

PREFACE	1
I. General stuff	1
1. Let's begin with a short sermon	1
2. Where does the Mac fit in?	1
3. Various connection methods	2
II. MacTCP	3
1. How to get it?	3
2. What is it?	3
3. Let's do it!	3
4. Details...	7
5. More details	8
6. Odds and ends	10
III. Applications	11
1. Terminal emulation	12
2. E-mail	12
3. FTP	14
4. Network news	14
5. Internet Gopher	15
6. Miscellaneous gadgets	15
IV. Sources	17
1. Downloading text files	17
2. What next?	18
3. Contacts for commercial products	19
4. Let's end with a short sermon...	20
Appendix A. Peculiarities of Mac file transfers	20
Appendix B. Mac networking: mini-tour	21
Appendix C. Dial-in access	23

MacTCP and related Macintosh software

revision 1.2.1, March 16, 1993

Copyright Eric Behr, Illinois State University, Mathematics Department

This document can be freely redistributed in whole or in part, provided that this copyright notice is included intact, and that no material profit is generated from such a transaction.

With sincere thanks to:

David N. Blank-Edelman	dnb@meshugge.media.mit.edu
Steve Dorner	sdorner@qualcomm.com
Patrick Hoepfner	hoepfner@heasfs.gsfc.nasa.gov
Peter N. Lewis	peter@cujo.curtin.edu.au
David S. Saunders	dave@intercon.com

as well as to several contributors to Usenet newsgroups, and to all those who sent me corrections and suggestions (but I'm the sole author of all mistakes, omissions and inanities)

Please send all comments and suggestions to behr@math.ilstu.edu or to ejbehr@rs6000.cmp.ilstu.edu. The newest version of these notes can be obtained by anonymous ftp to [spider.math.ilstu.edu](ftp://spider.math.ilstu.edu), or 138.87.132.21 (file /pub/mac/mac-tcp.txt), or from a gopher server on the same computer (directory "Various documents").

Significant changes from revision 1.1:

- generally brought the document in sync with MacTCP 1.1.1
- dropped remarks applicable specifically to MacTCP 1.0
- added e-mail address for site licenses
- changed the paragraph about MacTCP being bundled with certain free applications
- corrected class C address range
- changed the description of "disappearing LLAP icon" problem
- conflict at boot time with Dayna DOS Mounter
- mentioned apparent conflict with Super ATM
- mentioned PC Eudora
- Telnet 2.5+ now uses MacTCP or internal drivers
- modified description of LeeMail somewhat
- added separate section on gopher
- mentioned ircle, motd, tardis, hytelnet, archie
- added Webster's e-mail address and US phone
- added pointers to Comer's, Krol's and Kehoe's monographs
- updated the list of software on sumex
- mentioned DiskCopy
- mentioned Mac PPP, CCL and Calypso
- added pointer to SLIP info on [ftp.bio.indiana.edu](ftp://ftp.bio.indiana.edu)

PREFACE

Judging by questions and pleas for help, more and more people are playing with MacTCP, and some of them are having problems installing and configuring it.

I've spent many hours installing MacTCP on various Macs. I also devoted a large part of my free time to digging up various useful applications for use on the Internet. Hopefully some of my experiences will be helpful to you.

Please note that I'm not a networking expert. Some of the advice below simply stems from my paranoia and is not necessarily based on facts. There are many problems with MacTCP which I don't fully understand. In addition, please remember that I'm not a technical writer, and my style may not be to your liking —sorry about that.

Those readers who are new to the field of Macintosh networking might get something out of Appendix B —please look at it before you continue—.

I. General stuff

1. Let's begin with a short sermon

I started using the Internet several years ago, and it has become an important part of my life —for better or worse. I know that many people share my feelings about it. For the sake of all of us, please be considerate! There are lots of things you can mess up if you don't know what you are doing. Connecting your computer to the network which spans continents and is used by millions of people in their work is (or at least should be) a serious act.

Please follow this simple advice:

- when in doubt, don't do anything without consulting your local knowledgeable system administrator — chances are that if you don't, you will break more than you'll fix;
- read newsgroups such as comp.sys.mac.comm, comp.sys.mac.system, comp.sys.protocols.tcp-ip, comp.protocols.appletalk, especially the Frequently Asked Questions collections which are posted in most newsgroups from time to time;
- read the flaming manuals!
- if you are doing something especially tricky, find out how to access Internet documents (esp. RFC's) and read the relevant ones.

2. Where does the Mac fit in?

The Mac has always been a wonderful network machine. But for most people "Macintosh networking" means Apple's proprietary system called AppleTalk. The Internet uses a different networking "language". A major part of it is the protocol known by its acronym TCP/IP. To access the wonders of the Internet, a computer must understand TCP/IP.

Even though there are other ways to make the Mac speak TCP/IP, we will concentrate on the most popular and orthodox one, namely the **MacTCP driver** from Apple.

3. Various connection methods

The necessary software is only one part of a network connection. The other, obviously, is suitable hardware. A Macintosh can be connected to the outside world in many ways:

- with a LocalTalk or PhoneNet connector (all Macs)
- with a modem and suitable software (see Appendix C)
- with an internal Ethernet card (Mac SE and newer)
- with an external SCSI Ethernet adapter (all Macs)
- with an internal TokenRing card (Mac SE and newer)

Until recently MacTCP was not compatible with the last method of connection listed above. In mid-1992 Apple announced a Token Ring extension for use with MacTCP 1.1 and later, which does the trick.

The problem of providing reliable and convenient dial-in access would require a separate book. A short note on this subject is found in Appendix C.

Macs on Ethernet can usually connect to the outside TCP/IP world without a problem; for a long time TCP/IP has been the protocol of choice on Ethernet networks.

Macs on LocalTalk can communicate with the outside TCP/IP world only if the LocalTalk is connected to a larger network using an IP gateway such as FastPath (Shiva), Gatorbox (Cayman Systems), Multiport Gateway (Webster Computer Corp.), EtherRoute/TCP (Compatible Systems), or one of other equivalent devices; or if such a gateway is present elsewhere on your internet, and is visible to your Mac. None of the software routers like the Apple Internet Router or Liaison (Infosphere) currently provides TCP/IP services, so if that's what you use to connect to an Ethernet, you must also have a gateway which will translate the IP packets wrapped in AppleTalk (which your LocalTalk Mac sends) into genuine IP packets going to the outside world, and vice versa. We briefly describe some of the details in Appendix B.

Please note that AppleTalk and TCP/IP are two completely different animals, even though a single Mac can use both at the same time; configuring AppleTalk functions, such as printing over the network or AppleShare, will not be discussed here.

II. MacTCP

1. How to get it?

MacTCP (version 1.1.1 as of this writing; version 2.0 will be released soon) is marketed by APDA (800 282 2732). Don't expect your local Apple dealer to know much about it. Current price of a single copy is about \$100. Volume licenses are available, and some institutions may qualify for discounts. Call APDA, or send e-mail to sw.license@applelink.apple.com. If you already have an old version, you should definitely consider upgrading to the current one, even though it may be quite costly.

Some public domain TCP/IP applications used to come bundled with MacTCP. Developers used to pay Apple a rather stiff fee for a "distribution license". Apple has recently changed the rules in this area, significantly increasing those fees. The situation is still a bit murky, but it seems that it is impossible to obtain the newest version this way. MacTCP 1.1 can still be found bundled with older applications (e.g. Eudora 1.2.2). Note that MacTCP 1.1 may not work reliably, or at all, with System 7.1 or newer.

2. What is it?

For the user, it is a Control Panel, configured just as the Monitors or Sound panels. For applications, it is a set of procedures which allows them to communicate with other hosts on the network using the TCP/IP protocol. It is designed to be transparent in the sense that once it is properly configured, any correctly written application can make use of it without user intervention. Granted, life without MacTCP would be hell! In practice however, both MacTCP and the applications which use it have idiosyncrasies and bugs which make this software difficult to handle. In my own experience, the Mac SE is the most unstable MacTCP platform, followed closely by the LC, probably because of the peculiarities of their ROMs; but I admit that it is a statistical observation, with no theory at all to back it up.

Apple, in all its charming inconsistency, tells us to use their Installer even when adding a silly little printer driver, but at the same time does not provide an Installer script for the rather intricate procedure of installing and configuring MacTCP.

Here's the way I've successfully done it on many machines. These are purely empirical findings, so treat these procedures more like a rain dance than a rational troubleshooting guide.

3. Let's do it!

If you are installing MacTCP for the first time, or if you suddenly run into problems such as the dreaded message "Error opening TCP drivers - possibly no dynamic addressing", you should follow these steps for maximum effect.

My general advice is: start with virginal (or at least freshly updated) system files. Problems may result from improperly installed old version of AppleTalk or other resources. For example, I could never painlessly install new Ethernet drivers on any Mac which previously had drivers for the Nuvolink SC box installed on it (I don't mean to put down Nuvotech's products — they usually work fine). What's worse, some network applications seem to corrupt the System, and/or other important files, when they crash. So here we go:

- If you are having major problems, boot from a system floppy, and trash the following from the hard disk: MacTCP, AdminTCP, MacTCP Prep, and MacTCP DNR. These files are found in the System Folder proper (in pre-7 Systems) or in various subfolders, such as Control Panels, Preferences and Extensions. If problems persist, consider dumping at this stage not only the MacTCP files, but also the System, Finder, Multifinder, all network drivers and control panels (AppleTalk, EtherTalk, Network), and reinstalling fresh copies. This is almost certainly an overkill, but in some cases it is also a recipe for instant happiness. Of course, you'd better first back up all the files which seem important to you, such as non-standard extensions or fonts, which will get erased in the process! Then install a fresh system.
- Disable all INITs, and non-essential network-related CDEVs, especially those which may be requesting network services at boot time (such as Network Time). Again, this probably isn't necessary, but who knows... As is well-known, many INITs produce mind-boggling conflicts. When you succeed in making MacTCP work without them, you should later reinstall them one by one, checking - say - telnet after activating each of them. If MacTCP conks out, you'll know the culprit! If that happens, please publicize your findings, or at least send a note to me.
- If you will be using MacTCP over Ethernet, install network drivers according to manufacturer's instructions. For some reason I had better luck when I turned off AppleTalk in the Chooser before doing this. If you are installing Apple's drivers, make sure to use the latest Network Installer disk.

If you will be using MacTCP through the serial (printer) port, activate AppleTalk in the Chooser now. Some Ethernet drivers, and the MacTCP Token Ring extension, also require AppleTalk to be active for MacTCP to work.

- Configure the network drivers, if necessary. Most Ethernet drivers have no user-changeable settings at all. The Token Ring driver (configured via its Control Panel) should be set for the correct speed (4 or 16 Mbps). It also lets you set the hardware address of the adapter; if your site uses locally administered addresses, you need to get one from a local administrator, and set the driver accordingly.

If you need your Mac's Ethernet hardware address for debugging purposes or simply for your records, you can get at it by holding option down while clicking the "Ethernet built-in" icon in the MacTCP panel. Another method is to log on to a nearby Unix machine, do a "ping" to your Mac's IP address, and then tell the Unix host to list its "ARP cache": `—/etc/arp -a` on most systems—. The hardware address will appear next to your Mac's name as 12 hex digits divided into pairs.

Remember that Ethernet and Token Ring handle hardware addresses differently; if the Unix host and the Mac are on those two different types on networks, you need to translate each pair of hex digits into 8 binary digits, reverse their order, and translate the whole shebang back into 12 hex digits to get the real hardware address.

- Put fresh MacTCP, AdminTCP in the System folder. Make sure that you are using the newest version, and not some old copy obtained from shady sources. Under System 7+, they will go into the proper subfolder automatically if you drop the files on top of the System Folder icon.
- In the next few steps in our procedure, MacTCP will create two files: MacTCP DNR and MacTCP Prep (under System 7 the latter should end up in the Preferences folder). You did remove the old ones, didn't you? Since some virus protection utilities might think that the new files that MacTCP is installing in the System folder are viruses, it is a good idea to shut off all virus protection before you go any further.
- Configure MacTCP or AdminTCP (you will find more about that in later sections). Make sure you set the right network class and subnet mask if you use static addressing. You don't have to worry about that if you use a server such as FastPath or EtherRoute/TCP to assign addresses. For the time being, don't do anything with the "gateway" (should be 0.0.0.0) and "nameservers" (blank) fields.
- Remember to set the correct physical layer setting ("Ethernet Built-in", "LocalTalk", or "Token Ring") in the MacTCP panel (if it gives you that choice).

Let me reiterate that the physical link setting has nothing to do with the network resource (selected in the "Network" CDEV) which you will be using for AppleTalk functions, such as printing or AppleShare: we're dealing with strictly TCP/IP stuff.

If your Mac is using a direct Ethernet connection for TCP/IP, you normally **don't need to** (nor should you, in some specific cases) activate AppleTalk or choose anything in the Network CP at this point. As we mentioned before, there are exceptions to this: some Ethernet drivers, and the MacTCP Token Ring extension, do require AppleTalk to be turned on in the Chooser.

If you are using TCP/IP through the LocalTalk port, you must have activated AppleTalk in the Chooser before - otherwise that port is simply dead, and MacTCP doesn't know how to communicate with the outside world! On a Mac Plus, trying to reconfigure MacTCP with AppleTalk shut off will likely result in a horrifying "address error" or "illegal instruction" message. Don't worry —reboot, turn on AppleTalk and see if the new MacTCP settings are still there. If not, configure it one more time. Everything should be fine after that.

- Even though MacTCP 1.1 doesn't always tell you to do so, REBOOT now! Check that MacTCP DNR (and usually MacTCP Prep) files show up in the System folder.
- Older Macs, notably the Plus, become overworked while running System 7 (more precisely, the newest versions of AppleTalk) and MacTCP 1.1. Apple has released an unsupported patch which fixes the problem. You can get it from ftp.apple.com

(see section IV, Sources, below). Do **NOT** install this patch unless: you have a Plus, **and** you run AppleTalk vsn. 56 or higher, **and** you use MacTCP 1.1. Another workaround is to use a SCSI Ethernet adapter instead of a LocalTalk connection; the SCSI port can handle this load, while the serial port controller can't.

- Test MacTCP by running an application such as NCSA Telnet (the MacTCP version, of course). If the NCSA logo quietly disappears, you're in business. Try connecting to a **nearby** host by giving Telnet its decimalized IP address, such as 138.87.132.21.

As of this writing, newest versions of NCSA/BYU Telnet sometimes produce a system error when they're run on a Mac Plus under pre-7 System. This might indicate a serious problem, but in 9 cases out of 10 this happens only the first time you attempt to run Telnet. Reboot and try again.

- Assuming all went fine, you can now set the AppleTalk link in the Network panel if you wish (if AppleTalk was off, that is; this should also automatically turn on AppleTalk in the Chooser, but check it just in case). There are some exceptions here. For example if your Mac is on Ethernet, which in turn is connected to a Token Ring with the IBM 8209 bridge, activating AppleTalk might screw up TCP/IP. Details available on request.
- If you can't connect to remote hosts, check if the "default gateway" field in MacTCP changes from the original 0.0.0.0 to something else after you start telnet or other MacTCP software. If it doesn't, then your Mac is having trouble discovering a gateway which connects your LAN with the rest of the world. You need to tell MacTCP where it is: enter its numeric IP address in the "default gateway" box. You may also want to tell it where the local nameserver for the root domain is (put a period in the "Domain" field, enter the IP address in the box next to it), so you can check whether the Mac can use symbolic (not decimal) addresses. To be on the safe side, trash MacTCP DNR and MacTCP Prep again; be sure to reboot the computer.
- You should now be able to telnet to a remote host, and Telnet should be able to resolve symbolic addresses for you: e.g. nic.ddn.mil or sumex-aim.stanford.edu.

If after all this you are still having problems, first check the hardware (is Ethernet up? are plugs plugged in? switches switched on? gateways are gatewaying? etc) Next, consider the possibility that some other node is using your IP address. See if things work if you change the IP address to another one from the range assigned to you. When MacTCP loads, it in effect asks the network: "host with IP number so-and-so, please respond". Your Mac's IP address is used in that query. If there is no answer, all is good. But if something out there responds, MacTCP correctly assumes that there is a conflict in IP numbers, and returns an error code which most applications then translate to a less-than-helpful message like "MacTCP is not installed correctly", or "Error opening TCP drivers - possibly no dynamic addressing".

If all of the above doesn't help, buy a pair of tickets to Jamaica and invite your high-school flame along. After all, will you let a piece of steenkin' iron drive you nuts?!

4. Details...

In the description below, we use the MacTCP control panel. The only difference between it and AdminTCP is that the latter allows you to lock and unlock the former; other than that, they are functionally identical. If the MacTCP panel you have has been locked, you won't be able to make any changes to it; you have to unlock it with AdminTCP first, or simply make all changes from within AdminTCP.

After opening the MacTCP control panel, you should first set the physical link for the TCP/IP connection; for instance, if you use MacTCP on a LocalTalk network—whether gatewayed to an Ethernet, or isolated—select the "LocalTalk" icon. If you are directly on Ethernet, click the "Ethernet Built-in" icon.

Now click the More button. In the first stage, you will be setting only three things: address assignment mode, network class, and network subnet.

- If you are a "trusted and permanent" user who received a specially assigned IP address, click the "Manually" radio button. As a rule, it is best to have a permanent, static IP address. Some networks, however, have more nodes than addresses available to them. Such institutions resort to assigning addresses dynamically to those hosts (usually PCs) which request them; see the next subsection.

Please remember that choosing an address at random will at best make your connections unreliable; it can also provoke unrestrained wrath of some powerful people in your organization. Courts have been known to drop bodily harm charges under much weaker extenuating circumstances. Moreover, some network numbers are illegal on the Internet. If you pick one of those, you'll be asking for a lot of trouble—in particular, you'll be getting all e-mail addressed to the University of Mars for the rest of your life.

- The IP number you were assigned determines the network class. MacTCP is smart enough to figure it out by itself. You can also set it manually: addresses beginning with 1-127 are class A, 128-191 indicates class B, and 192-254 are class C. Make sure you know what you are doing.
- The netmask can be determined by stopping a friendly administrator in the hall and asking one or two simple questions. "Do we use subnetting?" And, "If so, how many bits form the subnet number?" If the first answer is "no", then slide the subnet mask indicator as far to the left as it will go. Such masks are 255.0.0.0 for class A networks, 255.255.0.0 for class B, and—guess what— 255.255.255.0 for class C.

If your site does use subnets, the plot thickens. A number of address bits is set aside as the subnet number. You have to enter the subnet mask (such as 255.255.255.0, eight bits determining the subnet in a class B network) in the appropriate field; or you can delicately slide the subnet indicator to the right until it marks the right place. For example, if 6 bits are reserved for the

subnet, you slide the thing from the leftmost position to the right by six little notches.

Be forewarned: it is important that all hosts which have the same network part in their address use identical subnet mask. You can avoid a lot of desperation in the future if you make sure you get this straight. 'Tis the time to read the manuals, perhaps?

- Get out of the "inside" panel by clicking OK. Type your IP address in the edit field in MacTCP. Go back to the subpanel to verify everything is as it's supposed to be. If anything has changed all by itself, you are probably trying to set a wrong network class.
- If you were told that "dynamic address assignment" is available to you, then all you need to do is click the Server radio button in MacTCP. Things should work automagically thanks to built-in intelligence of devices such as the FastPath, GatorBox, or EtherRoute/TCP. If there are problems, you have to get together with the person who maintains your gateway.
- Many people prefer to avoid the "dynamic" button in MacTCP. Others have been using it successfully. There are situations when Apple's "dynamic addressing" is called for, but don't use it unless you have to.

After you have entered all the required information, and you suspect that the user has itchy hands, it's time to click the "protected" box, close the panel and reboot right away. Remember to trash AdminTCP from his disk after you are done setting things up.

If the user is likely to heed your advice, but might accidentally change the settings, click the "protected" box in AdminTCP, tell him not to fiddle with AdminTCP, and leave the control panel on his disk. This will save you some time and trouble when you have to change his settings again.

If the prospective user is a TCP/IP sage, then you of course leave things wide open for him to play with the little buttons during long winter nights.

There are some other permutations involving, for example, locking just the network part of the address, but leaving the subnet and node numbers accessible. Please read the manual!

5. More details

After you have coerced MacTCP to work with this basic configuration, you may want to enter a few more things.

The "Gateway Address" field in the MacTCP control panel should contain the IP number of the default gateway, i.e. the place where MacTCP should send packets if it doesn't know what else to do with them (i.e. packets addressed to a remote network). On most networks you will leave that field set to 0.0.0.0, and MacTCP will locate the default gateway by means of a magic wand called Routing Information Protocol (RIP).

If your network for some reason does not implement RIP, you must enter a "real" address there. For example, if your Mac is on LocalTalk, you would use the address of a FastPath or another device which connects your network with the Ethernet. If you are directly on Ethernet, you should use the address of the router which connects that Ethernet to the campus network, or —if you don't have routers— the gateway which links your outfit with the rest of the world.

You will also want to tell MacTCP where to find nameservers. Going into the details of the Internet Domain Name System (DNS) would make this document twice as long, so we will stick with the basics. The Domain Name System is used by computers to convert the human-friendly names such as "spider.math.ilstu.edu" to numeric addresses (IP numbers) such as 138.87.132.21. On occasion, computers use this system to "reverse-map" numeric addresses, i.e. to verify that your numeric address does correspond to a name which recognized by the DNS.

If your network is connected to the Internet, your computer lives in some well-defined "domain". In my case, it is math.ilstu.edu, and a machine with address 138.87.132.21 is providing name service for that domain. I thus enter "math.ilstu.edu." (no quotes, but note the period at the end!) in the Domain field, and 138.87.132.21 in the Server field next to it. I also click the Default button. This informs MacTCP that my computer is in the math.ilstu.edu domain, so when I tell my Mac to telnet to "koala", it will query the nameserver about the address of the computer "koala.math.ilstu.edu". Here is the first row of edit fields on my Macs:

math.ilstu.edu.	138.87.132.21	<input checked="" type="radio"/>	("default" button on)
-----------------	---------------	----------------------------------	-----------------------

If the local nameserver doesn't keep a large table of addresses, or if it is less than reliable, you will probably want to put in some backup servers in fields below. In particular, one or more Domain fields could contain the period alone ("root domain", i.e. the entire Internet) with the address of a big, reliable machine next to it.

Clicking the "default" button by no means guarantees that this particular server will be consulted first; MacTCP first contacts those nameservers listed in the control panel which seem to match the domain name included in the query - and only if that fails, the default server is asked as a last resort. However, all domain name queries should generally go to your local nameserver first. You may want to enter your local nameserver's address in the second row as well, specifying the root domain (period) in the first field. In my case, the second row would thus look like this:

.	138.87.132.21	<input type="radio"/>	("default" button off)
---	---------------	-----------------------	------------------------

A small text file called Hosts can be put in the System Folder proper. I was told that some applications (e.g. VersaTerm) automatically create this file and modify it when needed. The Hosts file lets you enter additional information which MacTCP uses. For example, I could enter

spider.math.ilstu.edu.	IN	A	138.87.132.21
math.ilstu.edu.	IN	NS	spider.math.ilstu.edu.
rs6000.cmp.ilstu.edu.	IN	A	138.87.1.2
rs6000.	CNAME		rs6000.cmp.ilstu.edu.
ilstu.edu.	IN	NS	rs6000.cmp.ilstu.edu.

there to tell MacTCP that the computer spider.math.ilstu.edu has address 138.87.132.21, that it is a nameserver for my domain, that rs6000 in Computer Services has address 138.87.1.2, that I connect to it often enough so I want my Mac to use rs6000.cmp.ilstu.edu whenever I say "rs6000" (otherwise MacTCP would be looking for rs6000.math.ilstu.edu!), and that it is a nameserver for the domain directly above me.

The use of periods at the end of domain names guarantees that MacTCP treats them as "absolute names", and doesn't try to append anything to them. If you know what you are doing, you can work with relative names (without the period) saving some space, typing, and quite possibly reducing the amount of memory MacTCP consumes. Assuming that my Macs are placed in the domain "math.ilstu.edu" by using the 'default nameserver' button in MacTCP, I could simply use "spider" instead of the absolute "spider.math.ilstu.edu.", because that extension would be automatically appended to a relative name without the period.

For a list of specifications allowed in the Hosts file, see the MacTCP manual. I have also collected a few relevant Usenet postings in a file mac-dnr.txt, available by anonymous ftp from spider.math.ilstu.edu (directory /pub/mac).

After modifying the Hosts file or the nameserver information in the MacTCP control panel, make sure to trash the MacTCP DNR file, and reboot right away.

6. Odds and ends

We end this section with a few miscellaneous problems reported in Usenet newsgroups or in personal communication.

Sometimes the physical link icons (usually the "LocalTalk built-in" icon) mysteriously disappear from the MacTCP 1.1 control panel. This problem is fixed in the current version of MacTCP. If you are still using 1.1, read on. If the MacTCP panel does not show any choices at all, it might help to zap the parameter RAM by holding down:

command-option-p-r while restarting the mac (system 7)

command-option-shift while opening the Control Panel (pre-7 systems)

(reported by Mike Wiese of MIT in a Usenet posting). I haven't had much luck with this fix. Running System 7 and the TuneUp extension on a Mac broken in this fashion, I found that I had to reinstall the system and network files from scratch, then install, configure and test MacTCP, and finally install the TuneUp as the last step.

Stanley Dorst (skd@pop.pitt.edu) reports a conflict between MacTCP 1.1.1 and Dayna DOS MOUNTER 3.0. DOS MOUNTER must be made to load *after* MacTCP (by changing the name, most likely; I haven't tried this combination).

One of the gizmos with which our users are currently having problems is Adobe's SuperATM (which advertises its serial number on the network, and in the process appears to choke MacTCP). I don't have a solution for this.

Don't disregard the physical connection. I've seen several reports of strange problems which were eventually traced to a bad wire or transceiver. Some of Apple's "FriendlyNet" cables for use with the Quadras were involved. Borrow a working transceiver and cable from someone and try it before blaming everything on the software.

Are things messed up, and you are stuck? Make sure to contact the vendor of the network adapter. There may be a problem which they know about. You may need to get an updated driver from them.

It was reported that the Apple cache card (for the Ilci) used to cause problems with MacTCP and a lot of other things. Scream for a replacement!

Some cases of MacTCP applications freezing certain machines (e.g. older SE's) seem to be caused by bad karma between MacTCP and the system ROMs. Matthew D. Heys a while ago reported extensively on his experiments. He empirically demonstrated that the freezes happen after a "cold start" (i.e. powerup), but do not occur after the computer is restarted with the programmer's switch or with the "Restart" in the Finder's "Special" menu. This may or may not be fixed in MacTCP 1.1.1 —I haven't heard one way or another.

III. Applications

There are many commercial programs which use TCP/IP connectivity (InterCon's family of networking applications, or VersaTerm from Synergy Software). I will not review them

here because I have had little or no experience with them. Let the vendors speak for themselves. Some makers of similar commercial products are mentioned in Section IV for your convenience.

The list of public domain or shareware software which follows will hopefully be amended and expanded in the near future —please send in your favorites!

1. Terminal emulation

There are currently two popular freely available programs which let the Mac connect to other hosts as a terminal using TCP/IP (telnet): NCSA Telnet and its derivative, tn3270.

The first one is being developed at the National Center for Supercomputing Applications in Urbana-Champaign. It emulates a vt100 terminal and provides some Tektronix graphic terminal capabilities; it also implements an ftp server function, i.e. when it's running, you can ftp **from** other hosts **to** your Mac. NCSA's version was modified somewhat by the Brigham Young University; NCSA/BYU Telnet implements a simple ftp client function.

The second, tn3270 written at Brown University, is a variant of Telnet which provides the IBM 3270 terminal emulation. It also supports file transfers and printer sessions, but I don't know enough about our IBM big iron to actually try them out...

NCSA Telnet used to come in two versions: one which relied on MacTCP, and one which included built-in TCP/IP drivers. Starting with Telnet 2.5, the two have been merged into a single package.

Stanford University has developed a comprehensive package of TCP/IP applications collectively known as SU-Mac/IP. I didn't have a chance to try it. One advantage of Stanford's software is that it reportedly gives more informative error messages, which can be a great asset —especially when you are having problems! Contact macip@jessica.stanford.edu for conditions of a site license. If you know how to use MacBinary FTP, you can download the entire SU-Mac/IP documentation in MS Word format from [jessica.stanford.edu](http://jessica.stanford.edu/directory/netinfo/macip/manuals4.0/users) (directory [netinfo/macip/manuals4.0/users](http://jessica.stanford.edu/directory/netinfo/macip/manuals4.0/users)).

2. E-mail

There are several schemes in which a Mac can access Internet mail. The crudest way, of course, is to telnet to a host on which you have an account, and use that host's mail facilities. Another is to keep using whatever mail system you have on the AppleTalk network (e.g. QuickMail or Microsoft Mail), and then provide a SMTP gateway which will translate it to Internet mail; this tends to be expensive and sometimes unreliable.

By far the most popular and convenient system is the client/server method, in which one computer uses its powerful mail software and provides service to clients such as Macs or PCs. Macintosh users have the good fortune of being able to use some excellent mail

clients which work on a Mac. Eudora written by Steve Dorner leads the pack (in my humble opinion), with MIT's Techmail a close second.

Eudora and other similar clients allow the user to read, compose, and edit mail on the Macintosh desktop; it can print mail, save messages as Mac files, and attach Macintosh-specific files (say, formatted Word documents or even applications) to the letters using the Macintosh standard BinHex encoding scheme (see Appendix A). When it's time to process mail, Eudora contacts the server, uploads messages waiting to be sent and downloads those which the server received for you. The (supposedly) well-maintained and well-connected server computer handles the rest, so you don't need to know anything about Unix or any other alien operating system.

A special "Post Office Protocol" has been defined to handle this. There are two versions of it: POP2 and POP3. The server and the client have to agree on the version they speak. Most Mac clients, in particular Eudora, are POP3. If you succeed in sending mail, but attempts to check incoming messages fail, then it is very likely that your POP client and server use different versions of the protocol.

It is easiest to set up a POP server on a Unix computer (popper from UC Berkeley seems to be the way to go here), but there are also servers operating under other systems, including one (POP2) which runs on a Macintosh. If you use a VAX with VMS, and some TCP/IP package is already installed, chances are that it includes a POP server. There is also a public domain POP3 server for VMS available from Indiana University, which a dear friend of mine, a computer semi-literate, got up and running without much grief. Talk to your local system administrator.

For those of us who need to worry about PCs too, POPmail from University of Minnesota runs on MS-DOS machines and is compatible with both POP2 and POP3. A POP2 server running on a Mac is also available from that site. Finally, a PC version of Eudora can be found on Qualcomm's ftp server.

A Macintosh mail client is also included in Stanford's SU-Mac/IP package, available for a nominal fee.

In case you don't have a reliable machine which may be used as a POP server, don't worry; your Mac can be made into a full-blown SMTP mailer, and it then behaves like any other "real" Internet mail node. Lee Fyock's LeeMail does just that. Using it avoids the complications involved in maintaining a central mail computer, and in setting up a client-server system such as POP. Naturally, a Mac configured as an independent Internet mail host had better have reliable connectivity with the world at large (and a properly configured Domain Name Resolver).

Finally, a Swedish offering —MacPost— is something of a mixture of both methods; it has both a client and a server running on a Macintosh, but requires a Unix host as an intermediary for outside mail. The transactions between the individual clients and a MacPost server only use AppleTalk, and TCP/IP is only needed by the server Mac to communicate with the Unix host. I haven't used this one either, but who knows —it might be just the thing for you.

Steve Dorner occasionally posts his much more complete list of Mac POP clients (including commercial ones) to newsgroups such as `comp.protocols.appletalk` and `comp.sys.mac.comm`.

3. FTP

As we mentioned above, Telnet has limited ftp capability. But two other applications, XferIt and Fetch, are specifically designed to provide "FTP with a human face". Both present the directory on the remote host as a Macintosh dialog window, in which the user can select files to be downloaded. I've found minor quirks in both of them, but overall they are extremely impressive and useful. There is also HyperFTP, a HyperCard stack which is similar to the two applications. It is very solid, but relatively slow, and —naturally— it requires a reasonably recent version of HyperCard and consequently quite a bit of RAM.

As with telnet and mail, an ftp application is also available as part of the SU-Mac/IP software.

When using FTP from a Mac, you should realize that many anonymous FTP sites do not allow connections from hosts which are unknown to the Internet nameservers. To connect to such nodes, your Mac's IP address has to "reverse-map" to a legal Internet name, like `mac1.math.ilstu.edu`. The system administrator of your nameserver computer might be willing to enter your address in his database.

See Appendix A for a very short introduction to the intricacies of Macintosh file transfers. For more information on the procedures involved in FTP'ing Macintosh files, read the text file `ftp-primer.txt`, which is on `sumex.stanford.edu` in the directory `info-mac/reports`.

4. Network news

As with FTP, there are three applications and one HyperCard stack designed to let you access news: TheNews, Newswatcher, NetNews (stack), and Nuntius. I have not experimented with Nuntius, so I can only quote second-hand information: at least one of my correspondents swears by Nuntius as "the best newsreader I have seen on the Mac by far".

There are minor problems with each of the first three, compounded by the fact that this method of reading Usenet articles usually involves downloading humongous lists of articles over a slow connection such as overloaded LocalTalk, keeping track of read items, and so on. This is not for the faint of heart. Many people still stick to the old-fashioned method of logging on to a bigger host and reading news there. But when it works, it's worth it! You can finally organize the saved articles on your own disk, use your favorite word processor to write replies, etc. I hesitate to recommend specific choices, but personally I have found the newest NewsWatcher to be very stable, powerful and easy to use.

News clients require an address of a nearby friendly NNTP server. The server needs to be friendly in the sense that it must recognize your Mac as a host which is allowed to post news. This usually requires having a valid domain name (see above under FTP), and the server must have been configured to accept uploads from your computer. Even though many servers allow free read access but limit posting, the Mac clients will usually give up without explanation unless they are granted both permissions. This causes a lot of confusion among novice users, who later complain about "broken newsreaders".

5. Internet Gopher

The Internet Gopher searches "gopher servers" on various Internet hosts and retrieves various information such as directory entries, class schedules, campus newsletters, etc. One Macintosh client is a HyperCard stack written at University of Minnesota. It has been recently superseded by an application called TurboGopher, which definitely lives up to its name. Another standalone implementation is GopherApp from Indiana University.

Gopher is quickly becoming the standard way of providing access to distributed information such as campus directories, WAIS servers, on-line publications, etc. It is also the preferred method of accessing many of the anonymous ftp sites, such as the sumex archive at Stanford.

6. Miscellaneous gadgets

There are dozens of useful or simply nifty applications which use MacTCP. Here are some of them.

Network Time written by Peter Resnick is a CDEV which periodically contacts an Internet timeserver, like merit.edu or nic.ddn.mil, looks at the Map CDEV to determine your location (and hence time zone offset; I particularly like this elegant touch), and sets your Mac's clock. After installing it you can boast that your screen clock is accurate to a tiny fraction of a second.

A Chooser extension called Tardis is functionally similar to Network Time. It uses a Timelord server running under Unix or Mac OS to set the time on networked Macs. Another Chooser device, motd, will access a Unix motd server, providing a very useful "message of the day" service. The servers on the Unix side rely on the CAP libraries. Both of these gadgets use AppleTalk rather than TCP/IP, so they really don't belong here, but I think they are useful enough to deserve a short mention. CAP, motd, and tardis are all available from munnari.oz.au (and many other places).

Finger and fingerd by Peter Lewis give Macs the Unix "finger" capability. Finger sends a finger query to a remote host; it's even smart enough to automatically put people whom you "finger-ed" in a menu, so you don't have to type their address again. Fingerd simulates a finger daemon on your Macintosh, so other people can

finger you while you are simply working on your Mac.

The same author has released another fine product —Talk— which simulates a Unix "talk" client and daemon on the Macintosh.

Peter's latest product, `ftpd`, runs in the background under System 7 (with file sharing enabled) and turns the Mac into an ftp and gopher server.

Those who are familiar with the Internet Relay Chat system will want to try a full-blown IRC application, `ircle`. Chat (also by Peter Lewis), is a simple IRC-like server running on a Mac.

Wide Area Information Service is a fruit of a joint project of Apple Computer, Thinking Machines, Peat-Marwick and Dow Jones. There are several WAIS servers in operation; they can be queried from a Mac by means of an application called `WAISStation`. A WAIS server can also be accessed by telnet to `quake.think.com` (login name `wais`), but the interface is then much less friendly. Check it out! It's a glimpse at the things to come.

MIT's TechInfo is another impressive example of a fast and easy to use information retrieval system. MIT and a few other institutions run servers which store and search databases of campus events, faculty and student directories, etc. Both Mac and PC clients are available.

Unix users may be familiar with `hytelnet` ("Hyper Telnet"), which is somewhat similar in spirit to the gopher system. It provides a convenient way of exploring public Internet resources accessible by Telnet. A Macintosh implementation of `hytelnet`, based on HyperCard, can be requested by e-mail from Charles Burchill (`burchil@ccu.umanitoba.ca`).

Thanks to Chris McNeil, we finally have a Macintosh archie client. It searches archie servers, retrieving from them information about the contents of anonymous ftp archives.

If your Macs use the CAP AppleShare server, chances are you'll want to look at `MacDump`, which permits automated backups of your Mac volumes onto the CAP server.

A desk accessory called Client DA (written by Greg Anderson of UC Santa Cruz) connects to the Unix daemon "supersrv" (by Steven Grimm), which in turn uses the wonders of Unix to do whatever you want it to do. For example, a trivial Unix shell script invoked by `supersrv` allows our faculty to search all users on the system, or access the archie ftp search facility directly from their Macs. Client DA has some problems with smaller Macs, and I wish someone kept maintaining this wonderful gadget.

On the lighter side, we have `GameMaster`, which lets you play a variety of games over AppleTalk and TCP/IP networks.

There is also a Macintosh version of ping, called `Mping`; it is available from `ftp.apple.com` as one of the examples of a TCP/IP application (file `/dts/mac/netcomm/mactcp-1-1-examples.hqx`).

A bare-bones prototype of an SNMP management console can also be found on <ftp.apple.com>, in the file `/pub/apple-ip/snmpool1.0d7.sit.hqx`. It is rather crude, but I did use it to snatch some useful data from our network devices.

Please let me know what are your favorites in this category —I'll be happy to expand this section!

IV. Sources

There are some excellent sources of background information on the Internet in general, and TCP/IP in particular. Two classics which come to mind are:

"Zen and the Art of Internet" (by Brendan Kehoe),
 "Hitchhiker's Guide to the Internet" (by Ed Krol).

Both can be obtained by anonymous ftp from <ftp.uu.net> (directory `/inet/doc`). The files are Unix-compressed ASCII, so they should be downloaded using binary mode and then decoded with a Unix-style uncompress application. See the next section for a primer on ftp. Note that newer editions of "Zen" are only available commercially.

Another popular and very complete reference is "The Whole Internet User's Guide and Catalog", also by Ed Krol, published by O'Reilly and Associates.

A three-volume series "Internetworking with TCP/IP" by Douglas Comer is published by Prentice Hall, and contains everything you have ever wanted to know about this protocol.

1. Downloading text files

Use any account available to you on a well-connected host. Type `"ftp sumex-aim.stanford.edu"` (or if that doesn't work, try `"ftp 36.44.0.6"`). Type "anonymous" as login name, and your mail address as password. In 9 cases out of 10, archives such as this one accept "ftp" in place of "anonymous", which means that you can delay the onset of carpal tunnel syndrome. You will also discover that many sites will accept any password at all, but let's be nice to those folks who specifically ask for the real id.

Enter `"cd /info-mac/report"`, and then `"get ftp-primer.txt"`. When you see "Transfer complete", type "quit" and read the file you just downloaded.

After becoming skilled in using ftp, download some more text files from the `/info-mac` directory:

```
help/accessing-files.txt
report/how-do-i-find.txt
```

report/internet-access-11.hqx
report/ftp-sites.txt

A Macintosh communications FAQ ("frequently asked questions") list is an invaluable source of information. It is posted periodically in the Usenet newsgroup comp.sys.mac.comm; it is also available for anonymous ftp on rascal.ics.utexas.edu as communications.FAQ in the directory mac/faq.

The anonymous ftp site nnsc.nsf.net run by the National Science Foundation is a good source of background information about the Internet (see directories /nsfnnet and /info).

2. What next?

When you have learned how to download Macintosh executable files, it's time to go hunting for specific applications.

WARNING

The minute you start downloading files from the network, you become more susceptible to viral infection than before. I strongly suggest that you should first get and set up the wonderful free virus checker, Disinfectant. Its home is ftp.nwu.edu, in directory /pub/disinfectant; it can also be found in many other places, e.g. on sumex-aim.stanford.edu in the directory /info-mac/virus. You may then want to send its author, John Norstad, a nice thank-you note: we all owe him a great deal! If you have access to Usenet news, make it a habit to monitor the comp.sys.mac.announce group: it is probably the most reliable source of information about newly discovered viruses.

Here is a list of TCP/IP-related applications (most of which were mentioned above) which can be found on sumex in the directory /info-mac. There are many servers which maintain copies of /info-mac contents, such as wuarchive.wustl.edu in /mirrors/info-mac. Version numbers are current as of this writing.

comm/archie-client-09.hqx
comm/calypso-tool-10f1c2.hqx
comm/chat-11.hqx
comm/fern-mail-11.hqx
comm/fetch-21.hqx
comm/finger-135.hqx
comm/ftpd-201.hqx
comm/gopher-app-13b52.hqx
comm/hyperftp-13.hqx
comm/ircle-10.hqx
comm/lee-mail-124.hqx

comm/mac-ppp-10.hqx
comm/ncsa-telnet-25.hqx
comm/news-watcher-13d6.hqx

comm/nuntius-111d17.hqx
 comm/query-it-11.hqx
 comm/talk-106.hqx
 comm/the-news-222.hqx
 comm/turbo-gopher-105.hqx
 comm/xferit-14.hqx
 cp/network-time-111.hqx
 game/game-master.hqx

Software found on ftp servers other than sumex:

ClientDA:	ssyx.ucsc.edu (supersrv is there as well)
Eudora:	ftp.cso.uiuc.edu in /pub/mac/eudora (also on ftp.qualcomm.com)
GopherApp:	ftp.bio.indiana.edu in /util/gopher
MacDump:	bbn.com
MacPost:	pollux.lu.se in /pub/mac/comm
MacTCP+ patch:	ftp.apple.com in /dts/mac/netcomm
MacTCP T/R ext.:	ftp.apple.com in /dts/mac/netcomm
MPing:	as above, file name mactcp-1-1-examples.hqx
NCSA/BYU Telnet:	bert.cs.byu.edu
NetNews:	ftp.bio.indiana.edu in /util/mac
POPmail:	boombox.micro.umn.edu, in /pub/POPmail
popper:	ftp.cc.berkeley.edu
SU-Mac/IP:	send e-mail to macip@jessica.stanford.edu
TechInfo:	net-dist.mit.edu in /pub/TechInfo
TechMail:	net-dist.mit.edu in /pub/TechMail
tn3270:	brownvm.brown.edu
TurboGopher:	boombox.micro.umn.edu in /pub/gopher
WAIStation:	think.com in /wais

Note that Apple likes to distribute some of its software as "disk image" files. Such files have to be loaded into an application called DiskCopy (available on ftp.apple.com in /dts/utlis), which can then produce exact copies of an original master floppy disk.

A list of anonymous FTP sites (not only Macintosh-related) is maintained on pilot.njin.net. Many Unix hosts (and an increasing number of other machines) will also let you search an archie database of anonymous ftp sites.

This should keep you busy for now...

3. Contacts for commercial products

Please remember that I do not in any way endorse commercial products by listing the vendors here, and I'm not in any way associated with them. This is simply for your

convenience. Phone numbers are followed by an anonymous ftp server which can be used to get free upgrades and information, or an e-mail address.

Cayman Systems	(800) 473 4776 ftp.cayman.com
Compatible Systems	(800) 356 0283 csn.org, in /compatible
Farallon Computing	(800) 344 7489 farallon.com
Intercon Systems	(703) 709 9890 sales@intercon.com
Shiva (800) 458 3550	shiva.com
Synergy Software	(215) 779 0522 D2296@AppleLink.Apple.COM
Webster Computer Corp	
(Australia)	+61 3 764 1100
USA	(408) 954-8054 wcc@cup.portal.com

4. Let's end with a short sermon...

Many of the applications mentioned above are NOT in public domain. They are either shareware, or there are restrictions on their use and/or distribution. **PLEASE PAY FOR SHAREWARE YOU KEEP!!!** Author's address can almost always be found by pulling down the Apple menu and selecting "About..." Let's keep this wonderful, affordable software alive!

Appendix A. Peculiarities of Mac file transfers

Macintosh files differ from files on most other machines in that they consist of two parts. One contains data (text, executable program), and the other resources (icons, the file's creator code, etc.) I'm simplifying a little, but never mind. This complicated structure prevents us from sharing such files directly over the network.

Moreover, there is only one language which practically all computers understand: the ASCII code (plain text). Even though this isn't a terribly elegant solution, we simply bring everything to this lowest common denominator to assure compatibility. For example, in order to send a file to someone by the current e-mail systems, it has to be somehow encoded into an ASCII file.

The Mac community has pretty much agreed on a common standard for doing just that: BinHex. BinHex swallows a Mac file, icons, file creators and all, and converts all that into a plain text file filled with something that looks like garbage; it also performs the reverse procedure. So you need BinHex.

There is one obvious difficulty, however: how do you get a BinHex decoder (a Macintosh application!), when you don't have BinHex? You will also need software which will somehow let your Mac do FTP. The easiest way to "bootstrap" yourself is to simply get a copy of such a beast from a local Mac guru or a Mac User Group. If you're lucky, you will lay your hands on a utility which can not only transfer files, but also un-BinHex them —such as XferIt, Fetch, or one of the Gopher clients. You can then tell it to connect directly to, say, sumex, download the interesting BinHex'ed files, and decode

them while they arrive. Similarly, newer Gopher clients such as the GopherApp and TurboGopher can download and decode BinHex'ed files.

Another way is to get NCSA Telnet, log on to a friendly Internet machine, download the applications you need to that computer in **binary** form (e.g. the file binhex4.bin available on sumex in /info-mac/util) using the binary mode in ftp. Then connect back to your Mac using the Telnet FTP server and put the files on the Mac using the MacBinary mode... It sounds (and is) a bit complicated, but remember —this convoluted process is necessary only in the very beginning.

Once again, we have just licked the surface of this topic here. For more information, see the file /info-mac/report/ftp-primer.txt on sumex-aim.stanford.edu.

Appendix B. Mac networking: mini-tour

In the networking world, it is easy to drown in the alphabet soup and the sea of obscure terms. But understanding the process by which computers communicate helps troubleshoot problems. We will go through some elementary information in this appendix. Things may get a bit confusing, and you may want to read what follows two or three times until it makes sense...

For two digital devices to talk to each other, there must be a physical connection between them (a wire, optical fiber, radio link, etc.) and an agreement as to the logical organization of the information. In computerese, such mutual understanding is usually called a "communications protocol".

Apple's Macintoshes use a "native" set of protocols, collectively known as AppleTalk. The physical aspect of AppleTalk is, very simply, the kind of wires the device uses. If you stick a little round connector into the printer port of your Mac, or in the jack in your LaserWriter, you are using AppleTalk over Apple's original slow wiring, i.e. LocalTalk. Almost nobody uses that now —even Apple's own network uses Farallon's Phonenet system, but that is a technical detail. You are on LocalTalk.

If your Mac has an Ethernet card or external adapter attached, you will be using AppleTalk on a physical Ethernet network; that is called EtherTalk. Similarly, if you install a Token Ring adapter, the incarnation of the AppleTalk protocol you are dealing with is called TokenTalk.

The logical layer of AppleTalk handles details such as how two Macs can discover each other on the network, how individual nodes are uniquely identified, what should a Mac say to a printer or an AppleShare file server when it wants to use it, and so on. The devices you see in the Chooser (LaserWriters, file servers, or routers such as Netway or SNA/ps) are all AppleTalk devices. The beauty of AppleTalk is that you don't really care what physical method you use. You may see a printer on LocalTalk, or a LaserWriter IIg on Ethernet, or a Netway box on Token Ring—it doesn't matter.

But the nitty-gritty of how the actual network operates does vary from one kind of wire to another. The computer has to behave differently on each kind of network, but of course you don't want to know about that! Enter "network drivers": low-level pieces of system software which take care of that. When you put in an Ethernet card, you need to install the EtherTalk drivers in your system. Same with TokenTalk drivers. It's like speaking with someone over the phone, or on a walkie-talkie. The principle is different, but the message is the same.

There are more and more ways of making non-Apple devices speak AppleTalk: that's why there are MS DOS computers on LocalTalk networks, Novell AppleShare servers, and you can configure your Sun Sparcstation to print on an Apple LaserWriter. The Internet world, however, doesn't know the first thing about AppleTalk. It only understands the collection of protocols known as TCP/IP. What does TCP/IP have to do with AppleTalk? The answer is "not much", and "nothing" most of the time. Putting a Mac on a TCP/IP network is like dumping an Englishman in the center of Beijing: there is a language barrier.

MacTCP, the gadget we are discussing in this guide, allows Mac applications to use network interfaces —such as the built-in LocalTalk port or an Ethernet card— to transmit and receive data packets which contain TCP/IP information, and hence to communicate with the millions of other TCP/IP computers on this planet.

Just like Apple came up with specifications for sending the high-level (AppleTalk) data using the various low-level, network-specific "transport protocols" (Ethernet, Token Ring etc.), the Internet has standards for sending the high-level TCP/IP data using the low-level network mechanisms. Ethernet is the best choice, since the Internet protocols were "born" on Ethernet, so those standards are well-established.

Apple adopted a certain way of sending TCP/IP data wrapped inside LocalTalk packets, and MacTCP knows how to handle this. It puts ("encapsulates") TCP/IP information into a normal LocalTalk packet, and sends it out. That packet makes absolutely no sense to any AppleTalk device (it looks like it has garbage inside), except those which use MacTCP to do the reverse decoding.

A standard for transmitting TCP/IP data in Token Ring packets has also existed for quite some time. But MacTCP did not know about it, until Apple released an add-on "MacTCP Token Ring Extension", which —again— takes a TCP/IP packet and beats it into shape before sending it through a Token Ring card.

To make things more interesting, MacTCP used on Ethernet (or Token Ring) is capable of two different behaviors: it can take the TCP/IP data and spit it out unadorned, according to the usual, world-savvy IP-on-Ethernet or IP-on-Token Ring recipe. But it can also be set to use EtherTalk (respectively, TokenTalk), which means that it will activate the wrapping/unwrapping AppleTalk filter between the network interface and the application software! The general idea is **not** to use EtherTalk or TokenTalk in MacTCP. The reason should become clear soon.

To summarize, here are some scenarios. **(a)** Mac on Ethernet, MacTCP correctly installed, "Ethernet built-in" set in the MacTCP control panel; **(b)** Mac on Token Ring, MacTCP installed, the Token Ring driver installed and configured properly, TokenRing selected in MacTCP. All is peachy. A TCP/IP-aware application on the Mac (such as NCSA Telnet, etc. —see Chapter III) wants to communicate with a Cray at the Space Station "Freedom", which by the time I finish this will be in orbit. It tells MacTCP to send out a Telnet packet, MacTCP translates it into the standard format for Ethernet or Token Ring, some gateways down the road convert it to the TCP/IP over radio waves form, the packets gets to the Cray, it says "Aha! we've got a hacker trying to log in here", and so it goes.

Now for something more mundane. **(c)** Mac Plus on LocalTalk, MacTCP installed, "LocalTalk built-in" selected in the MacTCP panel (the only possible choice!). A TCP/IP "datagram" goes out after being wrapped into an AppleTalk packet! Now, the LocalTalk is no doubt connected to the outside world one way or another. If that's done using a relatively unsophisticated device, it will take the data and simply convert it to an equivalent EtherTalk, TokenTalk, or whatever packet. That one goes out alright, gets up to the Cray, but it now says: "Phooey, that's something I don't understand! It has some strange stuff inside! Let's quickly drop it on the floor." The reason is that most Internet hosts, like our Cray, are not instructed by their software to go deeper inside the packet and actually recognize that it was TCP/IP information wrapped inside AppleTalk... and why should they bother? What is needed is a more sophisticated gateway between the LocalTalk and the outside network.

Scenario **(d)**: Mac Plus on LocalTalk sends a TCP/IP-in-AppleTalk packet which is directed towards a "DDP-IP gateway", such as the Fastpath, Gatorbox, EtherRoute/TCP, etc. The gateway's software is smart enough to look under covers and see what is hiding inside. If it sees TCP/IP data wrapped inside AppleTalk, it strips the outer layer and passes the raw IP information in the standard format to the Ethernet network. All is well again! LocalTalk-to- Ethernet gateways like that are common, but equivalent ones for Token Ring are still scarce and expensive.

How about **(e)**: just like in scenarios (a) or (b), except MacTCP is set to use EtherTalk (or TokenTalk). Now regular TCP/IP packets will not be coming through —MacTCP simply ignores them! It expects AppleTalk packets only. It will be able to communicate with the Mac Plus in (c), but not much else. So let's just forget it and stop here...

Appendix C. Dial-in access

In our personal opinion, the most elegant way to connect a Mac to a TCP/IP network is AppleTalk Remote Access (ARA), a commercial product of Apple Computer, bundled with PowerBooks, and sold as a separate product.

ARA uses the Communications Toolbox (built into System 7, and installable in System 6.0.x) to ship AppleTalk packets over a modem to an ARA server, which is presumably

connected to a "real" network. MacTCP in turn uses the AppleTalk protocol to transmit "wrapped" TCP/IP packets (if it is configured to communicate via AppleTalk). This results in a two-stage translation: TCP/IP-to- AppleTalk, and AppleTalk-to-modem. The data have to be decoded by a reverse process at the other end.

This explains the only major drawback of ARA: speed. A 2400 baud modem is next to unusable in this configuration. But a 9600 baud connection should provide decent response even with the additional IP encapsulation.

The server Mac, whether it's on Ethernet or LocalTalk, spews out AppleTalk packets, from which the TCP/IP information has to be reconstructed by an IP gateway. If you don't have a gateway such as the Fastpath, GatorBox, EtherRoute, or MultiGate, you can't use ARA for TCP/IP access to the network.

Since for most of us the primary Internet application is e-mail, we will mention the "poor man's" dial-up connections which can be used with the mail client Eudora. You need to know how to edit a program's resources to patch Eudora in this fashion —a task which is not for the faint-hearted. The two alternatives are:

- i) modify "stock" Eudora to navigate through a simple modem connection;
- ii) use the Simon Fraser University version of Eudora and create an intelligent connection script.

The first method is described in the Eudora Q&A stack available from [ftp.qualcomm.com](ftp://qualcomm.com) and [ftp.cso.uiuc.edu](ftp://cso.uiuc.edu). Additional information is in an appendix in the Eudora documentation, available from the same sources. We have successfully used this scheme with a direct modem connection to a Unix POP server.

The second way is to learn a simple C-like programming language which the SFU Eudora understands, and use it to get through the maze of terminal servers and multiple logins. This version of Eudora is available from fraser.sfu.ca.

A much more "modular" and reliable option is to make use of a Communications Toolbox module which can interpret so-called CCL scripts. One such tool is Calypso, which acts as a buffer of sorts between the application itself (e.g. Eudora), and the actual communications tool such as the Apple Modem Tool. The progress of the connection is then controlled by a program (written in CCL) which

resides in an ordinary editable text file. Learning CCL is not hard, given that many sample CCL scripts can be found on sumex and elsewhere.

The most popular dial-in connection schemes, however, employ protocols developed specifically for that purpose, such as SLIP or PPP. I have seen a Communications Toolbox PPP tool, but lacking access to a PPP server I cannot say anything about it. SLIP was recently made available on the Mac in the form of several commercial offerings from Intercon, Synergy Software, and TriSoft. More information on SLIP can be found on <ftp.bio.indiana.edu> in the directory /util/slip.

A suitably configured SLIP connection gives the dial-in Macintosh all the functionality of a node attached directly to a TCP/IP network, even though it is of course usually much slower. For example, Alan Piszcz (alan@kaman.com) described his 9600 baud dial-in setup as follows:

REMOTE SITE					OFFICE			
<hr/>					<hr/>			
MAC	II--MacTCP--MacSLIP--Telebit===Telebit--PC	NFS--UNIX--SUN						
SPARC								
Sys. 7	1.1	1.0	T2500	T2500	SLIP 4.0	TCP/IP	4.1.1	

Last but definitely not least –Patrick Hoepfner kindly allowed me to include his introduction to the serial Internet protocols. Here goes:

SLIP or Serial Line Internet Protocol is a method of pushing IP packets down a asynchronous (usually phone) line. SLIP is a self described "non-standard" that was developed on the back of an envelope and implemented in 1/2 a day. SLIP simply sticks a single character in front of the IP packet. Because you have binary IP packets over this connection, this has to be an 8-bit clean connection. You have to have both modems using 8-bit, no parity, and no software flow control. Because of this, many may not want to invest in software to do this if your organization doesn't have modems that will support this.

SLIP requires that the Mac SLIP its IP packets. VersaTerm, MacSLIP, and InterCon supply SLIP software. The only difference (other than cost) is that VersaTerm SLIP and MacSLIP will work with any MacTCP software. InterCon uses its own proprietary TCP/IP driver and consequently it will SLIP only its own tools that use its TCP/IP drivers. But then again, InterCon can supply most all of the tools that you need. Since all of the SLIP products for the Mac are commercial products, you should refer to their documentation for information on setting it up on your Mac.

For a successful SLIP connection you have to have a machine that will host the SLIP connection. That can be a unix machine, a VMS machine, or even a terminal server. There are public domain versions

of SLIP for unix machines that are available, check your local archie server for more information. If you have a Vax running VMS and you have a TCP/IP package installed on it, chances are good that it too will host a SLIP connection. (See your local system administrator for more information.) And if you have a terminal server, chances are good that it too will support a SLIP connection (but not a CSLIP connection —see below for CSLIP information and see the terminal server's administrator for more details.)

CSLIP or Compressed SLIP compresses the TCP/IP headers that are sent with each packet. The header are much the same each time and when using CSLIP, each machine keeps a copy of the last header and only transmits the differences between the last one and the current one.

This can make the difference between a 4-5 byte header and the standard 40 byte header.

PPP or Point to Point Protocol is kind of like SLIP done right. Among other wondrous things that PPP can do is to send not just IP packets, but it is able to accept other types of protocols... Maybe we will see AppleTalk over PPP. Maybe with the next release of AppleTalk Remote Access and MacTCP??? Maybe the version after that...

How come I know so much about SLIP, CSLIP, and PPP? Other than being a genius and terribly handsome (and sometimes able to spell). I was pointed to two documents at "ftp.uu.net" in the "vendor/MorningStar/papers" directory. These unix compressed PostScript files are named "sug91-cheapIP.ps.Z" and "ppp-white-paper.ps.Z".