory · 3-2
- SI\$EXI
  - SI\$ABT · E-1
- SIEXIT · E-2
- SLIP · 1-6, 5-3
  - cabling · 7-4
  - log file · 7-3
  - script file commands · 7-2
  - script files · 3-16, 7-1
  - security · 5-4
  - setting Internet address in NETWRK. file · 3-16
  - setting up in CONFIG. file · 3-9, 3-16
  - TRMDEF for modem · 4-2
  - SLIPDn.LOG file · 5-3, 7-3
  - SLIPOK.n file · 5-3
  - SMEM · 3-2
  - SMEM statement · 4-4
  - SMTP · 3-18
    - spawning SMTPD server · 3-8
  - SMTP protocol · 1-7
  - SMTPD · 5-13
    - security · 5-17
  - SNTPD server · 3-9, 5-36
  - SNTPD. file · 3-5
  - Spawning
    - FTPdserver · 3-8
    - LPR server · 3-8
    - POP3D server · 3-8, 5-19
    - RWHODserver · 3-8
    - servers · 3-8
    - SMTPDserver · 3-8
    - TELNETserver · 3-8
    - TFTPDserver · 3-8
  - Spawning AlphaNET ITC tunneling server · 3-8
  - Spawning Electronic Mail server · 3-8
  - Spawning LPD server · 3-8
  - SSD (Software Security Device) · 2-3
  - STARTD command · 6-9
  - Subnetworks · 1-4
  - Synchronizing system time · 3-9, 5-36
  - System initialization file · 4-1
  - SYSTEM statement · 4-3
  - System uptime information · 3-9

---

## T

- TAME server · 3-8
- TAMED security · 5-36
- TAMED server · 5-35
- TCP · 1-7
  - ersatz device · 4-2
- TCP interface information · 6-18
- TCP message block size · 3-5
- TCP printer · 3-14, 5-20
- TCPEMU · 5-2
  - after booting · 4-5
  - allocating job · 4-1
  - defined · 5-1
  - defining job · 4-4
  - defining terminal · 4-2
  - IPCINI command · 8-1
  - memory requirements · 8-1
  - security · 5-3
- TCPEMU.MBD file · 3-5, 8-2
- TCPIP.ERZ file · 3-5, 3-13, 3-17
  - ersatz names · 3-5
- TCPLPD · 5-23
- TCPLPD: · 3-17
- TCPLPR: · 3-17, 5-20
- TCPMAL: · 3-18, 5-13

## TCPMON

- TCP/IP stack monitor · 6-14
- TCPMON command · 6-14
- TCPPOP: · 3-19, 5-18
- TDDD server · 3-8, 5-40
- TDDD. file · 3-5
- TELNET server · 5-4
- TELNET · 1-7
  - security · 5-8
  - spawning TELNET server · 3-8
- Terminal driver setup · 4-2
- Terminating applications · E-1
- terminating print requests · 6-18
- Terminating sessions · 6-8, 6-18
- TFTP · 1-7
  - security · 5-34
  - spawning TFTP server · 3-8
- TFTPD server · 5-33
- Time
  - synchronizing · 3-9, 5-36
- Time zone · See TIMZON. file
- TIMZON. file · 3-5, 3-13
- Tree View
  - FTP · 5-12
- TRMDEF statement · 4-2
  - for SLIP modem · 4-2
- TROUTE command · 6-22
- Tunneling
  - and SerialNet · 5-30

---

## U

- UDP protocol · 1-7
- UKILL command · 6-8, 6-18
- Uniform Resource Locator · 9-1
- UNIX file name conversion · 6-17
- Uptime information · 3-9
- URL · 9-1
- User names and passwords, FTP · 5-9
- User privilege level for FTP · 5-10
- Users file for FTP · See FTPUSR. file

---

## V

- Verifying the software · 2-3
- VIRTUAL DOMAINS · 9-10

---

## W

- WAIT command for SLIP · 7-2
- Web browser · 9-1
- Web documents · 9-1
  - counting hits · 9-4
- Web home page · 3-4
- Web Server · 5-24
- Web Servers · 3-20
- WEBCNT.LIT · 9-5
- WHOIS command · 6-18
- WHOIS. file · 3-14, 6-18
- World Wide Web server · 3-8